

République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur Et de La Recherche Scientifique



Université de Ghardaïa

Faculté des Sciences et Technologie

Département des Sciences et Technologie

N° d'ordre :

N° de série:

Projet de fin d'étude présenté en vue de l'obtention du diplôme de

MASTER

Domaine : Science et Technologie

Filière : Automatique

Spécialité : Automatique

Thème:

Process Supervising by the Microcontroller Arduino

PAR:

KHOBZI MOHAMMED

Jury:

Mr: KIFOUCHE Abdessalam	Maitre Assistant A, Univ. Ghardaia	Président
Mr: TOUAFEK Khaled	Maitre de Recherche A, URAER Ghardaia	Encadreur
Mr: HAMID OUDJANA Samir	Maitre Assistant A, URAER Ghardaia	Examineur
Mr REZAK Daoud	Maitre Assistant A, URAER Ghardaia	Examineur

ANNEE UNIVERSITAIRE: 2015/2016

• Summary

This memoir note is for manufacturing a prototype of a device that task full of fluid in the reservoirs or basins, measure their levels automatically, and show the results on the TV screen. The machine is a catcher for the fluid level and explorers to the presence of fluid connected microcontroller Arduino Inoue and the latter is plugged into microcontroller one full and one for emptying the tank or tub and bell-screen TV. It is committed to houses, stables, and even gardens. Used cutting industry who take the most are reused pieces can be found in the home and shops selling second-hand pieces easy to manufacture and device programming and even maintenance .it cost reasonable compared to what there is in the market that are less tasks are usually more expensive than this device.

• ملخص

هذه المذكرة هي عن صناعة نموذج أولي لجهاز يقوم مهمة ملئ السوائل في الخزانات أو الأحواض وقياس مستوياتها بطريقة آلية وإظهارها النتائج على شاشة تلفاز. الجهاز هو عبارة عن ملتقط لمستوى السوائل ومكتشفين لوجود السوائل موصولة بميكروكنترولر أردوينو اينو وهذا الأخير موصول بالكتروفانين واحد لمليء والاخر للتفريغ الخزان أو الحوض وجرس وشاشة تلفاز. الجهاز مخصص للبيوت والإسطبلات وحتى الحدائق. جل القطع المستعملة لصناعة الملتقطات هي قطع معاد استعمالها يمكن ايجادها في البيت ومحلات بيع القطع المستعملة الجهاز سهل الصنع والبرمجة وحتى الصيانة وتكلفته معقولة مقارنة مع ما يوجد في السوق التي تقوم بمهمات اقل وعادة ما تكون أعلى من هذا الجهاز.

• Résumé

Ce mémoire constitue un travail de recherche approfondi destiné à fabriqué un prototype d'un dispositif qui tâche pleine de liquide dans les réservoirs ou bassins et de mesurer leurs niveaux automatiquement et affiche les résultats sur l'écran du téléviseur. La machine est un receveur pour le niveau du liquide et des explorateurs à la présence de fluide relié microcontrôleur Arduino uno et celui-ci est branché sur des électrovannes à remplir et vidange du réservoir ou de la baignoire et la cloche à l'écran TV. Ce dispositif est désigné des maisons et des écuries et même des jardins. La plupart des morceaux peuvent être trouvés dans la maison et les magasins de vente de seconde main pièces .ce dispositif est faciles à fabriquer et à programmé et même à entretué et le coût raisonnable par rapport à ce qu'il ya dans le marché qui sont moins des tâches. Et que ils sont généralement plus chers que ce dispositif.

This humble project is dedicated

To my father and mother, my brothers and sisters and all my family members and to my Teachers and my project supervisor

D. Khaled TOUAFEK.

To my friends who keep encouraging me whenever I lose
motivation

Mohammed alamin and Hacene CHIKH BELHADJ

Neceradine HARIZ.

To my schoolmates

Mohammed MERMOURI

Nacreddine ELKIOUAS

Who helped me through this project

To all my friends and mates

Thank you for your support.

May Allah bless you all.

Contunent

• introduction	1
• contunent	2
1. Chapter 1: The state of art of the Microcontroller Arduino	3
1.1. The history of Arduino	3
1.2. .What is Arduino?	3
1.3. Why Arduino?	4
1.4. Exploring the Arduino Ecosystem	5
1.4.1. Arduino Functionality	6
1.4.2. Atmel Microcontroller	6
1.4.3. Programming Interfaces	7
1.4.4. General I/O and ADCs	7
1.4.5. Power Supplies	8
1.5. Arduino Boards	8
1.5.1. Downloading and Installing Arduino	10
1.5.2. Installing Arduino	10
1.5.2.1. Installing Arduino for Windows	10
1.6. Arduino Software (IDE)	12
1.6.1. . Writing Sketches	12
1.6.1.1. Breaking down example first program	20
1.7. shields	23
1.7.1. example of shields	23
1.7.1.1. The Arduino GSM Shield	23
1.7.1.1.1. Network operator requirements	24
1.7.1.1.2. SIM cards	24
1.7.1.1.3. Connecting the Shield	24

1.7.1.1.4. GSM Library	25
1.7.1.2. Arduino Wifi Shield	25
Conclusion	26
Chapter 2 : State of art about control	27
Introduction	27
2.1. Control	27
2.1.1. Control engineering	27
2.1.2. Control theory	28
2.1.3.The role of control theory	28
2.1.4.Control Modeling	29
2.1.4.1 Control Models	30
2.1.4.2Advanced Control Models	31
2.2.Physical Process	32
2.2.1. Natural Processes	32
2.2.2. Self-Regulated Processes	32
2.2.3. Man-made or Industrial Processes	33
2.3.Signals	33
2.3.1. Pneumatic Signals	33
2.3.2. Analog Signal	33
2.3.3. Digital Signal	33
2.3.4. Pulse	34
2.4. Process Control	34
2.4.1 Sequential Process Control	34
2.4.2. Continuous Process Control	35
2.5. Control system	35
2.5.1.Control Loop	36
2.5.2.Open-loop control	36
2.5.1.1.Closed-loop control	37
2.5.3.Technologies of artificial intelligence	37
2.6. Instruments	38
2.6.1. Continuous/Analog Instrumentation Device	39

2.7. Some Basic Sensors	40
2.6.2. Temperature Sensing	40
2.6.3. pressure sensor	41
2.6.4. Level sensors	41
2.7. actuators	42
Microprocessors	42
• Conclusion	43
Chapter 3:Supervision by the	44
microcontroller Arduino	
3.1. Introduction	44
3.2. The project architects	44
3.3. tasks of the project	44
3.3.1. The Main task	45
3.3.2. the additional task	45
○ For small gardans and green houses	46
○ For Burns and stabels:	46
3.4. Building sensor	47
3.4.1. Building the water level sensor	47
3.4.2. Building the water existans sensor F	47
3.4.3. Building the soil moist sensor E	48
3.1. The project scripts	48

3.4.4. The main task script	50
3.4.5. The main and the additional task script	50
3.5. The system in action	53
3.5.1. In homes	58
• conclusion	61

Introduction

Water covers 70% of the Earth, but only 3% of it is clean and suitable for human consumption. Even if you live in an area with ample rainfall, using water requires energy to process, pump, heat, re-pump, and re-process it. Fortunately, there are ways to save water for everyone from certified germaphobes to compost-toilet-level conservationists. The average family of four uses 450 liters of water a day, which is 164,000 a year. The responsibility relies on governments on citizens as well.

Controlling and monitoring the consumption of water will be very helpful to solve this huge problem and keeping useful water from demolishing .

As an engineer I consider this as contribution to reduce the catastrophic effects of the disaster. But the big part of this job is up on the shoulders of people who will use this.

- In the first chapter we will learn about the arduino project specifications ,boards, and its capabilities now we know what is it and how we can use it in our projects.
- The second chapter is about control and control system ,instruments and sensors , Process and we saw the mathematics behind control theory and the main classifications and models, parts of control system.
- In the final chapter we will see how can build a water level sensor and monitor based on simple physical principle which is electrical conductivity of fresh water using Arduino Uno ,a TV screen couple of electrical components and recycled materials that could be easily found in junk yards.

Chapter 1:

The state of art of the Microcontroller Arduino

1. The history of Arduino

Arduino started its life in Italy, at Interaction Design Institute Ivera (IDII), a graduate school for interaction design. This is a specific school of design education that focuses on how people interact with digital products, systems, and environments and how they in turn influence us.

In 2001, a project called Processing that was started by Casey Reas and Benjamin Fry aimed to get nonprogrammers into programming by making it quick and easy to produce onscreen visualizations and graphics. The project gave the user a digital sketchbook on which to try ideas and experiment with a very small investment of time. This project in turn inspired a similar project for experimenting in the physical world.

Building on the same principles as Processing, in 2003 Hernando Barragán started developing a microcontroller board called Wiring. This board was the predecessor to Arduino.

In common with the Processing project, the Wiring project also aimed to involve artists, designers, and other nontechnical people, but Wiring was designed to get people into electronics rather than programming. The Wiring board (shown in Figure 1) was less expensive than some other microcontrollers, such as the PIC and the Basic Stamp, but it was still a sizable investment for students to make.[1]



Figure 1.1: An early Wiring board.

2. What is Arduino?

Arduino is an open-source prototyping platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide.[16].[17]

3. Why Arduino?

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step by step

instructions of a kit, or sharing ideas online with other members of the Arduino community.[16]

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

Inexpensive - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than \$50[16].[3]

Cross-platform - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.[16]

Simple, clear programming environment - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.[3]

Open source and extensible software - The Arduino software is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.[3]

Open source and extensible hardware - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.[16].[4]

4. Exploring the Arduino Ecosystem

In your adventures with the Arduino, you'll depend on three main components for your projects:

- The Arduino board itself
- External hardware (including both shields and hand-made circuits, which you'll explore throughout this book)
- The Arduino integrated development environment, or Arduino IDE All these system components work in tandem to enable you do just about anything with your Arduino.[5]

You have a lot of options when it comes to Arduino development boards, but this book focuses on using official Arduino boards. Because the boards are all designed to be

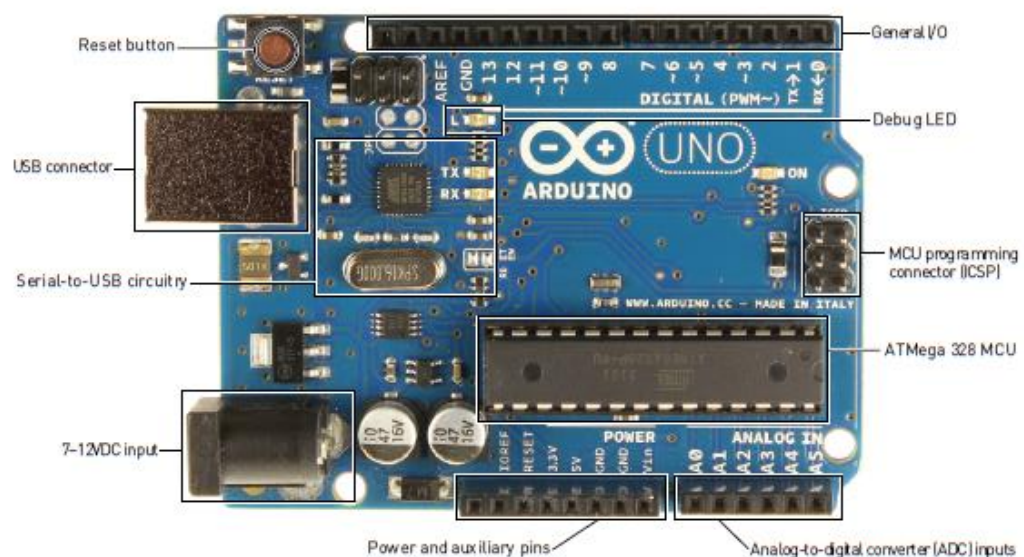


Figure 1.2 : Arduino Uno

programmable via the same IDE, you can generally use any of the modern Arduino boards to complete the projects in this book with zero or minor changes. However, when necessary, you'll see caveats about using different boards for various projects. The majority of the projects use the Arduino Uno. You start by exploring the basic functionality baked in to every Arduino board. Then you examine the differences between each modern board so that you can make an informed decision when choosing a board to use for your next project.[2].

4.1 Arduino Functionality

All Arduino boards have a few key capabilities and functions. Take a moment to examine the Arduino Uno (see Figure 1.2); it will be your base configuration. These are some key components that you'll be concerning yourself with:

- Atmel microcontroller
- USB programming/communication interface(s)
- Voltage regulator and power connections
- Breakout I/O pins
- Debug, Power, and RX/TX LEDs
- Reset button
- In-circuit serial programmer (ICSP) connector(s)[2].[5].[18

4.2 Atmel Microcontroller

At the heart of every Arduino is an Atmel microcontroller unit (MCU). Most Arduino boards, including the Arduino Uno, use an AVR ATmega microcontroller. The Arduino Uno in Figure 1-2 uses an ATmega 328p. The Due is an exception; it uses an ARM Cortex microcontroller. This microcontroller is responsible for holding all of your compiled code and executing the commands you specify. The Arduino programming language gives you access to microcontroller peripherals, including analog-to-digital converters (ADCs), general-purpose input/output (I/O) pins, communication buses (including I2C and SPI), and serial interfaces.[2]

4.3 Programming Interfaces

Ordinarily, ATmega microcontroller programs are written in C or Assembly and programmed via the ICSP interface using a dedicated programmer (see Figure 1-2). Perhaps the most important characteristic of an Arduino is that you can program it easily via USB, without using a separate programmer. This functionality is made possible by the Arduino bootloader. The bootloader is loaded onto the ATmega at the factory (using the ICSP header), which allows a serial USART (Universal Synchronous/Asynchronous Receiver/Transmitter) to load your program on the Arduino without using a separate programmer. (You can learn more about how the bootloader functions in “The Arduino Bootloader and Firmware Setup” sidebar.)

