Université Mohamed Khider de Biskra
Faculté des Sciences et de la Technologie
Département de Génie Electrique

MÉMOIRE DE MASTER

Sciences et Technologies
Electronique
Electronique des Systèmes embarqués

Réf. : .....

Présenté et soutenu par :
Gaouaoui Yahya

Le : .....

Pointeuse biométrique à base de Raspberry Pi

Jury :

M.	Boukerdine Salaheddine	MAA	Université de Biskra	Président
M.	Benelmir Okba	MCB	Université de Biskra	Examinateur
M.	Hezabra Adel	MAA	Université de Biskra	Rapporteur

Année universitaire : 2019 – 2020

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'enseignement Supérieur et de la recherche scientifique

Université Mohamed Khider Biskra
Faculté des Sciences et de la Technologie
Département de Génie Electrique
Filière : Electronique
Option: Electronique des systèmes embarqués

Mémoire de Fin d'Etudes
En vue de l'obtention du diplôme:

MASTER

Thème

Pointeuse biométrique à base de Raspberry Pi

Présenté par :
Gaouaoui Yahya

Avis favorable de l'encadreur :
Mr .Hezabra Adel

Avis favorable du Président du Jury
Mr . Boukerdine Salaheddine

Cachet et signature

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'enseignement Supérieur et de la Recherche scientifique

Université Mohamed Khider Biskra
Faculté des Sciences et de la Technologie
Département de Génie Electrique
Filière : Electronique
Option: Electronique des systèmes embarqués

Thème :

Pointeuse biométrique à base de Raspberry Pi

Proposé par : Mr .Hezabra Adel
Dirigé par : Mr .Hezabra Adel

ملخص

الهدف من هذا المشروع هو انجاز نظام تسجيل الحضور بيومتري يعتمد أساسا على صورة الوجه وبصمة الإصبع باستخدام لوحة تحكم راسبيري باي، بحيث يتم أخذ بصمة إصبع لكل موظف باستخدام لاقط بصمة رقميا و التقاط مجموعة من الصور باستخدام الكاميرا،كما يتم منح بعض الموظفين الخاصين رمز دخول. الغرض من استخدام طرق مختلفة لتسجيل الحضور هو زيادة أمان النظام وتجنب المشاكل التي تواجه عملية تسجيل الحضور او تجنب العدوى باستخدام الكاميرا في حالة انتشار فيروس معدي عن طريق اللمس مثلا. ويتم تنظيم أوقات دخول وخروج الموظفين وحساب أوقات العمل والغياب من خلال التطبيق الذي قمنا بتصميمه، والذي من خلاله استخراج تقرير شهريً مفصل عن ساعات العمل لكل موظف و بدقة.

Abstract

The objective of this project is to achieve a biometric attendance system that mainly depends on a face image and a fingerprint using a Raspberry Pi control panel, so that a fingerprint is taken for each employee using a digital fingerprint sensor or a pictures is taken using the camera, and some private employees are given an access code. The purpose of using different methods of recording attendance is to increase the system security and avoid problems facing the attendance registration process or avoid infection by using a camera in the event of an infectious virus spread by touch, for example. The entry and exit times of the employees and the calculation of working and absence times are organized through the application that we designed, through which a detailed monthly report on the working hours of each employee can be extracted accurately.

Dedicate

Praise be to God for the blessings, I dedicate this modest work accompanied by a deep love:

To my support in my life my dear mother

To my beautiful sister ***Najah*** *and my brothers* ***Abdallah*** *and* ***Lamine***

To my family, relatives, and those who have given me love and trust

To everyone who helped me and stood by me, especially ***Hashani Salim***

To my teachers who taught me, colleagues who accompanied me, and my friends who gave me honesty

Acknowledgements

Praise be to God for the blessings and i thank God for giving me all the patience, knowledge, courage and wisdom to do things in my life and my academic career.

