PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

UNIVERSITY AMAR TELIDJI – LAGHOUAT

Faculty of Technology

Electronics Department

MASTER REPORT

Presented by:

ALLAOUI KHADIDJA

FIELD: Technology

SECTOR: Electronics

OPTION: Electronics of embedded systems

Theme

Conception & realization of a Smartwatch controlled by Android Smartphone

Defense jury:

Mr. Aboubaker Hadj aissa	**Pr**	**President**
Mr. Belkhairi Mohammed	**MAA**	**Examiner**
Dr. GUEFFAF Hamza	**MCA**	**Reporter**

Section :2019/2020

Dedication

I dedicate my dissertation work to my family and many friends. A special feeling of gratitude to my loving parents, whose words of encouragement and push for tenacity ring in my ears.

To my uncle BENUIT MAHMOUD I appreciate all what you've done to me during this PROJECT, and to my two beloved families ALLAOUI and BENGUIT I am so lucky to have all of you by my side especially my dear MAYMA may Allah keep you always in a good health to guide as with your wise words

I also dedicate this thesis to my many friends especially my dear YOUSRA & LINA and all CHAOUI family for being there for me throughout the entire Master program. And to Feyrouz & Meriem who have supported me throughout the process

I dedicate this work and give special thanks to my supervisor GUEFFAF HAMZA I really am blessed to accomplish this project with such an intelligent and supportive professor like you.

Thank you, My love for you all can never be quantified.

Khadidja

Acknowledgment

First of all i am grateful to the God for the good health and wellbeing that were necessary to complete this project.

Apart from the efforts of myself, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been with me during this very intense year.

I would like to thank my supervisor **GUEFFAF HAMZA** for the thoughtful comments and recommendations on this dissertation, without your help and wise guidance this project would have not been the same, and especially your support through all the conditions that we've been through.

I cannot forget all of my other teachers for their help and the brilliant idea's that they gave to me, because thanks to their contribution this thesis has been realized.

Finally my biggest thanks to my family and friends for all the support they have shown me through this research.

Thank you all

Summary

Chapter Ⅱ : Wearable technology & SmartWatches

Chapter Ⅲ: ARDUINO, Programming & Application

Figures List

Chapter Ⅰ : Smart technology with Embedded systems

Chapter Ⅱ: Wearable technology & SmartWatches

Chapter Ⅲ: ARDUINO, Programming &Application

Tables List

Chapter Ⅱ: Wearable technology & SmartWatches

Chapter Ⅲ: ARDUINO, Programming & Application

General Introduction

An embedded system is typically some combination of hardware and software, either fixed in function or programmable, it system could be designed to support a specific function or functions within a larger system it include industrial control systems and machines automobiles, military systems such as avionics and weapons system, medical equipment, consumer products, smart phones, and building automation. There are beneficial uses of embedded systems in our daily life which are raising people's standard of living in many fields

The past year has certainly been an exciting time for the semiconductor industry. The shared enthusiasm among silicon companies has been on the rise based on the recent level of interest and demand for new products that complement the mobile experience typically based around smartphones and tablets. A main driver behind this phenomenon is wearable technology. Familiar items such as clothing, glasses, jewelry, and watches are being fitted with sensors, processors, and displays, technologies we wouldn't have expected to find inside these everyday objects before. The potential for this market is quite impressive

Wearable Embedded systems are cyber-physical systems that are carried from persons in their daily activities and can assist in quality of life through augmented sensing for the disabled, independent living for the elderly, and reduced healthcare costs. The implementation of wearable embedded systems is normally based on commercial available chipsets while recently efforts have focused on the development of e-textiles that will effectively disappear these systems in the person's clothes. The challenges faced with this type of embedded systems includes strict requirements for the size and energy resources together with severe constraints in computational power, while the highly variable environment due to person's movements and the body itself results to a complicated transmission channel for wireless connectivity affecting the bandwidth and the quality of communication.

The main object of my thesis is to create a smartwatch connected to a smartphone application using a Bluetooth HC06 component to send data to our arduino board that transfer it to an OLED screen, reaching to the main purpose of my final project I've divided my work to 3 chapters as follows :

- The 1st chapter discuss smart technology with embedded systems starting with some historical overview of the development of embedded systems and the latest trends of this technology
- The 2nd chapter I'll be talking about wearable devices and their importance and the last section will be about smart watches and their benefits in humans life, their performance and the different categories their used in
- The 3rd and the last chapter include the demonstration of Arduino cards, it types and the most used cards, also how to program on the arduino IDE.

Finlay, like any research work, a conclusion is obvious, it will take up the essential points of all the work and the perspectives envisaged. Then we'll present the bibliographic list which was used in the development of this thesis.

Ⅰ.1 Introduction:

Today, electronics are increasingly being replaced by programmed electronics. We also speak of an embedded system or embedded computing. Its purpose is to simplify electronic diagrams and therefore reduce the use of electronic components, thereby reducing the cost of the product manufacturing. It result more efficient and complex systems for reduced space. Electronics has become accessible to all people with the desire to explore more and get into the world of smart technology, what we will learn in this chapter is a mixture of electronics and the effect of smart technology on embedded systems. We are indeed going to talk about embedded electronic which is a sub-domain of electronics and which has the ability to combine the power of programming with the power of electronics.

Ⅰ.2 History:

One of the earliest electronic computing devices credited with the term “embedded system” and closer to our present conception of such was the Apollo Guidance Computer (AGC). Developed at the MIT Instrumentation Laboratory by a group of designers led by Charles Stark Draper in the early 1960s, the AGC was part of the guidance and navigation system used by NASA in the Apollo program for various spaceships. In its early days it was considered one of the riskiest items in the Apollo program due to the usage of the then newly developed monolithic integrated circuits.

The beginning of the decade of 1970 witnessed the development of the first micro- processor designs. By the end of 1971, almost simultaneously and independently, design teams working for Texas Instruments, Intel, and the US Navy had developed implementations of the first microprocessors. Gary Boone from Texas Instruments was awarded in 1973 the patent of the first single-chip microprocessor architecture for its 1971 design of the TMS1000 . This chip was a 4-bit CPU that incorporated in the same die 1K of ROM and 256 bits of RAM to offer a complete computer functionality in a single-chip, making it the first microcomputer-on-a-chip (a.k.a.microcontroller). TheTMS1000 was launched in September 1971 as a calculator chip with part number TMS1802N

Microprocessors-based embedded applications had grown to hundreds of millions of dollars. The list of initial players grew to more than a dozen of chip manufacturers that, besides Texas

Instruments and Intel, included Motorola, Zilog, Intersil, National Instruments, MOS Technology, and Signetics, to mention just a few of the most renowned. Remarkable parts include the Intel 8080 that eventually evolved into the famous 80×86/Pentium series, the Zilog Z-80, Motorola 6800 and MOS 6502. The evolution in CPU sizes continued through the 1980s and 1990s to 16-, 32-, and 64-bit designs, and now-a-days even some specialized CPUs crunching data at 128-bit widths. In terms of manufacturers and availability of processors, the list has grown to the point that it is possible to find over several dozens of different choices for processor sizes 32-bit and above, and hundreds of 16- and 8-bit processors. Examples of manufacturers available today include Texas Instruments, Intel, Microchip, Freescale (formerly Motorola), Zilog, Advanced Micro Devices, MIPS Technologies, ARM Limited, and the list goes on and on.

Nowadays microprocessor applications have grown in complexity; requiring applications to be broken into several interacting embedded systems. [1]

Ⅰ.3 Overview of embedded systems:

Embedded systems are a combination of software as well as hardware which is either programmable or fixed in capabilities. These systems are designed for a particular function. Automobiles, medical equipment, cameras, industrial machines, household appliances, airplanes, vending machines, toys, cellular phones and PDA (personal digital assistant) are among the myriad of possible hosts of the system. Embedded systems are the key enablers of smart devices which represent one of the technological marvels of the **21**st century. [2]

Ⅰ.3.1 Definition of a system:

A system is an arrangement in which all its unit assemble work together according to a set of rules. It can also be defined as a way of working, organizing or doing one or many tasks according to a fixed plan. For example, a watch is a time displaying system. Its components follow a set of rules to show time. If one of its parts fails, the watch will stop working. So we can say, in a system, all its subcomponents depend on each other. [3]

Ⅰ.3.2 Definition of embedded system:

As its name suggests, Embedded means something that is attached to another thing. An embedded system can be thought of as a computer hardware system having software embedded in it. It can be an independent system or it can be a part of a large system.

An embedded system is a microcontroller or microprocessor based system which is designed to perform a specific task. [3]

Ⅰ.3.3 Embedded system category:

- **Mandiri.(Standalone-device):**

  Can function independently of other hardware. Not integrated into other devices. Example: TiVo Box for recording TV broadcasts. While the DVR (digital video recorder) is an embedded system that integrates with a DVD player. Stand alone can also refer to software programs that do not require software other than the operating system to run it.

- **Real-Time:**

  An embedded system with specific tasks performed within a specific time period is called the real time system. Real time system consists of hard real time system and soft real time system.

- **Hard.real-time**

  Hard real time is a system that must perform tasks with the right deadlines. An example of a real-time hard drive system is a system that must open the valve within 30 milliseconds when the humidity of the air crosses a certain threshold. If the valve is not opened within 30 milliseconds it will cause havoc. The real-time hard system is often used as a controller for dedicated applications, having fixed time limits. Processing must be completed within defined constraints, or the system will fail.

- **Soft.real-time**

  Soft real time is a system that does not require deadlines. Example of soft real-time like DVD player, if given a command from the remote control it will experience delay for several milliseconds to run the command. This delay will not result in anything serious. Real-time soft systems have fewer hard time constraints, and do not support deadlines by using the deadline[3]

Ⅰ.3.4 Embedded systems and applications:

Nowadays, the world is becoming more and more digital, connected and automated. It is a technology that is often overlooked which is embedded system that are an essential part of electronics. If we talk about technology, we think about a mobile, laptop, computer, and cameras, etc. But we never think or talk about an embedded system that is running them. It is a part of a larger device and provides a particular function. [3]

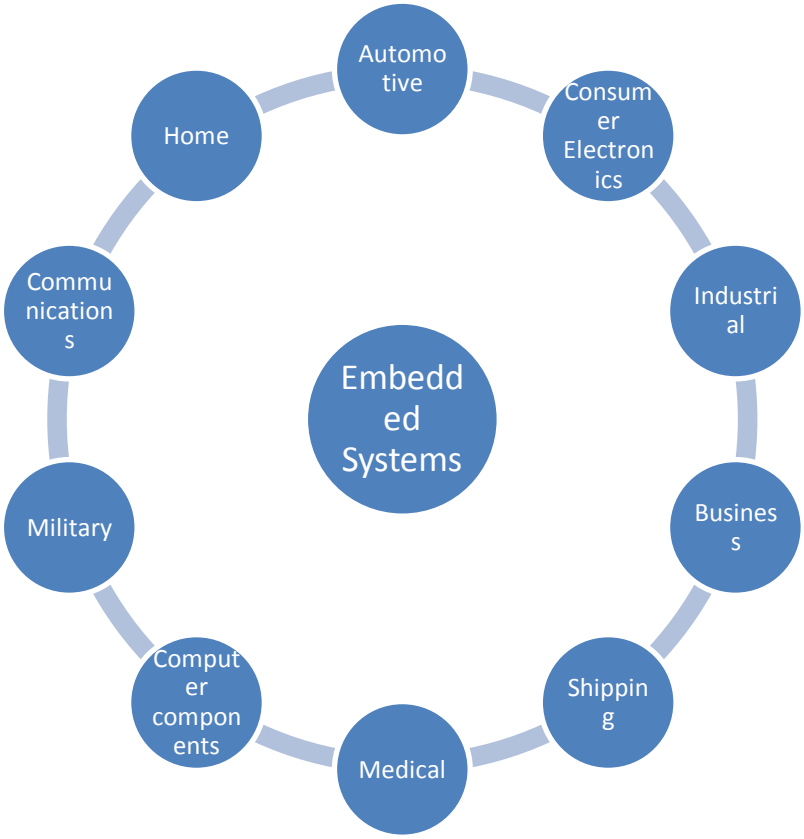


Fig Ⅰ.1. embedded system fields

Ⅰ.3.5 Latest technology in embedded systems and application:

Embedded systems and applications are severely increased, The growth of Internet of Things (IoT) is altering the dynamics of the smartphone device market. It has opened up an excellent avenue for a software system called embedded Java. Embedded sensors, image recognition, and near field communication payment technologies are the major IoT components which are embedded in smartphone devices. Such application is vital for the adoption of the integrated system and is expected to expand shortly attributed to the

advancements in the technology in coming years. the latest technology in embedded systems and application are:

- **Artificial intelligence:**

AI can be defined as the enabling of a machine to perform the logical analysis, obtain knowledge and adapt to an environment that varies. This technology is already being used in many applications such as self-driving cars, chatbots, personal voice assistant and super smart computing intensive.

- **Virtual reality and Augmented reality:**

Virtual reality is a way to generate realistic images, sound, and other sensations. VR with higher resolution will challenge available display and processor technology. On the other hand, augmented reality is the latest innovations in the electronics industry. Augmented Reality adds virtual stuff to the real world environment. These both technologies will become a big part of our world.

- **Deep Learning :**

It represents a rich and yet unexplored embedded systems market that has a range of applications from image processing to audio analysis

- **Embedded Security:**

With the rise of the Internet of Things, the focus of developers and manufacturers is on security. The advanced technologies for embedded security will emerge as crucial generators for identifying devices in an IoT network, and as microcontroller security solutions that isolate security operations from normal operations.

- **Cloud connectivity:**

Cloud connectivity technology is an important future market for embedded systems. These technologies are designed to simplify the process of connecting embedded systems with cloud-based services by reducing the underlying hardware complexities. [4]

Ⅰ.3.6 Key steps to create an embedded product:

The best way to start writing software that would directly affect physical objects is to explore such embedded platforms as **Arduino**, **Raspberry Pi**, or **Particle**, and in our case we'll be using the ARDUINO platform. To develop a viable product we take the following steps:

✓ Step 1. Learn C or C++:

Most of the embedded software is now written in two languages, C and C++. There is not much difference between C and C++ in terms of syntax. However, C++ has some additional features, like enhanced security and closeness to real-world applications, while C is considered to be more reliable and showing better performance and directly interacting with the hardware.

✓ Step 2. Learn Some Basic Electronics:

At least to the extent the user should understand what voltage, current, power, resistance, and ohms law are.

✓ Step 3. Get the Basic Equipment:

Embedded programmers actually interact with the physical world, so such things as soldering iron, Digital Multi-Meter (DMM), and a hardware debugger/ JTAG adapter (such as an ST-Link, or OLMEX adapter) or a Logic Analyzer would be of help.

✓ Step 4. Choose a Microcontroller and Tool chain:

To make our program run we need a microcontroller and for that we use a development board like arduino that contains usually a microcontroller next to other equipments that we might need

✓ Step 5. Understand the Datasheets:

Before actually sitting down to write the first line of our code, we need to understand the (end user) specifications.

✓ Step 6: Examine the components:

Analyze and pick up the components (*software and hardware*) required to make the product.

✓ Step 7: Design a product:

Designing is always the most critical phase of any development cycle. The peculiarity of the embedded programming is that we have to develop the hardware and software parts individually and integrate both.

✓ Step 8: Develop a prototype:

A prototype is a sample version created to test the concept which is developed according to the specifications using the selected hardware and software tool.

✓ Step 9: Test the application:

Now that the prototype it is possible to run we test cases to prove the possible potential of the application.

✓ Step 10: Deploy the application:

After testing the application, the result is checked in a real environment to realize the **Proof Of Concept**

✓ Step 11: Support and Upgrade

If needed, we should be ready to provide support and upgrade the application with new features[5]

Ⅰ.3.7 Programming Languages Popularity

The most popular programming language for embedded systems development has been the C language. It has been the standard option in this industry for too long. Recent statistics prove this fact to still be true for the present and the seeable future

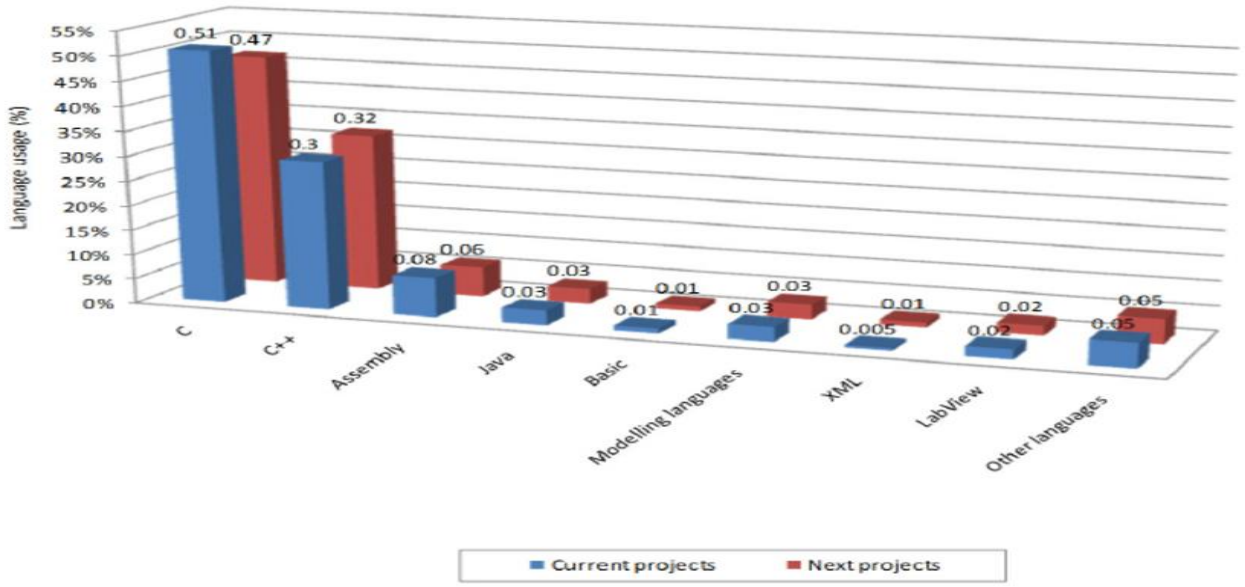


Fig Ⅰ.2. Programming language popularity

Note: the most popular programming language for embedded development is **C** followed by **C++** and **Assembly**. Other language options are slowly growing in traction as we need more and more OS-Based applications.

And to answer the most popular question in this case why C? Well There are many reasons why C is still dominating in the field of microcontrollers programming. I'll try to list some of them right now:

- The programmers aren't in need to know every specific detail about the microprocessor itself. Which is kind of the overwhelming thing that you'll have to deal with while programming in assembly? It gets tougher when you've got to work with multiple microprocessors with different architectures.
- C provides register-level programming and manipulation facilities on a low-level. Thanks to pointers
- Developing in C produces portable code and yet highly efficient.
- C Programming allows for resources and basic memory management. [5]

Ⅰ.3.8 Embedded programming:

Embedded programming is designing software for small computers that drive devices; essentially, it is the dominant methodology for microcontroller and microcomputer programming used in small facilities-handling devices like thermostats, handheld games or other small devices. [5]

Ⅰ.4 Embedded programming and IoT:

From the engineering perspective, the Internet of Things is an embedded microprocessor controlled system connected directly or indirectly to the web. The three pillars of the IoT are therefore embedded programming, network technology and information technology. The embedded system of a device collects data from a sensor and sends it to the cloud using a wifi module — basically, it means that you can turn your **embedded** device into an **IoT** device by simply giving it Internet access.

The IoT is everywhere, and so are embedded devices:

- **Industrial world,** such as industrial machinery and control, temperature monitoring, or cognitive anomaly detection — the recent challenges of embedded systems turned them towards automation.

Fig Ⅰ.3. Industrial machine

- **Healthcare,** including blood pressure monitors,

heartbeat monitors, and pacemakers.

Fig Ⅰ.4 blood pressure monitor

- **Aerospace and Defense with such applications** as flight control systems, actuation, air and thermal management,engine power control and many others.

- **Smart Homes,** including Home Security system, Setup Box,Digital Camera, Television, Microwave Oven, Air Conditioner, Refrigerator and much more. [4]

Ⅰ.5 Embedded Systems:

Once I've read the saying that every complex system in the world can be reduced to two ideas: software and hardware. An embedded system is not an exception: to understand how embedded programming works, we need to understand its hardware and software parts.

Ⅰ.5.1 Embedded Software Design and Development:

The embedded system software is an application-specific software that is dedicated to perform predesigned specific tasks repeatedly that users don't need to intervene the operation except the simple reset operation, once the embedded software is loaded into the system, the software is expected to run for a very long time by itself without any changes to the software. It developers must guarantee the reliability, safety, and correctness of the embedded software. The other challenges for software design include constraints on the software size, time-to-prototype, time-to-market, etc. An engineering discipline is needed for the embedded software design and development

Ⅰ.5.2 Hardware overview:

Ⅰ.5.2.1 Overall Architecture:

Every embedded system consists of customer-built hardware components supported by a Central Processing Unit (CPU), which is the heart of a microprocessor (µP) or microcontroller (µC). A microprocessor is a stand-alone CPU chip, and memory and I/O ports can be custom designed and expanded. There is no on-chip memory or ports on the microprocessor. A microprocessor chip is a single integrated circuit intended to operate for general purpose and can be embedded into embedded electronics hardware. A microcontroller is an integrated chip which comes with built-in memory, I/O ports, timers, and other components. Most embedded systems are built on microcontrollers, which run faster than a custom-built system with a microprocessor, because all components are integrated within a single chip. There is a wide variety of microprocessors and microcontrollers available.

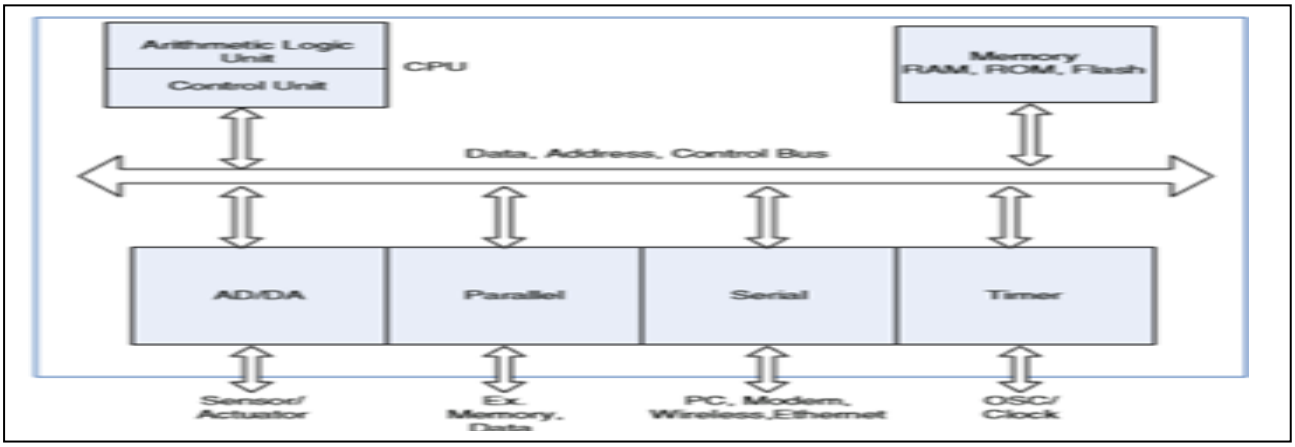


Fig Ⅰ.5. Overall architecture of embedded system

Ⅰ.5.2.2 Memory

Embedded system memory can be on-chip or off-chip. For same type memory, on- chip memory access is much fast than off-chip memory, but the size of on-chip memory is much smaller than the size of off-chip memory. Usually, it takes at least two I/O ports as external address lines plus a few control lines such as R/W and ALE control lines to enable the extended memory. For example, 16 bits of I/O port P0 and P2 are used to address 64k external memory in 8051. ($2^{16} = 2^6 \times 2^{10}$= 64k byte memory space)

- **RAM:**

The RAM is readable and writable, faster access and more expensive volatile storage, which can be used to store either data or code. Once the power is turned off, all information stored in the RAM will be lost. The 8051 memory is divided into Data Memory and Code Memory. Most of data is stored in Random Access Memory (RAM) and code is stored in Read Only Memory (ROM). This is due to the RAM constraint of the embedded system and the memory organization, The RAM chip can be SRAM (static) or DRAM (dynamic) depending on the manufacturer. SRAM is faster than DRAM, but is more expensive.

- **ROM:**

The ROM, EPROM, and Flash memory are all read-only type memories often used to store code in an embedded system. The embedded system code does not change after the code is loaded into memory. The ROM is programmed at the factory and cannot be changed over time. The newer microcontrollers come with EPROM or Flash instead of ROM. Most microcontroller development kits come with EPROM as well. EPROM and Flash memory are easier to rewrite than ROM. EPROM is an Erasable Programmable ROM in which the contents can be field programmed by a special burner and can be erased by a UV light bulb. The size of EPROM ranges up to 32kb in most embedded systems. Flash memory is an Electrically EPROM which can be programmed from soft- ware so that the developers don't need to physically remove the EPROM from the circuit to re-program it. It is much quicker and easier to re-write Flash than other types of EPROM. All ROM type memories are non-volatile. This is one of the reasons the embedded code firmware is stored in ROM type memory.

Ⅰ.5.2.3 Bus:

From the previous block diagrams, you can see all the CPU, memory, and I/O ports are connected by the buses. There are three type buses: data bus, address bus, and control bus. The bus is a pathway to collect all the microcontroller components. An 8-bit or 16-bit microcontroller has an 8-bit or 16-bit data bus to deliver data from one com- ponent to another component within the microcontroller

Ⅰ.5.2.4 I/O Ports:

The I/O ports are used to connect input and output devices. The common input devices for an embedded system include keypads, switches, buttons, knobs, and all kinds of sensors (light, temperature, pressure, etc). The output devices include Light Emitting Diodes (LED), Liquid Crystal Dis- plays (LCD), printers, alarms, actuators, etc. Some devices support both input and output, such as communication interfaces including Network Interface Cards (NIC), modems, and mobile phones

- **Parallel Port:**

A microcontroller is integrated with several parallel I/O ports, which are used to interface the microcontroller to outside devices such as switches, LCDs, keypads, and actuators. The parallel ports get or send 8 bits of data at a time to and from connected outside devices.