In the case of the Arduino Uno and Mega 2560, a secondary microcontroller (an ATmega 16U2 or 8U2 depending on your revision) serves as an interface between a USB cable and the

serial USART pins on the main microcontroller. The Arduino Leonardo, which uses an ATmega 32U4 as the main microcontroller, has USB baked right in, so a secondary microcontroller is not needed. In older

Arduino boards, an FTDI brand USB-to-serial chip was used as the interface between the ATmega's serial USART port and a USB connection.[2]

4.5. General I/O and ADCs

The part of the Arduino that you'll care the most about during your projects is the general-purpose I/O and ADC pins. All of these pins can be individually addressed via the programs you'll write. All of them can serve as digital inputs and outputs. The ADC pins can also act as analog inputs that can measure voltages between 0 and 5V (usually from resistive sensors). Many of these pins are also multiplexed to serve additional functions, which you will explore during your projects. These special functions include various communication interfaces, serial interfaces, pulse-width-modulated outputs, and external interrupts.[2].[4]

4.5. Power Supplies

For the majority of your projects, you will simply use the 5V power that is provided over your USB cable. However, when you're ready to untether your project from a computer, you have other power options. The Arduino can accept between 6V and 20V (7-12V recommend) via the direct current (DC) barrel jack connector, or into the Vin pin. The Arduino has built-in 5V and 3.3V regulators:

- 5V is used for all the logic on the board. In other words, when you toggle a digital I/O pin, you are toggling it between 5V and 0V.
- 3.3V is broken out to a pin to accommodate 3.3V shields and external circuitry.[2].

5. Arduino Boards

Covering all the available Arduino boards is impossible; there are many, and manufacturers are constantly releasing new ones with various features. The following section highlights some of the features in the official Arduino boards. The Uno (see Figure 1-3) is the flagship Arduino and will be used heavily in this book. It uses a 16U2 USB-to-serial converter chip and an ATmega 328p as the main MCU. It is available in both DIP and SMD versions (which defines whether the MCU is removable).



Figure 1-3: The Arduino Uno

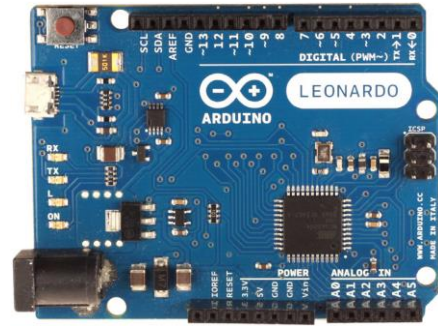


Figure 1-4: The Arduino Leonardo

The Leonardo (see Figure 1-4) uses the 32U4 as the main microcontroller, which has a USB interface built in. Therefore, it doesn't need a secondary MCU to perform the serial-to-USB conversion. This cuts down on the cost and enables you to do unique things like emulate a joystick or a keyboard instead of a simple serial device. The Mega 2560 (see Figure 1-5) employs an ATmega 2560 as the main MCU, which has 54 general I/Os to enable you to interface with many more devices.

The Mega also has more ADC channels, and has four hardware serial interfaces (unlike the one serial interface found on the Uno).

Unlike all the other Arduino variants, which use 8-bit AVR MCUs, the Due (see Figure 1-6) uses a 32-bit ARM Cortex M3 SAM3X MCU. The Due offers higher-precision ADCs, selectable resolution pulse-width modulation (PWM),

Digital-to-Analog Converters (DACs), a USB host connector, and an 84 MHz clock speed. The Nano (see Figure 1-7) is designed to be mounted right into a breadboard socket. Its small form factor makes it perfect for use in more finished projects.

The Mega ADK (see Figure 1-8) is very similar to the Mega 2560, except that it has USB host functionality, allowing it to connect to an Android phone so that it can communicate with apps that you write.

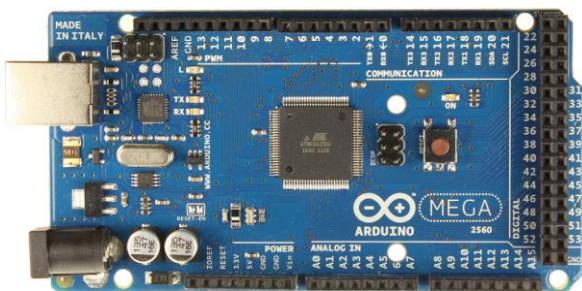


Figure 1-5: The Arduino Mega 2560

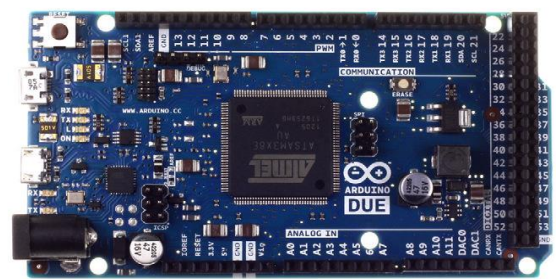


Figure 1-6: The Arduino Due

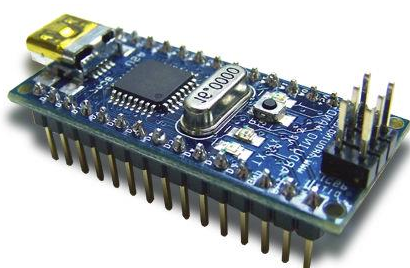


Figure 1-7: The Arduino Nano

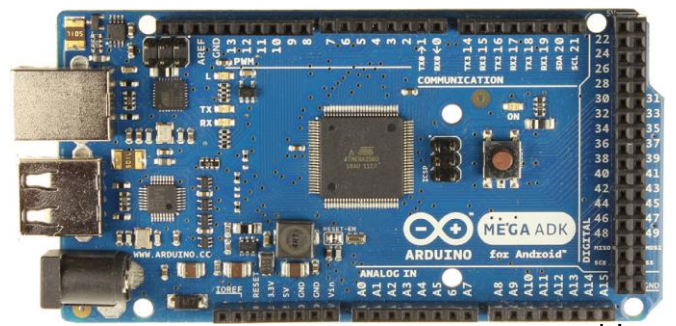


Figure 1-8: The Arduino Mega ADK

The LilyPad(see Figure 1-9) is unique because it is designed to be sewn into clothing. Using conductive thread, you can wire it up to sewable sensors, LEDs, and more. To keep size down, you need to program it using an FTDI cable.

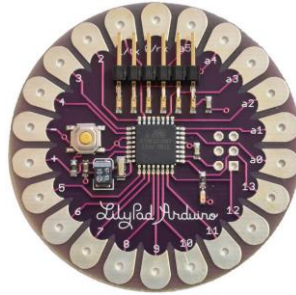


Figure 1-9: The LilyPad Arduino

As a result, you can find dozens and dozens of “Arduino compatible” devices available for sale that will work just fine with the Arduino IDE and all the projects you’ll do in this book. Some of the popular third-party boards include the Seeduino, the adafruit 32U4 breakout board, and the SparkFun Pro Mini Arduino boards. Many third-party boards are designed for very particular applications, with additional functionality already built in to the board.



Figure 1-10: Quadcopter and ArduPilot Mega controller

For example, the ArduPilot is an autopilot board for use in autonomous DIY quadcopters (Figure 1-10). You can even find Arduino-compatible circuitry baked in to consumer devices like the MakerBot Replicator and Replicator 2 3D printers. [2].

6. Some commonly used shields

There are a lot of extensions of arduino boards called “shields” that can make arduino to simply do more, this is some of them :

6.1. The Arduino GSM Shield

The Arduino GSM shield (Figure1-14) allows an Arduino board to connect to the internet, send and receive SMS, and make voice calls using the GSM library.

The shield will work with the Arduino Uno out of the box. The shield will work with the Mega, Mega ADK, Yun, and Leonardo boards with a minor modification. The Due is not supported at this time.

The GSM library is included with Arduino IDE 1.0.4 and later.

To use GPRS for internet access, and for the Arduino to request or serve webpages, you need to obtain the Access Point Name (APN) and a username/password from the network operator. See the information in Connecting to the Internet for more information about using the data capabilities of the shield. .[1]

Among other things, GSM supports outgoing and incoming voice calls, Simple Message System (SMS or text messaging), and data communication (via GPRS).

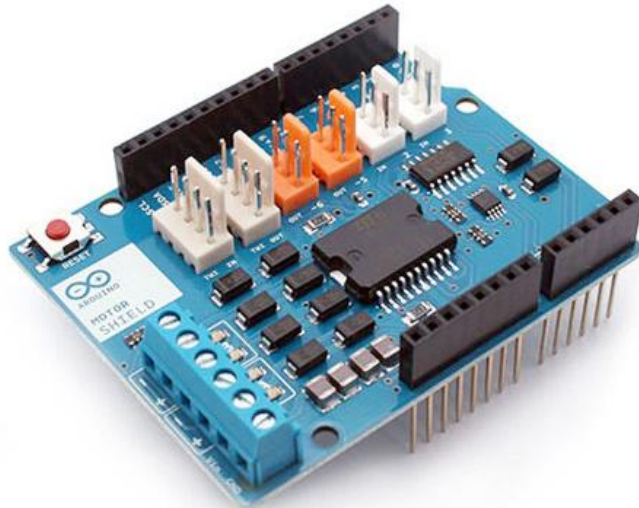
The Arduino GSM shield is aa GSM modem. From the mobile operator perspective, the Arduino GSM shield looks just like a mobile phone. From the Arduino perspective, the Arduino GSM shield looks just like a modem.[1].[6].[19].



Figure1-14: Arduino GSM Shield

6.2.The Arduino Motor Shield

Motor Shield is based on the L298 (datasheet), which is a dual full-bridge driver designed to drive inductive loads such as relays, solenoids, DC and stepping motors. It lets you drive two DC motors with your Arduino board, controlling the speed and direction of each one independently. You can also measure the motor current absorption of each motor, among other features.



The shield is TinkerKit compatible, which means you can quickly create projects by plugging TinkerKit modules to the board. [25].

6.3.The Yún Shield

The Yún Shield (Figure1-16) extends your Arduino & Genuino board with the power of a Linux based system that enables advanced network connections and applications.

Connection to your WiFi or wired network is simple thanks to the Yún Web Panel and the dedicated "YunFirstConfig" sketch. The Web panel allows you to manage your shield Preferences and upload your sketch (in .hex format) on the attached Arduino or Genuino board. The Yún Shield uses the Bridge library and so extends your board capabilities using the Linux processor, in the same way as the Yún board.

As always, every element of the platform – hardware, software and documentation – is freely available and open- that you can learn made and use its starting point for [26].



source. This means exactly how it's design as the your own projects.

Figure1-16: Arduino Yún Shield

6.4.Xbee Shield

The Xbee shield (Figure1-17) allows an Arduino board to communicate wirelessly using Zigbee. It is based on the Xbee module from MaxStream. The module can communicate up to 100 feet indoors or 300 feet outdoors (with line-of-sight). It can be used as a serial/usb replacement or you can put it into a command mode and configure it for a variety of broadcast and mesh networking options. The shields breaks out each of the Xbee's pins to a through-hole solder pad.

It also provides female pin headers for use of digital pins 2 to 7 and the analog inputs, which are covered by the shield (digital pins 8 to 13 are not obstructed by the shield, so you can use the headers on the board itself).

The Xbee shield was created in collaboration with Libelium, who developed it for use in their SquidBee motes (used for creating sensor networks). [27].

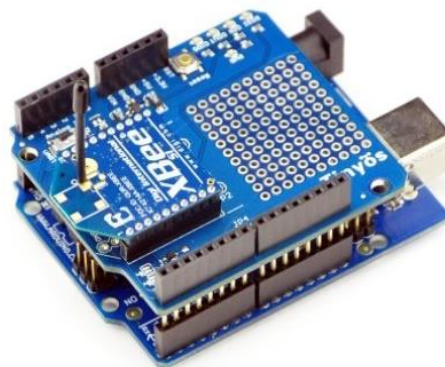


Figure1-17: Arduino Xbee Shield

- **Conclusion**

Arduino is cheap powerful prototyping platform an open source hardware and software originally created at Interaction Design Institute Ivera in Italy

There are a lot of existing boards developed for general purpose and for specific tasks like arduino due and arduino mega.

At the heart of the Most Arduino boards, including the Arduino Uno, use an AVR ATmega microcontroller.

The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users programs are written in C or Assembly and programmed via the ICSP interface using a dedicated programmer

There are a lot of extensions of arduino boards called “shields” that can make arduino to simply do more.

Chapter 2 : State of art about control

- **Introduction**

We are surrounded by electronic devices: mobile phones, computers, traffic light regulators, etc. Many of them work automatically with inputs provided by sensors scrutinizing the environment. Plane trips contain less and less manual drive. Electrical systems are managed with automatic circuits ensuring stability. Examples abound, and our technological society will use more and more such automatic devices under the pressure of various factors. Technological innovation and costs reduction increase the penetration of so-called “smart” devices. The world increase in demand for communications technologies and for energy requires more and more coordination in a global economy. Environmental protection fosters the need for soberness in the use of resources, hence of an optimized management.

