Remote Soil Moisture Monitor Using IoT

L.Rama Devi
Assistant Professor
SVIT ,
Secunderabad-500003.

D.Srivalli
Assistant Professor
SVIT ,
Secunderabad-500003.

Satya Sri N.S
Assistant Professor
SVIT,
Secunderabad-500003.

D.Badhru
Assistant Professor
SVIT ,
Secunderabad-500003.

Abstract: *Soil moisture sensors measure the volumetric water content in soil. Since the direct gravimetric measurement of free soil moisture requires removing, drying, and weighting of a sample, soil moisture sensors measure the volumetric water content indirectly by using some other property of the soil, such as electrical resistance, dielectric constant, or interaction with neutrons, as a proxy for the moisture content. The relation between the measured property and soil moisture must be calibrated and may vary depending on environmental factors such as soil type, temperature, or electric conductivity. Reflected microwave radiation is affected by the soil moisture and is used for remote sensing in hydrology and agriculture. Portable probe instruments can be used by farmers or gardeners. Soil moisture sensors typically refer to sensors that estimate volumetric water content. Another class of sensors measure another property of moisture in soils called water potential; these sensors are usually referred to as soil water potential sensors. Measuring soil moisture is important for agricultural applications to help farmers manage their irrigation systems more efficiently. Knowing the exact soil moisture conditions on their fields, not only are farmers able to generally use less water to grow a crop, they are also able to increase yields and the quality of the crop by improved management of soil moisture during critical plant growth stages*

Keywords: moisture, sensors, calibrated, hydrology.

I. Introduction

Soil moisture sensors measure the volumetric water content in soil.[1] Since the direct gravimetric measurement of free soil moisture requires removing, drying, and weighting of a sample, soil moisture sensors measure the volumetric water content indirectly by using some other property of the soil, such as electrical resistance, dielectric constant, or interaction with neutrons, as a proxy for the moisture content. Reflected microwave radiation

is affected by the soil moisture and is used for remote sensing in hydrology and agriculture. Portable probe instruments can be used by farmers or gardeners.

Soil moisture sensors typically refer to sensors that estimate volumetric water content. Another class of sensors measure another property of moisture in soils called water potential; these sensors are usually referred to as soil water potential sensors and include tensio meters and gypsum blocks.

> Measuring soil moisture is important for agricultural applications to help farmers manage their irrigation systems more efficiently. Knowing the exact soil moisture conditions on their fields, not only are farmers able to generally use less water to grow a crop, they are also able to increase yields and the quality of the crop by improved management of soil moisture during critical plant growth stages. Soil moisture sensors are used in numerous research applications, e.g. in agricultural science and horticulture including irrigation planning, climate research, or environmental science including solute transport studies and as auxiliary sensors for soil respiration measurements.

Acclaim SCX Soil Moisture Sensor and irrigation override controller.

> Conserve water with precision irrigation, upgrade your controller with the power of Digital TDT® moisture sensors.
>
> The SCX works with your existing electric valve irrigation timer, incorporating cutting edge moisture sensing technology to prevent over watering. A timer or clock has traditionally governed automated irrigation control systems. Day in and day out, rain or shine, these clocks faithfully water our landscapes.
>
> The weatherproof cabinet is suitable for outdoor installation. The SCX overcomes the problem of chronic over watering through use of the Acclima Digital TDT® Soil Moisture Sensor. The sensor measures soil moisture content every 10 minutes and will only allow a water cycle when the moisture drops below

a "turn on" threshold that you have set, suspending water cycles until your turf needs it. During the hottest part of the year, soil moisture is depleted more rapidly and the system will water more frequently. Then when the temperature cools, or if you get a big rainstorm, the system will inhibit watering until the moisture level in the ground falls below the "turn on" threshold. The SCX provides a history of the last seven irrigation attempts.

The proposed hardware of this system includes 8 bit AVR, Blue tooth module, and soil moisture sensors. The system is low cost & low power consuming so that anybody can afford it. The data monitored is collected at the server. It can be used in precision farming. The system should be designed in such a way that even illiterate villagers can operate it. They themselves can check different parameters of the soil like salinity, acidity, moisture etc. from time to time. During irrigation period, they have to monitor their distant pump house throughout the night as the electricity supply is not consistent. The system can be installed at the pump house located remotely from the village, it is interfaced with the pump starter & sensors are plugged at different location in the field for data acquisition. Using this system, they can switch on their pump from their home whenever they want.

Soil moisture sensor

The soil moisture sensor used is capacitive type. The sensor gives analog output of zero volt when there is 100% moisture and 5V for 0% moisture.

ABOUT ARDUINO

The Arduino Micro is a microcontroller board based on the ATmega32U4 (datasheet), developed in conjunction with Ad fruit. It has 20 digital input/output pins (of which 7 can be used as PWM outputs and 12 as analog inputs), a 16 MHz crystal oscillator, a micro USB connection, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a micro USB cable to get started. It has a form factor that enables it to be easily placed on a breadboard.