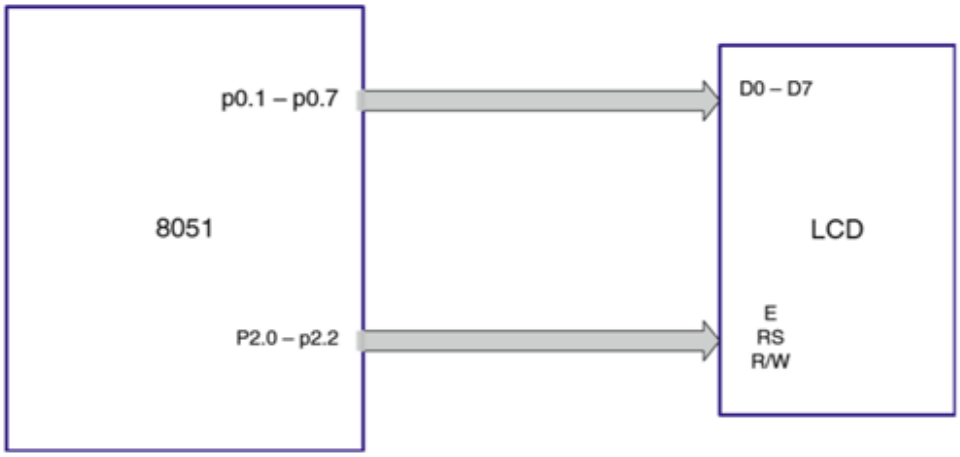


Fig Ⅰ.6.Parallel integration between MCU & LCD

- **Serial Ports:**

Some I/O ports such as the port D of the 68HC11 and port 3 of the 8051 can be used to connect an external serial device. Instead of 8 parallel lines in parallel I/O communication, the serial communication exchanges data one bit at a time with a single data line plus reference lines if needed. The typical application of serial ports is to connect a PC with a serial cable to take advantage of the PC for cross- platform embedded software debugging, testing, and deployment. Serial ports also allow data to flow between microcontrollers or between the microcontroller and any serial devices[4]

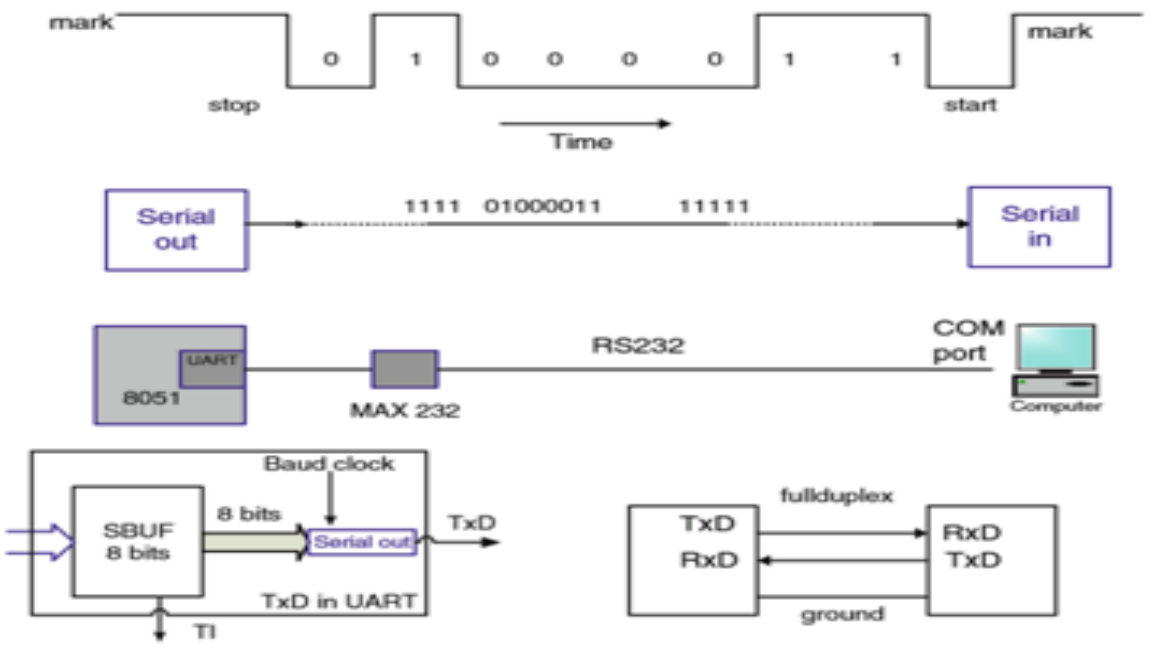


Fig Ⅰ. 7. serial communication between MCU & serial devices

Ⅰ.6. Conclusion:

In this chapter we've talked about embedded systems, some of it history and a wide description of them their applications and importance, and then we moved to the latest technologies in this field, and finally explained the embedded programming.

Chapter Ⅱ :

Wearable Technology & SmartWatches

Ⅱ.1. Introduction to wearables:

The past year has certainly been an exciting time for the semiconductor industry. The shared enthusiasm among silicon companies has been on the rise based on the recent level of interest and demand for new products that complement the mobile experience typically based around smartphones and tablets. [5]

A main driver behind this phenomenon is wearable technology. Familiar items such as clothing, glasses, jewelry, and watches are being fitted with sensors, processors, and displays technologies you wouldn't have expected to find inside these everyday objects before. The potential for this market is quite impressive; ABI Research projects that 32 million sports and fitness wristbands will have shipped in the past twelve months alone.

Wearables will mostly dominate in the fitness market. There has been a monumental rise in the world of fitness. Devices that track your progress will improve and become even more important and popular. You will be able to input your biological data and the wearable will create bespoke programs to improve your fitness.

Healthcare is another sector that looks set to be improved by new wearable technology. Wearables will be able to monitor all different kinds of health care issues. Health care companies are investing huge amounts of cash into these technologies so look out for huge developments over the next five years.

Wearable technology will allow us to become more interested in entertainment forms such as television and video games. Imagine being able to feel and smell things as the film progresses. Wearing technology that can allow us to physically take part in this fun and interacting activities could revolutionize the entertainment industry[11]

Ⅱ.2. History of wearable:

Fig Ⅱ.1.history if wearable technology

Wearable devices such as smart watches and fitbits have become the norm in recent years. As we become more aware of our health we are more inclined to monitor ourselves much closer than we ever did before. The younger people of the world have taken to wearable devices and the trend looks set to continue as technology improves. Let's take a quick look at the history of wearables and see what the future may hold.

If we want to get all technical about it, wearables were actually first invented in the 13th century when eyeglasses became available. 300 years later we had the first wearable clocks which led to the invention of the wristwatch. There have even been discoveries in China of abacus rings.

Moving on another 300 years, we saw the rise of the first wearable computers, calculators, and music devices. Listening to music while you walk has only been around for the last 30 years! The hearing aid hit stores during the 1980s creating the first bionic humans.

In 2002, bluetooth technology allowed wireless interactions with our devices and the market has grown massively since. Smartwatches and fitness devices know dominate the market. There are also other products out that can track patient seizures and exposure to sunlight. The wearable's industry looks set to expand.

We can expect wearables to be second only to smartphones in the consumer electronics industry over the next five years. Autonomous devices look set to really catch fire as processing power looks to improve vastly. App add-ons will make wearables more bespoke to users, much like mobile phones. [11]

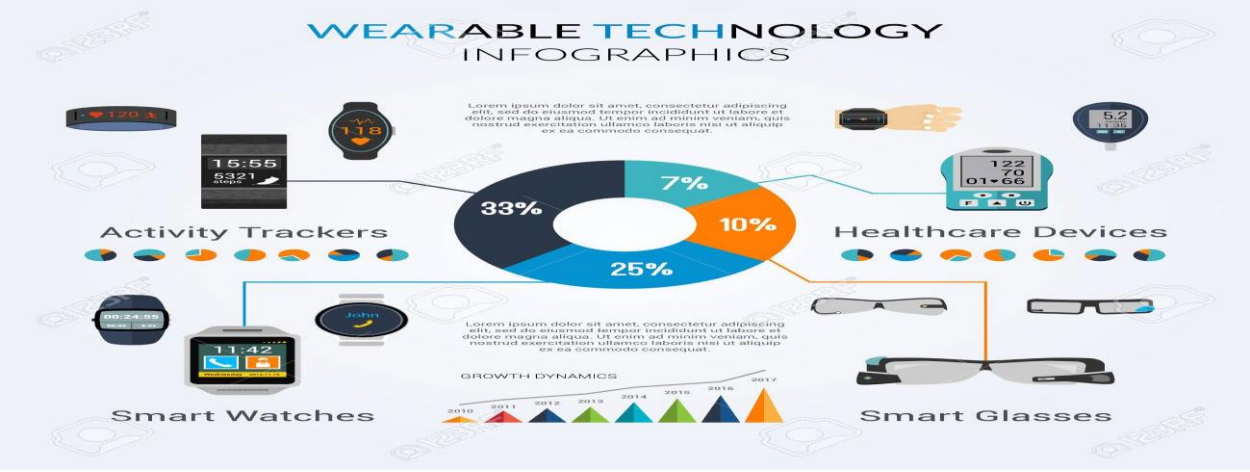
Ⅱ.3. Overview of wearables:

Wearable Embedded systems are cyber-physical systems that are carried from persons in their daily activities and can assist in quality of life through augmented sensing for the disabled, independent living for the elderly, and reduced healthcare costs. The implementation of wearable embedded systems is normally based on commercial available chipsets while recently efforts have focused on the development of e-textiles that will effectively disappear these systems in the person's clothes. The challenges faced with this type of embedded systems includes strict requirements for the size and energy resources together with severe constraints in computational power, while the highly variable environment due to person's movements and the body itself results to a complicated transmission channel for wireless connectivity affecting the bandwidth and the quality of communication. [12]

Ⅱ.3.1 Definition of wearables:

Wearables are electronic technology or devices incorporated into items that can be comfortably worn on a body. These wearable devices are used for tracking information on real time basis. They have motion sensors that take the snapshot of our day to day activity and sync them with mobile devices or laptop computers. After the invention of smartphones, wearable electronics are the next big innovation in the world of technology.

Even before the wearable technology entered the consumer market, these wearable devices were used in the field of military technology. In fact, these devices were an integral part of the medical and healthcare sector in the military forces. Devices like 'Wearable Motherboards' or 'Smart Shirts' used to monitor the health and wellbeing of the patients and send back information to the hub station in real time[12]



Ⅱ.3.2 Different types of Wearables devices:

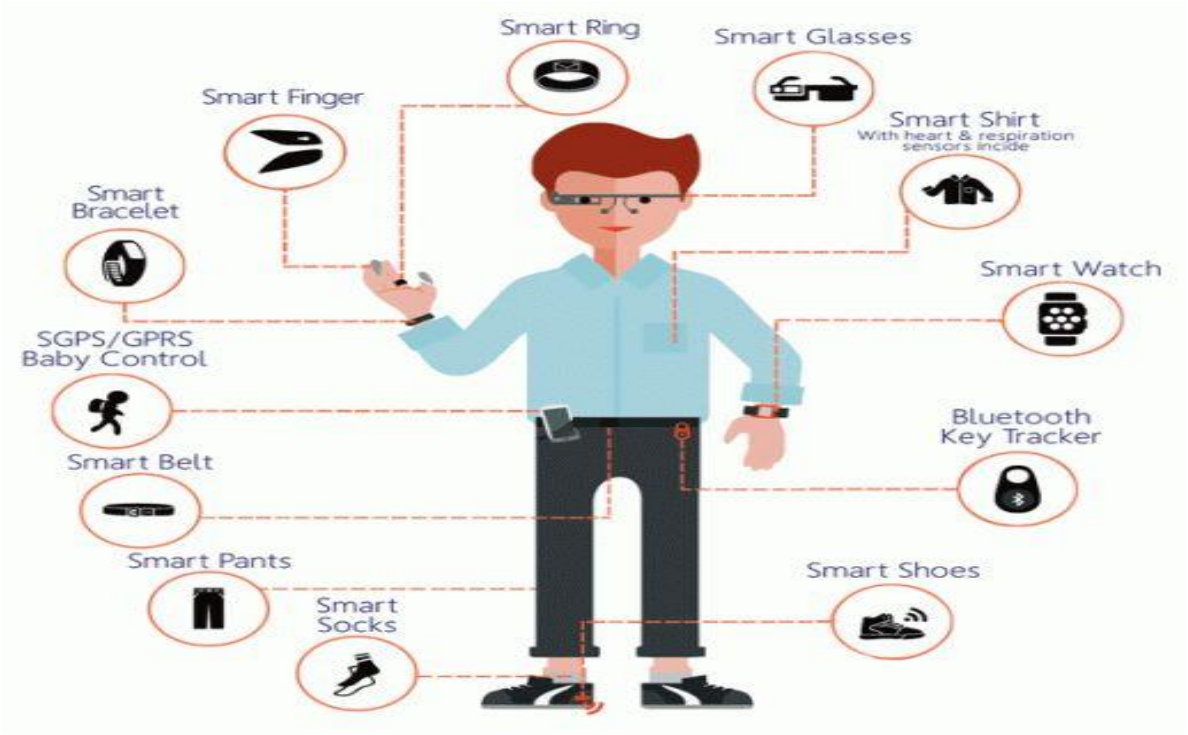


Fig Ⅱ.2. Types of wearables

• **Smart Watches:** A watch that does more than just telling time. It provides users notifications on their calls, messages, emails, social media updates, etc.

• **Fitness Tracker:** Helps keep a track of the number of steps the user walks each day and continuously monitors the heart rate. Using this information, the devices is able to calculate and report accurate data on calorie burn and exercise done by the user.

• **Head Mounted Display:** Takes you to a different world of virtual reality. It provides virtual information directly to your eyes.

• **Sports watches:** The wearable devices is especially built for sports personnel who love running, cycling, swimming etc. These devices come with GPS tracker and records information on the user's pace, heart rate etc.

• **Smart jewellery:** Smartwatches are designed as jewelries specially targeting women. These jewelries notify the users of their text messages, calls or emails when their phone is out of reach.

• **Smart Clothing:** The smart electronic devices are incorporated into the Wearable Clothing to give an interesting and fashionable look.

• **Implantable:** These wearable electronics are surgically implanted under the skin. These are usually used for medical reasons like tracking contraception's, insulin levels etc. [13]

Ⅱ.3.3 Importance of wearable technology:

The Wearable technology aims to influence the fields of health and medicine, fitness, aging, disability, education, transportation, enterprise, finance, gaming, music, etc. The goal is to smoothly enter the daily lives of individuals and become a functional part of them.

The hands-free nature of the wearable computing devices makes it very useful for businesses. Tracking the emergency and rescue team becomes easy thus making the workplace more efficient and safe. Hands-free access to important data and information through smart glasses and smart watches helps researcher, engineers, and technicians to be more efficient at their work. [13]

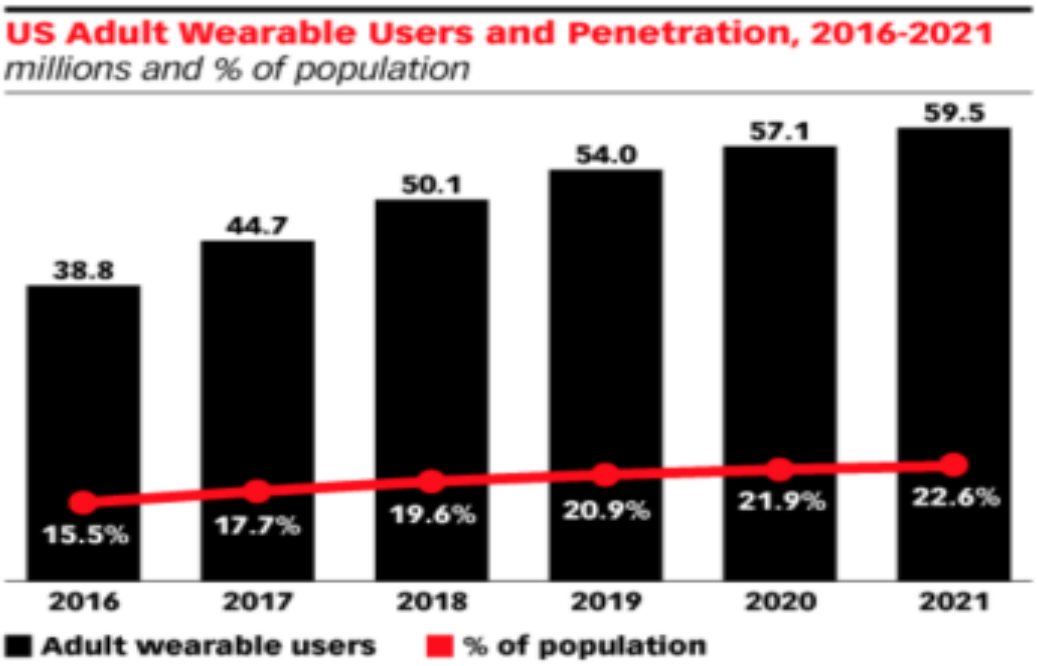


Fig Ⅱ.3. the increase of the use of wearables

Ⅱ.3.4 Key Challenges of Wearables:

The biggest challenge for the wearable industry is to get a sustainable customer engagement. Many wearable electronics are short lived because of it:

- Short term customer engagement
- Bad quality
- Pain to sync with smartphones
- Poor battery life
- Uncomfortable and awful design
- UX problems

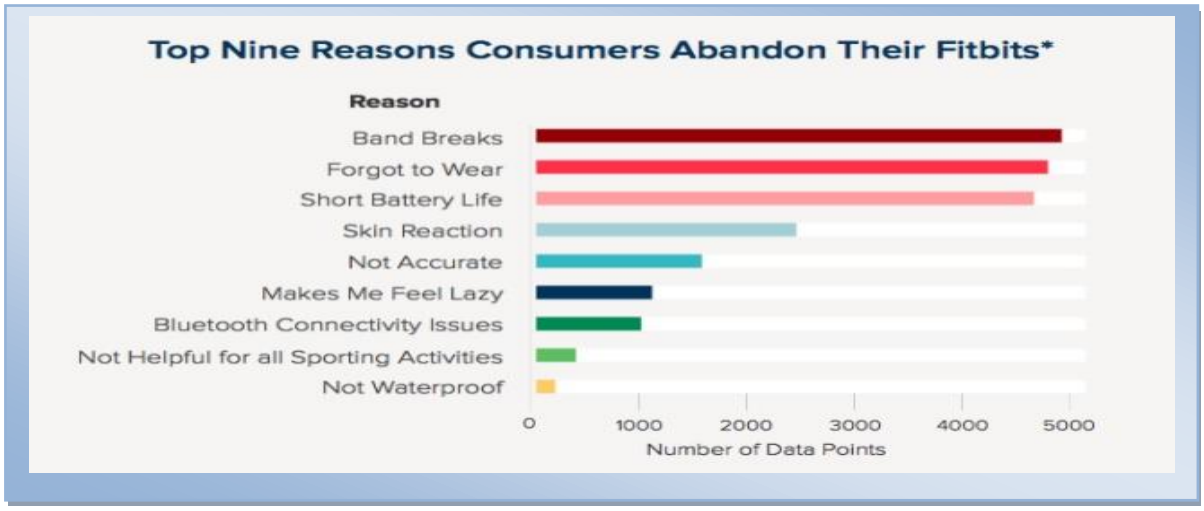


Fig Ⅱ.4. challenges that facing wearables

These are some of the functional reasons which put the user off the device. However, we might find wearable devices that are very strong functionally and physically and still bomb at the market simply because they failed to create any meaningful impact on the users, their live habits or behaviors

With the wider use of wearable technology, enterprises see wearable computing as one of the biggest opportunity to drive efficiency and improve communication and workflow. Like mobile technology, wearable technology is looked upon as a disruption in the world of business, however, the growth and popularity of a technology always comes with concerns on the risk that it brings on data security and privacy. According to a report by ***Pricewaterhouse*** Cooper, 86% of users expresses concerns on the use of wearables and the risk of data security

breach. Companies should have very strong policies and procedures in place before introducing wearable technology for the company usage. [13]

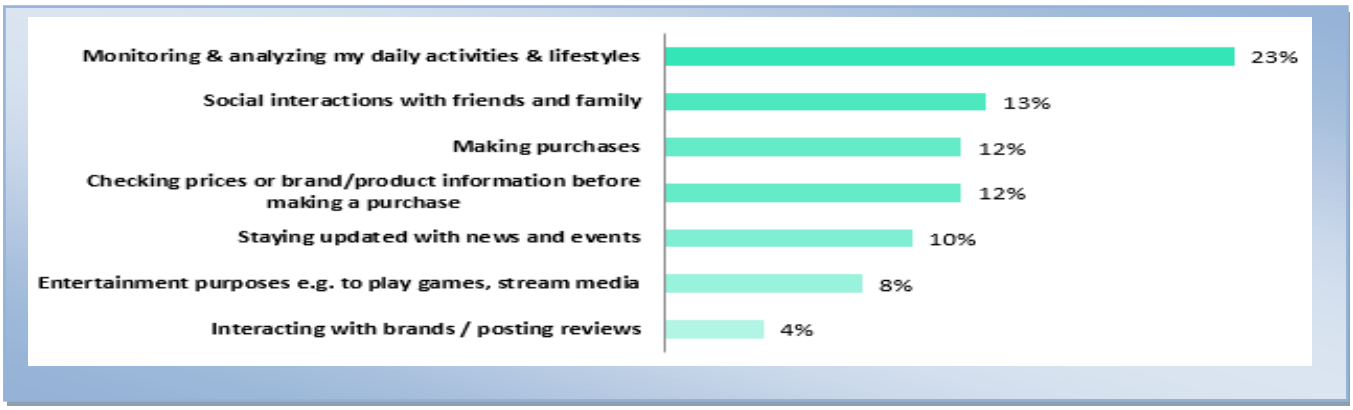


Fig Ⅱ.5. wearable technology trends

Ⅱ.3.5 Requirement of wearables:

Ⅱ.3.5.1. Aesthetics: The wearable device needs to be stylish and blend with existing fashion accessories such as ornaments, watches, and glasses. Aesthetics are so important that top semiconductor companies such as Intel are partnering with the fashion industry to make these devices fashionable.

Fig Ⅱ.6.fashionable wearables

Capacitive touch sensing is a key technology that helps to improve the aesthetics. Capacitive UI need to work on a variety of form factors including curved surfaces, be tolerant of liquids, and be able to sense under thick overlays. Cypress' CapSense and TrueTouch technologies make such requirements realizable.

Ⅱ.3.5.2. Size:

As we saw earlier, these devices must be small so that they can easily fit on to a wearable. Nevertheless, at the same time they must integrate more features in the same space.

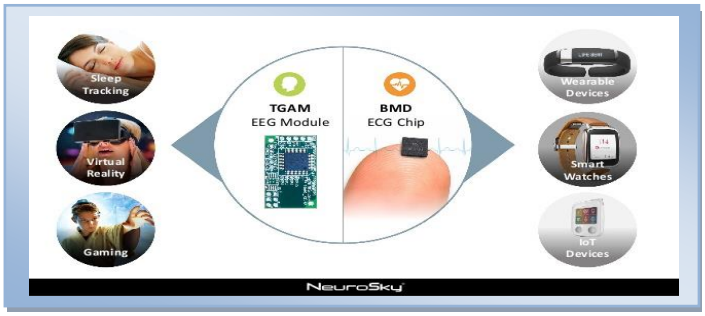


Fig Ⅱ.7.sizes of wearables

Technologies such as System-on-Chip (SoC) and chip scale packages (CSP) help to shrink the size. For example, Cypress offers PSoC (Programmable System-on-Chip) devices in multiple packaging options including WLCSP.

Ⅱ.3.5.3. Water tolerance:

Wearable devices are going to be everywhere the human body can go. Therefore it is important to design these devices to be tolerant of the environmental conditions such as water droplets, moisture, and sweat.

Fig Ⅱ.8.waterproof wearables

Ⅱ.3.5.4. Power consumption:

Since wearable devices are battery powered, reducing the power consumption of these devices poses unique challenges. Wearable devices, unlike other mobile devices, are required to be always on and always connected because most of these are monitoring devices. For

example, a smart watch needs to always show the time and be connected to a mobile phone through a wireless link such as Bluetooth in order to receive alerts; a pedometer is required to continuously count the steps and report it to a mobile phone app; and similarly a heart rate monitor needs to be always monitoring and reporting. Yet battery capacity is inherently limited due to the requirement to reduce the overall size

Therefore these devices need to operate at ultra-low power to conserve the battery life. This requirement drives special needs in MCU and firmware algorithms. 32-bit ARM architecture is a popular CPU technology for wearable devices as it provides best performance and energy efficiency.

Ⅱ.3.5.5. Wireless communication:

Wireless connectivity is important for wearable devices

as they need to interact with one or more other devices.

Depending on the type and features offered,

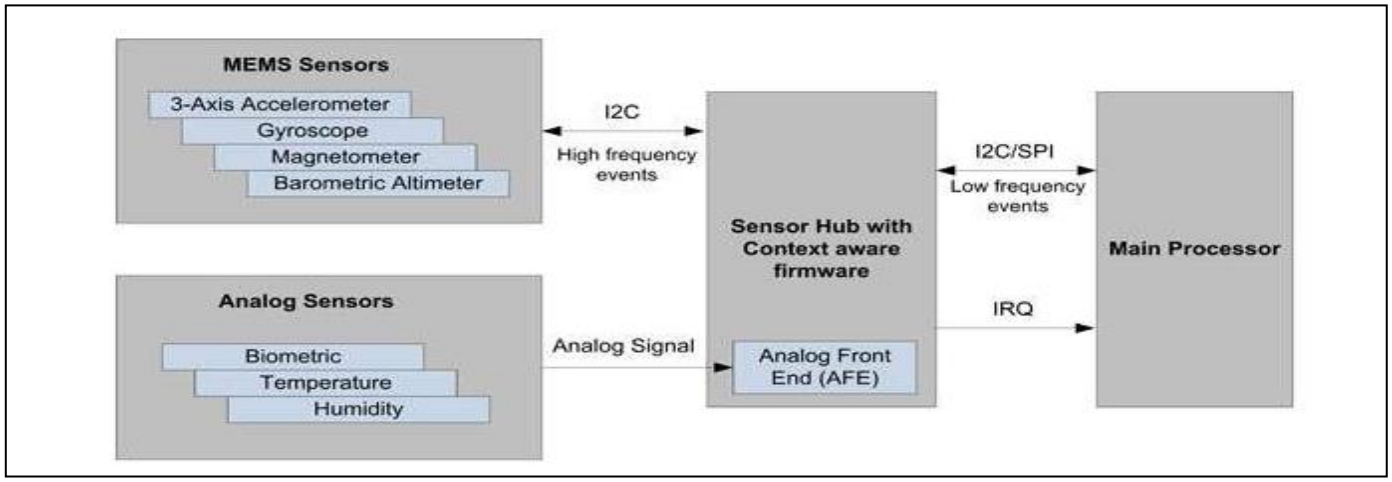
Fig Ⅱ.9.wireless communication

The device is required to support different wireless protocols such as Wi-Fi, ANT+, Bluetooth Low Energy (BLE), and IEEE 802.15.4 based proprietary protocol. Some devices are required to support multiple protocols. For example, a wrist watch communicates with a heart rate monitoring chest strap using a proprietary wireless protocol and also it communicates to a running application in a mobile phone using BLE.