Control Theory is at the heart of Information and Communication Technologies of complex systems; it can contribute to answer such challenges.[11]

2.1. Control

Control is used to modify the behavior of a system so it behaves in a specific desirable way over time. For example, we may want the speed of a car on the highway to remain as close as possible to 60 miles per hour in spite of possible hills or adverse wind; or we may want an aircraft to follow a desired altitude, heading, and velocity profile independent of wind gusts; or we may want the temperature and pressure in a reactor vessel in a chemical process plant to be maintained at desired levels. All these are being accomplished today by control methods and the above are examples of what automatic control systems are designed to do, without human intervention. Control is used whenever quantities such as speed, altitude, temperature, or voltage must be made to behave in some desirable way over time.[5]

2.1.1 Control engineering

Control engineering or control systems engineering is the engineering discipline that applies control theory to design systems with desired behaviors. The practice uses sensors to measure the output performance of the device being controlled and those measurements can be used to give feedback to the input actuators that can make corrections toward desired performance. When a device is designed to perform without the need of human inputs for correction it is called automatic control (such as cruise control for regulating the speed of a car). Multi-

disciplinary in nature, control systems engineering activities focus on implementation of control systems mainly derived by mathematical modeling of systems of a diverse range.

2.1.2. Control theory

Control theory is an interdisciplinary branch of engineering and mathematics that deals with the behavior of dynamical systems with inputs, and how their behavior is modified by feedback. The usual objective of control theory is to control a system, often called the plant, so its output follows a desired control signal, called the reference, which may be a fixed or changing value. To do this a controller is designed, which monitors the output and compares it with the reference. The difference between actual and desired output, called the error signal, is applied as feedback to the input of the system, to bring the actual output closer to the reference. Some topics studied in control theory are stability (whether the output will converge to the reference value or oscillate about it), controllability and observability.

Although a major application of control theory is in control systems engineering, which deals with the design of process control systems for industry, other applications range far beyond this. As the general theory of feedback systems, control theory is useful wherever feedback occurs. A few examples are in physiology, electronics, climate modeling, machine design, ecosystems, navigation, neural networks, predator-prey interaction, gene expression, and production theory. .[14]

Control theory is a theory that deals with influencing the behavior of dynamical systems an interdisciplinary subfield of science, which originated in engineering and mathematics, and evolved into use by the social sciences, such as economics, psychology, sociology, criminology and in the financial system.

Control systems may be thought of as having four functions: measure, compare, compute and correct. These four functions are completed by five elements: detector, transducer, transmitter, controller and final control element. The measuring function is completed by the detector, transducer and transmitter. In practical applications these three elements are typically contained in one unit. A standard example of a measuring unit is a resistance thermometer.[14]

2.1.3. The role of control theory

To design a controller that makes a system behave in a desirable manner, we need a way to predict the behavior of the quantities of interest over time, specifically how they change in response to different inputs.

Mathematical models are most often used to predict future behavior, and control system design methodologies are based on such models. Understanding control theory requires engineers to be well versed in basic mathematical concepts and skills, such as solving differential equations and using Laplace transform. The role of control theory is to help us gain insight on how and why feedback control systems work and how to *systematically* deal with various design and analysis issues. Specifically, the following issues are of both practical importance and theoretical interest:

1. Stability and stability margins of closed-loop systems.
2. How fast and smooth the error between the output and the set point is driven to zero.
3. How well the control system handles unexpected external disturbances, sensor noises, and internal dynamic changes.[5]

2.1.4. Control Modeling

Five types of modeling methodologies have been employed to represent physical components and relationships in the study of control systems:

1. . Mathematical equations, in particular, differential equations, which are the basis of classical control theory (transfer functions are a common form of these equations)
2. Mathematical equations that are used on state variables of multivariable systems and associated with modern control theory
3. Block diagrams
4. Signal flow graphs
5. Functional analysis representations (data flow diagram and entity relationships)

Mathematical models are employed when detailed relationships are necessary. To simplify the analysis of mathematical equations, we usually approximate them by linear, ordinary differential equations. For instance, a characteristic differential equation of a control loop

$$\frac{d^2x}{dt^2} + 2\alpha \frac{dx}{dt} + \beta^2x = f(t)$$

model may have the form with initial conditions of the system given as

$$\begin{aligned}x(0) &= X_0 \\x'(0) &= V_0\end{aligned}$$

where $x(t)$ is a time function of the controlled output variable, its first and second derivatives over time specify the temporal nature of the system, α and β are parameters of the system properties, $f(t)$ specifies the input function, and X_0 and V_0 are specified constants.

Mathematical equations such as this example are developed to describe the performance of a given system. Usually an equation or a transfer function is determined for each system component.

Then a model is formulated by appropriately combining the individual components. This process is often simplified by applying Laplace and Fourier transforms. A graph representation by block diagrams (see Figure 2.1) is usually applied to define the connections between components.

Once a mathematical model is formulated, the control system characteristics can be analytically or empirically determined. The basic characteristics that are the object of the control system design are:

1. Response time
2. Relative stability
3. Control accuracy

They can be expressed either as functions of frequency, called frequency domain specifications, or as functions of time, called time domain specifications. To develop the specifications the mathematical equations have to be solved. Modern computer software, such as MATLAB has provided convenient tools for solving the equations.

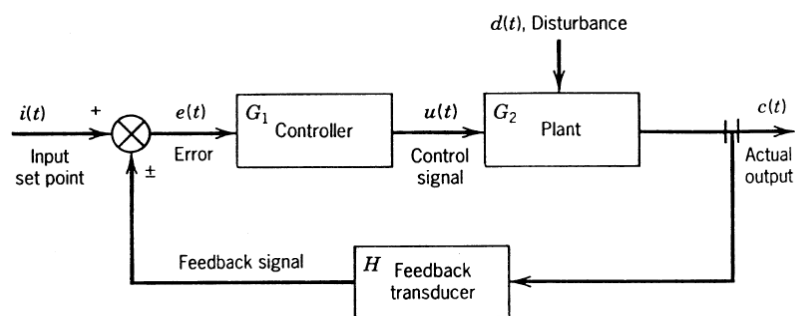


Figure 2.1 block diagram of feedback loop

2.1.4.1. Control Models

Unlike the open-loop control, which basically provides a transfer function for the input signals to actuators, the feedback control systems receive feedback signals from sensors then compare the signals with the set point. The controller can then control the plant to the desired set point according to the feedback signal. There are five basic feedback control models:

1. *On/off control*: In on/off control, if the $e(t)$ is smaller than 0, the controller may activate the plant; otherwise the controller stays still. Most household temperature thermostats follow this model.
2. *Proportional (PE) control*: In PE control, the output is proportional to the $e(t)$ value, i.e., $e(t) = KPe(t)$. In PE, the plant responds as soon as the error signal is non-zero. The output will not stop exactly at the set point. When it approaches the set point, the $e(t)$ becomes smaller. Eventually, the output is too small to overcome opposing force (e.g., friction). Attempts to reduce this small $e(t)$, also called steady state error, by increasing KP can only cause more overshoot error.
3. *Proportional-integral (PI) control*: PI control tries to solve the problem of steady state error. In PI, output $=KPe(t) + KI \int e(t) dt$. The integral of the error signal will have grown to a certain value and will continue to grow as soon as the steady state error exists. The plant can thus be drawn to close the steady state error.
4. *Proportional-derivative (PD) control*: PD control modifies the rate of response of the feedback control system in order to prevent overshoot. In PD,

$$\text{Output} = K_p e(t) + K_D \frac{d(e(t))}{dt}$$

When the $e(t)$ gets smaller, a negative derivative results. Therefore, overshoot is prevented.

5. *Proportional-integral-derivative (PID) control*: PID control takes advantage of PE, PI, and PD controls by finding the gains (KP , KI , and KD) to balance the proportional response, steady state reset ability, and rate of response control, so the plant can be well controlled.

2.1.4.2. Advanced Control Models Based on the control models introduced before researchers have developed various advanced control models for special needs. Table 2.1 shows the application domains and examples of the models, rather than the complicated theoretical control diagram.[19]

TABLE 1 System Characteristics and Examples of Advanced Control Models		
Control Models	When to Apply (System Characteristics)	Examples
(1) Nested Control Loops	More than one characteristic of a system's output to be controlled at the same time	Robot controllers
(2) Directed Synthesis Control	Time lag between output and feedback reception is long	Satellite guidance systems
(3) Adaptive Directed Synthesis Control	Allow a direct synthesis control system to be adjusted when actual feedback is received	Modern machine tools
(4) Feedforward Control; Cascade Control	Set points are given by the sensed conditions upstream of the process.	Chemical process control
(5) Ratio Control	Use a sensor in one stream to control the processes in one or more parallel streams	Chemical process control
(6) Multiple Output Control	A single control provides the output signal to a group of parallel actuators. Then the result of the multiple actuation is fed back to the controller.	Complex hydraulic cylinders
(7) Constraint Control	More than one controller in the system. Under normal conditions, one of the controllers does the controlling, but if a preset limit is exceeded at a sensor, the second controller overrides the first and takes over control.	Chemical process control

Table 2.1 characteristics and examples of Advanced

2.2. Physical Process

Physical process is a series of actions, operations, changes, or functions that takes place within bringing about changes or producing an output or a result. Also, the physical process as a sequence of interdependent operations or actions which, at every stage, consume one or more inputs or resources to convert same into outputs or results to reach a known goal or the desired end result.

Process as used in the terms process control and process industry, refers to the methods of changing or refining raw materials to create end products. The raw materials, which either pass through or remain in a liquid, gaseous, or slurry (a mix of solids and liquids) state during the process, are transferred, measured, mixed, heated or cooled, filtered, stored, or handled in some other way to produce the end product.

Process industries include the chemical industry, the oil and gas industry, the food and beverage industry, the pharmaceutical industry, the water treatment industry, and the power industry.[6]

Whatever we see and work within reality are all physical processes. The physical processes can be broadly divided into three categories as follows:

1. Natural processes

2. Self-regulated processes
3. Man-made or industrial processes.

2.2.1. Natural Processes

Natural processes are presented by or produced by nature. The best example is a human body that, generally, does not need any external assistance to regulate its body parameters (e.g., the body temperature) irrespective of the effects of the surrounding environmental conditions. The human body maintains or regulates all its parameters. Typically, in natural processes, no abnormal behavior is present in most of the conditions.

2.2.2. Self-Regulated Processes

Self-regulated processes are not natural but do not need any external assistance for their regulation. The best example is a domestic geyser in which the water level in the geyser is always maintained irrespective of its water temperature. All natural processes are self-regulated, but the reverse is not true.

2.2.3. Man-made or Industrial Processes

Man-made or industrial processes⁵ are systematic series of physical, mechanical, chemical, or other kinds of operations that produce a result. They manufacture goods or provide services. These processes are not always self-regulating and may need external regulation on a continuous basis.

Typical examples of goods are any products manufactured by an industrial process, such as food, chemical, engineering⁷. Examples of typical services include the supply of electricity, water, gas, etc., to consumers in a municipal locality.[7]

2.3. Signals

2.3.1. Pneumatic Signals

Pneumatic signals are signals produced by changing the air pressure in a signal pipe in proportion to the measured change in a process variable. The common industry standard pneumatic signal range is 3–15 psig. The 3 corresponds to the lower range value (LRV) and the 15 corresponds to the upper range value (URV). Pneumatic signaling is still common. However, since the advent of electronic instruments

in the 1960s, the lower costs involved in running electrical signal wire through a plant as opposed to running pressurized air tubes has made pneumatic signal technology less attractive.[6]

2.3.2. Analog Signal

Signal amplitudes are represented by voltage or current amplitudes in analog systems.

Analog processing means that the data, such as signal linearization, from the sensor is conditioned, and corrections that are made for temperature variations are all performed using analog circuits. Analog processing also controls the actuators and feedback loops. The most common current transmission range is 4 to 20 mA, where 0 mA is a fault indication.

2.3.3. Digital Signal

Signal amplitudes are represented by binary numbers in digital systems. Since variables are analog in nature, and the output from the sensor needs to be in a digital format, an analog to digital converter (ADC) must be used, or the sensor's output must be directly converted into a digital signal using switching techniques. Once digitized, the signal will be processed using digital techniques, which have many advantages over analog techniques, and few, if any, disadvantages. Some of the advantages of digital signals are: data storage, transmission of signals without loss of integrity, reduced power requirements, storage of set points, control of multiple variables, and the flexibility and ease of program changes. The output of a digital system may have to be converted back into an analog format for actuator control, using either a digital to analog converter (DAC) or width modulation techniques.[8]

2.3.4. Pulse

signals consist of trains of pulses, each pulse being equivalent to a fleeting discrete signal. They are associated with rotary devices such as turbine meters and agitator shafts. A known number of electrical pulses are generated with each revolution. Counting of the pulses with respect to time yields an average shaft speed. Relative to analogues and discretely, pulse signals are not very common.

2.4. Process Control

Process control refers to the methods that are used to control process variables when manufacturing a product. For example, factors such as the proportion of one ingredient to another, the temperature of the materials, how well the ingredients are mixed, and the pressure under which the materials are held can significantly impact the quality of an end product. Manufacturers control the production process for three reasons:

- Reduce variability
- Increase efficiency
- Ensure safety.[20]

Process control can take two forms:

2.4.1 Sequential Process Control

Control systems can be sequential in nature, or can use continuous measurement; both systems normally use a form of feedback for control. Sequential control is an event-based process, in which the completion of one event follows the completion of another, until a process is complete, as by the sensing devices. Figure (figure2.2) shows an example of a process using a sequencer for mixing liquids in a set ratio [11]. The sequence of events is as follows:

1. Open valve A to fill tank A.
2. When tank A is full, a feedback signal from the level sensor tells the sequencer to turn valve A Off.
3. Open valve B to fill tank B.
4. When tank B is full, a feedback signal from the level sensor tells the sequencer to turn valve B Off.
5. When valves A and B are closed, valves C and D are opened to let measured quantities of liquids A and B into mixing tank C.
6. When tanks A and B are empty, valves C and D are turned Off.
7. After C and D are closed, start mixing motor, run for set period.
8. Turn Off mixing motor.
9. Open valve F to use mixture.
10. The sequence can then be repeated after tank C is empty and Valve F is turned Off.