The Micro board is similar to the Arduino Leonardo in that the ATmega32U4 has built-in USB communication, eliminating the need for a secondary processor. This allows the Micro to appear to a connected computer as a mouse and keyboard, in addition to a virtual (CDC) serial / COM port.

The Arduino Micro differ from other Arduino boards in that they use a single microcontroller to both run your sketches and for USB communication with the computer. The Uno and other boards use separate microcontrollers for these two functions, meaning that the USB connection to the computer remains established regardless of the state of the main microcontroller. By combining these two functions onto a single processor, the Leonardo allows for more flexibility in its communication with the computer. It also helps to lower the cost of the board by removing the need for an additional processor.

Serial re-enumeration on reset.

Since the boards do not have a dedicated chip to handle serial communication, it means that the serial port is virtual -- it's a software routine, both on your operating system, and on the board itself. Just as your computer creates an instance of the serial port driver when you plug in any Arduino, the Leonardo/Micro creates a serial instance whenever it runs its bootloader. The board is an instance of USB's Connected Device Class (CDC) driver.

This means that every time you reset the board, the USB serial connection will be broken and re-established. The board will disappear from the list of serial ports, and the list will re-enumerate. Any program that has an open serial connection to the Leonardo will lose its connection. This is in contrast to the Arduino Uno, with which you can reset the main processor (the ATmega328P) without closing the USB connection (which is maintained by the secondary ATmega8U2 or ATmega16U2 processor). This difference has implications for driver installation, uploading, and communication.

No reset when you open the serial port.

Unlike the Arduino Uno, the Leonardo and Micro won't restart your sketch when you open a serial port on the computer. That means you won't see serial data that's already been sent to the computer by the board, including, for example, most data sent in the setup () function.

This change means that if you're using any Serial print (), println () or write() statements in your setup, they won't show up when you open the serial monitor.

Separation of USB and serial communication.

On the Leonardo and Micro, the main **Serial** class refers to the virtual serial driver on the board for connection to your computer over USB. It's not connected to the physical pins 0 and 1 as it is on the Uno and earlier boards. To use the hardware serial port (pins 0 and 1, RX and TX), use Serial1. (See the Serial reference pages for more information.)

Writing Sketches

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension.ino.the editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

Verify
Checks your code for errors compiling it.

Upload
Compiles your code and uploads it to the configured board. See uploading below for details.

Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"

New
Creates a new sketch.

Open
Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketchbook menu instead.

Save
Saves your sketch.

Serial Monitor
Opens the serial monitor.

Additional commands are found within the five menus: **File**, **Edit**, **Sketch**, **Tools**, **Help**. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

File

- *New*
  Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.
- *Open*
  Allows to load a sketch file browsing through the computer drives and folders.
- *Open Recent*
  Provides a short list of the most recent sketches, ready to be opened.
- *Sketchbook*
  Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.
- *Examples*
  Any example provided by the Arduino Software (IDE) or library shows up in this menu item. All the examples are structured in a tree that allows easy access by topic or library.
- *Close*
  Closes the instance of the Arduino Software from which it is clicked.
- *Save*
  Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as.." window.
- *Save as...*
  Allows to save the current sketch with a different name.
- *Page Setup*
  It shows the Page Setup window for printing.
- *Print*
  Sends the current sketch to the printer

according to the settings defined in Page Setup.

- *Preferences*
  Opens the Preferences window where some settings of the IDE may be customized, as the language of the IDE interface.

- *Quit*
  Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.

Edit

- *Undo/Redo*
  Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.

- *Cut*
  Removes the selected text from the editor and places it into the clipboard.

- *Copy*
  Duplicates the selected text in the editor and places it into the clipboard.

- *Copy for Forum*
  Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax coloring.

- *Copy as HTML*
  Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages.

- *Paste*
  Puts the contents of the clipboard at the cursor position, in the editor.

- *Select All*
  Selects and highlights the whole content of the editor.

- *Comment/Uncomment*
  Puts or removes the // comment marker at the beginning of each selected line.

- *Increase/Decrease Indent*
  Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.

- *Find*
  Opens the Find and Replace window where you can specify text to search inside the current sketch according to several options.

- *Find Next*
  Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position.

- *Find Previous*
  Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.

Sketch

- *Verify/Compile*
  Checks your sketch for errors compiling it; it will report memory usage for code and variables in the console area.

- *Upload*
  Compiles and loads the binary file onto the configured board through the configured Port.

- *Upload Using Programmer*
  This will overwrite the bootloader on the board; you will need to use Tools > Burn Bootloader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a *Tools -> Burn Bootloader* command must be executed.

- *Export Compiled Binary*
  Saves a. hex file that may be kept as archive or sent to the board using other tools.

- *Show Sketch Folder*
  Opens the current sketch folder.

- *Include Library*
  Adds a library to your sketch by inserting #include statements at the start of your code. For more details, see libraries below. Additionally, from this menu item you can access the Library Manager and import new libraries from .zip files.

- *Add File...*
  Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu accessible

clicking on the small triangle icon below the serial monitor one on the right side of the toolbar.