Ⅱ.3.5.6. Selecting the right App or microprocessor:

Selection of the main processor is driven by the type and features of the device. Except for advanced infotainment devices which requires an application processor, MCUs can address

most of the types of wearable devices. Also the latest MCUs integrate most of the functions in a single chip. This is important in reducing the overall size of a wearable device and BOM cost. For example, an ARM cortex-M controller can power a simple wrist band but a smart watch requires an application processor in order to run a complex operating system such as Android. **Fig Ⅱ.10. the role of a sensor hub in a wearable system.**



need a specific operating system. For example, a simple wrist watch that monitors temperature, measures movement using a 3-axis accelerometer, and displays time on a monochromatic segment LCD display can run with a lightweight RTOS, whereas a smart watch that is designed to be an extension of your mobile phone needs to run an advanced operating system such as Android. At the same time, sensor hubs require special firmware[13]

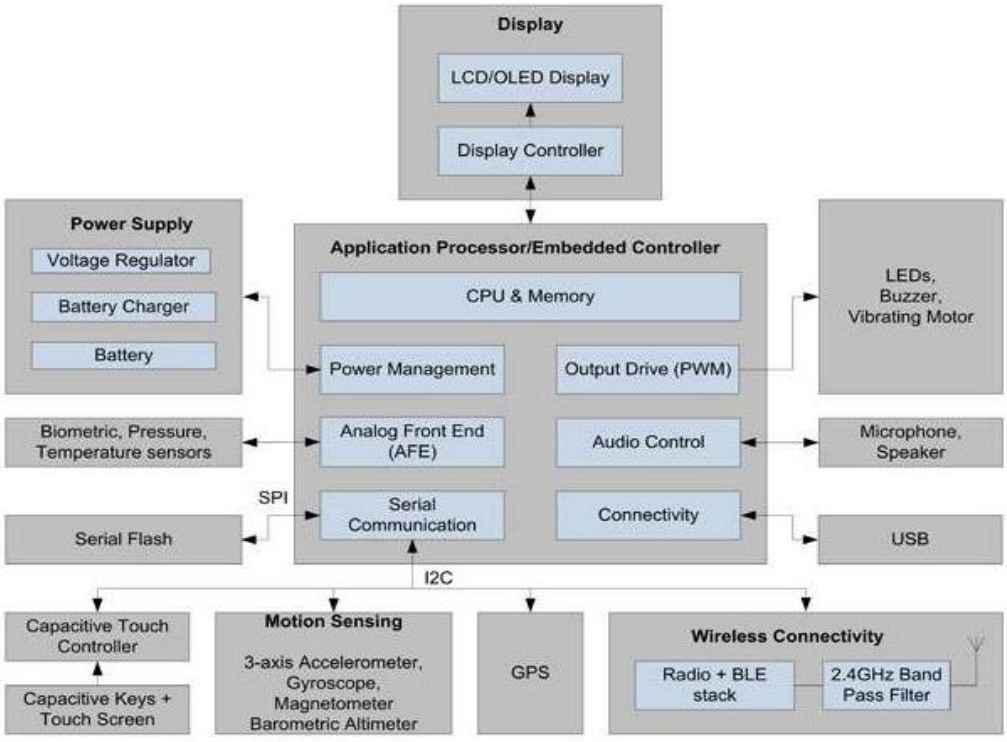


Fig Ⅱ.11. block diagram shows the components in a wearable electronic device

Ⅱ.4. Introduction to Smart Watches:

Technology nowadays is moving along at a furious pace, developers compete with each other by implementing more and more innovative solutions which make our life easier. Recently the new trend namely wearable devices is catching on, they are not as popular as smart phones yet but the statistics shows that the industry is growing very fast especially the industry of smart watches

There's no doubt that with the popularity of smartphones that can manage virtually every aspect of our lives, the trend in technology is to get more and more "connectivity" into smaller and smaller packages.

Simultaneously, wrist watches have become a lesson in technological redundancy for many people. Ask a friend for the time of day and they're just as likely to glance at their smartphone as they are to look at an actual wrist-bound timepiece. The newest wave of smart watches aims to change all that

Smart watches can run apps and play back all sorts of digital media, like audio tracks or radio streamed to Bluetooth headphones. Many of these watches have touch screens, which allow us to access functions like a calculator, thermometer, compass and more.

Fig Ⅱ.12. Smartwatch connected to smartphone

Just as with our smartphone, Internet access enables a smart watch with whole world of potential capabilities, like message notifications, GPS navigation and calendar synchronization. And of course, a Bluetooth connection to our phone means the watch can help us place calls or send and receive messages. [14]

Ⅱ.6. History of smart watches:

To many people, these newfangled watches might look like a brand new technology. The truth, however, is that smart watches have been lurking on the fringes of gadgetry for a long time. The very first may have been Microsoft's UC-2000, a digital watch released in 1984 that could be programmed in BASIC via its keypad. In 2002, Microsoft introduced a technology called **Smart Personal Object Technology**, or **SPOT**, designed to add new purpose to everyday objects by integrating so-called smart software. Watches built around SPOT were discontinued in 2008, but the idea of the smart watch lived on.

Fig Ⅱ.13. History of smart watches

-In fact, the first smart watch dates back to **1927**, when the first Wristlet Route Indicator was launched. There was no GPS built into that model, but a scroll map cartridge allowed users to set their route in some European countries

-The first electric digital watch arrived in **1972** boasting LED, the first model was wrapped up in gold. Users had to push a button to see the time on this magic device, conceived and fabricated by **the *Hamilton Watch Company***. Ten years later, in **1982**, ***Seiko*** presented its own first TV-watch model – Seiko TV Watch, which needed an adapter, a large receiver box, in order to show TV images below the digital time display. The display was also only in black and white. The next year, Seiko launched the Data-2000, a watch with calendar entries and a calculator. This model was followed by a few other Seiko devices. Much later, in **1995**, ***Seiko*** conceived its Seiko **MessageWatch** which could display updates on a variety of subjects ranging from sports scores, weather forecasts, and stock prices, all in black and white.

Breitling Emergency Watch was developed in **1995**. It could send a distress signal that could be picked up from anywhere within 90 nautical miles. In **1998**, the first Linux-powered

watch was built. It was designed to communicate wirelessly with personal computers, cell phones, and other wireless-enabled devices. **Fossil's Palm Pilot** appeared in **2002**. It featured a small display, 2 MB of internal memory, memo pad, an address book, a to-do list, and a calculator.

More recently in **2010**, ***Nike*** Fuel band, ***Sony*** Smartwatch, ***Pebble***, ***Samsung*** **Galaxy Gear**, ***Samsung*** **Gear Fit**, Android Wear, ***Moto*** 360, and ***Samsung*** Gear S made their entry. Thus, ***Apple*** Watch is far from being the first smartwatch in the world. [35]

Ⅱ.7. Definition of smart watches:

A smartwatch is a digital watch that provides many other features besides timekeeping, for example: - Examples include monitoring heart rate

- tracking our activities
- providing reminders throughout the day

Like a Smartphone, a smartwatch has a touchscreen display, which allows us to perform actions by tapping or swiping on the screen. Modern Smartwatches include several apps, similar to apps

Fig Ⅱ.14. smartwatch functionalities

for smartphones and tablets. These apps provide additional functionality, such as displaying weather information, listing stock prices, and displaying maps and directions. Most

smartwatches can also be used to make phone calls and send and receive text messages

. **Fig Ⅱ.15. smartwatches additional functionalities**

While these apps run directly on the smartwatch, they require a smartphone to function. This is because the data is first received by the phone, then sent to watch. Most smartwatches do not include Wi-Fi and they do not have a SIM card for cellular data. Therefore, most apps rely on a compatible smartphone to provide data over a Bluetooth connection. For example, the text messaging app on our smartwatch may allow us to dictate and send a text message, but the actual message is sent using our phone. If our watch is not within range of our phone's Bluetooth signal, the message will not be sent.

Since smartwatches rely on smartphones for a large percentage of their functionality, they are generally considered a smartphone accessory rather than a **standalone device.** [30]

Ⅱ.8. Functions of smart watches:

Most smartwatches, whether they're intended for daily use (***Apple Watch***) or for specific purposes (***the Garmin Fenix***) offer a suite of standard features:

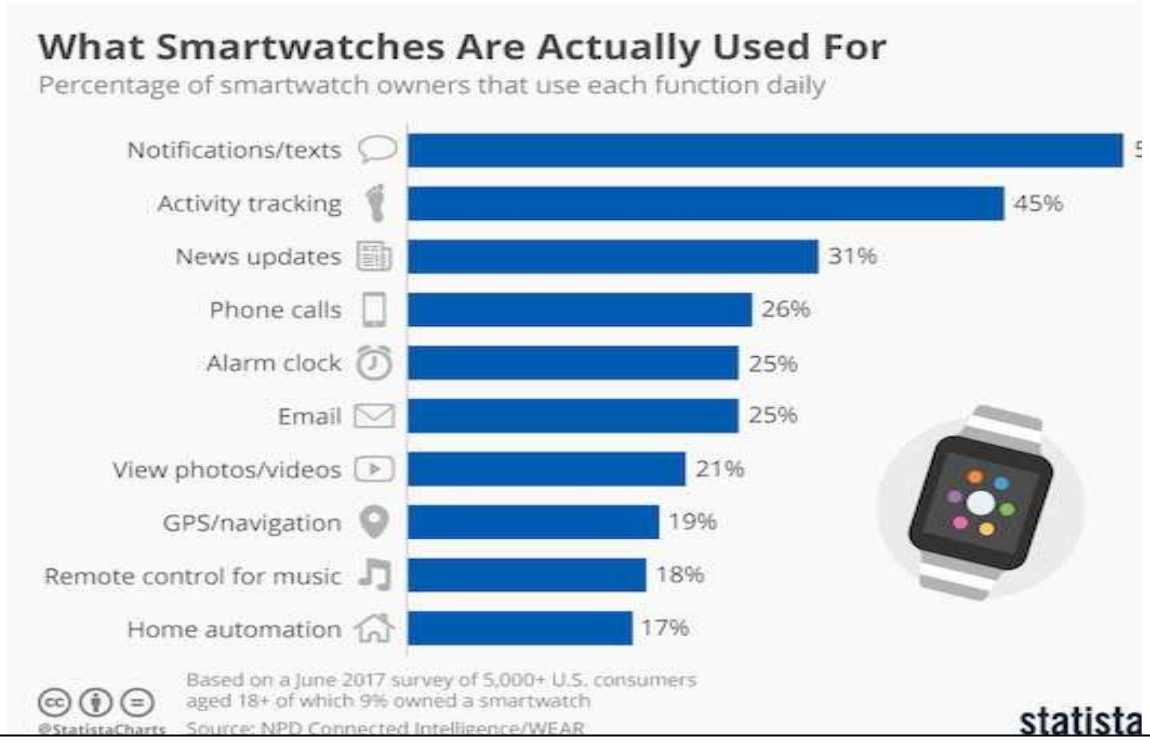


Fig Ⅱ.16. SmartWatches uses

- **Notifications**: Smartphones display notifications to alert us of important events or activities. The types of notifications differ; devices connected to a smartphone may simply mirror the phone's notifications on our wrist, but other smartwatches display notifications that only a wearable could provide. For example, the newest Apple Watch includes a fall sensor. If we fall while wearing the watch, the watch senses our subsequent movement if it doesn't detect any, it'll send a series of escalating notifications. Fail to respond to the notification, and the watch will assume we're injured and alert authorities on our behalf.
- **Apps**: Beyond displaying notifications from our phone, a smartwatch is only as good as the apps it supports. App ecosystems vary, and they're tied to either Apple's or Google's environments. Smartwatches with a dedicated purpose, such as for hiking or diving, generally support the apps they need to accomplish that purpose without the opportunity to add other kinds of apps.
- **Media management**: Most smartwatches paired with smartphones can manage media playback for us. For example, when we're listening to music on an iPhone using Apple's AirPods, we can use our Apple Watch to change volume and tracks.
- **Answer messages by voice**: Modern smartwatches running either the watch OS or Wear OS operating systems support voice dictation.
- **Fitness tracking**: If we're a hard-core athlete, a dedicated fitness band is likely a better choice than a smartwatch. Still, many smartwatches include a heart rate monitor and a pedometer to help track our workouts.
- **GPS**: Most smartwatches include a **GPS** for tracking our location or receiving location-specific alerts.
- **Good battery life**: Modern smartwatches feature batteries that will get us through the day, with normal use, with a bit of juice still left to go. Battery use varies; the Apple Watch typically gets 18 hours of normal use on a single charge, while the Pebble gets two or three days

Ⅱ.9. Advantages and drawbacks of smart watch:

Smartwatches aren't going anywhere anytime soon, so it's important to learn how to adapt and develop better habits when wearing one. we should aim to reap all the advantages of a smartwatch while minimizing the downsides of wearing one.

Ⅱ.9.1. Advantages:

- Encourage a healthy lifestyle:

We can often track calories burned, steps walked,

how long we've spent exercising, and how often

we stand on a given day. All of these are effective at motivating us into staying active and keeping our numbers up.

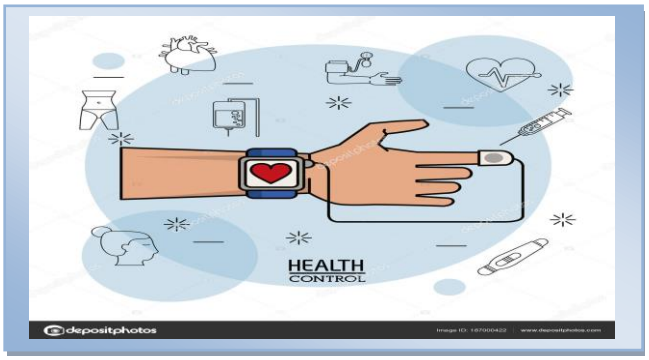


Fig Ⅱ.17. Smartwatch for healthy life

Many smartwatches also include notifications and alerts reminding us to stay active. It's one thing to *know* staying fit is important, but a dedicated device strapped to our wrist that keeps us in check adds another level of accountability.

- Connectivity to other smart devices:

Fig Ⅱ.18.Connectivity of smartwatch

If we have an iPhone, Airpods, and any smart home gadgets, we can pair our watch with them as a controller and receive notifications. Say we're listening to music on our iPhone in another

room. If we want to play the next song or change the volume, we can use our Apple watch to do that. We can also respond to text messages, view push notifications, and even play music without the use of our phone.

A smartwatch is a smart device, which means having the ability to connect to other intelligent devices is a must.

- Greater access to notification:

Just about any notification, we would get on our phone can get sent to our smartwatch. It isn't always convenient

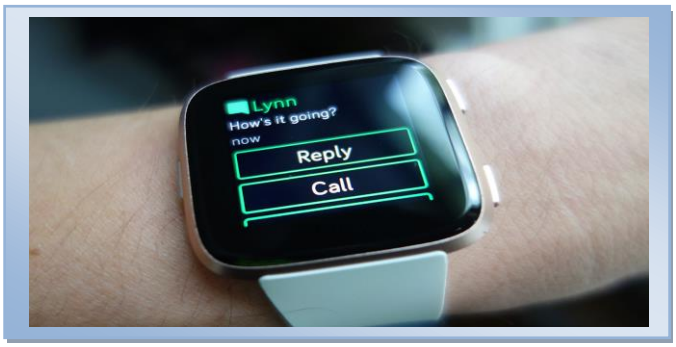


Fig Ⅱ.19.notification access by smartwatch

to carry our phone around with us, but a smartwatch will inevitably follow us because it's strapped to our wrist.

If we're busy and can't check our phone, we can take a quick look at our smartwatch instead to see what's going on. Even better, we can often respond to these notifications directly on our smartwatch.

Ⅱ.9.2. Drawbacks:

- Fitness data isn't perfect:

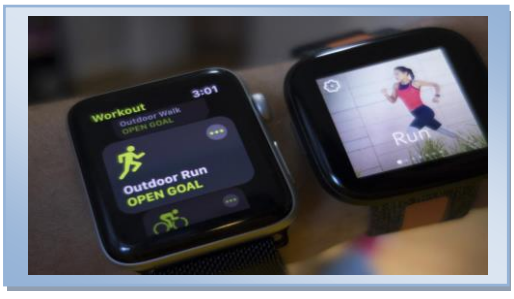


Fig Ⅱ.20. SmartWatch in fitness life

While smartwatches help motivate us to be healthy, the data they collect is not always perfectly accurate. Something to keep in mind is that these devices are not individually calibrated for everyone wearing them. Instead, they're designed to provide data on a widespread, one-size-fits-all basis. Because of this, a smartwatch can't provide us with accurate, individualized data that fully reflects our situation. This doesn't mean that the number of calories we burn is completely wrong, but there's certainly room for mistakes

Another figure that tends to get improperly reported is heart rate. Smartwatches are meant to provide our heart rate at an exact moment, but oftentimes there is a delay and it isn't always correct..

- Smartwatches leads to distracted driving:

Smartphones have been the recent culprit for distracted driving, but the rising trend of smartwatches is making the chance of distraction greater. We can put our phone in our glove box or the backseat, but that won't do you much good if your wrist is buzzing and lighting up with a notification. It isn't common for smartwatch wearers to take it off before they start driving. While it *is* easy to do, the action doesn't come naturally. [34]

Ⅱ.10. Popular applications used for smartwatches:

Ⅱ.10.1. Applications to set up a smartwatch:

Before getting started with the setup process, there are some important considerations we need to put in place in order to achieve a successful setup procedure. This will ensure that there isn't unnecessary problems while setting up the Smartwatch of our choice.

First of all the battery of both smartphone and smartwatch should be full before starting our set up process. Also we need to make sure that Bluetooth is turned ON in the phone

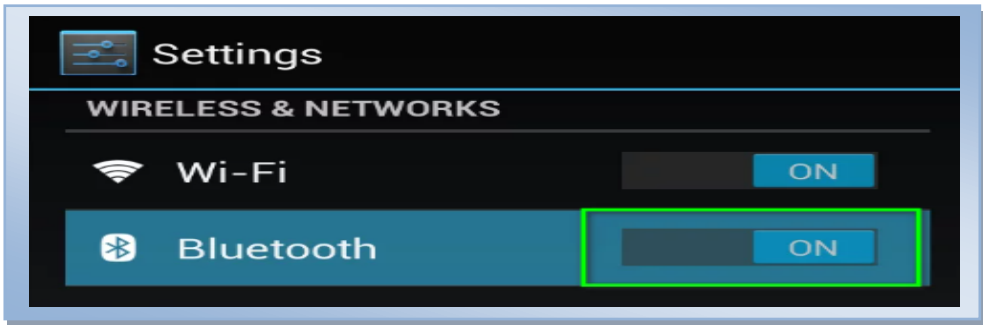


Fig Ⅱ.21. Activating Bluetooth to connect with smartwatch

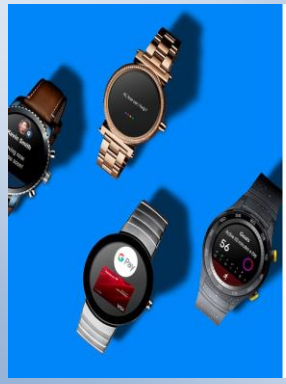
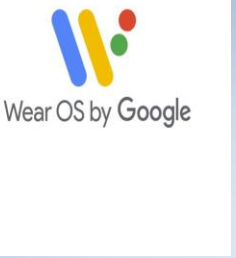
Android smartwatch	Apple watch
Android wear OS	Don’t need a specific application it just need the latest version of iOS
 	

Table Ⅱ.1.comparison between Android and Apple watch App’s

Ⅱ.10.2. Best wearable application to download on a smartwatch:

Some of the smartwatch manufacturers include out of the box functionality with the default apps. But still, there are a few functionalities that can be add by installing third-party best wearable apps that claim to make our day far managed and controlled.

a. CALM:

- Lets the users experience a serene sense of meditation

with the help of relevant sound and pictures

Fig Ⅱ.22.Calm application

- Best for reducing stress and anxiety
- Brings sound sleep
- Integrated with sleep stories and relaxing music gallery for sound sleep
- Incorporated with breathing exercises
- Helps track our daily progress

b. PARKING:

- Capable of quickly and automatically determining when and where we parked our vehicle
- Automatic parking detection with the help of car's Bluetooth
- Multiple navigations are available for car parking
- Excellent smartwatch support
- Comes with beautiful home screen widget

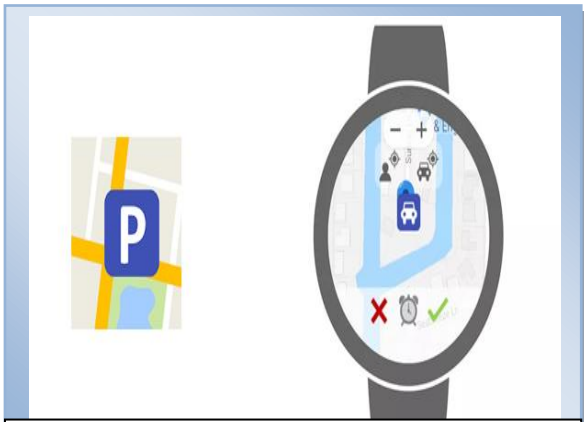


Fig Ⅱ.23.Parking application

c. ACCUWEATHER:

- Gives us access to severe weather alerts/warnings

to keep us safe from natural disasters like strong hurricane

- Render today's detailed temperature information
- Using the "RealFeel Temperature" feature tell us about

the facts that how the weather looks and how it feels

- Facility to have Live weather updates (24/7) for every minute information

Fig Ⅱ.24.Accuweather application

d. BRING:

Fig Ⅱ.25. Bring application

- Permits to collect the loyalty card details to eliminate the need of a separate app
- Provides the option to share the created shopping list with other family members and friends
- Incorporated with easy to understand icons that help prepare grocery list easily
- Users can also have access to add items by using voice transcription functionality of their smartwatches

e. CityMapper:

- Offers detailed information about arrival and departure

timing for public transit options

- The app allows to create routes for mixed transportation types

. For example, a cab ride to a subway station

- Provides Uber details that further saves time and money
- Check all the nearby departure in real-time
- Provides step by step direction to our rout

Fig Ⅱ.26.CityMapper application

Apart from these the above-mentioned wearable apps, there are many other apps like Runkeeper, Duolingo, Recordr, Spotify, Google Maps, and Calculator that can be easily install and use efficiently from our digital wristwatch.

Ⅱ.11. Manufacturers of smartwatches:

Samsung, Sony, and LG were the first to introduce smartwatches in the global market. Later, Apple took its first step in the smartwatch industry with iWatch and unsurprisingly became the leading producer of disruptive smart watches. Fitbit, Fossil, and Guess opted to enter the Android Wear project as well. Consequently, the following are the top 10 brands that are dominating the global smartwatch market, as per the market research experts at Technavio.

A. Apple:

Coming on the heels of the successful Apple Watch launch in 2015 and with regular upgrades, it is clear that Apple is on top of its game in a way like none other. The secret lies in their product line and design standards

Pro	Con
○ Display is always on ○ Very large on-board storage ○ WatchOS 6 has great features	○ Short battery life ○ No sleep tracking ○ Pricey

Table Ⅱ.2. Advantages and drawbacks of Apple watch

B. Samsung group:

Samsung has not only been a popular name in the production of mobile phones and similar technologies, but it has a diverse business portfolio that includes advanced semiconductors, petrochemicals, plant construction, skyscrapers, alongside fashion, finance and medicine. The company is a leading global brand in electronic manufacturing and digital media. Samsung offers a variety of products as smart wearable devices under its Gear portfolio which includes the Gear, Gear 2 Neo, Gear 2, Gear Fit, and Gear S range of smart watches.

Pro	Con
○ Great battery life ○ Useful rotating bezel	○ Expensive ○ Annoying charger ○ Poor voice assistant technology.

Table Ⅱ.3. Advantages and drawbacks of Samsung Watch

C. Lenovo group:

Lenovo group, has expanded its business globally , Lenovo's smart wearable product featured as Moto 360 Sport offers a slew of specifications such as Wi-Fi connectivity, Android Wear OS, which includes heart rate sensors, and dual digital mics.

D. Garmin:

Along with its subsidiaries, Garmin designs, develops, manufactures, and markets handhelds; portable, wrist-based, and fixed-mount GPS-enabled products alongside navigation and communication products globally

E. Fitbit:

Fitbit was founded in 2007 to deliver exclusive personal wireless communication solutions that redesign the way we stay connected to the world. It manufactures and sells activity trackers for fitness as well as develops sensors and wireless technology for the health and fitness sector.

Pro	Con
○ Battery life is great ○ Always-On AMOLED display ○ Great sleep tracking features	○ No GPS ○ No Google Fit or Apple Health integration

Table Ⅱ.4. Advantages and drawbacks of the Fitbit Watch

F. LG Electronics:

LG together with Google announced their first Android Wear-based smartwatch in June 2014 imaginatively named the LG G Watch. It followed that up with a pair of slick looking Google-powered smartwatches

G. Huwaei technologies:

Huawei has officially entered the sphere of smartwatches with the announcement of Huawei Watch. The affordable luxury watch is not only smart, but the frame's timeless design can be tailored with 40 face options

H.Fossil Group:

Since introducing its first smartwatch in the fall of 2015, Fossil Group- an American fashion brand, has launched more than 150 touch screen smart watches, hybrid smart watches, activity trackers in 20 languages throughout 40 countries for brands including Diesel, Chap, Emporio Armani, Kate Spade New York, Michael Kors, Skagen, and Misfit

Pro	Con
○ Sleek minimalistic design ○ Pretty fast ○ Decent battery life	○ Poor speaker quality

Table Ⅱ.5. Advantages and drawbacks of Fossil smartwatch

Ⅱ.12. Smartwatch forcasts and statistics :

Although still in the early stages of diffusion, smartwatches represent the most popular type of wearable devices. Yet, little is known about why some people are more likely to adopt smartwatches than others. To deepen the understanding of underlying factors prompting adoption behavior, the authors develop a theoretical model grounded in technology acceptance and social psychology literatures. Empirical results reveal perceived usefulness and visibility as important factors that drive adoption intention, suggesting that smartwatches represent a type of 'fashnology' (i.e., fashion and technology). The magnitude of these antecedents is influenced by an individual's perception of viewing smartwatches as a technology and/or as a fashion accessory. Theoretical and managerial implications are discussed.

Smartwatches can be considered as a disruptive innovation, which imply a radical change in consumer behavior in order to adopt it. Thus, being this one of the reasons why its adoption rate is still at very low levels.

About a half of those who don't own a smartwatch would like to have one (53%), of which 46% would prefer an Apple. Also, perceptions regarding design are quite positive, contrary to expected, with 87% of respondents who agreed to generally like smartwatches design. About 60% of respondents are happy with the current features offered by smartwatch manufacturers and in most of the cases they would use it for fitness/wellness, daily-life or business.

In general expectation for the future of smartwatches are positive, with 49% thinking they will be the future (vs. 19% who don't agree, and 32% who don't know), and 39% assuming they will replace regular wristwatches eventually (vs. 26% who don't agree, and 35% who don't know). [36]

Ⅱ.13. Conclusion:

In this chapter we explained wearable technology, starting from it meaning moving to wearable devices popularity and their benefits in human life, then in the second par of this chapter we focused on smartwatches and their functionality, and we mentioned some of applications and famous Ⅱ brands to make it easy in case of choosing the perfect smartwatch, and also we showed some statistics that concerns the future of smartwatches.

Chapter Ⅲ : ARDUINO, Programming & Application

Ⅲ.1. Introduction to Arduino:

Arduino is the world's leading open-source hardware and software ecosystem. The Company offers a range of software tools, hardware platforms and documentation enabling almost anybody to be creative with technology.