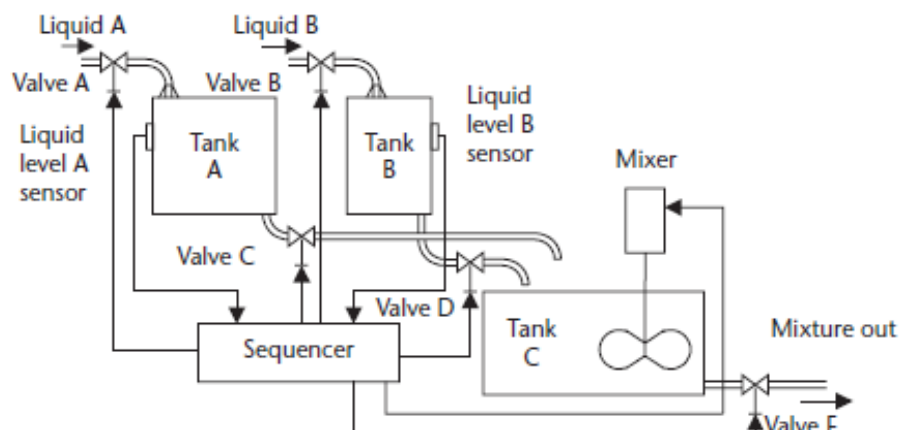


Figure 2.2: Sequential Process Control

2.4.2. Continuous Process Control

Continuous process control falls into two categories: (1) elementary On/Off action, and (2) continuous control action.

On/Off action is used in applications where the system has high inertia, which prevents the system from rapid cycling. This type of control only has only two states, On and Off; hence, its name. This type of control has been in use for many decades.[12].

2.5. Control system

A **control system** is an interconnection of components forming a system configuration that will provide a desired system response. The basis for analysis of a system is the foundation provided by linear system, which assumes a cause effect relationship for the components of a system. A component or process to be controlled can be represented by a block as shown in Figure 2.3.[24]

2.5.1. The Control Loop

Imagine you are sitting in a cabin in front of a small fire on a cold winter evening. You feel uncomfortably cold, so you

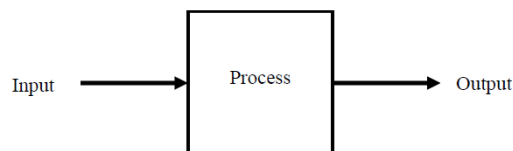


Figure 2.3:Process Control

throw another log on the fire. This is an example of a *control loop*. In the control loop, a variable (temperature) fell below the set point (your comfort level), and you took action to bring the process back into the desired condition by adding fuel to the fire. The control loop will now remain static until the temperature again rises above or falls below your comfort level. Control loops in the process control industry work in the same way, requiring three tasks to occur:

- Measurement
- Comparison
- Adjustment

In Figure 2.4, a level transmitter (LT) measures the level in the tank and transmits a signal associated with the level reading to a controller (LIC). The controller compares the reading to a predetermined value, in this case, the maximum tank level established by the plant operator,

and finds that the values are equal. The controller then sends a signal to the device that can bring the tank level back to a lower level—a valve at the bottom of the tank. The valve opens to let some liquid out of the tank. Many different instruments and devices may or may not be used in control loops (e.g., transmitters, sensors, controllers, valves, pumps), but the three tasks of measurement, comparison, and adjustment are always present.[20].

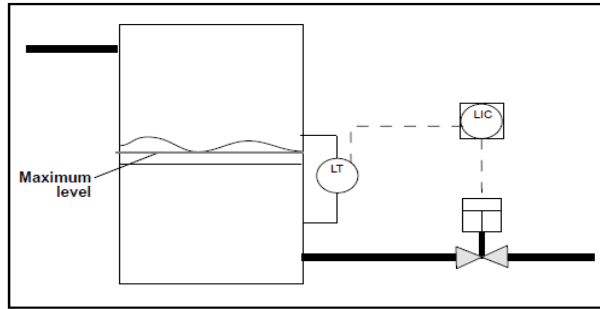


Figure 2.4:an example of a Process Control

2.5.1.1.Open-loop control

An open-loop control system utilizes a controller or control actuator to obtain the desired response as shown in Figure 2.5 The ,open-loop control system utilizes an actuating device to control the process directly without using device. An example of an open-loop control system is an electric toaster.

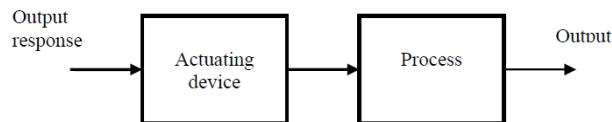


Figure 2.5: Open-loop control system (no feedback)

2.5.1.2.Closed-loop control

A **closed-loop control system** (Figure 2.6) utilizes an additional measure of the actual output to compare the actual output with the desired output response. The measure of the output is called the **feedback signal**. A feedback control system is a control system that tends to maintain a relationship of one system variable to another by comparing functions of these variables and using the difference as a means of control. As the system is becoming more

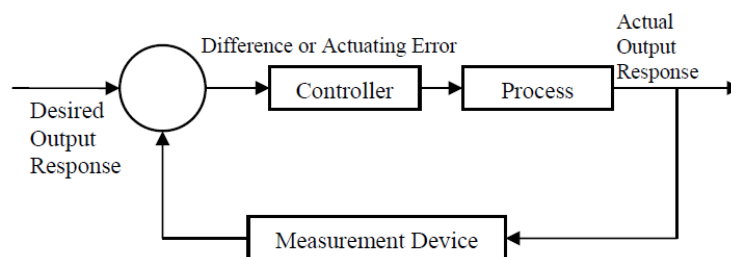


Figure 2.6: Closed-loop feedback control system

complex, the interrelationship of many controlled variables may be considered in the control scheme. An example of closed-loop control system is a person steering an automobile by looking at the auto's location on the road and making the appropriate adjustments.[20].

2.5.2. Technologies of artificial intelligence

Automation technologies have been bestowed intelligence by the invention of computers and the evolution of artificial intelligence theories. Because of the introduction of the technologies of artificial intelligence, automated systems, from the perspective of control, can intelligently plan, actuate, and control their operations in a reasonable time limit by handling / sensing much more environmental input information (see the horizontal axis in Figure 2.6). Meanwhile, artificial intelligence increases the decision making complexity of automation technology, but the cost of the system that is automated by the technologies of artificial intelligence is relatively low compared with the system automated by the automatic control theory (see the vertical axis in Figure 2.6). Figure 2.6 shows a roadmap of how various automation technologies influence the development of automation systems in two axes. In the following subsections, those technologies of artificial intelligence, including neural networks(NN), genetic algorithms (GA), knowledge-based systems (KBS), fuzzy control, and the hybrid of the above-mentioned technologies will be introduced in general. [19]

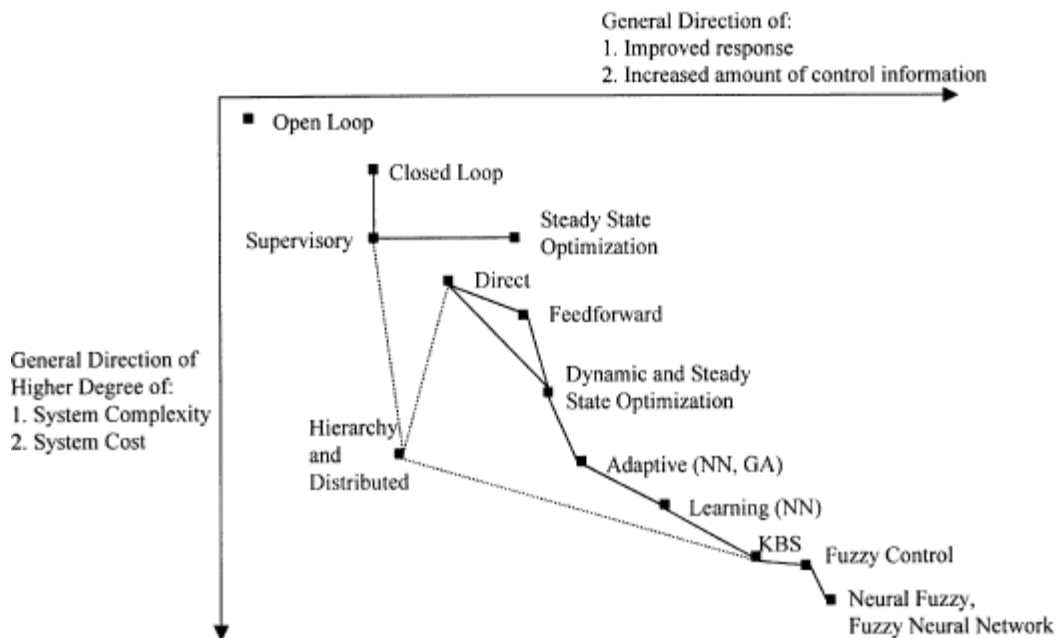


Figure 2.7: road-map of control models

2.6. Instruments

Instruments are manmade devices that measure the parameters of physical variables. They may be analog, digital, or a combination of the two. Nowadays, most instruments are digital because of their advantages over analog counterparts.

However, the front ends of many instruments are still analog; that is, most signals from sensors and transducers and the first stage of signal processing are still analog. Nevertheless, it is important to mention that in recent years digital instruments operating purely on digital principles have been developing fast. Today's smart sensors based on digital principles, contain the complete signal condition circuits and the sensors in a single chip. The outputs of smart sensors can directly be interfaced with other digital devices.

Sensors and transducers are the most basic and primary elements of all instruments. They respond to physical variations to produce continuous outputs that convey useful information. The outputs may be in the forms of amplitudes, frequencies, or phase differences of currents, voltages, power, or energy. As in the case of all signal-bearing systems, in both analog and digital instruments, there are useful signals that respond to the physical phenomena and unwanted signals that are imposed as various forms of noise.

In the last 10 to 15 years or so, due to rapid progress in integrated circuit (IC) technology and the availability of low-cost analog and digital components and microprocessors, considerable progress in digital instruments has taken place.[9]

2.6.1. Continuous/Analog Instrumentation Devices

Figure 3.1 illustrates the functional components and the general structure of continuous/analog instrumentation devices, for both input and output.

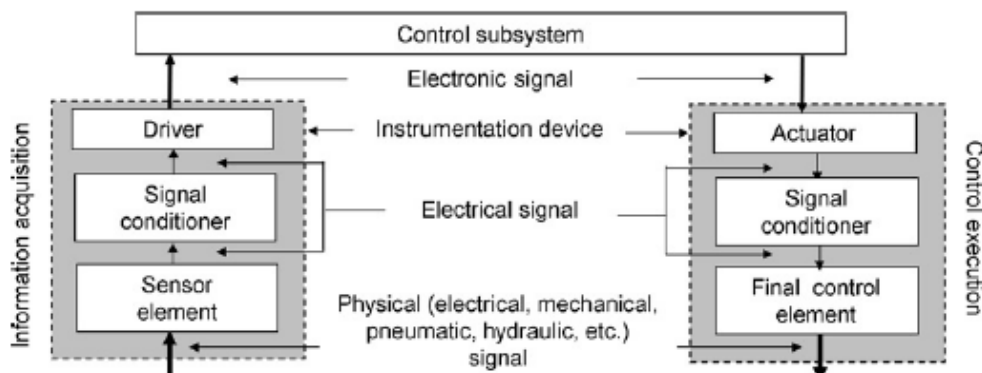


Figure 2.8: Analog inputs and output Instrumentation Devices

In Figure 2.8, the structure and the input/output relationship of an analog input instrumentation device is illustrated.

The analog input instrumentation device processes the received physical signal generated by the process with the following functional components:

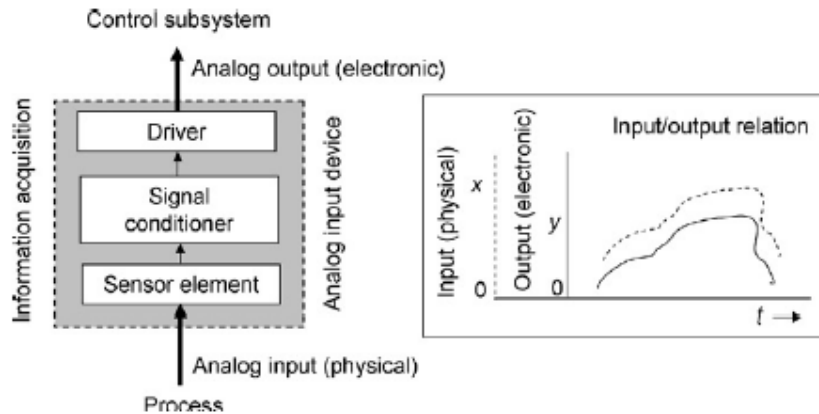


Figure 2.9: Analog inputs and output Instrumentation Devices-input/output relationship

- *Sensor*: It converts an analog or continuous physical signal into its electrical equivalent signal through the transduction process.
- *Signal conditioner*: Used in the intermediate stage, the signal conditioner prepares or manipulates (isolating, compensating, linearizing, amplifying, filtering, offset correcting, etc.) the weak and noisy sensor output to meet the requirements of the next stage for further processing.
- *Driver*: Strengthens the signal received from the signal conditioner to an appropriate form and drives it for transmission to the control subsystem.

The instrumentation device Figure 2.10 is called a **transducer** if its output is suitable for transmission (in voltage or current form) over a relatively short distance. The device is called a **transmitter** if the output is suitable for transmission (in current form) over a relatively long distance.