Tools

- *Auto Format*
  This formats your code nicely: i.e. indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more.

- *Archive Sketch*
  Archives a copy of the current sketch in .zip format. The archive is placed in the same directory as the sketch.

- *Fix Encoding & Reload*
  Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.

- *Serial Monitor*
  Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over serial port opening.

- *Board*
  Select the board that you're using. See below for descriptions of the various boards.

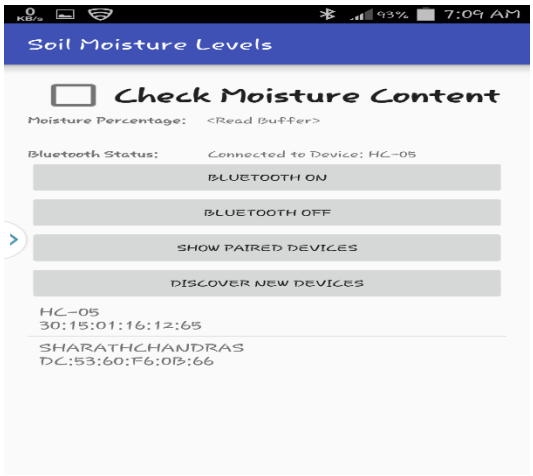
- *Port*
  This menu contains all the serial devices

(real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.

- *Programmer*
  For selecting a Hardware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't need this, but if you're burning a bootloader to a new microcontroller, you will use this.

- *Burn Bootloader*
  The items in this menu allow you to burn a bootloader onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino or Genuino board but is useful if you purchase a new ATmega microcontroller (which normally come without a bootloader). Ensure that you've selected the correct board from the Boards menu before burning the bootloader on the target board. This command also set the right fuses.

-

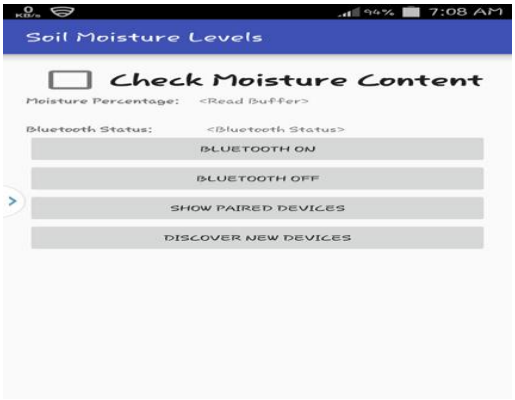


Help

Here you find easy access to a number of documents that come with the Arduino Software (IDE). You have access to Getting Started, Reference, this guide to the IDE and other documents locally, without an internet connection. The documents are a local copy of the online ones and may link back to our online website.

- Find in Reference
This is the only interactive function of the Help menu: it directly selects the relevant page in the local copy of the Reference for the function or command under the cursor.

RESULT ANALYSIS



8.CONCLUSION

This Soil Moisture Sensor based agriculture monitoring system serves as a reliable and efficient system for monitoring agricultural parameters. The corrective action can be taken. Wireless monitoring of field not only allows user to reduce the human power, but it also allows user to see accurate changes in it. It is cheaper in cost and consumes less power. The GDP per capita in agro sector can be increased. This project can be extended for cattle monitoring. Automatic irrigation system uses an 8051-series microcontroller which is programmed to receive the input signal of varying moisture condition of the soil through the sensing arrangement. This is achieved by using an op-amp as comparator which acts as interface between the sensing arrangement and the microcontroller. Once the controller receives this signal, it generates an output that drives a relay for operating the water pump. An LCD display is also interfaced to the microcontroller to display status of the soil and water pump. The sensing arrangement is made by using two stiff metallic rods inserted into the field at a distance. Connections from the metallic rods are interfaced to the control unit. The concept in future can be enhanced by integrating GSM technology, such that whenever the water pump switches ON/OFF, an SMS is delivered to the concerned person regarding the status of the pump. We can also control the pump through SMS.

11. REFERENCES

https://developer.android.com/studio/intro/index.html
https://developer.android.com/guide/index.html
https://github.com/bauerjj/Android-Simple-Bluetooth-Example
http://android-er.blogspot.in/2015/07/android-example-to-communicate-with.html
https://create.arduino.cc/projecthub/evive-an-opensource-embedded-platoform/plant-monitoring-and-watering-system-using-evive-d48677
https://www.tindie.com/products/miceuz/i2c-soil-moisture-sensor/
https://atmelcorporation.wordpress.com/tag/esp8266/
http://www.electroschematics.com/12065/arduino-soil-moisture-sensor-module/
http://www.instructables.com/id/Arduino-NanoSoil-Moisture-SensorLCD/?ALLSTEPS
http://www.c-sharpcorner.com/UploadFile/167ad2/arduino-soil-moisture/
https://circuitdigest.com/microcontroller-projects/arduino-automatic-plant-watering-system
http://runtimeprojects.com/2016/03/plants-monitor-soil-moisture-sensor/
http://turf-tec.com/Acclima-SCX-lit.html