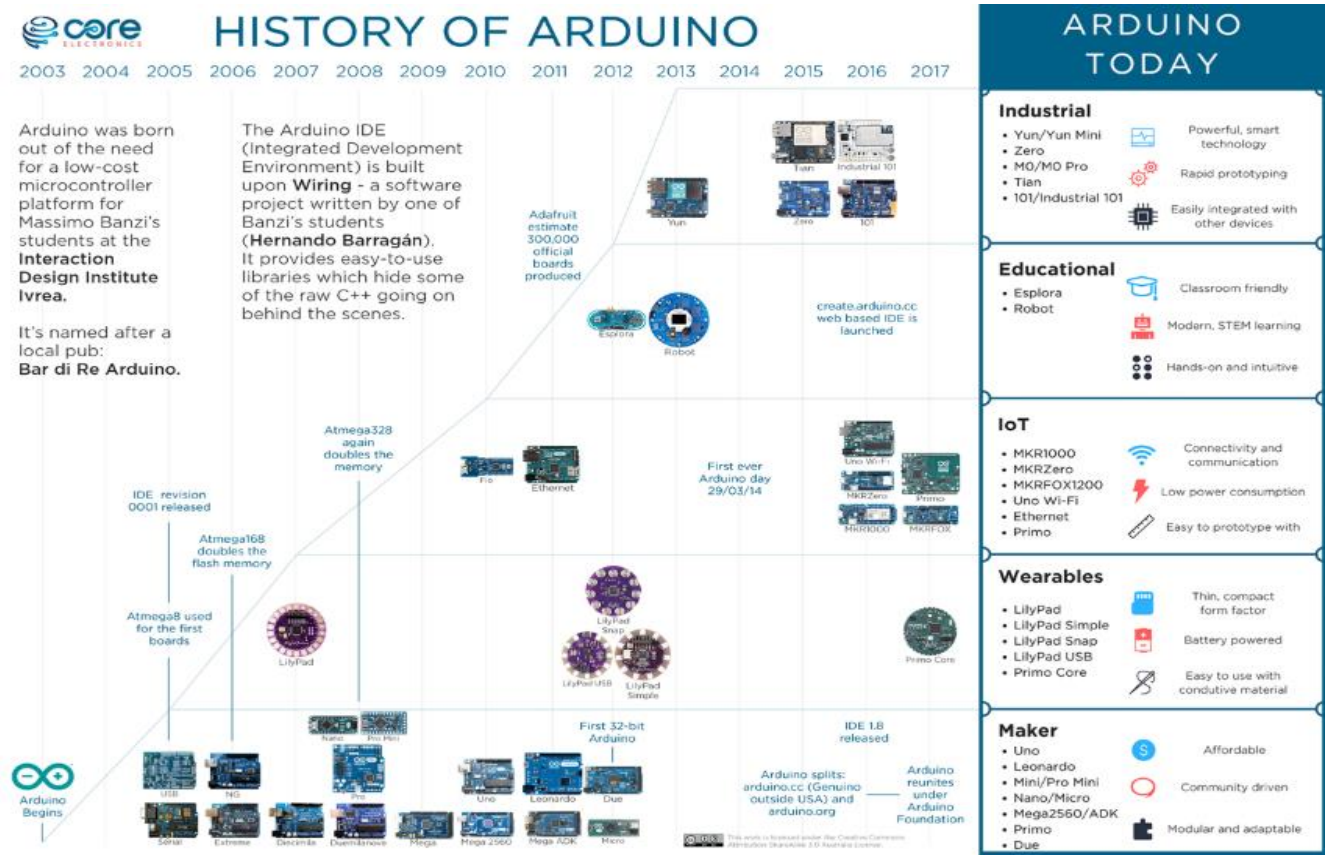
It is an open-source microcontroller development board. In plain English, we can use the Arduino to read sensors and control things like motors and lights. This allows us to upload programs to this board which can then interact with things in the real world. With this, we can make devices which respond and react to the world at large.

Basically, if there is something that is in any way controlled by electricity, the Arduino can interface with it in some manner. And even if it is not controlled by electricity, we can probably still use things which are (like motors and electromagnets), to interface with it. [17]

Ⅲ.2. History of Arduino:

Originally started as a research project by Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, and David Mellis at the Interaction Design Institute of Ivrea in the early 2000s, it builds upon the Processing project, a language for learning how to code within the context of the visual arts developed by Casey Reas and Ben Fry as well as a thesis project by Hernando Barragan about the Wiring board.

The name Arduino originates from the name of a bar it is also the name of an Italian king, historical figure of the city "Arduin d'Ivrée", or an Italian male first name which means "strong friend"[16]



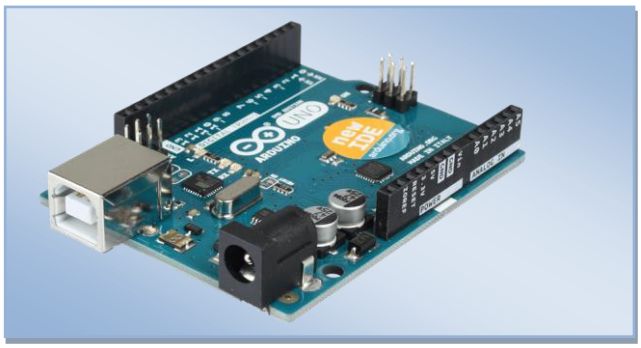
FigⅢ.1. Timeline of developpement of the ARDUINO

Ⅲ.3. Definition of Arduino:

Basically, it is a small development board with a brain

(also known as a microcontroller) that can be connected

to electrical circuits. This makes it easy to read inputs



figⅢ.2.Arduino board

read data from the outside – and control outputs, send a command to the outside. The brain of this board (Arduino Uno) is an ATmega328p chip where we can store our programs that will tell the Arduino what to do.

Ⅲ.4. Different types of Arduino:

There are a number of different types of Arduino to choose from. This is a brief overview of some of the more common types of Arduino boards:

Ⅲ.4.1.Arduino Uno

The most common version of Arduino is the Arduino Uno. This board is what most people are talking about when they refer to an Arduino. **Arduino NG, Diecimila, and the Duemilanove (Legacy Versions)**

Legacy versions of the Arduino Uno product line

consist of the NG, Diecimila, and the Duemilanove.

The important thing to note about legacy boards is that they lack particular feature of the Arduino Uno. Some key differences:

FigⅢ.3.Arduino UNO board

- The Diecimila and NG use an ATMEGA168 chips (as opposed to the more powerful ATMEGA328),
- Both the Diecimila and NG have a jumper next to the USB port and require manual selection of either USB or battery power.
- The Arduino NG requires that you hold the rest button on the board for a few seconds prior to uploading a program.

Ⅲ.4.2.Arduino Mega 2560

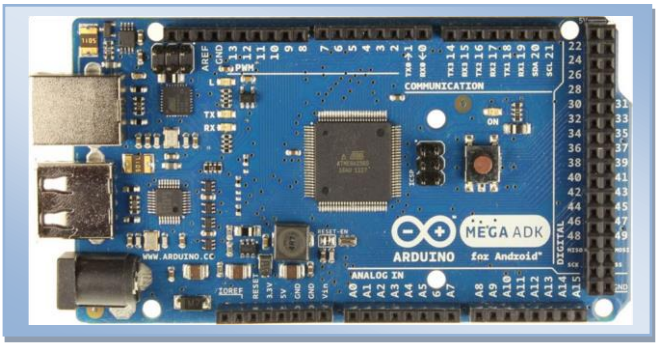
The Arduino Mega 2560 is the second most commonly encountered version of the Arduino family The Arduino Mega is like the Arduino Uno's beefier older 0 brother. It boasts 256 KB of memory (8 times more than the Uno).

FigⅢ.4.Arduino Mrga2560 board

It also had 54 input and output pins, 16 of which are analog pins, and 14 of which can do PWM. However, all of the added functionality comes at the cost of a slightly larger circuit board. It may make your project more powerful, but it will also make your project larger. Check out the official Arduino Mega 2560 page for more details.

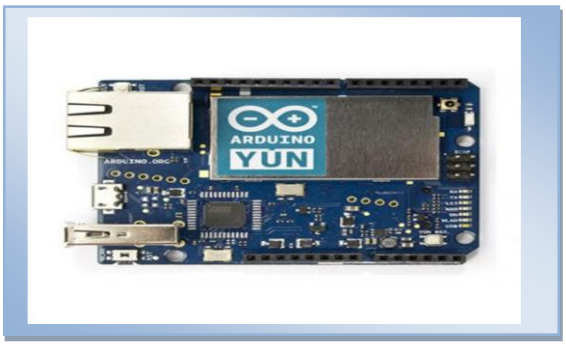
Ⅲ.4.3.Arduino Mega ADK

This specialized version of the Arduino is basically an Arduino Mega that has been specifically designed for interfacing with Android smartphones. This too is now a legacy version.



Ⅲ.4.4.Arduino Yun :

The Arduino Yun uses an ATMega32U4 chip instead of the ATmega328. However, what really sets it apartis the addition of the Atheros AR9331 microprocessor.

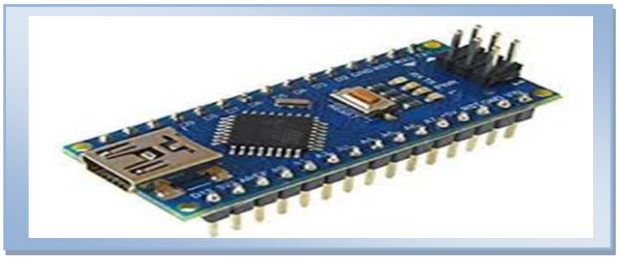


figⅢ.6.Arduino Yun board

This extra chip allows this board to run Linux in addition to the normal Arduino operating system. If all of that were not enough, it also has onboard wifi capability. In other words, we can program the board to do stuff like we would with any other Arduino, but we can also access the Linux side of the board to connect to the internet via wifi. The Arduino-side and Linux-side can then easily communicate back and forth with each other. This makes this board extremely powerful and versatile.

Ⅲ.4.5. Arduino Nano:

If we want to go smaller than the standard Arduino board, the Arduino Nano is the ideal Based on a surface mount

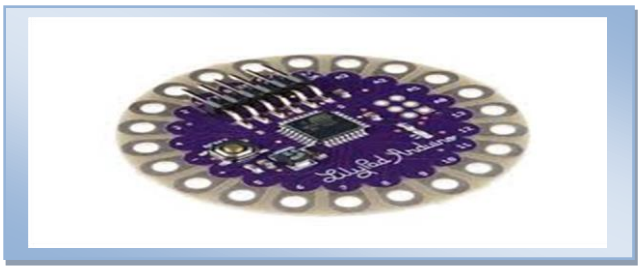


FigⅢ.7.Arduino Nano board

ATmega328 chip, this version of the Arduino has been shrunk down to a small footprint capable of fitting into tight spaces. It can also be inserted directly into a breadboard, making it easy to prototype with.

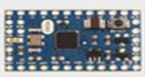
Ⅲ.4.6.Arduino Lilypad:

The Lilypad was designed for wearable and e-textile applications. It is intended to be sewn to fabric and connected to other suable components using conductive thread.



FigⅢ.8.Arduino LilyPad board

This board requires the use of a special FTDI-USB TTL serial programming cable. [19]

							
Fabricante	Arduino	Arduino	Arduino	Arduino	Arduino	Arduino	Arduino
Modelo	Pro Mini	Nano	Uno	Mega / Mega 2560	Leonardo	Micro	Due
Microcontrolador	AVR Atmega 168 ó 328 8bits	AVR ATmega 168 ó 328 8bits	AVR ATmega 328 8bits	AVR ATmega2560 8bits	AVR ATmega 32u4 8bits	AVR ATmega 32u4 8bits	ARM SAM3X8E Cortex-M3 32bits
Frecuencia	16Mhz	16Mhz	16Mhz	16Mhz	16Mhz	16Mhz	84Mhz
Memoria RAM	2KiB	2KiB	2KiB	8KiB	2.5KiB	2.5KiB	96KiB (64+32KiB)
Memoria EEPROM	1KiB	1KiB	1KiB	4KiB	1KiB	1KiB	0
Memoria FLASH	16 ó 32KiB	16 ó 32KiB	32KiB	128 ó 256KiB	32KiB	32KiB	512KiB
Pines digitales entradas/salidas	14/14	14/14	14/14	54/54	20/20	20/20	54/54
Tensión/corriente pines digitales	3.3v ó 5v 40mA	5v 40mA	5v 40mA	5v 40mA	5v 40mA	5v 40mA	3.3v 3~15mA (130mA entre todos)
Pines analógicos entradas/salidas	6/0	8/0	6/0	16/0	12/0	12/0	12/2
Tensión/resolución pines analógicos	3.3v ó 5v 10bits (1024 valores)	5v 10bits (1024 valores)	5v 10bits (1024 valores)	5v 10bits (1024 valores)	5v 10bits (1024 valores)	5v 10bits (1024 valores)	3.3v 12bits (4096 valores)

TableⅢ.1. Characteristics of some Arduino boards

Ⅲ.5. Benefits of using an Arduino board:

- huge documentation and support
- large library collection
- simplified and user-friendly programming language
- low power consumption
- highly customizable
- format of the text SMS is simple to understand and write

- can be used for real time applications
- very easy to extend it and has tons of user contributed shields and libraries, Shields are available to do pretty much anything

Like any electronic component there is advantages but also there is some drawbacks.

Ⅲ.5.1.Drawbacks of Arduino:

- lack of speed control makes the robot unstable at times
- it require power supply
- choice of lines is made in the hardware abstraction and cannot be changed by software[16]

Ⅲ.6. Applications of Arduino:

When it comes to the applications of Arduino they are limitless, but here is the common apps:[21]

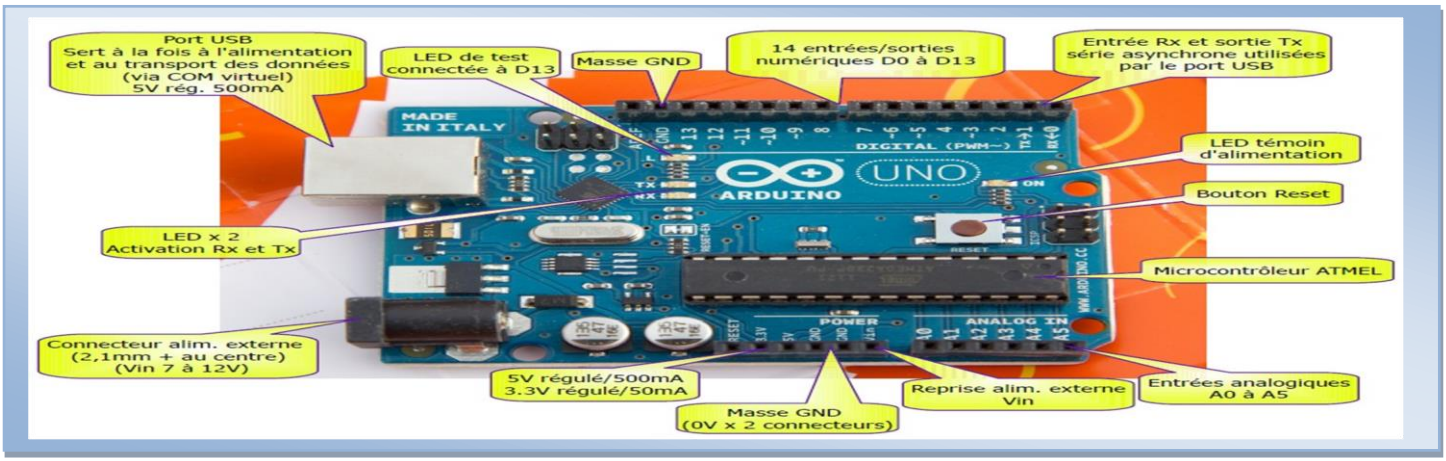
- Alarm system
- Smartwatches
- Automation system developpement
- Home automation
- Robotics
- Sensor prototyping
- Smart glasses

Ⅲ.7. Technical specification:

In order to set up our project which is a smartwatch based on Arduino NANO board we used two boards , we first started testing our programs and circuit using the Arduino UNO board

Then after checking on all the components and the march of our smartwatch we swipe up the UNO board by the NANO board due to it small size.

Ⅲ.7.1. Exploring the Arduino UNO board:



figⅢ.9.Arduino UNO board demonstration

- **Digital pins:** Arduino has 14 digital pins, labeled from 0 to 13 that can act as inputs or outputs.
  - ✓ When set as inputs, these pins can read voltage. They can only read two states: HIGH or LOW.
  - ✓ When set as outputs, these pins can apply voltage. They can only apply 5V (HIGH) or 0V (LOW).

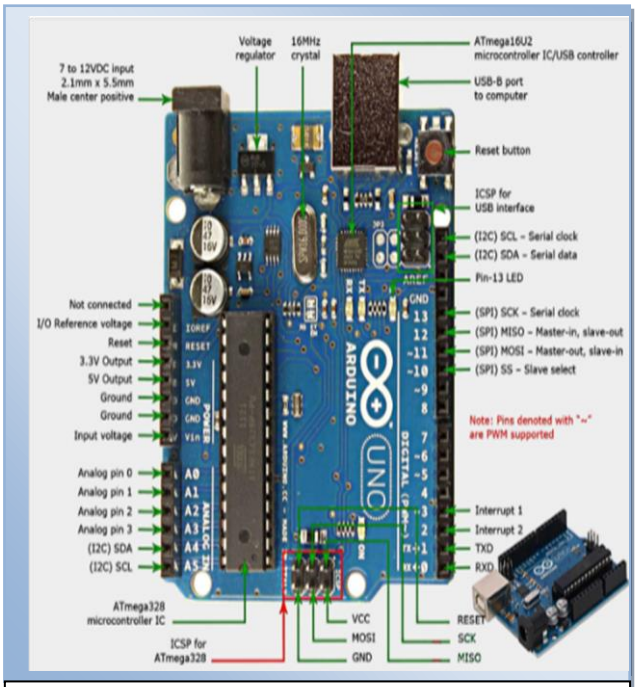


Fig Ⅲ.10. Arduino UNO board pins

- **PWM pins:** These are digital pins marked with a (pins 11, 10, 9, 6, 5 and 3). PWM stands for "pulse width modulation" and allows the digital pins output "fake" varying amounts of voltage.

- **TX and RX pins**: digital pins 0 and 1. The T stands for "transmit" and the R for "receive". The Arduino uses these pins to communicate with other electronics via serial .Arduino also uses these pins to communicate with our computer when uploading new code.
- **LED attached to digital pin 13:** This is useful for an easy debugging of the Arduino sketches.
- **TX and RX LEDs**: these leds blink when there are information being sent between the computer and the Arduino.
- **Analog pins**: the analog pins are labeled from A0 to A5 and are often used to read analog sensors. They can read different amounts of voltage between 0 and 5V. Additionally, they can also be used as digital output/input pins like the digital pins.
- **Power pins**: the Arduino provides 3.3V or 5V through these pins. This is really useful since most components require 3.3V or 5V to operate. The pins labelled as "GND" are the ground pins.
- **Reset button:** when we press that button the program that is currently being run in our Arduino restarts. There is also a Reset pin next to the power pins that acts as reset button. When we apply a small voltage to that pin, it will reset the Arduino.

Microcontroller	ATmega328
Operating Voltage	5V
Input voltage (recommended)	7-12 V
Input voltage (limits)	6-20V
DIGITAL I / O Pins	14 (6 of which provide the PWM output)
Analog input pins	¬6
DC Current per I / O Pin	40 Ma
DC Current for 3.3 Pin	50 Ma
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

TableⅢ.2.characteristics of Arduino UNO board

- **Power ON LED:** will be on since power is applied to the Arduino.
- **USB jack:** we need a male USB A to male USB B cable to upload programs from the computer to the Arduino board. This cable also powers our Arduino**.**

- **Power jack:** we can power the Arduino through the power jack. The recommended input voltage is 7V to 12V. There are several ways to power up the Arduino:
  >rechargeable batteries
  >disposable batteries
  > wall-warts and solar panel[26]

Name	Arduino Nano	Arduino Uno
MCU	Atmega328p/Atmega 168.	Atmega328p
Power	5V	5V
Input Voltage	7 -12 V	7 – 12 V
Maximum Current Rating	40mA	40mA
Clock Frequency	16MHz	16MHz
Flash Memory	16KB/32KB	32KB
USB	Mini	Standard
USART	Yes	Yes
SRAM	1KB/2KB	2KB
PWM	6 out of 14 digital pins	6 out of 14 digital pins
GPIO	14	14
Analog Pins	8	6
EEPROM	512bytes/1KB	1KB

Difference between Arduino Uno and Arduino Nano

www.TheEngineeringProjects.com

Table Ⅲ.3. Difference between Arduino UNO and NANO boards

Ⅲ.7.2.Exploring the Arduino Nano board:

- It comes with exactly the same functionality as in Arduino UNO but quite in small size.
- It comes with an operating voltage of 5V however the input voltage can vary from 7 to 12V.
- Arduino Nano Pinout contains 14 digital pins 8 analog Pins, 2 Reset Pins & 6 Power Pins.

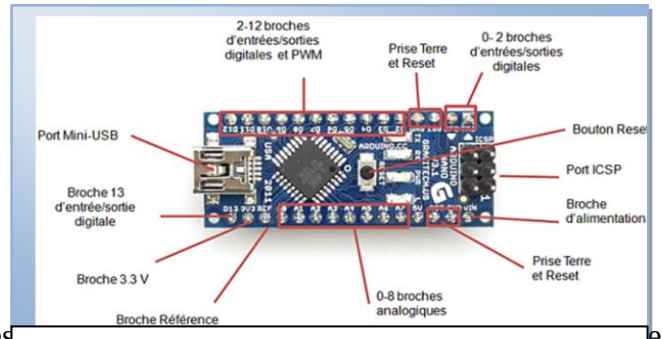


Fig Ⅲ.11.Arduino Nano board demonstration

- Each of thes[illegible]e functions but their main function is t[illegible]
- They are acted as input pins when they are interfaced with sensors, but if you are driving some load then use them as output.

- Functions like pinMode() and digitalWrite() are used to control the operations of digital pins while analogRead() is used to control analog pins.
- The analog pins come with a total resolution of 10bits which measure the value from zero to 5V.
- Arduino Nano comes with a crystal oscillator of frequency 16 MHz. It is used to produce a clock of precise frequency using constant voltage.
- There is one limitation using Arduino Nano i.e. it doesn't come with DC power jack, means you cannot supply external power source through a battery.
- This board doesn't use standard USB for connection with a computer, instead, it comes with Mini USB support.
- Tiny size and breadboard friendly nature make this device an ideal choice for most of the applications where a size of the electronic components are of great concern.
- Flash memory is 16KB or 32KB that all depends on the Atmega board i.e Atmega168 comes with 16KB of flash memory while Atmega328 comes with a flash memory of 32KB. Flash memory is used for storing code. The 2KB of memory out of total flash memory is used for a boot loader. [20]

Ⅲ.7.3. Control an Output and Read an Input:

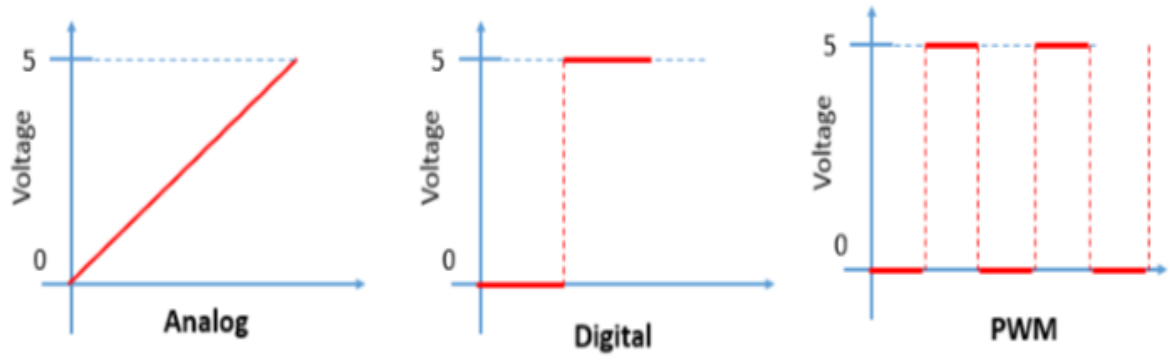
An Arduino board contains digital pins, analog pins and PWM pins.

Difference between digital, analog and PWM

- In **digital pins**, we have just two possible states, which are on or off. These can also be referred as High or Low, 1 or 0 and 5V or 0V.

For example, if a LED is on, then, its state is High or 1 or 5V. If it is off, the it's Low, or 0 or 0V.

- In **analog pins**, we have unlimited possible states between 0 and 1023. This allows us to read sensor values. For example, with a light sensor, if it is very dark, we'll read 1023, if it is very bright we'll read 0 If there is a brightness between dark and very bright we'll read a value between 0 and 1023.
- **PWM** pins are digital pins, so they output either 0 or 5V. However these pins can output "fake" intermediate voltage values between 0 and 5V, because they can perform "Pulse Width Modulation" (PWM). PWM allows to "simulate" varying levels of power by oscillating the output voltage of the Arduino



FigⅢ.12. Different voltages in the three modes

- **Controlling an output**
  - To control a digital output we use the digitalWrite() function and between brackets we write, the pin we want to control, and then HIGH or LOW.
  - To control a PWM pin we use the analogWrite() function and between brackets we write the pin we want to control and a number between 0 and 255.
- **Reading an input**
  - To read an analog input you use the function analogRead() and for a digital input we use digitalRead(). [23]

Ⅲ.8. Programming:

Ⅲ.8.1. Downloading the Arduino IDE:

The Arduino IDE (Integrated Development Environment) is where we develop our programs that will tell the Arduino what to do. We can load new programs onto the main chip, the ATmega328p, via USB using the Arduino IDE, then Select which Operating System we're using and download it. Then, simply follow the installation wizard to install the Arduino IDE. When we first open the Arduino IDE, we should see something similar to the figure below: [38]

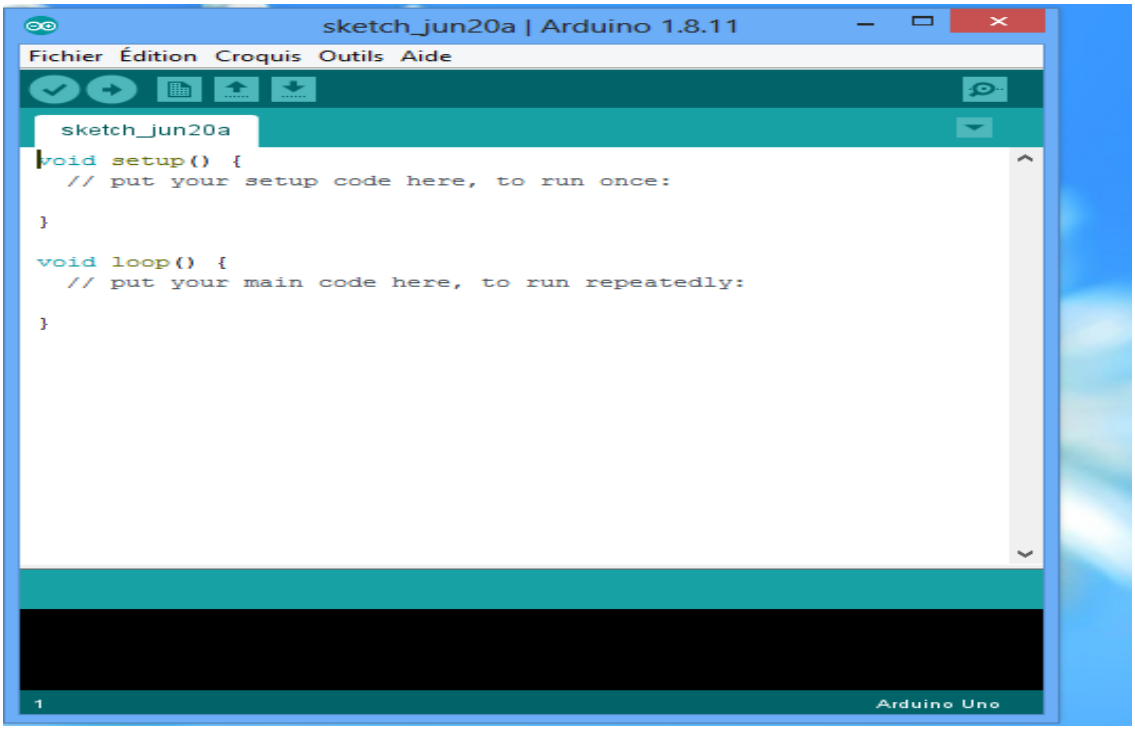
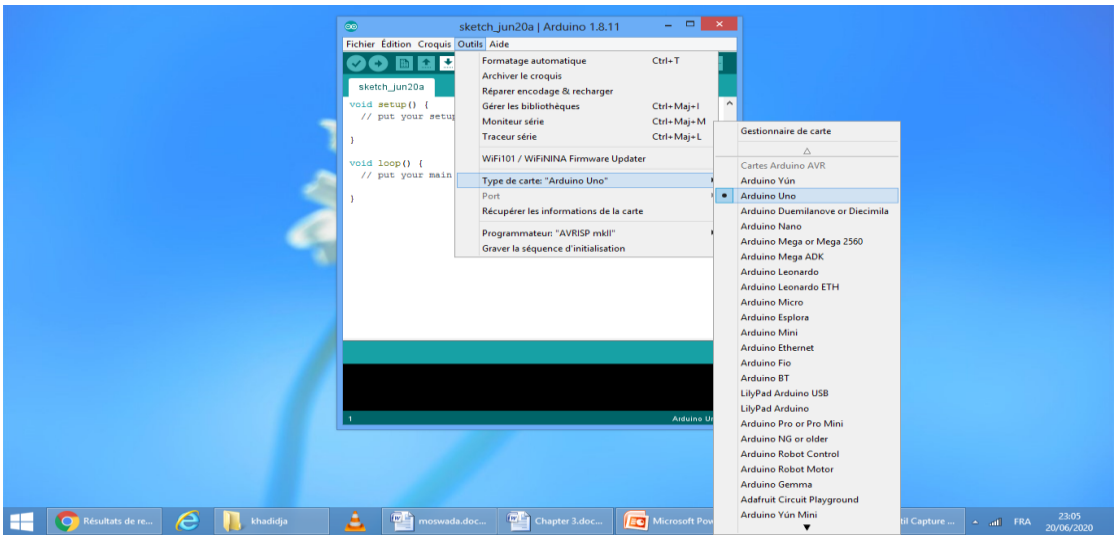