Figure 2.9 illustrates the functions of the components of the analog input instrumentation device.[20]

Figure 2.11 illustrates industry examples of four commonly used analog input instrumentation devices in process industry (temperature transmitter, flow transmitter, level transmitter, and differential pressure transmitter). These devices receive the physical signals (temperature, flow, level, differential pressure) and generate corresponding electronic output in current form suitable for the control subsystems [6].[9]

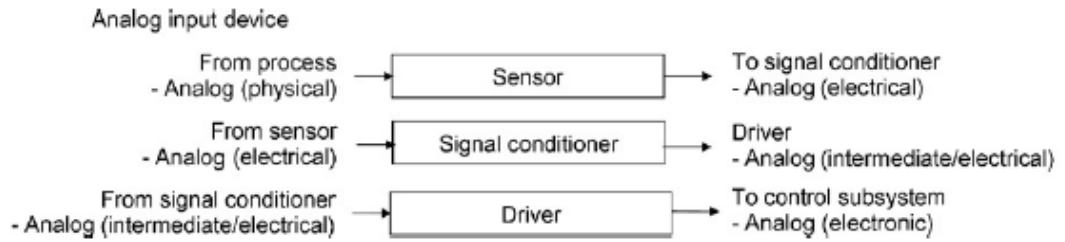


Figure 2.10: Analog inputs and output Instrumentation Devices-components

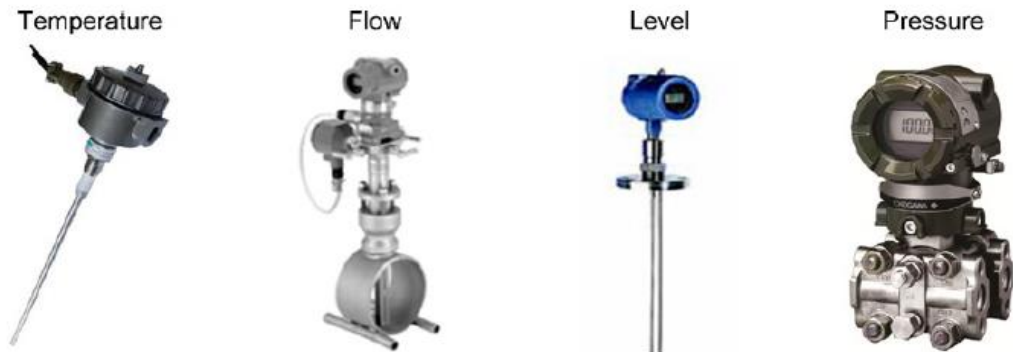


Figure 2.11: Analog inputs and output Instrumentation Devices- examples

2.7. Some Basic Sensors

2.7.2. Temperature Sensing

A number of semiconductor parameters vary linearly with temperature, and can be used for temperature sensing. These parameters are diode or transistor junction voltages, zener diode voltages, or polysilicon resistors.

In an integrated circuit, the fact that the differential base emitter voltage between two transistors operating at different current densities (bandgap) is directly proportional to temperature is normally used to measure temperature.

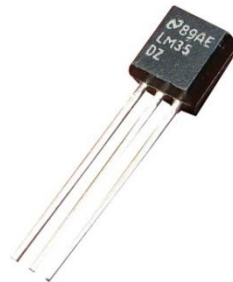


Figure 2.12: LM35 Temperature Sensor

The output is normally adjusted to give a sensitivity of 10 mV per degree (C, F, or K). These devices can operate over a very wide of Temperature Sensor Example of Temperature Sensor (see figure 2.12) .[8].[15].

2.7.3. pressure sensor

A **pressure sensor** measures pressure, typically of gases or liquids. Pressure is an expression of the force required to stop a fluid from expanding, and is usually stated in terms of force per unit area. A pressure sensor usually acts as a transducer; it generates a signal as a function of the pressure imposed. For the purposes of this article, such a signal is electrical. Pressure sensors are used for control and monitoring in thousands of everyday applications. Pressure sensors can also be used to indirectly measure other variables such as fluid/gas flow, speed, water level, and altitude. Example of Pressure sensors Figure 2.13.

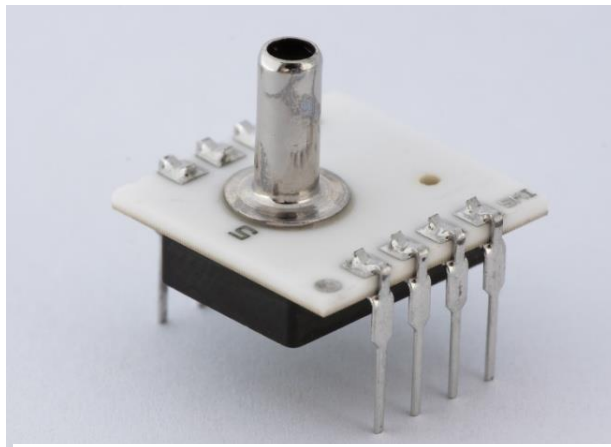


Figure 2.13: LM35 Digital air pressure sensor

2.7.4. Level sensors

Level sensors detect the level of liquids and other fluids and fluidized solids, including slurries, granular materials, and powders that exhibit an upper free surface. Substances that flow become essentially horizontal in their containers



Figure 2.14 Capacitance level sensor

(or other physical boundaries) because of gravity whereas most bulk solids pile at an angle of repose to a peak. The substance to be measured can be inside a container or can be in its natural form (e.g., a river or a lake). The level measurement can be either continuous or point values. Continuous level sensors measure level within a specified range and determine the exact amount of substance in a certain place, while point-level sensors only indicate whether the substance is above or below the sensing point. Generally the latter detect levels that are excessively high or low.[23]

2.8. **actuators**

An **actuator** is a type of motor that is responsible for moving or controlling a mechanism or system. It is operated by a source of energy, typically electric current, hydraulic fluid pressure, or pneumatic pressure, and converts that energy into motion. An actuator is the mechanism by which a control system acts upon an environment. The control system can be simple (a fixed mechanical or electronic system), software-based (e.g. a printer driver, robot control system), a human, or any other input.[8].[9]

2.9. **Microprocessors**

Many instruments and almost all control systems are microprocessor based.

A microprocessor is essentially a very large scale integrated (VLSI) transistorized circuit on a single silicon chip. The so-called von Neumann machine, which is the classical microprocessor architecture, is depicted in Figure 2.15. This shows the functional relationship of the major hardware elements of a typical microprocessor, and how they are interconnected by the address, control and data buses.

The architecture can be divided functionally into three parts. First, the central processing unit (CPU), contained within the broken line, which consists of the control unit, the program counter (PC), the instruction register (IR), the arithmetic/logic unit (ALU) and its registers, and the register stack. Second, the main memory system consisting of random access memory (RAM) and read only memory (ROM). And third, the input/output system, consisting of the memory address register (MAR), the memory buffer (MBR) and various other devices for communications purposes.

The register set consists of the IR, PC, ALU registers and the stack, the stack consisting of a number of special purpose and general purpose registers.

An important point to appreciate is the distinction between the set's functions and its physical location. This distinction is common to many aspects of microprocessor architecture.

Architectures other than that of Figure 9.1 exist. These enable, for example, parallel processing and/or reduced instruction set computing (RISC). However, they have yet to find their way into process automation to any significant extent.[10]

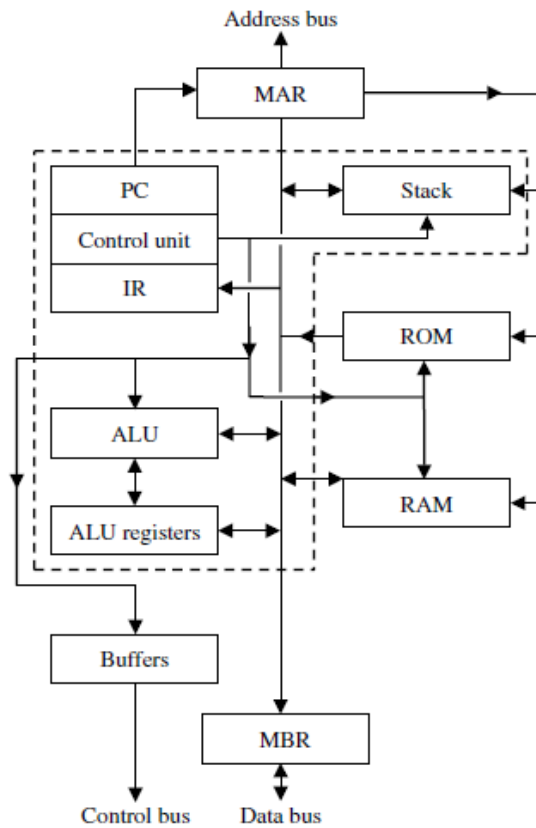


Figure 2.15: elements of a typical classic microprocessor

- **Conclusion**

In this chapter we learned about control and control system, instruments and sensors .we are surrounded by electronic devices: mobile phones, computers, traffic light regulators, etc. Many of them work automatically with inputs provided by sensors scrutinizing the environment, Process and we saw the mathematics behind control theory and the mean classifications and models, parts of control system. control systems engineering is the engineering discipline that applies control theory to design systems with desired behaviors. Automation technologies have been bestowed intelligence by the invention of computers and the evolution of artificial intelligence theories.

Chapter 3:

Supervision by the microcontroller Arduino

3.1. introduction

When it comes to conserving water, small adjustments can have a big impact. There are many simple things we can do to save water.as automation engineer what you can do to help protect our water supplies. In this chapter illustrate how the hardware and software

3.2. The project architects

In this chapter, we will present our contribution that attempt to establish a solution, the water level monitor based on Arduino.

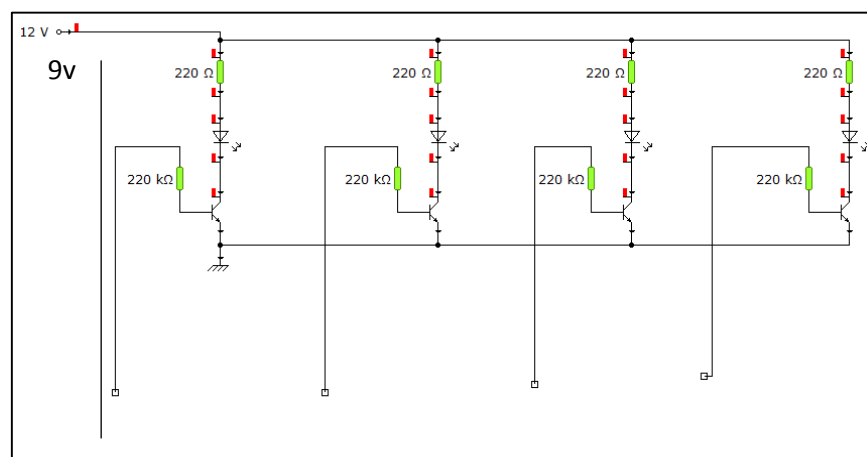
This systeme could be used in indivudial or collaclative habitats,hospitals,schools and universities...etc or gardans and even green houses, burns,stabels...etc.

By exploring the electrical conductivity propriety of fresh water to detect the water level in the water container or soil moist.

To build a level sensor we used a simple circuit based on a BC548 NPN transistors (figure 3.1)

The results will be shown in a TV screen. Rather then using TVscreen we can use arduino's GSM shiled to send status messeges to the owner's sellphone or wifi shiled to send e-mails to the owner's pc or laptop.

The systeme has a main task and an additional task,in the houses or habitats ,no need of the additional task.



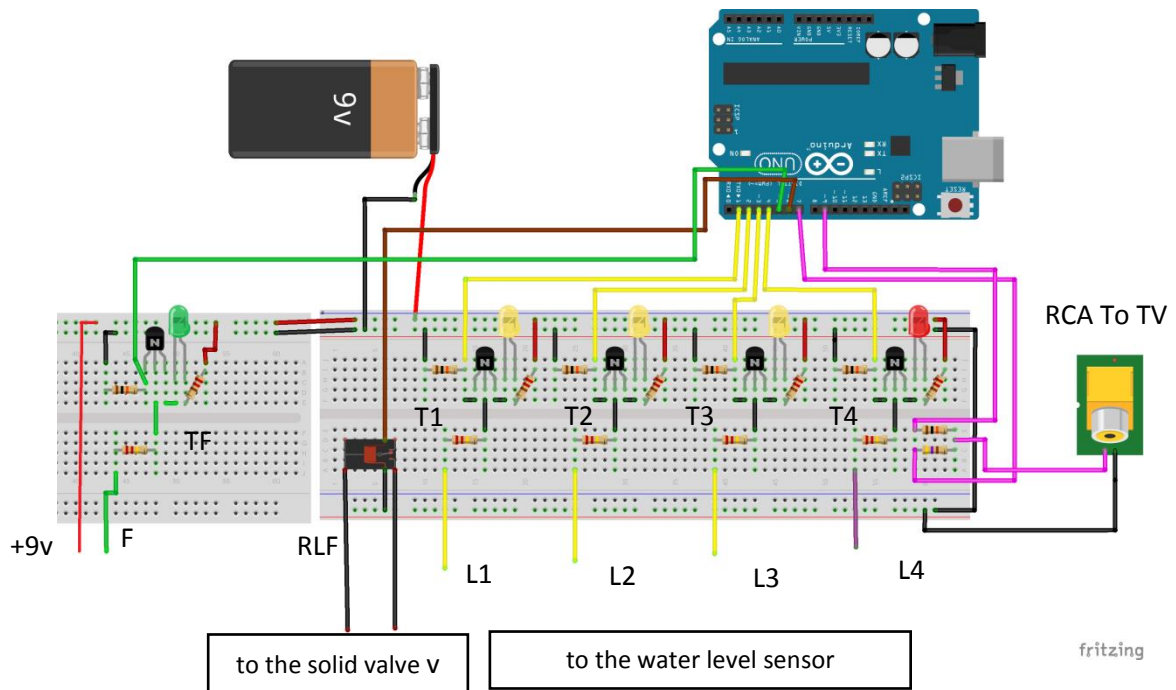
Figuer 3.1 the level sensor circuit

3.3. tasks of the project

3.3.1. The Main task

This task is all about sensing the presence of water filling the water tank

- 1, 2, 3, 4, are the base pins of the transistors (T1, T2, T3 and T4) submerged in the water tank, connected to pins 1 to 4 respectively.
- A is the +12v terminal.
- F is the base of the transistor TF, connected to pin 6 .
- RCA jack ,connected to pin 7 and 9 this jack must be conncted to a TV
- RLF is relay that controls the irrigation solid valve connected to pin 6.