I thank my supervisor Mr. ***Hezabra Adel*** *for his trouble with me, the advice he gave me, and the complete supervision that honored me.*

*I thank all the professors and teachers of the faculty of Science and Technology for the effort they have made to educate me. I also thank all the administration workers. I extend a special thanks to all the workers in the laboratory, especially Mr.****Laadjel Hamza****, and I also thank Mr.* ***Menadi Samir****.*

I thank the jury who were excellent teachers

List of tables:

List of figures:

List of abbreviation:

DNA	**D**eoxyribo**N**ucleic **A**cid
CMOS	**C**omplementary **M**etal–**O**xide–**S**emiconductor
CCD	**C**harge **C**oupled **D**evice
ATM	**A**utomated **T**eller **M**achine
PDA	**P**ersonal **D**igital **A**ssistant
BioID	**B**iometric **I**dentification
WiFi	**W**ireless **F**idelity
USB	**U**niversal **S**erial **B**us
RFID	**R**adio-**F**requency **I**dentification
ZKBioSecurity	Biometric system web based security platform developed by ZKTeco
ZKBioTime	ZKTeco Attendance Management Software User Manual
ZKFace	Face algorithm of biometric access control terminal by ZKTeco
ZKFinger	Fingerprint algorithm of biometric access control terminal by ZKTeco
ZKPalmVein	Palm vein algorithm of biometric access control terminal by ZKTeco
UI	**U**ser **I**nterface
ARM	**A**corn **R**ISC **M**achine
IoT	**I**nternet **o**f **T**hings
OS	**O**perating **S**ystem
RISC	**R**educed-**I**nstruction-**S**et **C**omputing
GUI	**G**raphical **U**ser **I**nterface
NOOBS	**N**ew **O**ut **O**f **B**ox **S**oftware
GPIO	**G**eneral **P**urpose **I**nput/**O**utput
HDMI	**H**igh-**D**efinition **M**ultimedia **I**nterface
RGB	**R**ed, **G**reen and **B**lue
DPI	**P**arallel **D**isplay **I**nterface
GPU	**G**raphics **P**rocessing **U**nit
LCD	**L**iquid **C**rystal **D**isplay
DSI	**D**isplay **S**erial **I**nterface
PCB	**P**rinted **C**ircuit **B**oard
GND	**G**round

FKMS	**F**requency and **K**ey **M**anagement **S**ystem
DRM	**D**irect **R**endering **M**anager
KMS	**K**ernel **M**ode **S**etting
Cmd	**C**ommand
PWM	**P**ulse **W**idth **M**odulation
I2C	**I**nter-**I**ntegrated **C**ircuit
SOC	**S**ystem **O**n a **C**hip
GHz	**G**igahertz
V	**V**olt
SDRAM	**S**ynchronous **D**ynamic **R**andom **A**ccess **M**emory
OTP	**O**ne-**T**ime **P**rogrammable
ROM	**R**ead-**O**nly **M**emory
SD	**S**ecure **D**igital
NAND	**N**OT **AND**
SPI	**S**erial **P**eripheral **I**nterface
OTG	USB **O**n-**T**he-**G**o
LAN951x	**L**ocal **A**rea **N**etwork (USB hub and 10/100 Ethernet controller)
MSD	**M**icro **S**ecure **D**igital
DHCP	**D**ynamic **H**ost **C**onfiguration **P**rotocol
TFTP	**T**rivial **F**ile **T**ransfer **P**rotocol
PC	**P**ersonal **C**omputer
VID	**V**isual **I**dentification
PID	**P**roportional-**I**ntegral-**D**erivative
OTG	**O**n-**T**he-**G**o
FAT	**F**ile **A**llocation **T**able
MMC	**M**ulti **M**edia **C**ard
SMSC	**S**ystem **M**aintenance **S**upport **C**enter
UART	**U**niversal **A**synchronous **R**eceiver-**T**ransmitter
API	**A**pplication **P**rogramming **I**nterface
SDA	**S**erial **D**ata **L**ine
SCL	**S**erial **C**lock **L**ine
CS	**C**hip **S**elect