Fig Ⅲ.13. Arduino IDE

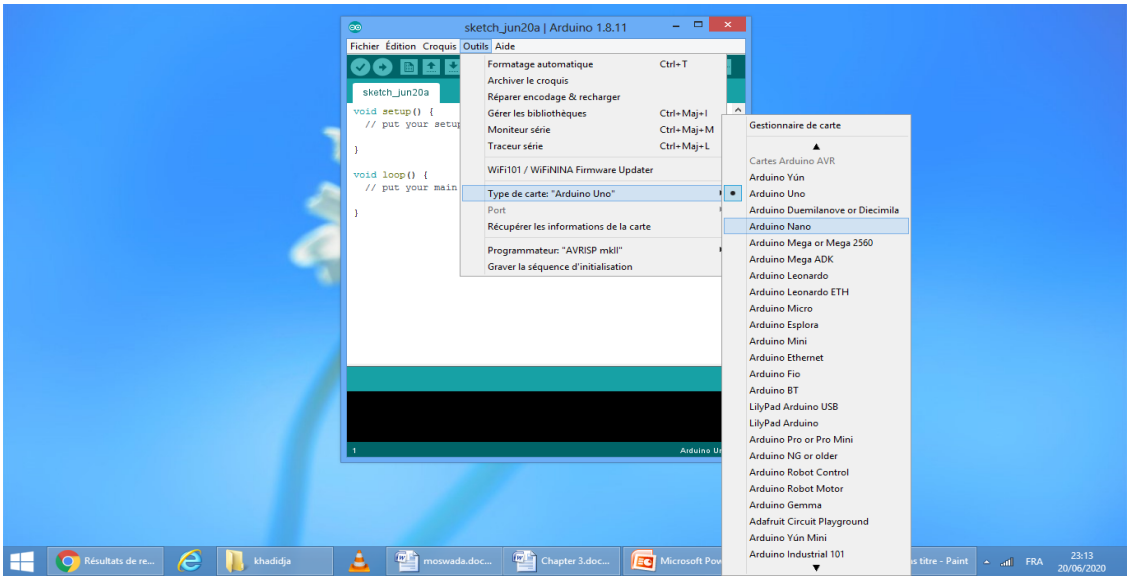
Ⅲ.8.2. Connecting the Arduino:

Connect our Arduino UNO to our computer via USB. After connecting the Arduino with a USB cable, we need to make sure that the Arduino IDE has selected the right board.

In our case, we're using Arduino Uno in the first try , so we should go to Tools ➔ Board➔ Arduino/ Uno. And in the last try we use Arduino NANO board [38]

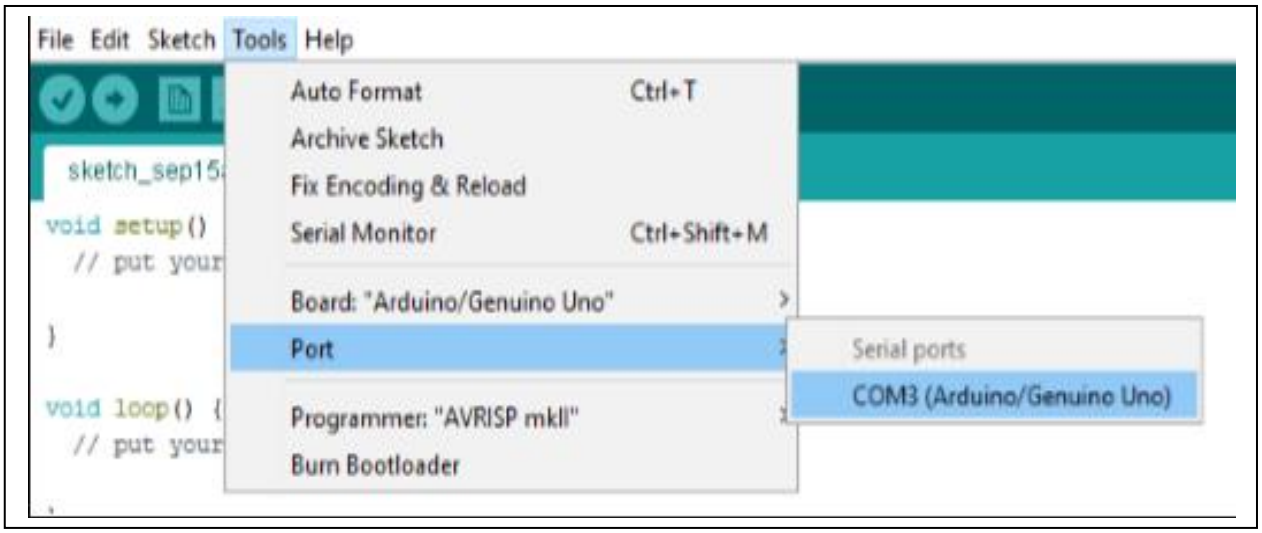


FigⅢ.14.selecting the Arduino UNO board



FigⅢ.15. Select the Arduino NANO board

Then, we should select the serial port where our Arduino is connected to. Go to Tools➔ Port and select the right port.

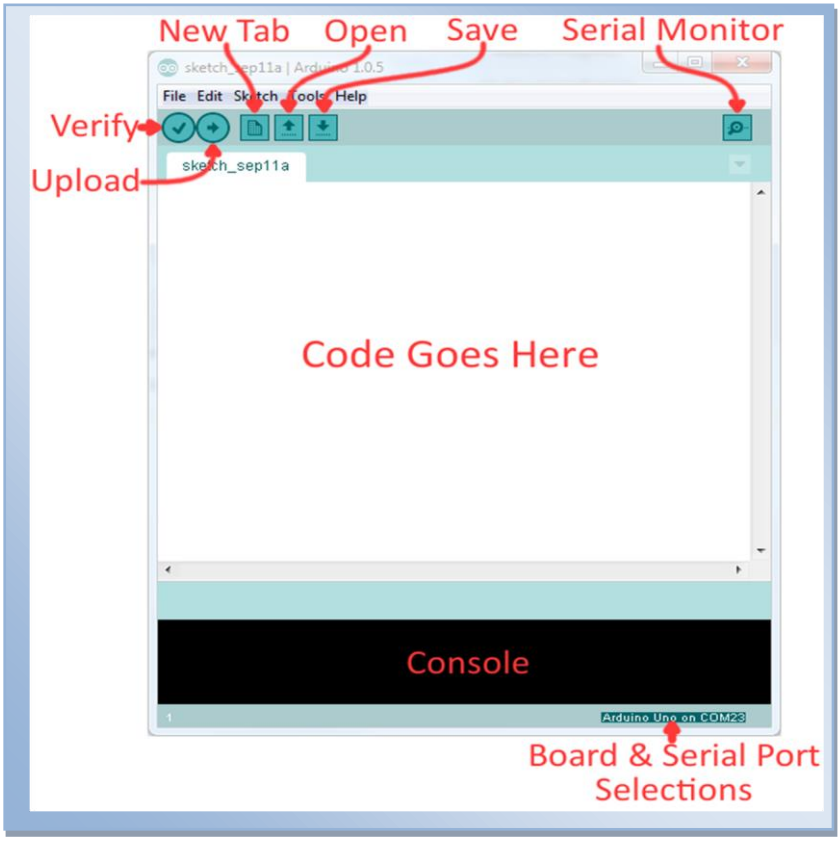


FigⅢ.16. Selecting the right PORT

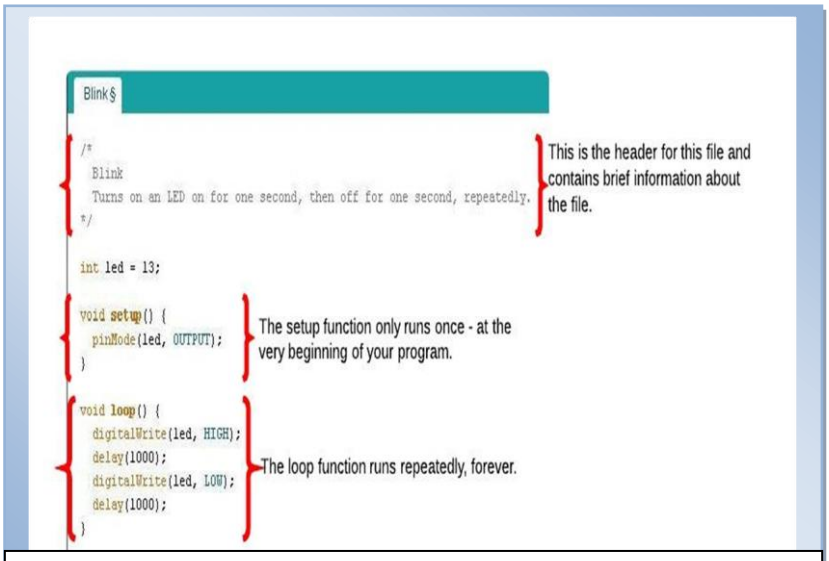
Ⅲ.8.3. Writing sketches:

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension into 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 righthand corner of the window displays the configured board and serial port. The toolbar buttons allow us to verify and upload programs, create, open, and save sketches, and open the serial monitor. [38]



FigⅢ.17.Arduino IDE interface



FigⅢ.18.IDE code writing steps

Ⅲ.8.4. Arduino Programming language:

Ⅲ.8.4.1. Introduction to the Arduino language:

Due to their simplicity, the programs we write using the Arduino IDE are called **sketches**. In their essence, they are text files written in Arduino language. To save and upload them to our Arduino board, we will need to use the **.ino** extension. There are three main parts that make up the Arduino programming language. First of all, we have **functions** that allow us to control our board. Using functions, we can analyze characters, perform mathematical operations, and perform various other tasks – e.g., **digitalRead()** and **digitalWrite()** to read or write a value to a certain pin.

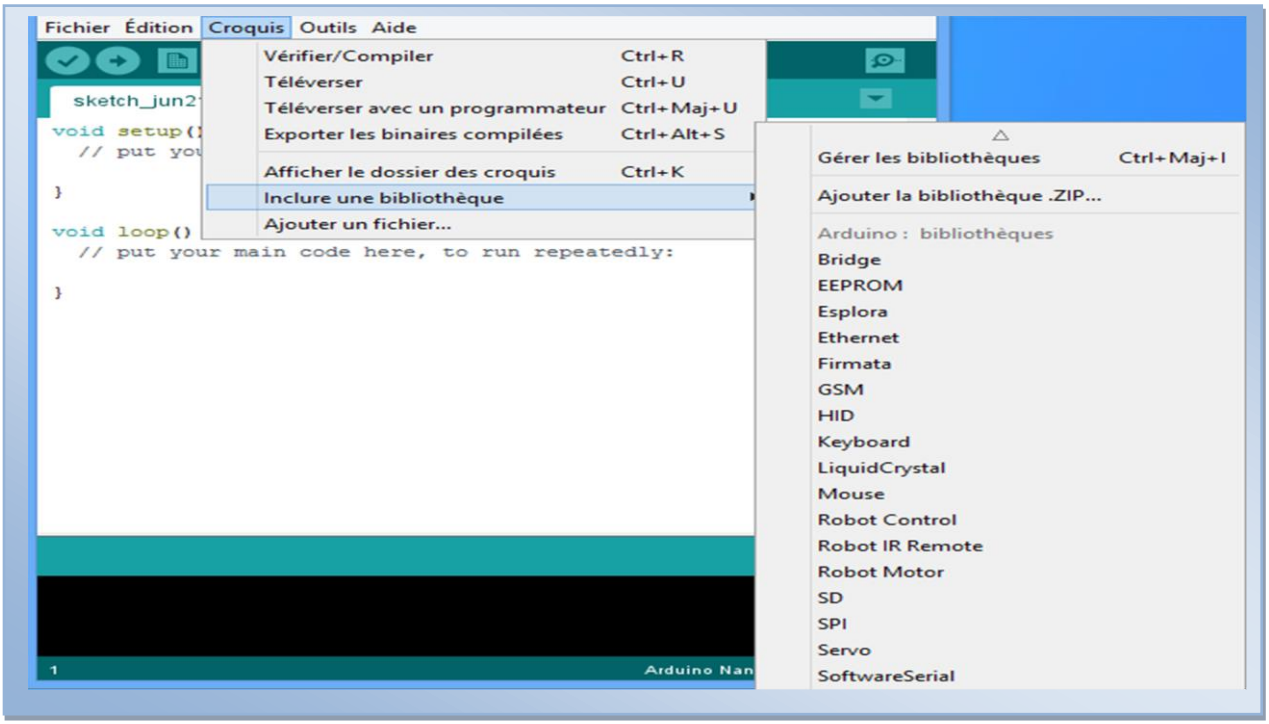
There are two functions that every sketch written in Arduino language contains. Those are **setUp()** and **loop()**. A sketch always starts with **setUp()**, which executes once after power-up or reset the board. After creating it, **loop()** ;to loop the program repeatedly until power-off or reset the board.

Next, we have the Arduino **values** that represent constants and variables. Most of the data types (array, bool, char, float, etc.) are similar to those of C++. We can perform type conversion as well. The last part of the Arduino language is called **structure**. It contains small code elements, such as operators. [9]

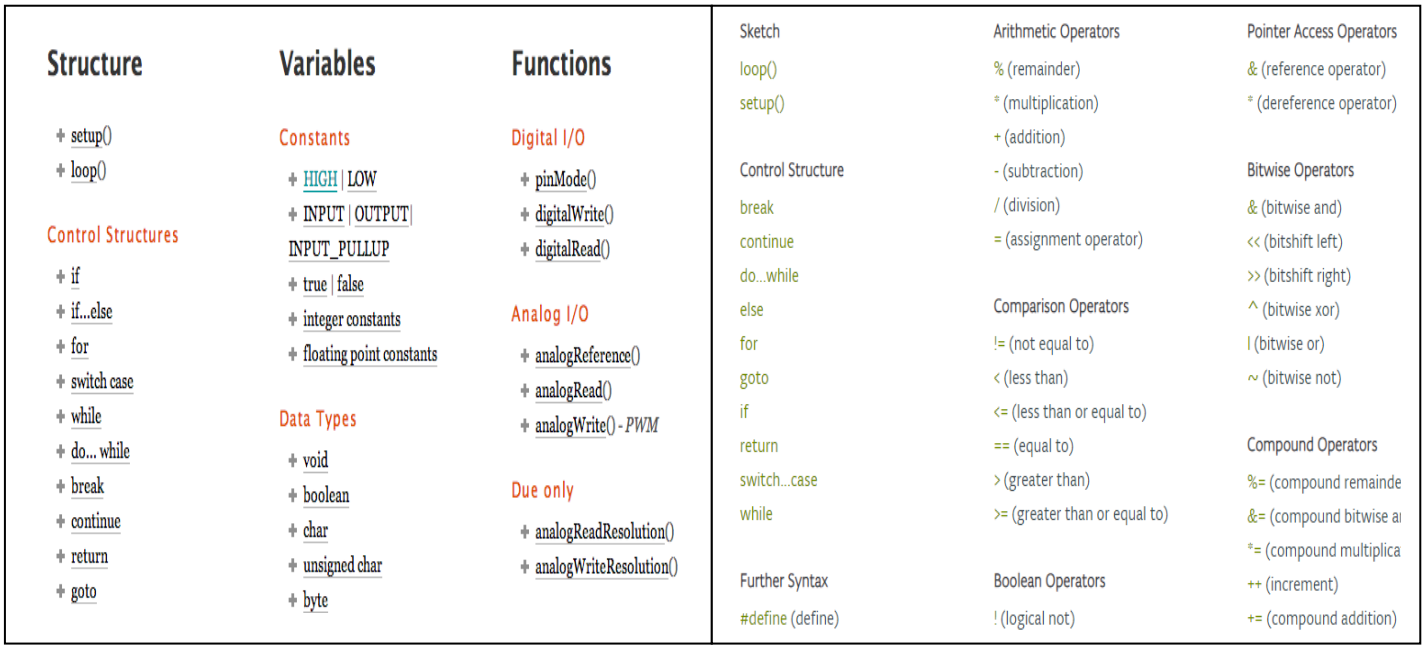
Ⅲ.8.4.2. Extending the Arduino programming language:

Like most other coding languages, the Arduino language allows to import **external libraries**. To put it shortly, a library is a set of prewritten code that provides us with extra features. If the built-in libraries are not enough we can download them online or even write our own.

We can use both C libraries and ones that are Arduino-specific. After choosing one, we'll need to install it using the Library Manager in the Arduino IDE. To include a specific library in the sketch, use the **#include** statement and name the library we need to use. [17]



FigⅢ.19. Including library in Arduino



FigⅢ.20. Arduino IDE References

Ⅲ.9. Connecting Components with Arduino NANO board:

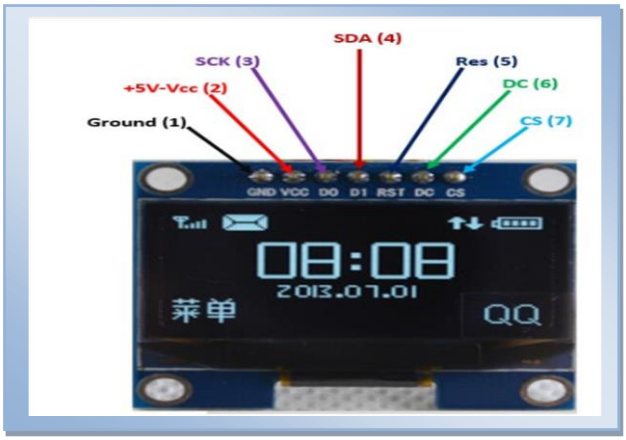
In the making of our project which is build a smartwatch that is based on OLED display in order to interface with Android phone and display some basic information from the Android smartphone like Time, date, network strength and battery status. This project provides a basic idea and framework to build an Arduino based SmartWatch using Arduino NANO board and a Bluetooth HC-06 to send informations from android phone to OLED display, Android smartphone already has inbuilt Bluetooth to send the data and at receiving end we have used Bluetooth module HC-06 with Arduino. Bluetooth module HC-05 can also be used in place of HC-06, and of corse this smartwatch needs to be power up by a battery a 3.7v LI-ON battery is the perfect for our project but finding it in our country seems to be impossible and ordering it too because of some strict delivering conditions, so we went with a normal 9v battery although it size haven't been helpful but it powered our watch

We've already talked about the Arduino NANO board and it characteristics earlier in this chapter, now we are going to explain how to connect it with the other components, but first let's talk about the characteristics of these components. [39]

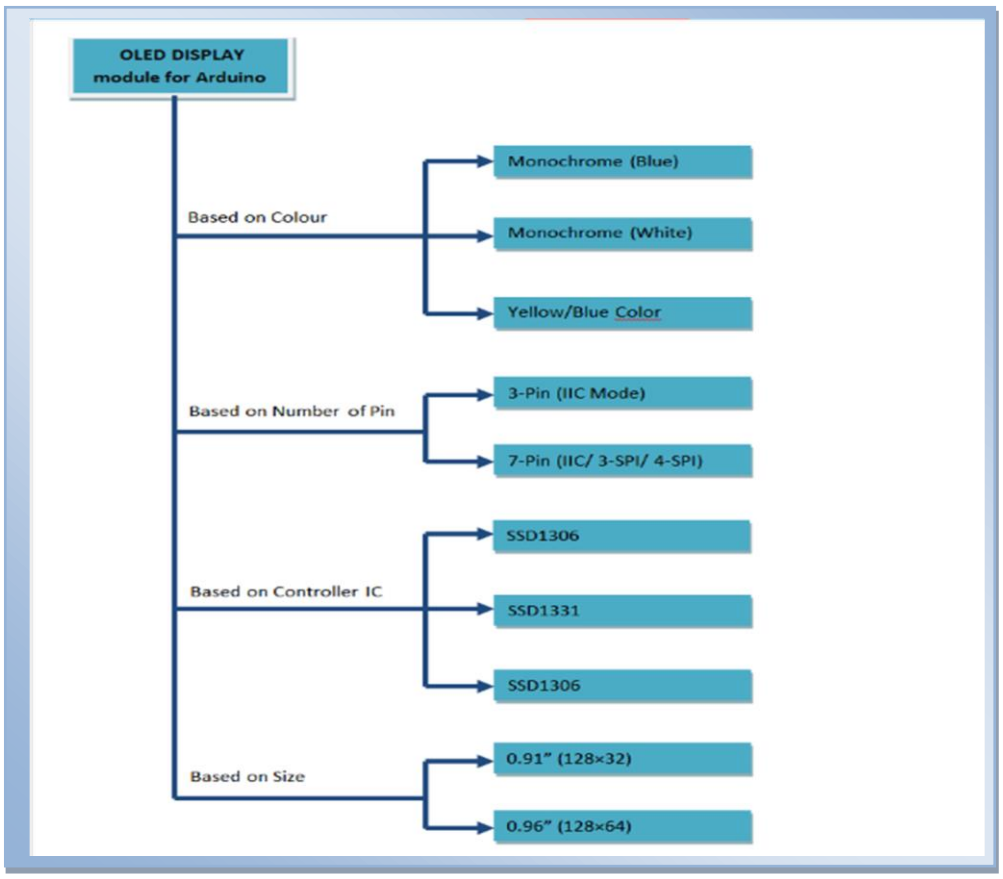
Ⅲ.9.1. OLED display:

Let's understand what these **OLED displays** mean. The term OLED stands for "*Organic Light emitting diode"* it uses the same technology that is used in most of our televisions but has fewer pixels compared to them. It is real fun to have these cool looking display modules to be interfaced with the Microcontrollers since it will make our projects look cool.

There are a lot of OLED display modules available in the market, each with its own classification. So before we buy one make sure which one would suit our project much better. The most commonly used types are classified below:



FigⅢ.22. OLED display pins



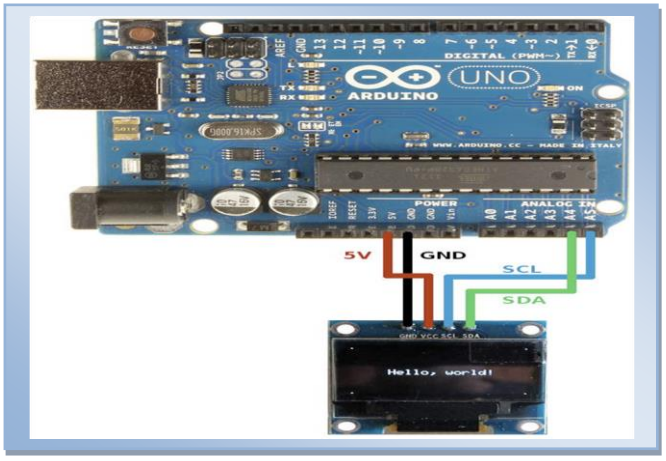
FigⅢ.2. Types of OLED display module

In our project we used the Monochrome 4-pin SSD1306 OLED display. This Display can only work with the I2C mode. [37]

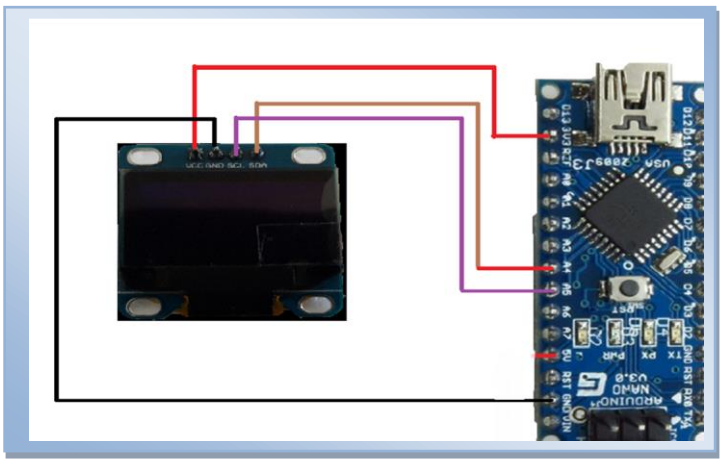
Ⅲ.9.1.1. SSD1306 OLED display Applications:

- Used in consumer electronics.
- Used for Smartwatch, mobile phone, and MP3 displays.
- Small level gaming displays.
- Wide range of viewing angle enable to be used in low light.

Ⅲ.9.1.2. Hardware and connection:



FigⅢ.23. Arduino UNO connected with OLED



FigⅢ.24. Arduino NANO connected with OLED

SHT Sensor Pins	Arduino Pins
VCC	5V or 3.3V
GND	GND
SDA	A4
SCL	A5

TableⅢ.4 Pins configuration for UNO board & OLED display

SSD1306 OLED Pins	Arduino NANO Pins
VCC	3.3V
GND	GND
SCL	A5
SDA	A4

TableⅢ.5. Pins configuration for NANO board & OLED display

Ⅲ.9.1.3. Programming the SSD1306 OLED display for Arduino:

Once the connections are ready we can start programming our Arduino NANO board, Arduino community has already given us a lot of Libraries which can be directly used to make this a lot simpler. We've tried out a few libraries and found that the **Adafruit** Library and **GFX Graphics** Library was very easy to use and had a handful of graphical options hence we will use the same in this project

- ✓ **Step 1:** Download the Adafruit Library and the GFX library from Github

1. Adafruit Library
2. GFX Graphics Library

- ✓ **Step 2:** download two Zip files.Add them to Arduino by following

Sketch ➔Include Library ➔ Add Zip library as shown below.

Then select the library we just downloaded. We can select only one library at a time, hence we have to repeat this step again.