Figuer 3.2 The maintask schematics

The transistors (T1, T2, T3, T4, TF), in this case are behaving as switches, Whene the circuit bitween the A terminal and one or more of the bases terminals of the transistors, the transistor turn on.

First of all, the arduino will check if there is water in the filling tube using the sensor TF.

Then the arduino will check the first level (lower level)using the sensor T1 , if tank is emty,the arduino will turn on the filling pump .arduino will show on tv screen the following “ *Filling the first level...* ” and draws a rectangular none filled.

Now, the first level is full, the arduino will check the second level, if it is empty, the arduino will keep the filling pump on and draws a rectangle filled in white to the quarter .

when the second level is full, the arduino will check the third level, if it is empty, the arduino will keep the filling pump on. The arduino will show on the tv screen the following “*the second level is full. Filling the third level...*” and draws a rectangle filled in white to the half .

when the third level is full, the arduino will check the fourth level, if it is empty the arduino will keep the filling pump on. The arduino will show on tv screen the following “*the third level is full. Filling the fourth level...*” and draws a rectangle filled in white to the 3 quarters.

Now the tank is full...the arduino will show on the TV screen the following “*the tank is full...*” and draws a rectangle filled in white.

3.3.2. the additional task

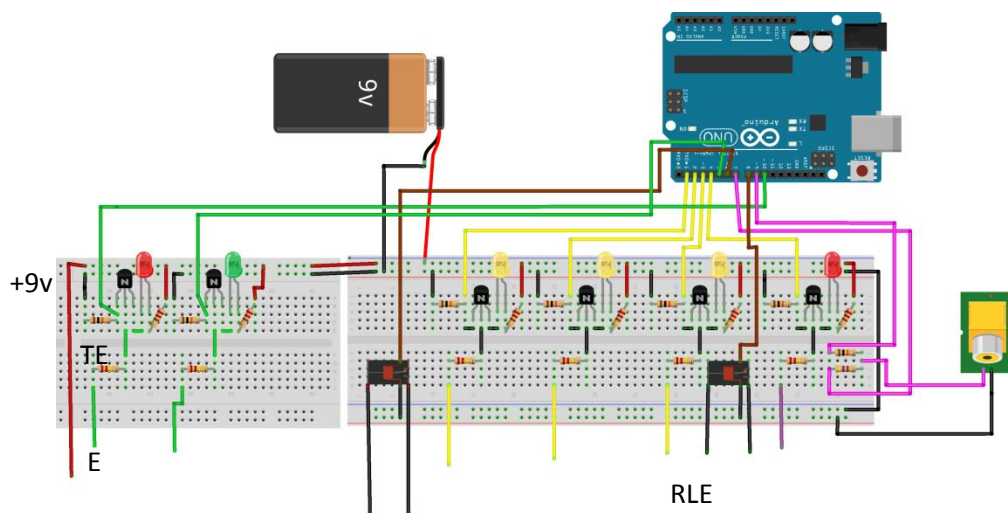


Figure 3.3. the main and additional tasks schematics

○ For small gardens and green houses

we add the following instructions to the previous schematic

- E is the base of the transistor TE, connected to pin 10 buried in the soil.
- RLE is relay that controls the filling pump , connected to pin 8.

the arduino will check soil moist by the sensor TE, if it is dry, the arduino will turn on relay that opens the irrigation solid valve ,arduino will writ on tv screen the following “*irrigating the soil...*”.

When the soil is well-irrigated, the arduino will shut the irrigation solid valve and will write on tv screen the following "*the soil is well-irrigated*".

○ **For Burns and stabels:**

we add the fallowing pieces to the previous schematic

- E is the base of the transistor TE, connected to pin 10 submerged in water Trough

the arduino will check water Trough by the sensor TE if it is emty, the arduino will turn on relay that opens the irrigation solid valve ,arduino will writ on tv screen the fallowing "*filling the water trough...*". When the soil is well-irrigated, the arduino will shut the irrigation solid valve and will writ on tv screen the fallowing "*the water troughis full*".

the arduino will read the analogic signals that comes from the tempearatur sensor and prossuse them then show the results on the tv screen.

3.4. Building sensor

3.4.1. Building the water level sensor

The requiriments:

- 2m of PVC tube.
- 2PVC elbow fittings.
- 2PVC tee fittings.
- 3 PVC crossfittings.
- 10 PVC cabs.
- 10 Stainless steel (inox) nuts.
- 5 different color wires.
- 10 Ring Terminals.

The sensor must be submerged in side the water trough as shown in the Figure3.4 and Figure3.5



Figure3.5: the water level sensor in reality

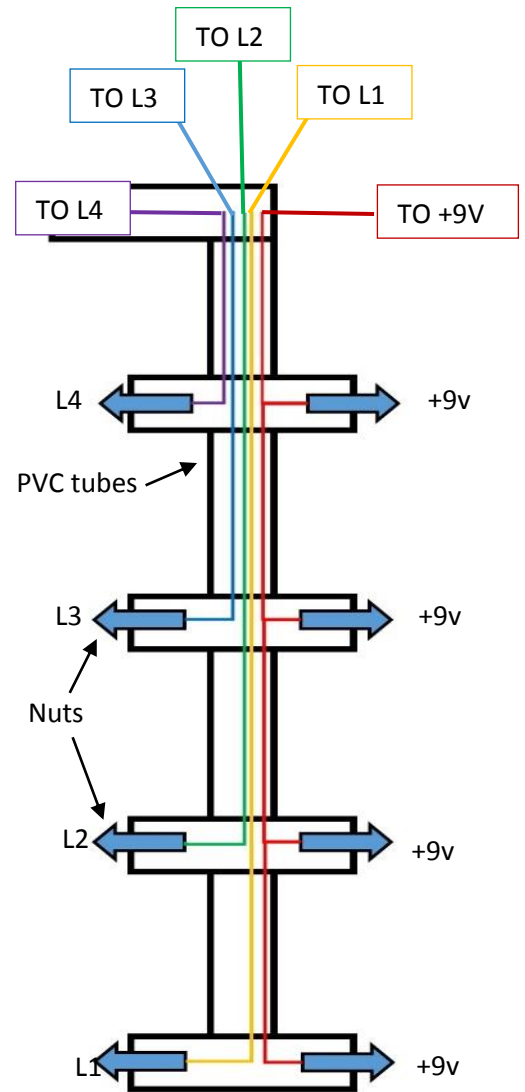


Figure3.4: the water level sensor architecture

3.4.2. Building the water existans sensor F

the same thing about this but we make sure that this sensor sties horisently when it is working is much effcint

- The requirments:
 - 25cm of PVC tube.
 - 2 PVC elbow fitting.
 - 2 PVC tee fittings.
 - 2 Stainless steelnuts.
 - 2 different color wires.
 - 2 Ring Terminals.

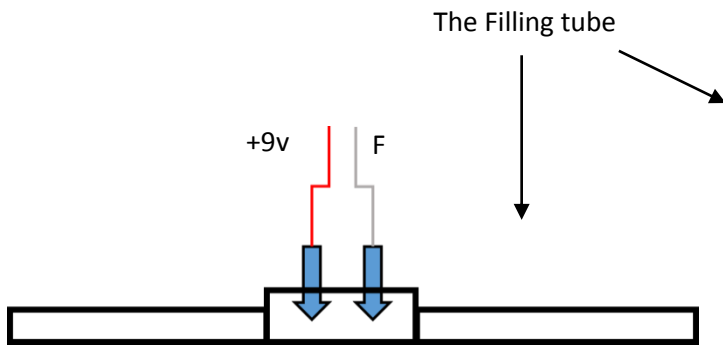


Figure3.6 the existans sensor in realty



Figure3.5 the existans sensor architecture

3.4.3. Building the soil moist sensor E

The requirments:

- 25cm of PVC tube.
- 1 PVC elbow fitting.
- 2 PVC tee fittings.
- 2 PVC cabs.
- 2Stainless steel nuts.
- 2 different color wires.
- 2 Ring Terminals.

The sensor must be baried in the soil or submerged in side the water trough like in the imege

Figure 3.6 and Figure 3.7

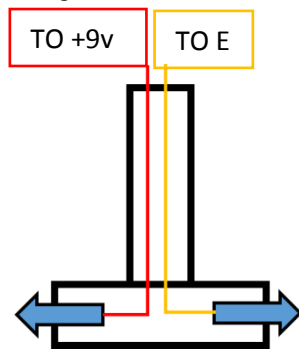


Figure3.8 : the soil moist sensor architecture



Figure3.7: the soil moist sensor in reality

3.5. The project scripts

Libraries are a collection of code that makes it easy for you to connect to a sensor, display, module, etc. For example in our case, the TV out library makes it easy to talk to character TV screen. We have two scripts:

3.5.1. The main task script

```
#include <TVout.h>
#include <video_gen.h>
TVout TV;
int L1 = 1;
int L2 = 2;
int L3 = 3;
int L4 = 4;
int Wf = 5;
int Rlf = 6;
void setup() {
    // put your setup code here, to run once:
    TV.begin(_PAL); // for PAL system
    pinMode(Rlf, OUTPUT);
```

```

{
  digitalWrite(R1f, LOW);
  TV.clear_screen();
  TV.set_cursor(2, 5);
  TV.print("the water tank is full");
  TV.draw_rect(66, 2, 20, 60, 1, 2);
  TV.draw_rect(66, 2, 20, 60, 1, 1);
  delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == LOW &&
digitalRead(L2) == LOW && digitalRead(L3) == LOW &&
digitalRead(L4) == LOW);
{
  digitalWrite(R1f, LOW);
  TV.clear_screen();
  TV.set_cursor(2, 5);
  TV.print("the water tank is empty");
  TV.set_cursor(2, 20);
  TV.print("there is no water in the filling tube....");
  TV.draw_rect(66, 2, 20, 60, 1, 0);
  delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH &&
digitalRead(L2) == LOW && digitalRead(L3) == LOW &&
digitalRead(L4) == LOW);
{
  digitalWrite(R1f, LOW);
  TV.clear_screen();
  TV.set_cursor(2, 5);
  TV.print("the water tank is 1/4 full");
  TV.set_cursor(2, 20);
  TV.print("there is no water in the filling tube....");
  TV.draw_rect(66, 2, 20, 60, 1, 0);
  TV.draw_rect(66, 39, 20, 60, 2, 0);
  delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH &&
digitalRead(L2) == HIGH && digitalRead(L3) == LOW &&
digitalRead(L4) == LOW);
{
  digitalWrite(R1f, LOW);

```


```

TV.clear_screen();
TV.set_cursor(2, 5);
TV.print("the water tank is 2/4 full");
TV.set_cursor(2, 20);
TV.print("there is no water in the filling tube. ...");
TV.draw_rect(66, 2, 20, 60, 1, 0);
TV.draw_rect(66, 28, 20, 60, 2, 0);
delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH &&
digitalRead(L2) == HIGH && digitalRead(L3) == HIGH &&
digitalRead(L4) == LOW);
{
digitalWrite(Rlf, LOW);
TV.clear_screen();
TV.set_cursor(2, 5);
TV.print("the water tank is 3/4 full");
TV.set_cursor(2, 20);
TV.print("there is no water in the filling tube...");
TV.draw_rect(66, 2, 20, 60, 1, 0);
TV.draw_rect(66, 2, 20, 60, 15, 0);
delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH &&
digitalRead(L2) == HIGH && digitalRead(L3) == HIGH &&
digitalRead(L4) == HIGH);
{
digitalWrite(Rlf, LOW);
TV.clear_screen();
TV.set_cursor(2, 5);
TV.print("the water tank is full");
TV.set_cursor(2, 20);
TV.print("there is no water in the filling tube...");
TV.draw_rect(66, 2, 20, 60, 1, 0);
TV.draw_rect(66, 2, 20, 60, 2, 0);
delay(500);
}
}
}

```

3.5.2. The main and the additional task script

```
#include <TVout.h>
#include <video_gen.h>
TVout TV;
int L1 = 1;
int L2 = 2;
int L3 = 3;
int L4 = 4;
int Wf = 5;
```




```

    delay(500);
}
if (digitalRead(Wf) == HIGH && digitalRead(L1) == HIGH && digitalRead(L2) ==
HIGH && digitalRead(L3) == LOW && digitalRead(L4) == LOW&&digitalRead(Wf) ==
HIGH);
{
    digitalWrite(Rlf , HIGH);
    TV.clear_screen();
    TV.set_cursor(2, 5);
    TV.print("the water tank is 2/4 full");
    TV.set_cursor(2, 20);
    TV.print("filling the third(3)level...");
    TV.draw_rect(66, 2, 20, 60, 2, 2);
    TV.draw_rect(66, 15, 20, 50, 1, 1); //Draws a rectangle
    delay(500);
}
if (digitalRead(Wf) == HIGH && digitalRead(L1) == HIGH && digitalRead(L2) ==
HIGH && digitalRead(L3) == HIGH && digitalRead(L4) == LOW&&digitalRead(Wf) ==
HIGH);
{
    digitalWrite(Rlf , HIGH);
    TV.clear_screen();
    TV.set_cursor(2, 5);
    TV.print("the water tank is 3/4 full");
    TV.set_cursor(2, 20);
    TV.print("filling the fourth(4)level...");
    TV.draw_rect(66, 2, 20, 60, 1, 2);
    TV.draw_rect(66, 2, 20, 60, 1, 1);
    delay(500);
}
if (digitalRead(Wf) == HIGH && digitalRead(L1) == HIGH && digitalRead(L2) ==
HIGH && digitalRead(L3) == HIGH && digitalRead(L4) == HIGH&&digitalRead(Wf) ==
HIGH);
{
    digitalWrite(Rlf, LOW);
    TV.clear_screen();
    TV.set_cursor(2, 5);
    TV.print("the water tank is full");
    TV.draw_rect(66, 2, 20, 60, 1, 2);
    TV.draw_rect(66, 2, 20, 60, 1, 1);
    delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == LOW && digitalRead(L2) == LOW
&& digitalRead(L3) == LOW && digitalRead(L4) == LOW&&digitalRead(Wf) == HIGH);
{
    digitalWrite(Rlf, LOW);
    TV.clear_screen();
    TV.set_cursor(2, 5);
    TV.print("the water tank is emty");
}

```

```

    TV.set_cursor(2, 20);
    TV.print("filling the first level...");
    TV.draw_rect(66, 2, 20, 60, 1, 0);
    delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH && digitalRead(L2) == LOW
&& digitalRead(L3) == LOW && digitalRead(L4) == LOW&&digitalRead(Wf) == HIGH);
{
    digitalWrite(Rlf, LOW);
    TV.clear_screen();
    TV.set_cursor(2, 5);
    TV.print("the water tank is 1/4 full");
    TV.set_cursor(2, 20);
    TV.print("filling the first level...");
    TV.draw_rect(66, 2, 20, 60, 1, 0);
    TV.draw_rect(66, 39, 20, 60, 2, 0);
    delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH && digitalRead(L2) == HIGH
&& digitalRead(L3) == LOW && digitalRead(L4) == LOW&&digitalRead(Wf) == HIGH);
{
    digitalWrite(Rlf, LOW);
    TV.clear_screen();
    TV.set_cursor(2, 5);
    TV.print("the water tank is 2/4 full");
    TV.set_cursor(2, 20);
    TV.print("filling the first level...");
    TV.draw_rect(66, 2, 20, 60, 1, 0);
    TV.draw_rect(66, 28, 20, 60, 2, 0);
    delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH && digitalRead(L2) ==
HIGH && digitalRead(L3) == HIGH && digitalRead(L4) == LOW&&digitalRead(Wf) ==
HIGH);
{
    digitalWrite(Rlf, LOW);
    TV.clear_screen();
    TV.set_cursor(2, 5);
    TV.print("the water tank is 3/4 full");
    TV.set_cursor(2, 20);
    TV.print("there is no water in the filling tube...");
    TV.draw_rect(66, 2, 20, 60, 1, 0);
    TV.draw_rect(66, 2, 20, 60, 15, 0);
    delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH && digitalRead(L2) ==
HIGH && digitalRead(L3) == HIGH && digitalRead(L4) == HIGH&&digitalRead(Wf) ==
HIGH);
{

```

```

digitalWrite(R1f, LOW);
TV.clear_screen();
TV.set_cursor(2, 5);
TV.print("the water tank is full");
TV.set_cursor(2, 20);
TV.print("there is no water in the filling tube...");
TV.draw_rect(66, 2, 20, 60, 1, 0);
TV.draw_rect(66, 2, 20, 60, 2, 0);
delay(500);
}
if (digitalRead(Wf) == HIGH && digitalRead(L1) == LOW && digitalRead(L2) ==
LOW && digitalRead(L3) == LOW && digitalRead(L4) == LOW&&digitalRead(Wf) == LOW);
{
digitalWrite(R1f , HIGH);// turn the filling valve on
digitalWrite(R1e , HIGH);// turn the epmtying valve on
TV.clear_screen();// clearing the tv screen
TV.set_cursor(2, 5);//shifting the cursor x=2 , y=5
TV.print("the water tank is emty"); // printing on the tv screen"the water
tank is emty"
TV.set_cursor(2, 20);//shifting the cursor x=2 , y=20
TV.print("filling the first(1)level...");//printing on the tv screen"filling
the first(1)level..."
TV.draw_rect(66, 2, 20, 50, 1, 0);//Draws a rectangle(x,y,w,h,colour,
fill)with the top-left corner at x,y; width w, height h, colour and optional fill
colour.
delay(500);
}
if (digitalRead(Wf) == HIGH && digitalRead(L1) == HIGH && digitalRead(L2) ==
LOW && digitalRead(L3) == LOW && digitalRead(L4) == LOW&&digitalRead(Wf) == LOW);
{
digitalWrite(R1f , HIGH);// turn the filling valve on
digitalWrite(R1e , HIGH);// turn the epmtying valve on
TV.clear_screen();
TV.set_cursor(2, 5);
TV.print("the water tank is 1/4 full");// printing on the tv screen"the water
tank is 1/4 full"
TV.set_cursor(2, 20);
TV.print("filling the secand(2)level...");
TV.draw_rect(66, 2, 20, 50, 1, 1);
TV.draw_rect(66, 28, 20, 50, 1, 0); //Draws a rectangle
delay(500);
}
if (digitalRead(Wf) == HIGH && digitalRead(L1) == HIGH && digitalRead(L2) ==
HIGH && digitalRead(L3) == LOW && digitalRead(L4) == LOW&&digitalRead(Wf) == LOW);
{
digitalWrite(R1f , HIGH);// turn the filling valve on
digitalWrite(R1e , HIGH);// turn the epmtying valve on
TV.clear_screen();
TV.set_cursor(2, 5);

```

```

    TV.print("the water tank is 2/4 full");// printing on the tv screen"the water
tank is 2/4 full"
    TV.set_cursor(2, 20);
    TV.print("filling the third(3)level...");
    TV.draw_rect(66, 2, 20, 60, 2, 2);
    TV.draw_rect(66, 15, 20, 50, 1, 1); //Draws a rectangle
    delay(500);
}
    if (digitalRead(Wf) == HIGH && digitalRead(L1) == HIGH && digitalRead(L2) ==
HIGH && digitalRead(L3) == HIGH && digitalRead(L4) == LOW&&digitalRead(Wf) ==
LOW);
    {
        digitalWrite(Rlf , HIGH);// turn the filling valve on
        digitalWrite(Rle , HIGH);// turn the epmtying valve on
        TV.clear_screen();
        TV.set_cursor(2, 5);
        TV.print("the water tank is 3/4 full");// printing on the tv screen"the water
tank is 3/4 full"
        TV.set_cursor(2, 20);
        TV.print("filling the fourth(4)level...");
        TV.draw_rect(66, 2, 20, 60, 1, 2);
        TV.draw_rect(66, 2, 20, 60, 1, 1);
        delay(500);
    }
    if (digitalRead(Wf) == HIGH && digitalRead(L1) == HIGH && digitalRead(L2) ==
HIGH && digitalRead(L3) == HIGH && digitalRead(L4) == HIGH&&digitalRead(Wf) ==
LOW);
    {
        digitalWrite(Rlf, LOW);
        digitalWrite(Rle , HIGH);// turn the epmtying valve
        TV.clear_screen();
        TV.set_cursor(2, 5);
        TV.print("the water tank is full");// printing on the tv screen"the water
tank is full"
        TV.draw_rect(66, 2, 20, 60, 1, 2);
        TV.draw_rect(66, 2, 20, 60, 1, 1);
        delay(500);
    }
    if (digitalRead(Wf) == LOW && digitalRead(L1) == LOW && digitalRead(L2) == LOW
&& digitalRead(L3) == LOW && digitalRead(L4) == LOW&&digitalRead(Wf) == LOW);
    {
        digitalWrite(Rlf, LOW);
        digitalWrite(Rle , LOW);// turn the epmtying valve
        TV.clear_screen();
        TV.set_cursor(2, 5);
        TV.print("the water tank is emty");
        TV.set_cursor(2, 20);
        TV.print("there is no water in the filling tube...");
        TV.draw_rect(66, 2, 20, 60, 1, 0);
    }

```

```

    delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH && digitalRead(L2) == LOW
&& digitalRead(L3) == LOW && digitalRead(L4) == LOW&&digitalRead(Wf) == LOW);
{
    digitalWrite(Rlf, LOW);
    digitalWrite(Rle , HIGH);// turn the epmtying valve
    TV.clear_screen();
    TV.set_cursor(2, 5);
    TV.print("the water tank is 1/4 full");
    TV.set_cursor(2, 20);
    TV.print("there is no water in the filling tube...");
    TV.draw_rect(66, 2, 20, 60, 1, 0);
    TV.draw_rect(66, 39, 20, 60, 2, 0);
    delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH && digitalRead(L2) == HIGH
&& digitalRead(L3) == LOW && digitalRead(L4) == LOW&&digitalRead(Wf) == LOW);
{
    digitalWrite(Rlf, LOW);
    digitalWrite(Rle , HIGH);// turn the epmtying valve
    TV.clear_screen();
    TV.set_cursor(2, 5);
    TV.print("the water tank is 2/4 full");
    TV.set_cursor(2, 20);
    TV.print("there is no water in the filling tube...");
    TV.draw_rect(66, 2, 20, 60, 1, 0);
    TV.draw_rect(66, 28, 20, 60, 2, 0);
    delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH && digitalRead(L2) ==
HIGH && digitalRead(L3) == HIGH && digitalRead(L4) == LOW&&digitalRead(Wf) ==
LOW);
{
    digitalWrite(Rlf, LOW);
    digitalWrite(Rle , HIGH);// turn the epmtying valve
    TV.clear_screen();
    TV.set_cursor(2, 5);
    TV.print("the water tank is 3/4 full");
    TV.set_cursor(2, 20);
    TV.print("there is no water in the filling tube...");
    TV.draw_rect(66, 2, 20, 60, 1, 0);
    TV.draw_rect(66, 2, 20, 60, 15, 0);
    delay(500);
}
if (digitalRead(Wf) == LOW && digitalRead(L1) == HIGH && digitalRead(L2) ==
HIGH && digitalRead(L3) == HIGH && digitalRead(L4) == HIGH&&digitalRead(Wf) ==
LOW);
{

```

```

digitalWrite(R1f, LOW);
digitalWrite(R1e , HIGH);// turn the emptying valve
TV.clear_screen();
TV.set_cursor(2, 5);
TV.print("the water tank is full");
TV.set_cursor(2, 20);
TV.print("there is no water in the filling tube...");
TV.draw_rect(66, 2, 20, 60, 1, 0);
TV.draw_rect(66, 2, 20, 60, 2, 0);
delay(500);
}

```

3.6. The system in action

3.6.1. In homes

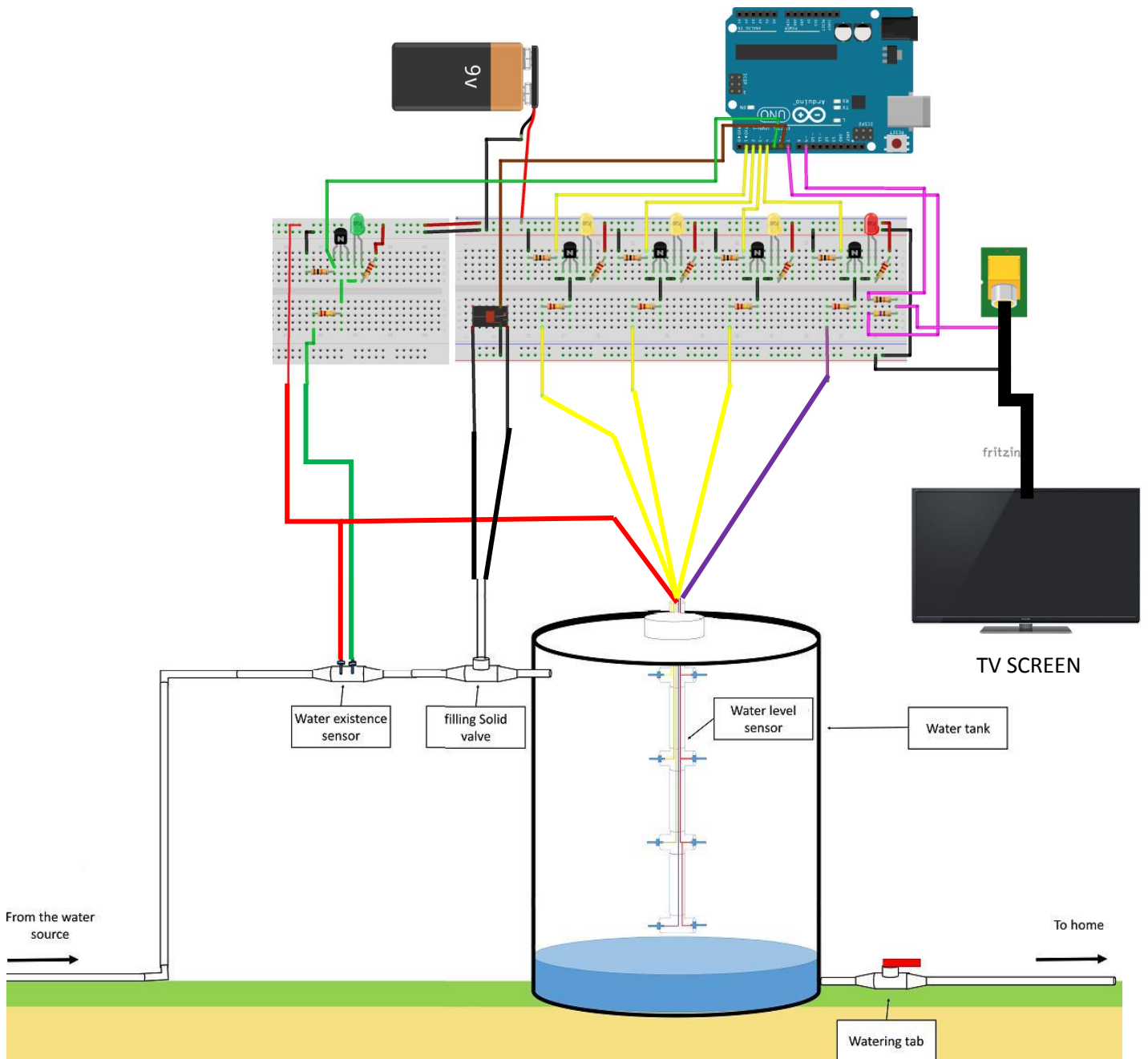


Figure 3.9 the water level monitor

To test the project we must put it to action. In figure 3.7 shows the whole operation. First, we tested the water exists sensor, the water level sensor without wiring it with the arduino.