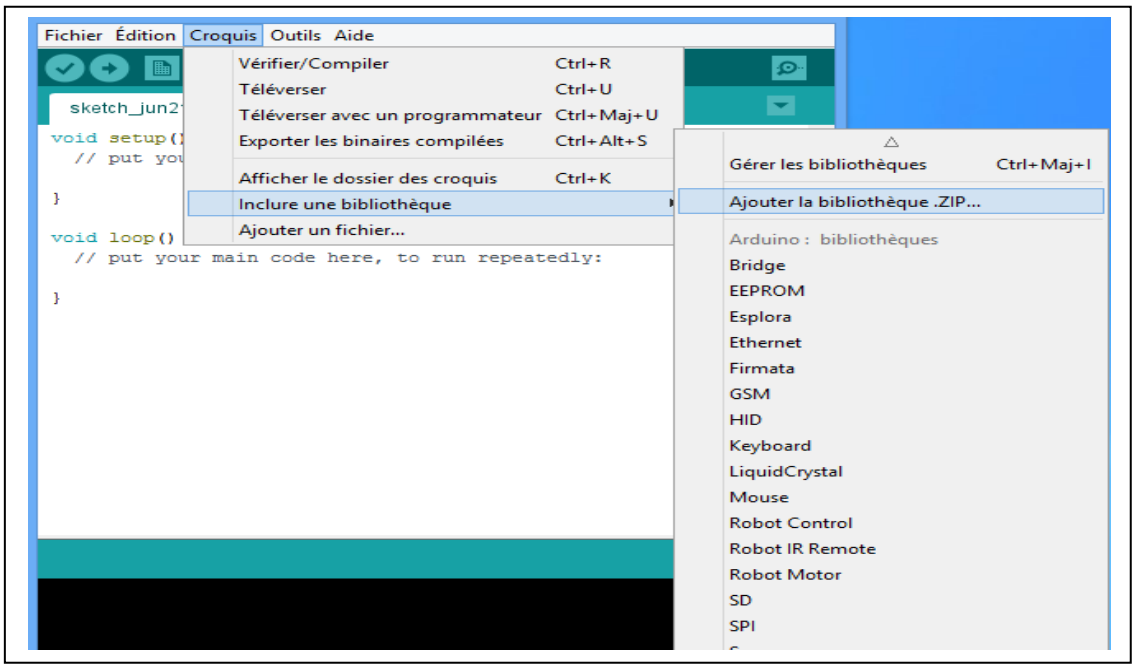
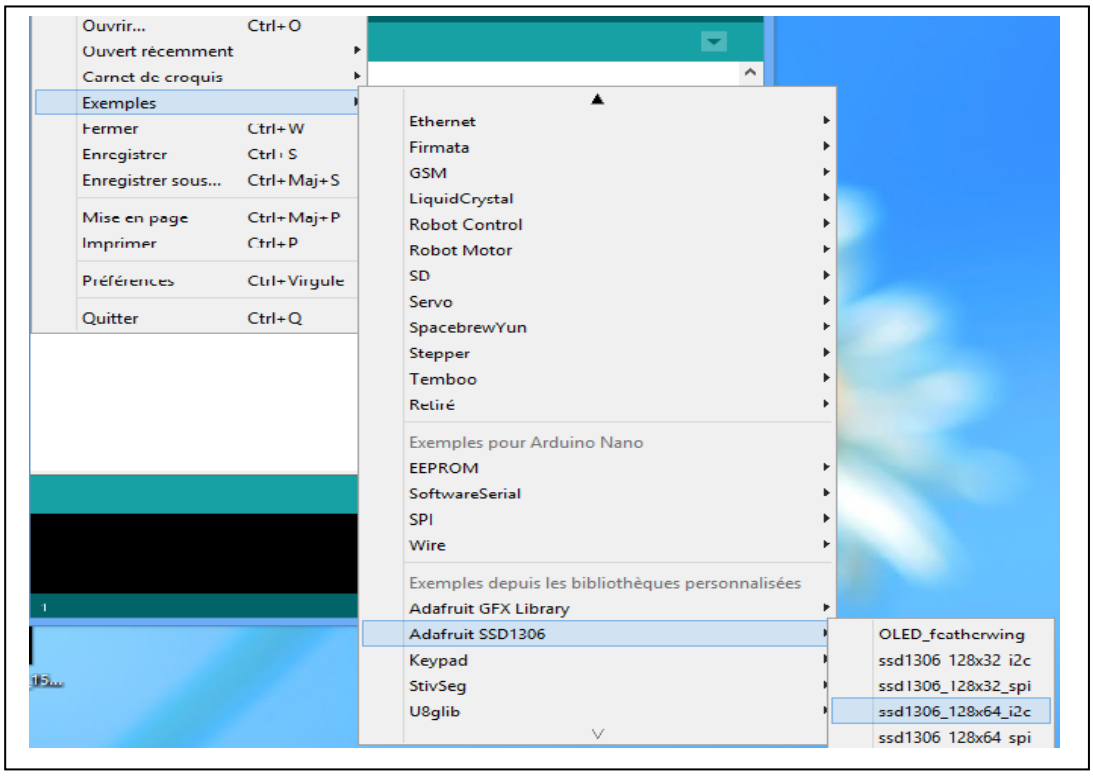


Fig Ⅲ.25. Including Zip library to Arduino

✓ **Step 3:** Launch the example Program by *selecting* File ➔Examples ➔Adafruit SSD1306 ➔ SSD1306_128*64_i2c.ino as shown in the image below.



FigⅢ.26. lunching the example program

Moving to our program now and after including all the required libraries:

```
#include <SoftwareSerial.h>
#include <math.h>
#include "bitmap.h"
#include "U8glib.h"
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
```

Fig Ⅲ.27. Including libraries in the program

After that we define some macros and variables for OLED display operations:

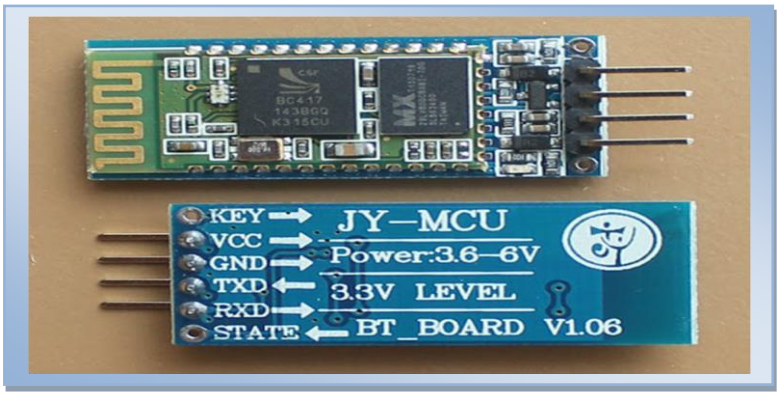
```
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
#define OLED_RESET     -1 // Reset pin # (or -1 if sharing Arduino reset pin)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
#define NUMFLAKES     10 // Number of snowflakes in the animation example
#define LOGO_HEIGHT   16
#define LOGO_WIDTH    16
U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE|U8G_I2C_OPT_DEV_0);  // I2C / TWI
```

Ⅲ.9.2. Bluetooth HC-06:

Bluetooth can operate in the following two modes:

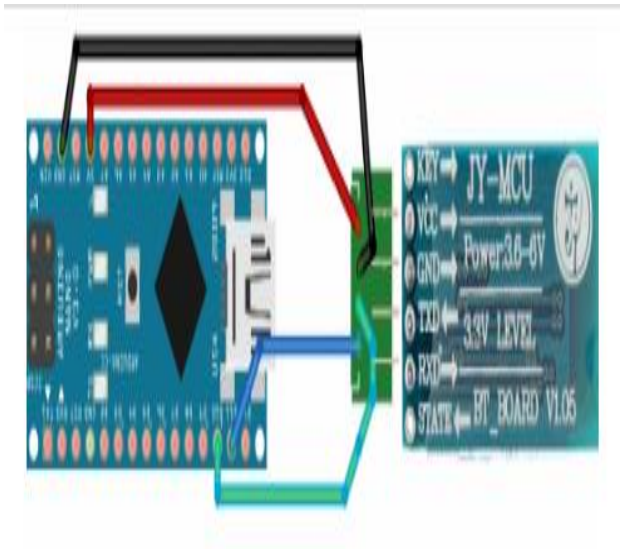
- ✓ Command Mode
- ✓ Operating Mode

In **Command Mode** we will be able to configure the Bluetooth properties like the name of the Bluetooth signal, its password, the operating baud rate etc.



FigⅢ.28. Bluetooth module HC-06

The **Operating Mode** is the one in which we will be able to send and receive data between the PIC Microcontroller and the Bluetooth module. Hence in this tutorial we will be toying only with the Operating Mode. The Command mode will be left to the default settings.



FigⅢ.29.Module BT wired up with Arduino NANO board

Module Bluetooth HC-06	Arduino NANO
GND	GND
VCC	5V
TXD	D10
RXD	D11

TableⅢ.6.Pins configuration for BT & NANO

Ⅲ.9.2.1:Bluetooth HC-06 Applications:

- Hobby projects
- Engineering applications & robotics
- Mobile Phone Accessories
- Computer Peripherals
- Sports and Leisure Equipment
- USB Dongles

- **Ⅲ.9.3. Power supply:**

FigⅢ.30. Li-On 3.7 battery

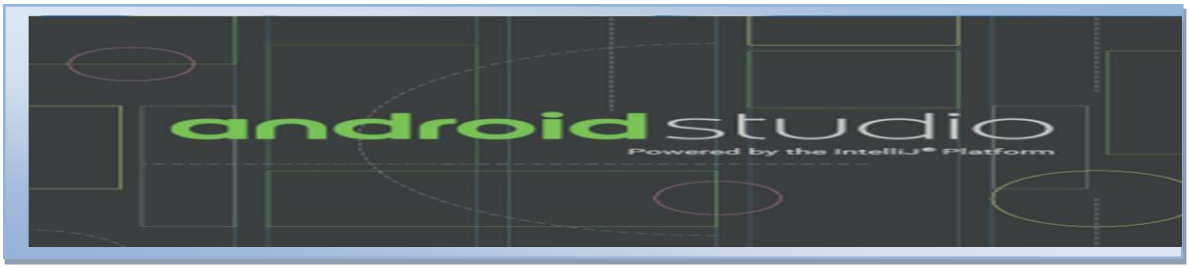
Here we have used an **Arduino NANO** board to control all the operations. The reason to choose Arduino NANO is that it can operate at 3.3v power supply. The **4 pin OLED** and **Bluetooth module HC-06** can also work on 3.3v, so all of these modules can be powered by a single 3.7v Li-on. Li-on battery is the compact and light weight battery, and is the perfect choice for wearable devices, As we said earlier is that it isn't available so we had to go with a normal 9v battery to power up our smartwatch .

FigⅢ.31. 9V battery

Now one question arises about power supply is that here all the modules are working on 3.3v but our works on 9v which may damage the modules.So to solve this problem we have applied battery's 9v supply to a raw pin of Arduino NANO which can convert that voltage to 3.3v.

Ⅲ.10. Android Application for sending data to Arduino over Bluetooth:

In this project we tried to create our own application to install on the smartphone and help connecting with our smartwatch, this application would've worked on android phones so the first thing we needed to do is to install the **ANDROID STUDIO** application

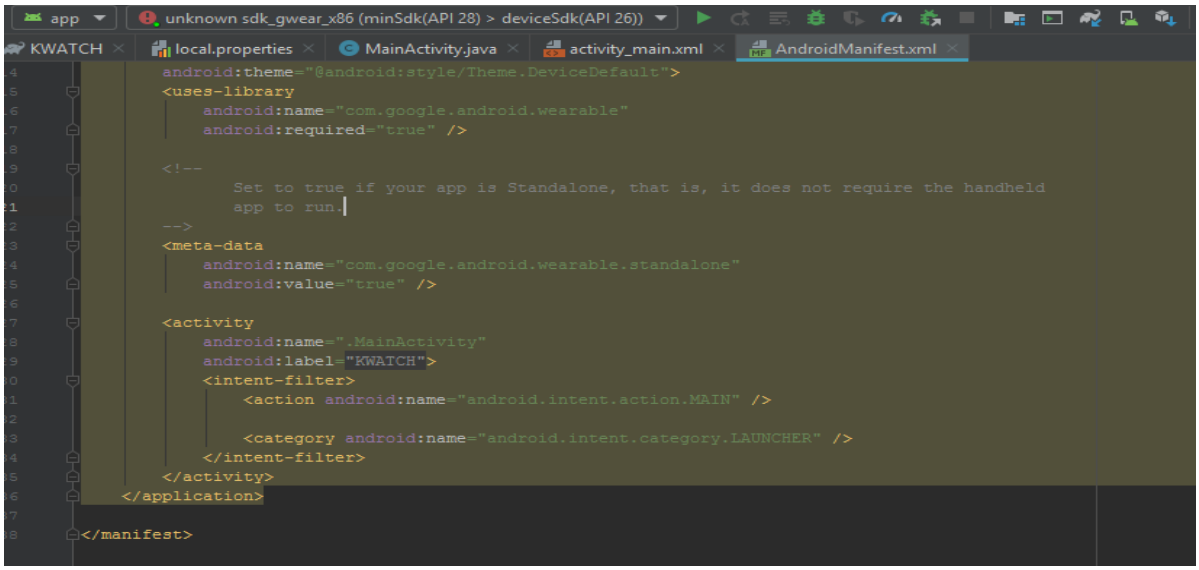


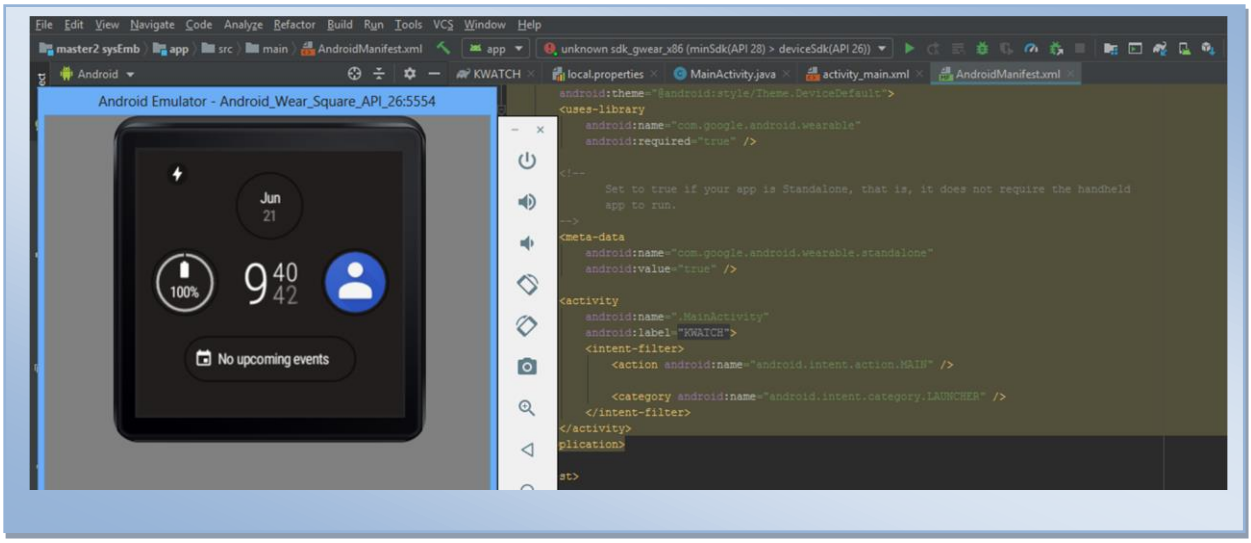
Ⅲ.10.1. Launch the emulator and run ANDROID STUDIO:

To use the emulator, we must configure an Android Virtual Device, referred to as an AVD. Confirm that we have the latest version of the **Android SDK Platform-tools** from the SDK Manager.

Configure an AVD and run the app as follows:

1. In Android Studio, open the Android Virtual Device Manager by selecting **Tools > AVD Manager**.
2. Click **Create Virtual Device**.
3. In the **Category** pane, select **Wear** and choose a hardware profile. Click **Next**.
4. Select the **O** image to download. For example, select the image with the **Release Name** of O, the **API Level** of 26, and the **Target** of "Android 8.0 (Wear OS)". Click **Next** and then click **Finish**.
5. Close the Android Virtual Device Manager.
6. In the Android Studio toolbar, select the AVD we just created from the target device drop-down menu, then click **Run**  .





FigⅢ.32. Program simulation result in Android studio

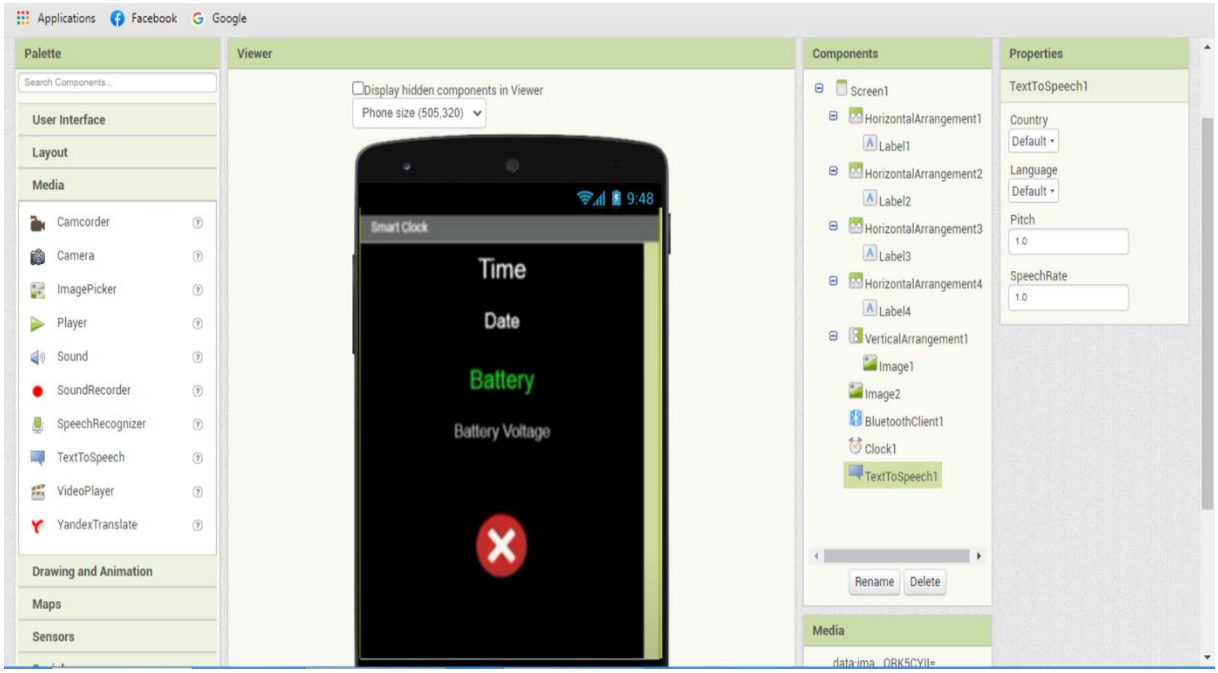
Unfortunately our Oled display wasn't quite developed which lead me to stop the simulation of this application and go look for another ready application from google store, I have found 'RetroWatch' application that seemed a great one so I've download it then started the simulation with it matched arduino code and once everything was set up, the data that was generated wasn't enough so we had to change the application another time but this time we've created our own application that reflects data that we need to show.

Ⅲ.10.2. Launch the emulator and run MIT APP INVENTOR:

We used for the **MIT A2 app inventor** which is a **free, cloud-based service** that allows us to make our own mobile apps using a blocks-based programming language.

Ⅲ.10.2.1User Interface:

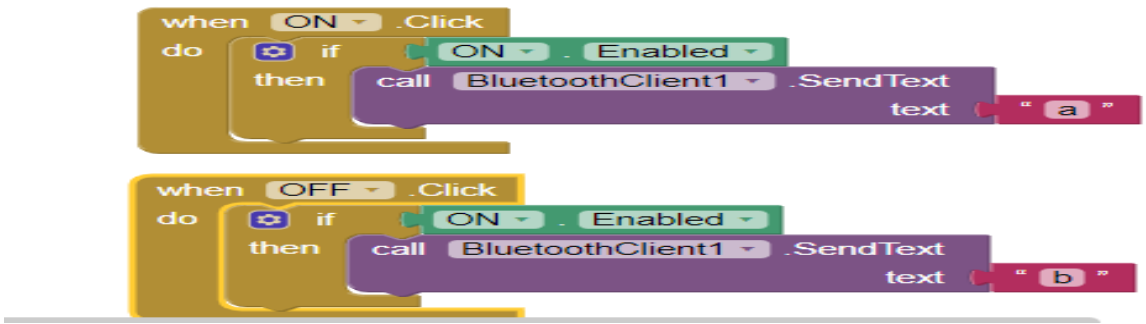
The user interface consists of labels to show location data and buttons to initiate events. Some labels just show static text, e.g., GPS Label is the text "GPS:" that appears in the user interface. Others, such as Current Lat Label, will display dynamic data one the location sensor gets its readings. For these labels, a default value is set (0) here in the Component Designer



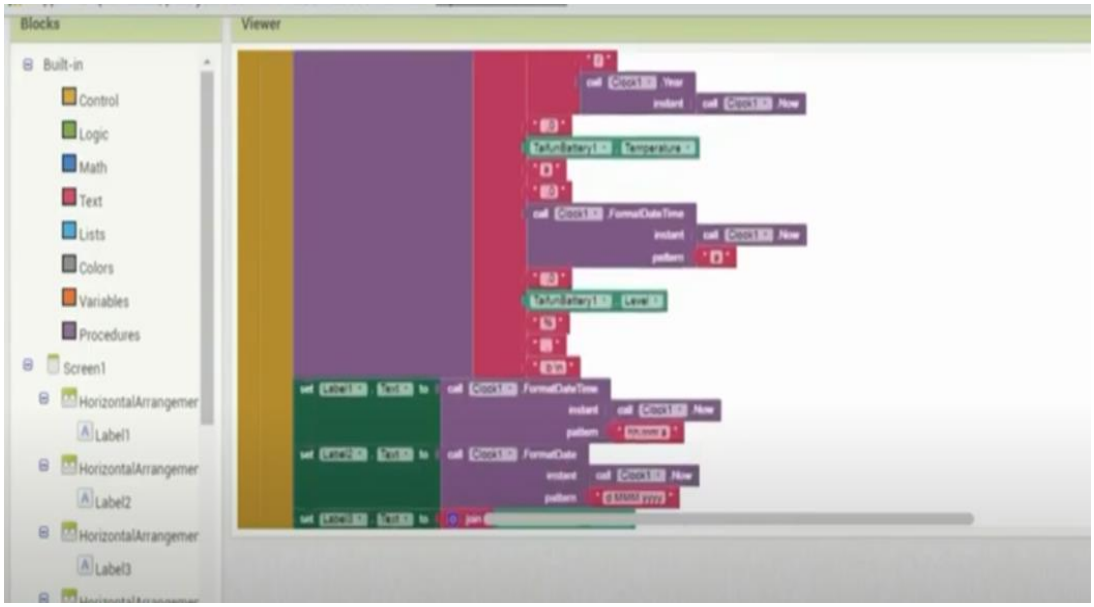
FigⅢ.33. Application designing block

Ⅲ.10.2.2. The App's Behavior:

Here are the blocks for the Android, Smart Clock application:



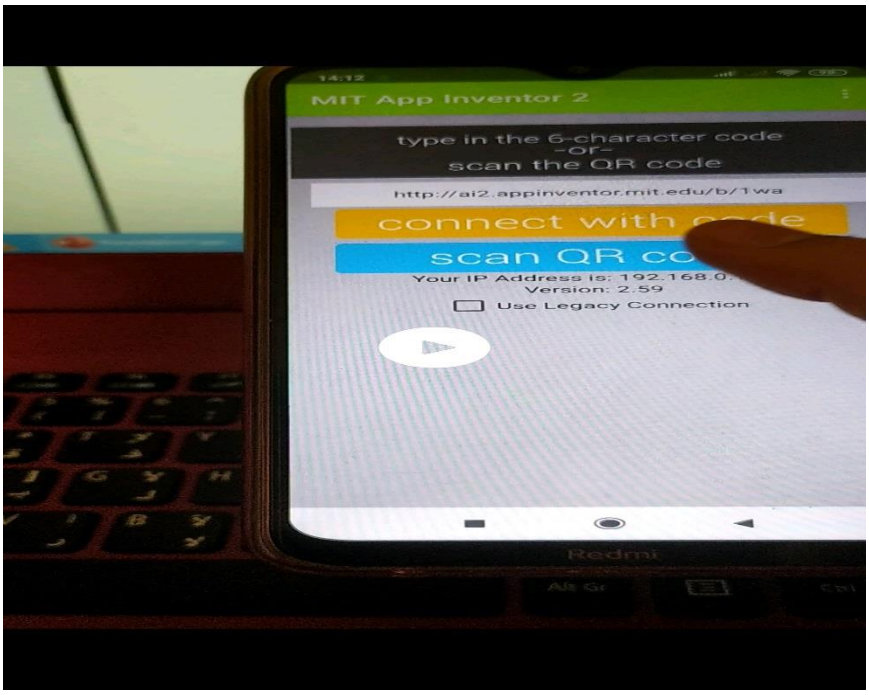
FigⅢ.34. blocks for the Smart Clock application (Bluetooth connection)



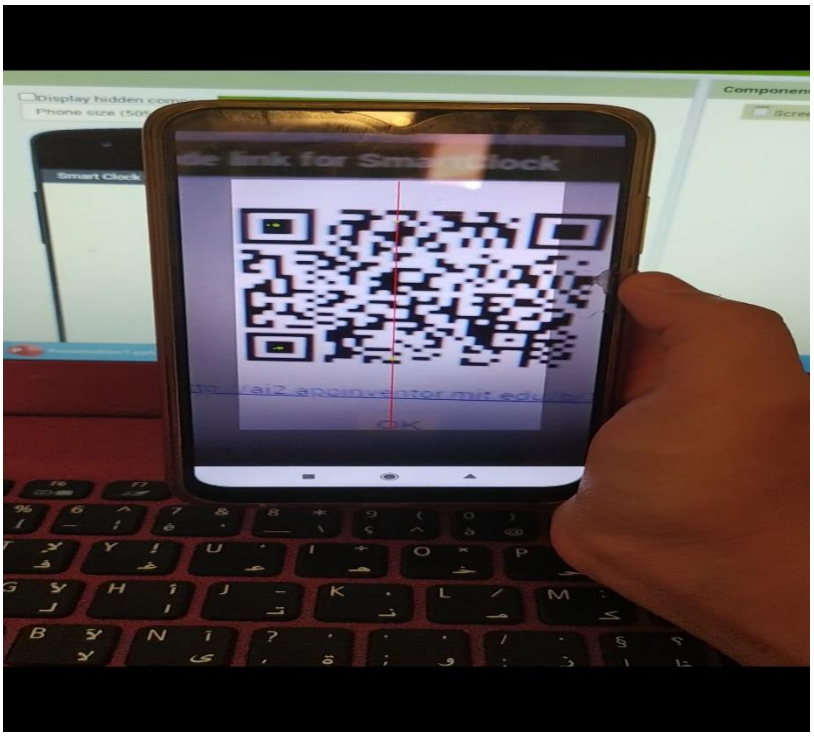
FigⅢ.35. blocks for the Smart Clock application

Ⅲ.10.2. Scan the Sample App to the Phone:

Scan the following barcode onto our phone to install and run the sample app before this step we must install the MIT AI2 on our phone

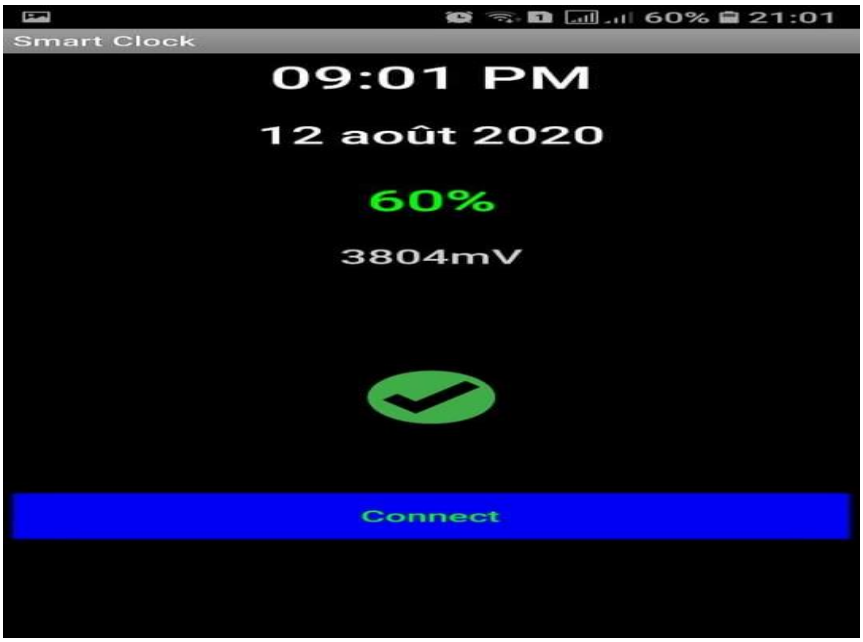


FigⅢ.36. MIT App inventor2 Application



FigⅢ.37. scanning code from pc using phone application

After the application is loaded on our phone and ready to be used, we can activate our phone Bluetooth to connect with our smartwatch Bluetooth HC06 device.