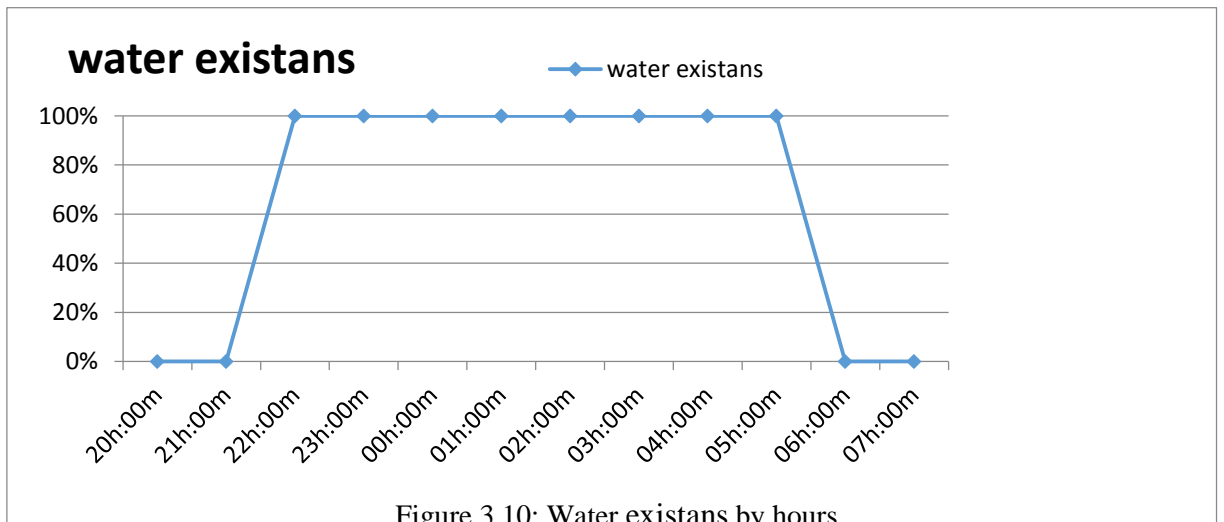


Figure 3.10: Water existans by hours

The water exists sensor detected water at 21h:00m and until 06h:00m the day after as we can see in figure 3.9. In my case, water shift is between 20h:50m and until 06h:12m, which is pretty amusing.

Then we tested the water level sensor, the sensor detects levels while the tank is filling. We could see clearly that the LEDs are glowing one by one until the filling is complete in the same period. See figure 3.10 later on the day exactly at 12h:23, I took a TV screen figure 3.12.

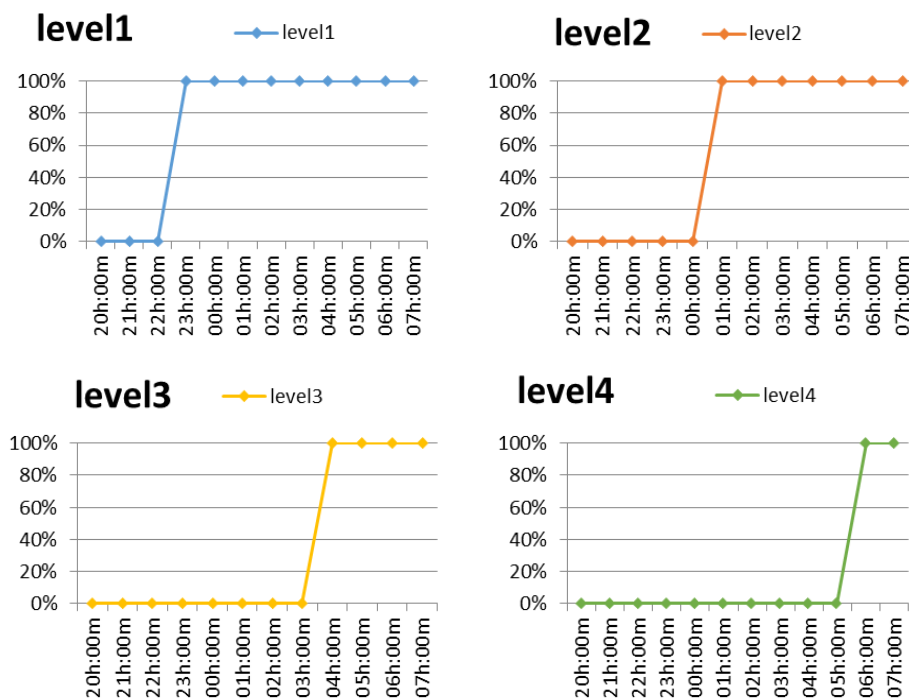


Figure 3.11: water levels per hour



Figure 3.12: Water level sensor In action

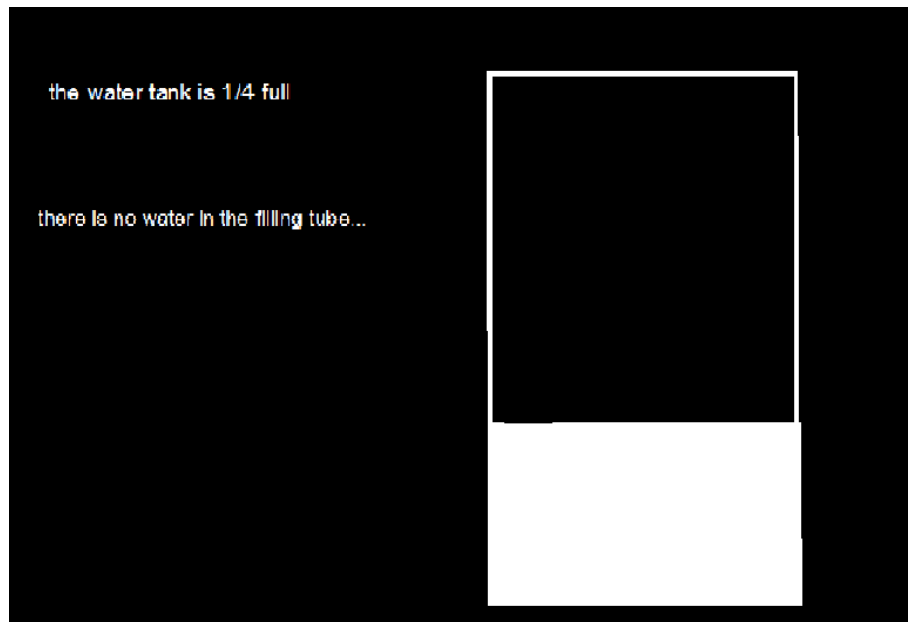


Figure 3.13: TV screen shot

3. Conclusion

- In this chapter we saw how can build a water level sensor and monitor based on simple physical principle which is electrical conductivity of fresh water using Adriano Uno
- This system could be used in individual or collaclative habitats, hospitals, schools and universities or gardens and even green houses, burns, stables...etc.
- By exploring the electrical conductivity propriety of fresh water to detect the water level in the water container or soil moist. , with a Adriano Uno, TV screen, couple of electrical components and recycled materials and tubes and fitting that could be easily found in junk yards.

• Bibliography

[1]	John nesy,Arduino, for Dummies , 2013 John Wiley & Sons, Ltd, Chichester, West Sussex, England
[2]	Exploring Arduino Tools and Techniques for Engineering Wizardry
[3]	Kimmo karvinen, tero karvinen,make, Arduino bots and gadgets. o'reilly 2014
[4]	James .a .langdridge Arduino robotics projects ,2014 by John Wiley & Sons, Inc., Indianapolis, Indiana
[5]	MARTIN EVANS, JOSHUA NOBLE, JORDAN HOCHENBAUM .Arduino in action. ©2013 by ning Publications Co.
[6]	James .a .langdridge .Arduino sketches. 2015 by John Wiley & Sons, Inc., Indianapolis, Indiana
[7]	Brigitte d'Andréa-Novel, Michel De Lara Control Theory for Engineers - A Primer - (Springer, 2013)
[8]	Marco jill,The Electronics Engineers' Handbook, 5th Edition 2005
[9]	Handbook industrial engineering 2007
[10]	Instrumentation & Control Process Control Fundamentals
[11]	Overview of industrial process automation Elsevier (2011)
[12]	Introduction to Instrumentation, Sensors, and Process Control – William C. Dunn (Artech House, 2006)
[13]	Instruments and Control Process Fundamentals of Control. 2006 PAControl.com
[14]	Béla G. Lipták ,Instrument Engineers' Handbook 4th ed Vol. 2 - Process Control and Optimization (CRC, ISA, 2006)
[15]	Process Automation Handbook – Jonathan Love (Springer, 2007)
[16]	Michael McRoberts ,Beginning Arduino, technology in action 2013
[17]	https://www.arduino.cc/en/Guide/Introduction
[18]	https://www.arduino.cc/en/Guide/Environment
[19]	https://www.arduino.cc/en/Guide/ArduinoGSMShield
[20]	https://en.wikipedia.org/wiki/Control_theory
[21]	https://www.site.uottawa.ca/~rhabash/ELG4152LN01.pdf
[22]	https://en.wikipedia.org/wiki/Pressure_sensor
[23]	https://en.wikipedia.org/wiki/Level_sensor
[24]	https://www.geeker.co.nz/sensors/temperature/lm35-temperature-sensor.html
[25]	https://www.arduino.cc/en/Main/ArduinoMotorShieldR3
[26]	https://www.arduino.cc/en/Main/ArduinoYunShield
[27]	https://www.arduino.cc/en/Main/ArduinoXbeeShield

Conclusion

At the end of this project we have learned about the Arduino as a power full tool to solve a lots problems after this brief explaining of control know we have the keys of the future of industrial and home automation.

We learned how to make a simple water level sensor and simple water prisons sensor , soil moist sensor with basic principles of electronic.

This is as I mentioned earlier is a contribution to solve one of the most biggest problems of our plant ;but the huge part of saving our plant is up on the shoulders of all mankind.

List of figures

Figure 1.1: An early Wiring board
Figure 1.2 : Arduino Uno
Figure 1-3: The Arduino Uno
Figure 1-4: The Arduino Leonardo
Figure 1-5: The Arduino Mega 2560
Figure 1-6: The Arduino Due
Figure 1-7: The Arduino Nano
Figure 1-8: The Arduino Mega ADK
Figure 1-9: The LilyPad Arduino
Figure 1-10: Quadcopter and ArduPilot Mega controller
Figure 1.11: An A-B USB cable and Arduino Uno.
Figure 1.12:New hard ware found —or not, as the case may be.
Figure 1-13: Installing drivers in Device Manager
Figure 1-14: A beautiful Turquoise Arduino window in Windows 7
Figure 1-13: The components of the Blink program
Figure1-14: Arduino GSM Shield
Figure1-15: Arduino motor Shield
Figure1-16: Arduino Yún Shield
Figure 2.1 block diagram of feedback loop
Table 2.1 characteristics and examples of Advanced Control Models
Figure 2.2: Sequential Process Control
Figure 2.3:Process Control
Figure 2.4:an example of Process Control
Figure 2.5: Open-loop control system (no feedback)
Figure 2.6: Closed-loop feedback control system
Figure 2.7: road-map of control models
Figure 2.8:Analog inputs and output Instrumentation Devices
Figure 2.9:Analog inputs and output Instrumentation Devices-input/output relationship
Figure 2.10:Analog inputs and output Instrumentation Devices-components
Figure 2.11:Analog inputs and output Instrumentation Devices- examples
Figure 2.12: LM35 Temperature Sensor

Figure 2.13: LM35 Digital air pressure sensor
Figure 2.14 Capacitance level sensor
Figure 2.15: classical microprocessors architectuer
Figuer 3.1 the level sensor circuit
Figuer 3.2 The mean task schmatics
Figure 3.3 the mean and additional tasks schematics
Figure 3.4: the water level sensor architecture
Figure 3.5: the water level sensor in reality
Figure 3.6 the existans sensor in realty
Figure 3.7 : the soil moist sensor architecture
Figure 3.7 : the soil moist sensor in reality
figuer 3.8: the water level monitor
figuer 3.9: Water existans by hours
figuer 3.10: water levels per hour
figuer 3.11: Water level sensor In action
figuer 3.12: TVscreen shot