FigⅢ.38. Smart-Clock app on phone screen

Ⅲ.11. Simulation With ISIS PROTEUS:

III.11.1. Definition:

Proteus Professional is a software suite for electronics. Developed by the company Absenter Electronics, the software included in Proteus Professional enables CAD (Computer Aided Construction) in the electronic field. Two main software's make up this software suite: (ISIS, ARES, PROSPICE) and VSM.

This software suite is very well known in the field of electronics. Many companies and training organizations (including high school and university) use this software suite. Besides the popularity of the tool, Proteus Professional has other advantages:

- Pack containing software that is easy and quick to understand and use
- Technical support is efficient
- The virtual prototyping tool helps reduce hardware and software costs when designing a project.

➢ ISIS:

Proteus Professional ISIS software is primarily known for editing electrical schematics. In addition, the software also makes it possible to simulate these diagrams which makes it possible to detect certain errors from the design stage. Indirectly, the electrical circuits designed with this software can be used in documentations because the software allows to control the majority of the graphic aspect of the circuits.

➢ ARES:

ARES software is an editing and routing tool that perfectly complements ISIS. An electrical diagram produced on ISIS can then be easily imported into ARES to produce the PCB (Printed circuit board) of the electronic card. Although editing a PCB is more efficient when done manually, this software allows you to automatically place components and perform routing automatically.

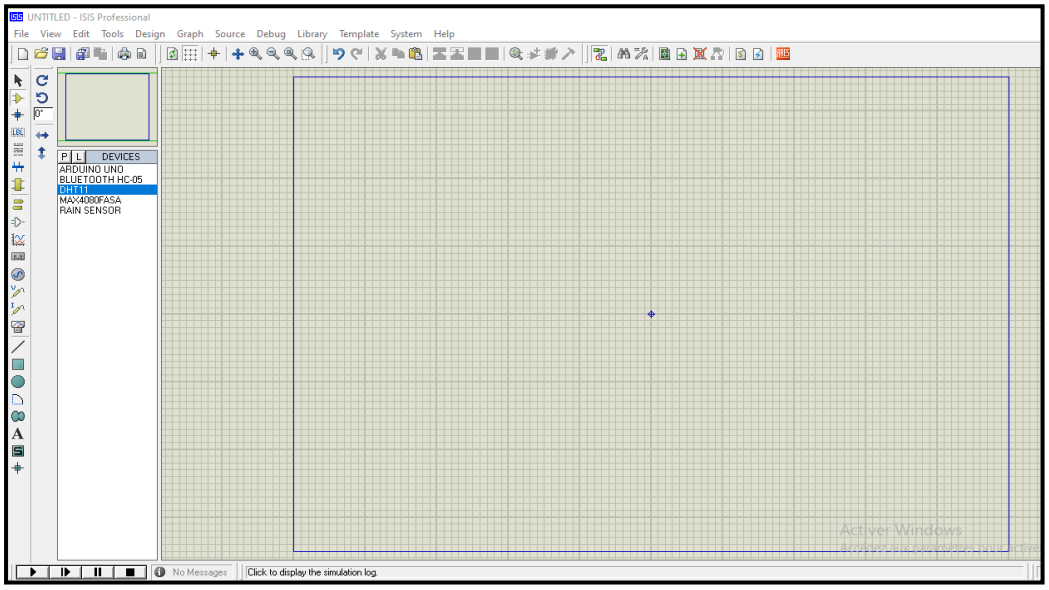


Fig III.39. ISIS PROTEUS software interface

III.11.2. Download the necessary libraries for this project:

- Step 01: added Arduino.
- Step 02: add Bluetooth

So, to add these libraries in Proteus we followed these steps:

1. First, download the library from the links given below:

Arduino Library for Proteus

» Arduino UNO
» Arduino Méga 2560
» Arduino Méga 1280
» Arduino Nano
» Arduino Mini
» Arduino Pro Mini

Bluetooth Library for Proteus

» BluetoothTEP.IDX
» BluetoothTEP.LIB

2. Download the file to the desktop

3. Unzip the .zip file on the desktop.

4. Cut the two files obtained from the desktop

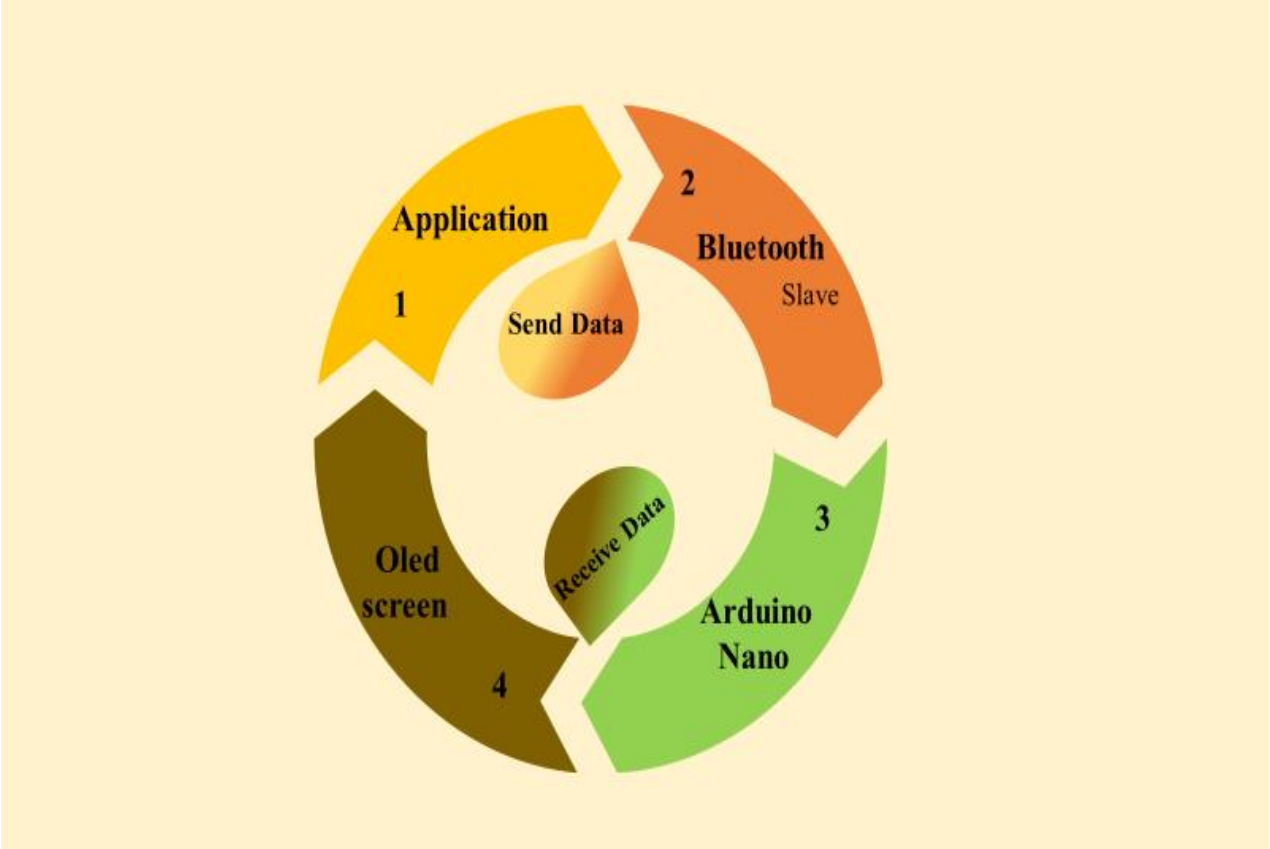
5. Navigate to C: \ Program Files (x86) \ Labcenter Electronics \ Proteus 7 Professional \ LIBRARY.

6. Paste the files here and you're done!

7. Open Proteus and click “Choose from Libraries”.

8. Search for “Arduino, BT, DHT ..”, you will see the boards and components in the results bar. Select the board of your choice and start simulating

III.11.3. Synoptic diagram:



FigⅢ.40. smartwatch's synoptic diagram

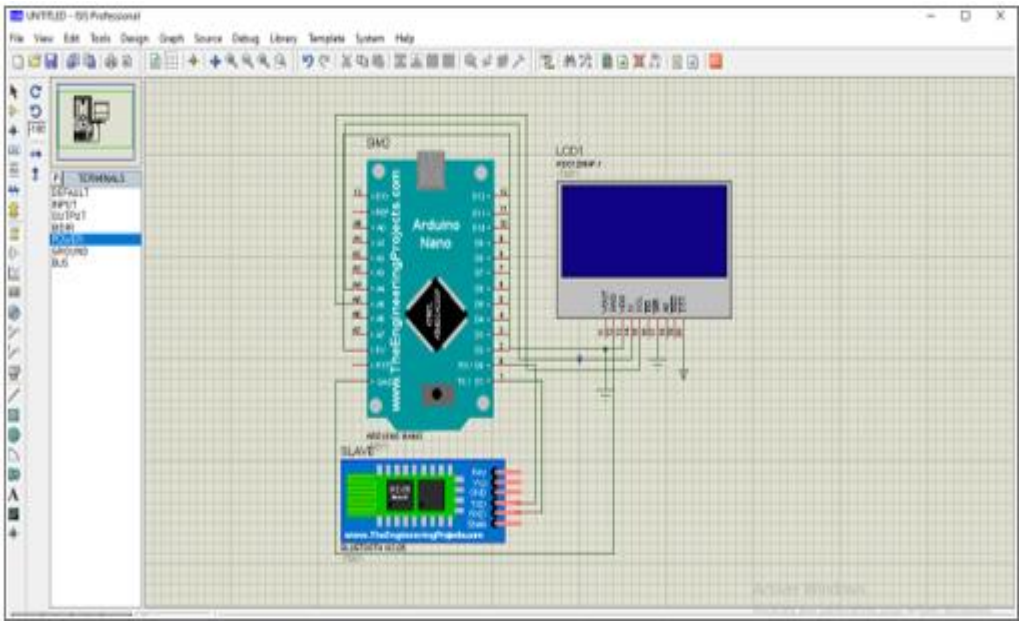


Figure III.41. Schematic of Smartwatch with PROTEUS

Ⅲ.11. Project description:

Ⅲ.11.1. Different stages of assembly:

The basic idea of this is to create a smartwatch that can be connected to our android smartphone by Bluetooth after connecting all the needed components for this project the next step was to create an android application but due to quarantine we haven't finish all the simulation needed to run our watch with Android studio especially with the OLED screen that we have.

a. Before improvement:

So our first experience was to download an android application called 'RetroWatch', then we wrote the code on our IDE in order to display the time on our smartwatch as a first step, and once we connected our watch to the android phone by activating the Bluetooth and running the program on IDE, the information that appeared on the watch screen was time and thanks to the application functions we could change the mode of the time data, there was three modes :

- Digital mode
- Analog mode
- Mixed mode: it shows both Analog & digital mode

But this data wasn't enough for our project we wanted to add more data to our smartwatch.

b. After improvement:

So we installed an application from google store that helped running our smartwatch it is the 'SMART CLOCK' application so easy to work with and manipulate, it shows on our OLED screen the time in the digital mode with the indication of whether it is before midday(AM) or after midday (PM),beside that it shows the date too, it also reflects the battery and signal level and when it is first powered On it shows my name 'Allaoui Khadidja', next to those the temperature degree also shows up on our smartwatch screen, so we can say that our watch use Bluetooth for data that is downloaded from our android smartphone

Ⅲ.11.2. Programming Code:

```
#include <SoftwareSerial.h>

SoftwareSerial BTserial(2,3); //hc06 tx to 2,rx to 3

String tim;

String dat;

String temp;

String lev;

String ampm;

#include "U8glib.h"

U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE|U8G_I2C_OPT_DEV_0); // oled Scl to A5, Sda to A4

const uint8_t fullsignal[] PROGMEM= {

  0xFC, 0x03, 0x84, 0x03, 0x58, 0x1B, 0x30, 0x1B, 0x30, 0xDB, 0x30, 0xDB, 0x36, 0xDB, 0x36, 0xDB,

0x36, 0xDB, 0x36, 0xDB, 0x36, 0xDB, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00

};

const uint8_t batt[] PROGMEM= {

0x00, 0x00, 0x00, 0x00, 0x7F, 0xFF, 0x40, 0x01, 0xDB, 0x6D, 0xDB, 0x6D, 0xDB, 0x6D, 0xDB, 0x6D,

0xDB, 0x6D, 0xDB, 0x6D, 0xDB, 0x6D, 0x40, 0x01, 0x7F, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00

} ;

const uint8_t intro[] PROGMEM= {

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
```

```
0x00, 0x07, 0x81, 0xFB, 0xE1, 0xFC, 0x7E, 0xE0, 0x01, 0xFC, 0x7F, 0x1F, 0x06, 0xF8, 0x00, 0x00,
0x00, 0x0F, 0xC0, 0xF9, 0xF0, 0xF8, 0x3E, 0x70, 0x00, 0xF8, 0x3F, 0x0E, 0x1C, 0x38, 0x00, 0x00,
0x00, 0x0F, 0xC0, 0xF8, 0xF8, 0xF8, 0x3E, 0x38, 0x00, 0xF8, 0x1F, 0x84, 0x3C, 0x18, 0x00, 0x00,
0x00, 0x0F, 0xC0, 0xF8, 0xF8, 0xF8, 0x3E, 0x3C, 0x00, 0xF8, 0x1F, 0xC4, 0x3C, 0x18, 0x00, 0x00,
0x00, 0x1F, 0xC0, 0xF8, 0xF8, 0xF8, 0x3E, 0x3C, 0x00, 0xF8, 0x1F, 0xC4, 0x7C, 0x08, 0x00, 0x00,
0x00, 0x1F, 0xE0, 0xF8, 0xF8, 0xF8, 0x3E, 0x3E, 0x00, 0xF8, 0x17, 0xE4, 0x7C, 0x08, 0x00, 0x00,
0x00, 0x1F, 0xC0, 0xF8, 0xF8, 0xF8, 0x3E, 0x3C, 0x00, 0xF8, 0x1F, 0xC4, 0x7C, 0x08, 0x00, 0x00,
0x00, 0x1F, 0xE0, 0xF8, 0xF8, 0xF8, 0x3E, 0x3E, 0x00, 0xF8, 0x17, 0xE4, 0x7C, 0x08, 0x00, 0x00,
0x00, 0x1B, 0xE0, 0xF8, 0xF0, 0xF8, 0x3E, 0x3E, 0x00, 0xF8, 0x17, 0xE4, 0x7C, 0x00, 0x00, 0x00,
0x00, 0x31, 0xF0, 0xF8, 0xF8, 0xF8, 0x3E, 0x3E, 0x00, 0xF8, 0x10, 0xFC, 0x7C, 0x00, 0x00, 0x00,
0x00, 0x7F, 0xF0, 0xF8, 0xF8, 0xF8, 0x3E, 0x3C, 0x00, 0xF8, 0x10, 0xFC, 0x7C, 0x08, 0x60, 0x00,
0x00, 0x61, 0xF0, 0xF8, 0xF8, 0xF8, 0x3E, 0x3C, 0x00, 0xF8, 0x10, 0x7C, 0x7C, 0x08, 0xF0, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};
void draw(void) {
   u8g.setFont(u8g_font_8x13);
  u8g.setPrintPos(18,10);
  u8g.print("Khadidja Allaoui");
  u8g.drawBitmapP( 0,15, 16, 64,intro);
}
void setup(){
  BTserial.begin(9600);
  u8g.firstPage();
```

```
  u8g.firstPage();
  do {
   draw(); // show intro on startup
        }
  while( u8g.nextPage() );
  delay(2000);    }
void loop() {
   while (BTserial.available() > 0) {
    tim=BTserial.readStringUntil(','); // writes in the string all the inputs till a comma
          BTserial.read();
    dat=BTserial.readStringUntil(',');
          BTserial.read();
    temp=BTserial.readStringUntil(',');
          BTserial.read();
    ampm=BTserial.readStringUntil(',');
          BTserial.read();
    lev=BTserial.readStringUntil(',');
          BTserial.read();
    String notused= BTserial.readStringUntil('\n'); // null to emit /n
    }
   u8g.firstPage();
  do {
     u8g.setFont(u8g_font_unifont);
     u8g.setPrintPos(85,12); //Print phone battery LEVEL
  u8g.print(lev);
  u8g.setPrintPos(32,60); //Print Date
  u8g.print(dat);
```

```
    u8g.drawBitmapP(2,0,2,16,fullsignal);// Print signal icon
    u8g.drawBitmapP(112,0,2,16,batt); // Print battery icon


u8g.setFont(u8g_font_freedoomr25n);
  u8g.setPrintPos(16,44); //Print TIME
  u8g.print(tim);
u8g.setFont(u8g_font_6x10);
  u8g.setPrintPos(106,40); //Print am/pm
  u8g.print(ampm);
  u8g.setPrintPos(106,30); //Print Temp from phone
  u8g.print(temp);
  u8g.setPrintPos(119,26); //Print degree
  u8g.print(".");
  } while( u8g.nextPage());
 //delay(1000);
}
```

Ⅲ.11.2.1. Programming code description:

After including the required libraries and defining some variables for our components, we included <SoftwareSerial> library to allow serial communication and replicate the functionality of the hardwired Rx &Tx lines of the Bluetooth HC-06 device to connect with our Arduino board.

Then we used the string variable to the sequence of letters such as:

- tim ; for time
- dat ; for date
- temp ; for temperature
- lev ; for both signal and battery level
- am,pm ; for the midday indicators

Another library is included to our program <u8glib.h> which is a graphics library with support for many different monochrome displays and for our case it's for our SSD1603 OLED display to get it connected with our Arduino.

Then we declared our data in PROGMEM to store it flash memory such as : full signal [], batt[], intro[].

In order to print my name on the OLED screen we used the void draw(void) function and when our watch is powered my name shows up.

In the void setup() function we wrote our code that allowed to move from intro page to the other page on our OLED screen once our program starts running.

Because we are using an Android application to help communicating between our smartphone and our smartwatch by Bluetooth, the exchanged data can be many in a short time in a two-way communication and for this reason we used in our void loop() function <serial.readStringUntil()> that provides reading a text string until a character is found, in our case the terminator is (',') and the string that we declared earlier in this program.

Then we started printing the icons for the data that are going to show up on our smartwatch screen as the time with it indicator (am/pm), temperature icon with the degree sign, also the battery and signal icons all these using <U8g.setPrintPos> instruction.

So we can say that our program provides the communication between our Android Smartphone and our SmartWatch using "Smart Clock" , and it manage setting the icons for the data screening on our smartwatch.

Ⅲ.12. Conclusion:

In this part of the work we described the progress of the different stages of our application. We have given a global block diagram to explain how the system works. This achievement allowed us to get to know the electronic components and we learned Arduino's programming bases.

Our smartwatch is working in perfect way showing all the data we needed to exchange with our android smartphone, finally we can say that we achieved our goal

General Conclusion & Perspectives

Since embedded systems are everywhere and we cannot get away from them, so in this thesis we tried to present them and their everyday evolution especially wearable devices that represent a huge part in our lives and with their interesting programming part we tried to realize one of the most successful wearable and most used one, which is a SmartWatch based on Arduino Nano and Bluetooth HC06 in order to send informations from an Android smartphone .

During this studies that we've been doing we figure out that embedded systems are used in a variety of technologies across industries even in medical field, and looking to the development of this technology especially for wearable devices and some of the most common functions for wristwear devices like smartwatches and health and fitness trackers include monitoring steps, distance travelled, steps climbed, calories burned, activity statistics, sleep time and quality, heart rate, fitness level, workout routes, GPS location etc. These devices are not only helpful in tracking these types of user habits, but the technology is also beneficial to businesses and various industries, like in healthcare and manufacturing, for improving workplace productivity, quickly solving issues and ensuring the efficiency of organizations. Even though its small size might cause some difficulties in running some notifications with just simple button or a finger touch.

The work that we've accomplished in this thesis is a smartwatch based on an Arduino Nano board and a Bluetooth HC-06 that helps sending informations from the smartphone to our watch and reflecting it in an OLED screen using our android application "Smart Clock", so we described the performance of this project also the impact of the entire components on our circuit and their operating principal

The results obtained indicate that our Smartwatch once it is connected to the Android smartphone by the Bluetooth HC-06 device and using the application 'SmartClock' to control it, our OLED screen shows the informations below:

- Time in the Digital mode with the indicator of whether it is morning timing 'Am' or evening timing 'Pm'
- It also shows the Date and temperature of the day
- It shows the battery level

We wanted to develop this smartwatch more, and add more applications to it but due to the quarantine and the shortage of components especially the power supply device we reached this much of work, not just this difficulties that we had during this project we also didn't finish the application that we wanted to create for own watch using the Android studio application and we had to download more than one application to finally end up inventing the 'Smart Clock' application that it code match right with the codes of the components of our watch, but in spite of all this problems we reached our goal

We hope to open doors into more additions to this watch and make it more suitable and practicable for the everyday uses.

Bibliography

Reference:

[1] Ibrahim, D. (2013). Designing embedded systems with 32-bit PIC microcontrollers and MikroC. Newnes.

[2] Dey, N., & Mukherjee, A. (2018). Embedded systems and robotics with open source tools. CRC Press.

[3] Jiménez, M., Palomera, R., & Couvertier, I. (2013). Introduction to embedded systems. Springer.

[4] https://www.microcontrollertips.com/real-time-operating-systems-for-wearable-devices-in-the-iot/

[5] https://www.edgefxkits.com/blog/embedded-systems-development-process/

[6] https://www.maxfizz.com/components-of-embedded-system/

[7] https://iot.electronicsforu.com/content/tech-trends/ai-and-security-in-embedded-system-design/

[8] Bamberger, K. A. (2009). Technologies of compliance: Risk and regulation in a digital age. Tex. L. Rev., 88, 669.

[9] https://techmeetups.com/most-popular-and-influential-programming-languages-of-2018/

[10] https://www.colorado.edu/oai/embedded-software-hardware-architecture

[11] https://www.microcontrollertips.com/real-time-operating-systems-for-wearable-devices-in-the-iot/

[12] https:// www.EmbeddedSoftwareWorks.com/

[13] Jiménez, M., Palomera, R., & Couvertier, I. (2014). Assembly Language Programming. In Introduction to Embedded Systems (pp. 155-218). Springer, New York, NY.

[14] Rawassizadeh, R., Price, B. A., & Petre, M. (2014). Wearables: Has the age of smartwatches finally arrived?. Communications of the ACM, 58(1), 45-47.

[15] Pearson, J., Robinson, S., & Jones, M. (2015, April). It's about time: Smartwatches as public displays. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (pp. 1257-1266).

[16] https://pecquery.wixsite.com/arduino-passion/copie-de-le-detecteur-a-ultrasons-h-1

[17] https://store.arduino.cc/due

[18] https://store.arduino.cc/arduino-mini-05

[19] https://www.arduino.cc/en/Main/arduinoBoardDiecimila

[20] https://www.gotronic.fr/art-module-arduino-nano-12422.htm.

[21] C. Tavernier, « Arduino applications avancées ». Version Dunod.

[22] X.HINAULT. www.mon-club-elec.fr

[23] http://www.acm.uiuc.edu/sigbot/tutorials/ 2009-11-17-arduino-basics.

[24] S.V.D.Reyvanth, G.Shirish, « PID controller using Arduino ».

[25] http://www.generationrobots.com/fr/ 152-arduino.

[26] http://arduino.cc/en/Main/ArduinoBoardUno

[27] https://forum.arduino.cc/index.php?topic=352904.0

[28] https://www.govtech.com/fs/news/8-Mind-blowing-Uses-of-Wearable-Technology-Seriously.html

[29] Godfrey, A. (2017). Wearables for independent living in older adults: Gait and falls. Maturitas, 100, 16-26.

[30] Li, J., Ma, Q., Chan, A. H., & Man, S. S. (2019). Health monitoring through wearable technologies for older adults: Smart wearables acceptance model. Applied ergonomics, 75, 162-169.

[31] Lee, J., Kim, D., Ryoo, H. Y., & Shin, B. S. (2016). Sustainable wearables: Wearable technology for enhancing the quality of human life. Sustainability, 8(5), 466.

[32] Hurford, R. D. (2009). Types of smart clothes and wearable technology. In *Smart clothes and wearable technology* (pp. 25-44). Woodhead Publishing.

[33] McCann, J., & Bryson, D. (Eds.). (2009). Smart clothes and wearable technology. Elsevier.

[34] https://www.smartgeekwrist.com/advantages-of-smart-watch/

[35] https://gearpatrol.com/2018/10/10/smartwatch-history/

[36] https://www.bloomberg.com/news/articles/2018-01-08/a-concise-history-of-the-smartwatch

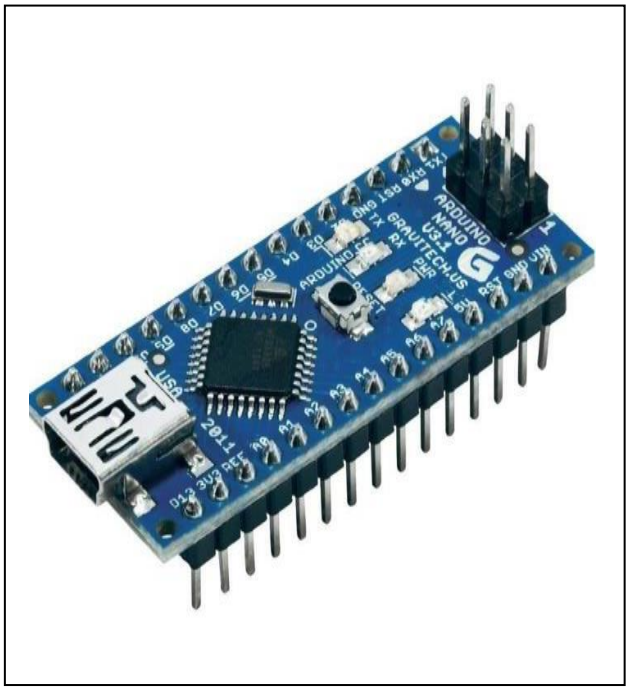
[37] https://techbeacon.com/app-dev-testing/smartwatch-oss-want-do-more-what-means-developers

[38] https://code.tutsplus.com/articles/wearable-development-platforms--cms-30213

[39] https://hackernoon.com/smartwatch-app-development-company-redefining-business-solutions-aa620906185d

[40]https://www.nordicsemi.com/news/2018/05/nrf51822%20supports%20mmt%20swissconnect%20smartwatch%20platform

1. Components used in this project:



Arduino Nano board

Bluetooth HC-06

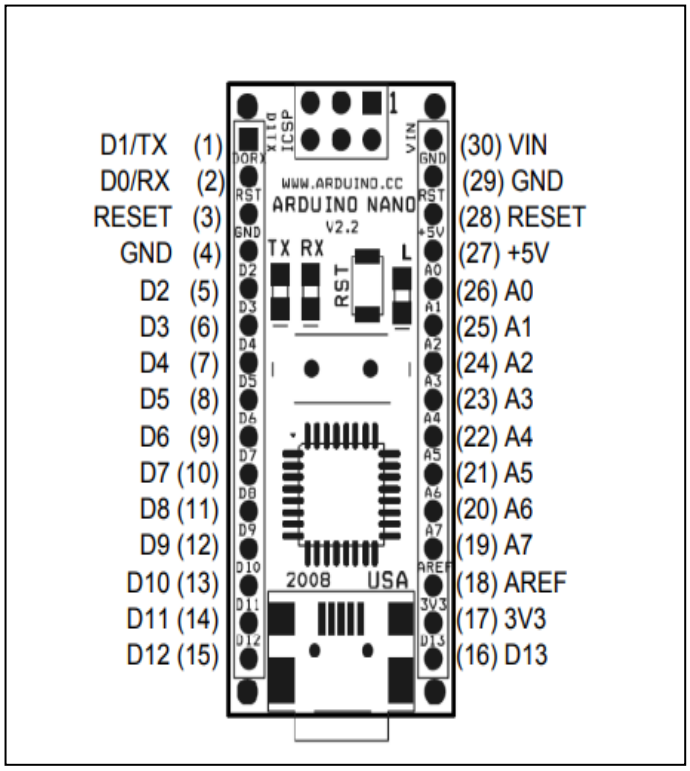
OLED ssd1306

9v Battery

2. Datasheet:

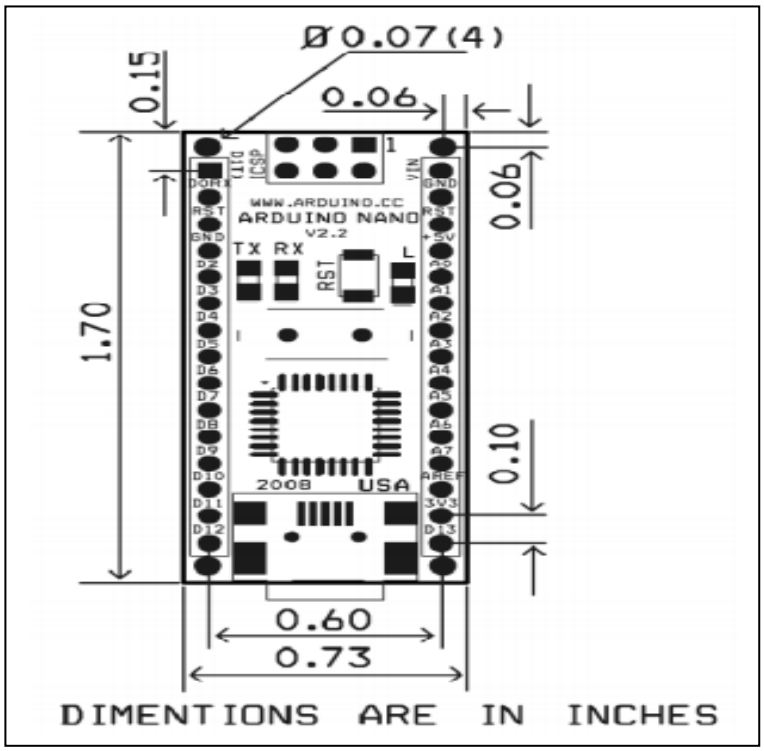
2.1. Arduino Nano board:

Arduino Nano Pin Layout



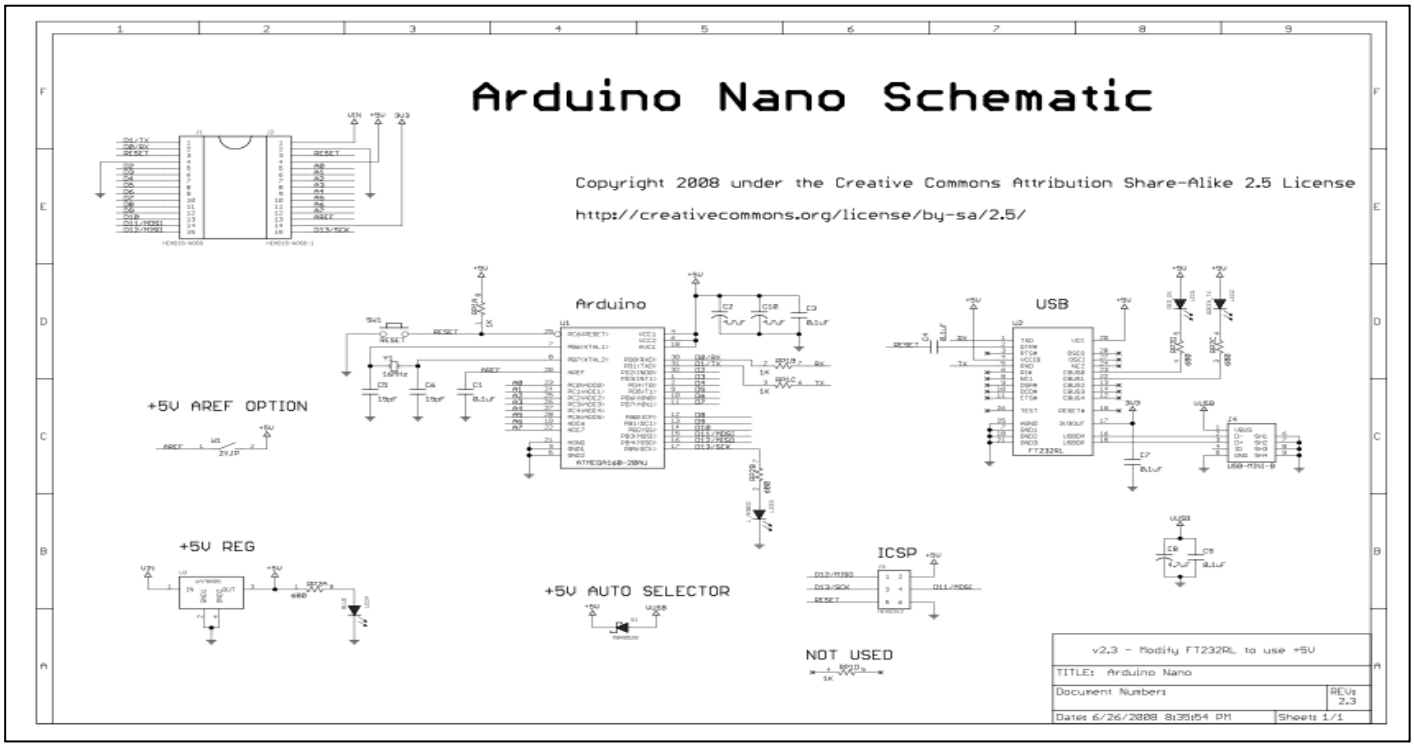
Pin No.	Name	Type	Description
1-2, 5-16	D0-D13	I/O	Digital input/output port 0 to 13
3, 28	RESET	Input	Reset (active low)
4, 29	GND	PWR	Supply ground
17	3V3	Output	+3.3V output (from FTDI)
18	AREF	Input	ADC reference
19-26	A7-A0	Input	Analog input channel 0 to 7
27	+5V	Output or Input	+5V output (from on-board regulator) or +5V (input from external power supply)
30	VIN	PWR	Supply voltage

Arduino Nano Mechanical Drawing



Arduino Nano Bill of Material

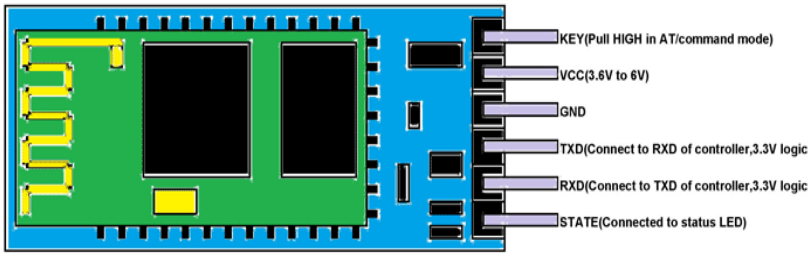
Item Number	Qty.	Ref. Dest.	Description	Mfg. P/N	MFG	Vendor P/N	Vendor
1	5	C1,C3,C4,C7,C9	Capacitor, 0.1uF 50V 10% Ceramic X7R 0805	C0805C104K5RACTU	Kemet	80-C0805C104K5R	Mouser
2	3	C2,C8,C10	Capacitor, 4.7uF 10V 10% Tantalum Case A	T491A475K010AT	Kemet	80-T491A475K010	Mouser
3	2	C5,C6	Capacitor, 18pF 50V 5% Ceramic NOP/COG 0805	C0805C180J5GACTU	Kemet	80-C0805C180J5G	Mouser
4	1	D1	Diode, Schottky 0.5A 20V	MBR0520LT1G	ONSemi	863-MBR0520LT1G	Mouser
5	1	J1,J2	Headers, 36PS 1 Row	68000-136HLF	FCI	649-68000-136HLF	Mouser
6	1	J4	Connector, Mini-B Recept Rt. Angle	67503-1020	Molex	538-67503-1020	Mouser
7	1	J5	Headers, 72PS 2 Rows	67996-272HLF	FCI	649-67996-272HLF	Mouser
8	1	LD1	LED, Super Bright RED 100mcd 640nm 120degree 0805	APT2012SRCPRV	Kingbright	604-APT2012SRCPRV	Mouser
9	1	LD2	LED, Super Bright GREEN 50mcd 570nm 110degree 0805	APHCM2012CGCK-F01	Kingbright	604-APHCM2012CGCK	Mouser
10	1	LD3	LED, Super Bright ORANGE 160mcd 601nm 110degree 0805	APHCM2012SECK-F01	Kingbright	04-APHCM2012SECK	Mouser
11	1	LD4	LED, Super Bright BLUE 80mcd 470nm 110degree 0805	LTST-C170TBKT	Lite-On Inc	160-1579-1-ND	Digikey
12	1	R1	Resistor Pack, 1K +/-5% 62.5mW 4RES SMD	YC164-JR-071KL	Yageo	YC164J-1.0KCT-ND	Digikey
13	1	R2	Resistor Pack, 680 +/-5% 62.5mW 4RES SMD	YC164-JR-07680RL	Yageo	YC164J-680CT-ND	Digikey
14	1	SW1	Switch, Momentary Tact SPST 150gf 3.0x2.5mm	B3U-1000P	Omron	SW1020CT-ND	Digikey
15	1	U1	IC, Microcontroller RISC 16kB Flash, 0.5kB EEPROM, 23 I/O Pins	ATmega168-20AU	Atmel	556-ATMEGA168-20AU	Mouser
16	1	U2	IC, USB to SERIAL UART 28 Pins SSOP	FT232RL	FTDI	895-FT232RL	Mouser
17	1	U3	IC, Voltage regulator 5V, 500mA SOT-223	UA78M05CDCYRG3	TI	595-UA78M05CDCYRG3	Mouser
18	1	Y1	Cystal, 16MHz +/-20ppm HC-49/US Low Profile	ABL-16.000MHZ-B2	Abracon	815-ABL-16-B2	Mouser



2.2. Bluetooth HC -06:

➢ HC-06 Features and Electrical characteristics

- Bluetooth protocol: Bluetooth V2.0 protocol standard
- Power Level: Class2(+6dBm)
- Band: 2.40GHz—2.48GHz, ISM Band
- Receiver sensitivity: -85dBm
- USB protocol: USB v1.1/2.0
- Modulation mode: Gauss frequency Shift Keying
- Safety feature: Authentication and encryption
- Operating voltage range:+3.3V to +6V
- Operating temperature range: -20ºC to +55ºC
- Operating Current: 40mA



Bluetooth HC-06 pinout

- **Pin configuration**

HC-06 module has six pins as shown in the pinout. In them we only need to use four for successfully interfacing the module. Some breakout boards will only leave four output pins only because of this reason.

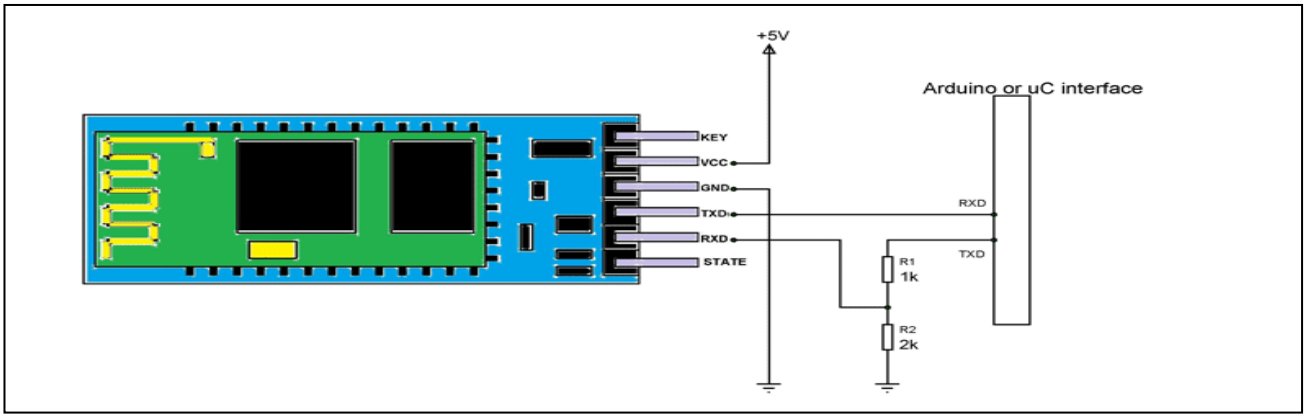
Pin	Name	Function
1	Key	The pin state determines whether the module works in AT command mode or normal mode [High=AT commands receiving mode(Commands response mode), Low or NC= Bluetooth module normally working]
2	Vcc	+5V Positive supply needs to be given to this pin for powering the module
3	Gnd	Connect to ground
4	TXD	Serial data is transmitted by module through this pin (at 9600bps by default), 3.3V logic
5	RXD	Serial data is received by module through this pin (at 9600bps by default),3.3V logic
6	State	The pin is connected to the LED on the board to represent the state of the module

- **Similar Bluetooth Modules:**

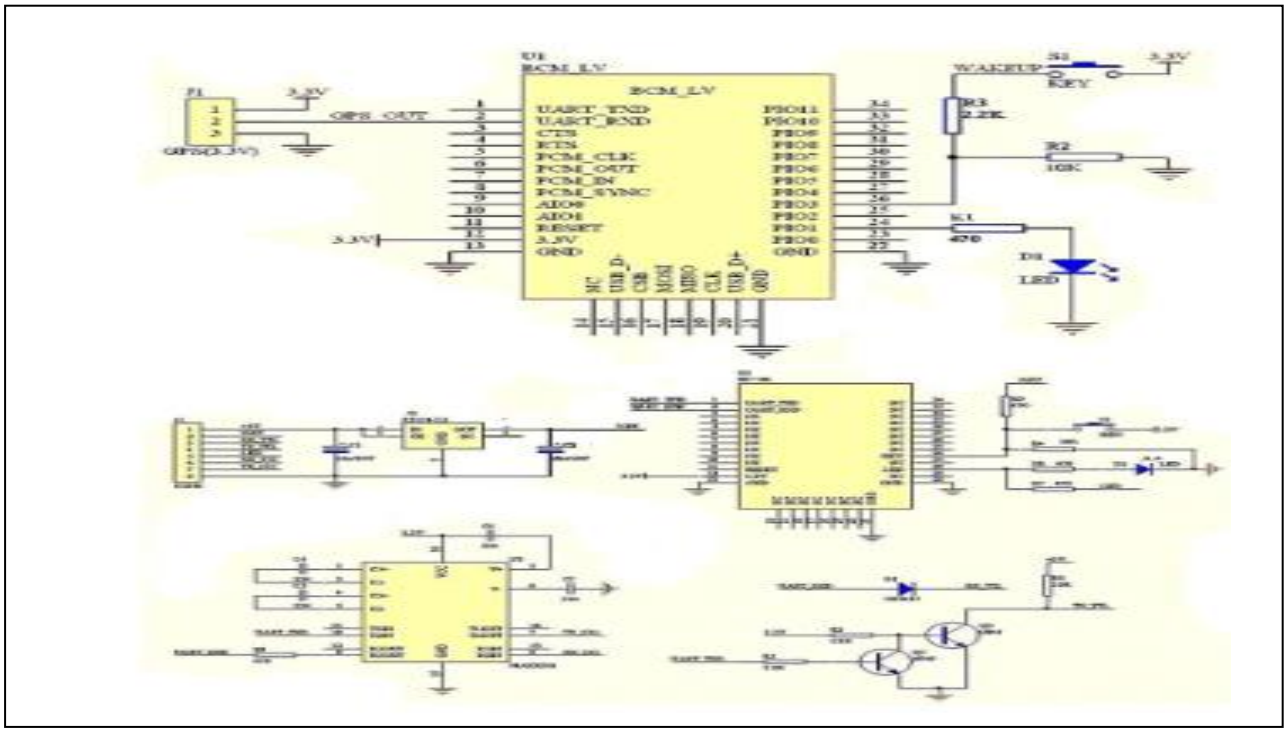
HC-04, HC-02, HC-05, HC-03

- **How to use HC-06 Bluetooth Module:**

The communication with this HC-06 module is done through **UART interface**. The data is sent to the module or received from the module though this interface. So we can connect the module to any microcontroller or directly to PC which has RS232 port (UART interface). A typical interface circuit of the module to an arduino is shown below.



Bluetooth HC-06 schematic module



2.3. OLED ssd1603 display:

This is an OLED monochrome 128x64 dot matrix display module with I2C Interface. Comparing to LCD, OLED screens are way more competitive, which has a number of advantages such as high brightness, self-emission, high contrast ratio, wide viewing angle, wide temperature range, and low power consumption.

- **Features:**
  - ✓ Interface: I2C (3.3V logic level)
  - ✓ Resolution: 128*64
  - ✓ Angle of view: >160 degree
  - ✓ Display color: White Display
  - ✓ Dimension: 0.96inch
  - ✓ Driver IC: SSD1306
  - ✓ Power supply: 3.3V~5VDC
  - ✓ Operating temperature: -20℃~70℃
  - ✓ Application: smart watch, MP3, thermometer, instruments, DIY projects, etc.

OLED display pins configuration

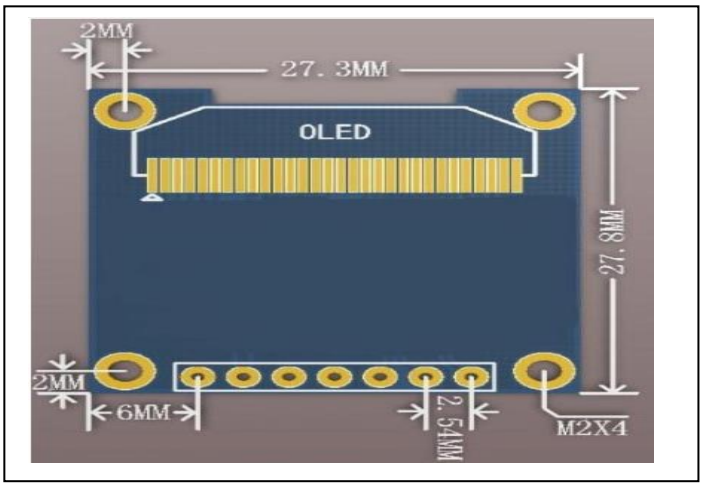
Pin No:	Pin Name:	Description
1	Ground (Gnd)	Connected to the ground of the circuit
2	Supply (Vdd,Vcc,5V)	Can be powered by either 3.3V or 5V
3	SCK (D0,SCL,CLK)	The display supports both IIC and SPI, for which clock is supplied through this pin
4	SDA (D1,MOSI)	This is the data pin of the both, it can either be used for IIC or for SPI
5	RES(RST,RESET)	When held to ground momentarily this pin resets the module
6	DC (A0)	This is command pin, can either be used for SPI or for IIC
7	Chip Select (CS)	Normally held low, used only when more than one SPI device is connected to MCU

Other types of OLED display module

S.No:	Classification	Types
1	Based on Colour	Monochrome(Blue)
		Monochrome(White)
		Yellow/Blue Colour
2	Based on number of Pin	3-Pin (supports only IIC)
		7-Pin (supports IIC and SPI)
3	Based on interface IC	SSD1306
		SSD1331
4	Based on size	0.91" (128×32)
		0.96" (128×64)

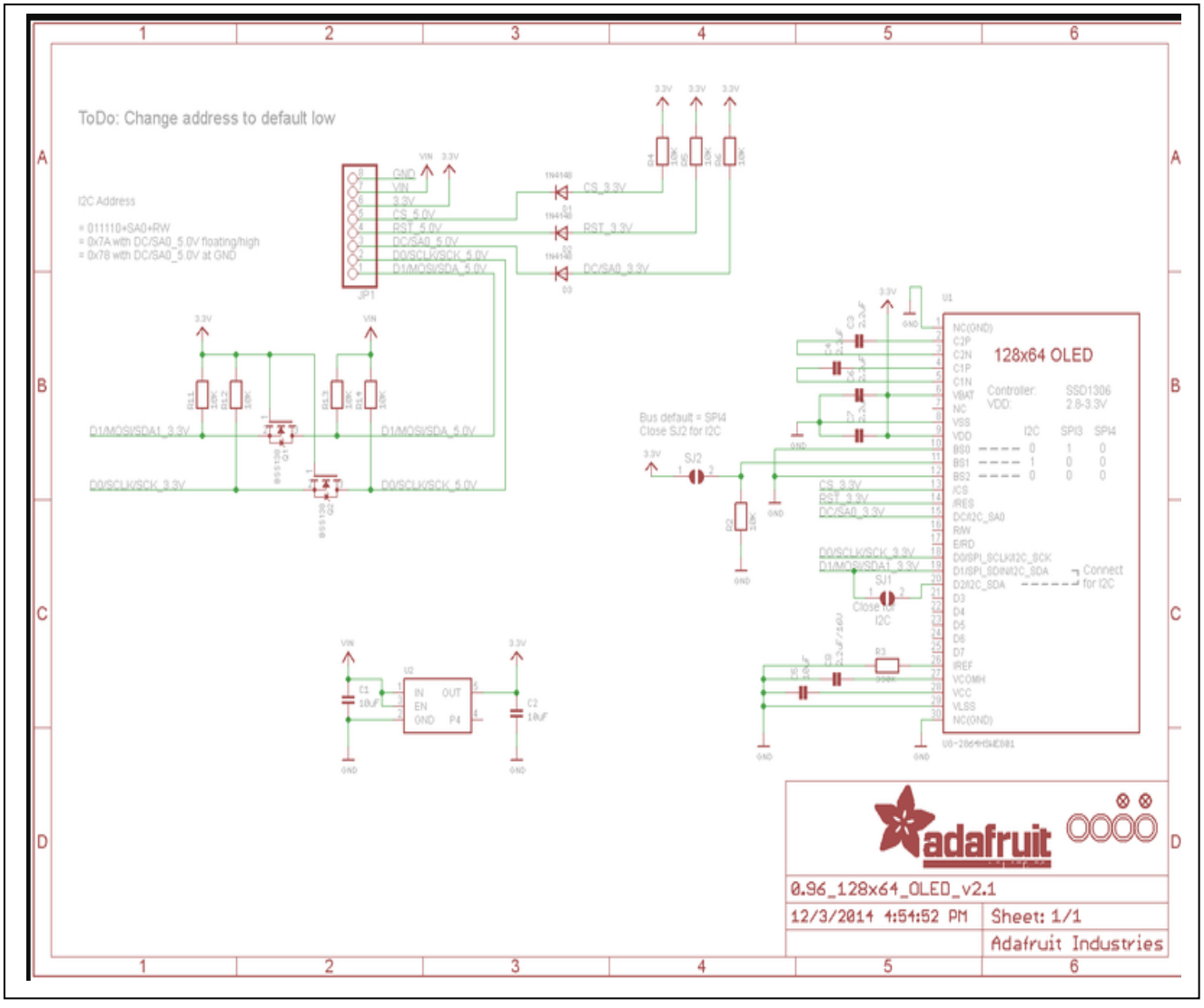
2D-model of Baseboard:

When purchased, the **SSD1306 OLED display** comes with this baseboard. We can use the following dimensional details for our PCB or mounting purposes



OLED 2D module

OLED ssd1603 I2C display schematic



The goal of this project is to design a SmartWatch related to a Smartphone using an android application “Smart-Clock” that we’ve invent , we used in this design an Arduino Nano board after studying all the other boards we used the Nano board because of it suitability especially the size affect, then we identified the other different components necessary for the project, their characteristics, their method of connection and the programming method of each component including the OLED screen, the Bluetooth HC-06 and finally the power supply device, then the controlling program was accomplished by IDE using the C++ language

Keywords: Arduino Nano, Bluetooth HC-06, OLED, IDE

Le but de ce projet est de concevoir une SmartWatch liée à un smartphone Android par Bluetooth, nous avons utilisé dans cette conception une carte Arduino Nano après avoir étudié toutes les autres cartes, nous avons utilisé la carte Nano en raison de sa pertinence, en particulier l’effet de la taille , puis nous avons identifié les autres différents composants nécessaires au projet, leurs caractéristiques, leur mode de connexion et la méthode de programmation de chaque composant dont l'écran OLED, le Bluetooth HC-06 et enfin le dispositif d'alimentation, en plus des composants nous avons installé l'application ‘Smart clock’ qui a aidé à surveiller notre montre, puis le programme de contrôle a été accompli par IDE en utilisant le language C ++

Mots-clés: Arduino Nano, Bluetooth HC-06, OLED, IDE

الهدف من هذا المشروع هو تصميم ساعة ذكية تتعلق بهاتف ذكي يعمل بنظام Android بواسطة Bluetooth ، وقد استخدمنا في هذا التصميم لوحة Arduino Nano بعد دراسة جميع اللوحات الأخرى , استخدمنا لوحة Nano بسبب ملاءمتها خاصة تأثير الحجم ، ثم حددنا المكونات المختلفة الأخرى اللازمة للمشروع ، وخصائصها ، وطريقة اتصالها وطريقة البرمجة لكل مكون بما في ذلك شاشة OLED ، و Bluetooth HC-06 وأخيرًا جهاز تزويد الطاقة ، إلى جانب المكونات التي قمنا بتثبيت تطبيق "Smart clock" التي ساعدت في التحكم بساعتنا ، ثم تم إنجاز برنامج التحكم بواسطة IDE باستخدام لغة C ++

الكلمات المفتاحية : اردوينو نانو ، بلوتوث HC-06، شاشة OLED، IDE