MOSI	**M**aster **O**utput, **S**lave **I**nput
MISO	**M**aster **I**nput, **S**lave **O**utput
SCLK	**S**erial **C**lock
CSI	**C**amera **S**erial **I**nterface
OCR	**O**ptical **C**haracter **R**ecognition
X-ray	Energetic high-frequency electromagnetic radiation
DIP	**D**igital **I**mage **P**rocessing
SONAR	**S**ound **N**avigation and **R**anging
LBPH	**L**ocal **B**inary **P**attern **H**istogram
FTIR	**F**rustrated **T**otal **I**nternal **R**eflection
LED	**L**ight **E**mitting **D**iode
ANSI	**A**merican **N**ational **S**tandard for **I**nformation **S**ystems
NIST	**N**ational **I**nstitute of **S**tandards and **T**echnology
ITL	**I**nformation **T**echnology **L**aboratory
ACN	**A**ccess **C**ode **N**umber
TTL	**T**ransistor-**T**ransistor **L**ogic
DC	**D**irect **C**urrent
MCU	**M**icrocontroller **U**nit
RST	**D**evice **R**ese**t**
TX	Asynchronous serial data transmit
RX	Asynchronous serial data receive
CTS	**C**lear **T**o **S**end
RTS	**R**eady **T**o **S**end
DSR	**D**ata **S**et **R**eady
DTR	**D**ata **T**erminal **R**eady
DCD	**D**ata **C**arrier **D**etect
RI	**R**ing **I**ndicator
ID	**I**dentifier

Contents:

General Introduction

In the past, the process of registering attendance was tiring and time consuming, as registration was manually with all people and we do not forget about fraud and lack of attendance at the time. This problem has disappeared with the development of technology, as there are many systems of attendance registration, these systems are placed anywhere in the workplace and takes only seconds to record attendance. The development of biometric sensors has increased the ease and accuracy of the work of these systems, and due to the difference in the biometric features of people, there are many biometric sensors currently in use, including: the face sensor, fingerprint, sound, iris and signature. The most widespread and used systems are those that use the fingerprint sensor, but there is a problem when the user's finger is damaged, In this work we designed a system that records attendance using three methods, by recognizing the face, fingerprint or using code to avoid problems that may happen to the attendance registration process.

This thesis is organized as follows:

- Chapter 1: This chapter talks about the biometric features of a person and the various biosensors that exist. We also learn about the various systems made and currently used, and how these systems work.
- Chapter 2: This chapter talks about the used control unit which is Raspberry Pi and some general information about these units, then we mention the basic information that must be known to use the Raspberry Pi 3 model B unit.
- Chapter 3:This chapter talks about digital image processing and some basic information for this process, because the process of recognizing the face and the fingerprint by taking a picture and then extracting the information in the image and recognizing it through the digital image processing process. As for the face recognition process, we used a **Haar** classifier to detect the face. As for the fingerprint recognition process, we talked about two ways to extract the fingerprint features.
- Chapter 4:This chapter talks about the system that we designed for the attendance registration process, which consists of two parts hard and soft, where we will talk about the components of hard part and the way it works and link it with the control unit, also we will talk about the way to use soft part.

Chapter 1:

Biometric Attendance System

I.1. Introduction:

Attendance management system is a system that records the time of workers entering and leaving according to a specific time and during specific days. These modern systems are easy to use and avoid all kinds of fraud, they work to identify people through the various features that humans carry, their use takes only seconds or once a person crosses in front of them. These modern systems reduced many of the problems they were facing in the past, where they were recording with the same person in the papers, then it developed a little and they started to register in one machine with the papers, however this process takes a lot of work time. Then it evolved over time with the discovery of biometric sensors, which helped a lot and are the basis of these machines. The accuracy of the biometric systems lies in the accuracy of the biometric sensors. Now there are various of biometric attendance system which use one or many biometric sensors

I.2. Biometric system:

A biometric system is a system that allows the recognition of a certain characteristic of an individual using mathematical algorithms and biometric data. There are several uses of biometric systems [1]. The word biometrics is derived from the Greek words bio and metric. Where bio means life and metric means to measure. Biometrics are used to identify his or her physical and behavioral characteristics of a person. This method of identification is chosen over traditional methods, including PIN numbers and passwords for its exactness and case sensitiveness. Based on the designing, this system can be used as an identification system or authentication system. These systems are divided into various types which include vein pattern, fingerprints, hand geometry, DNA, voice pattern, iris pattern, signature dynamics and face detection [2].

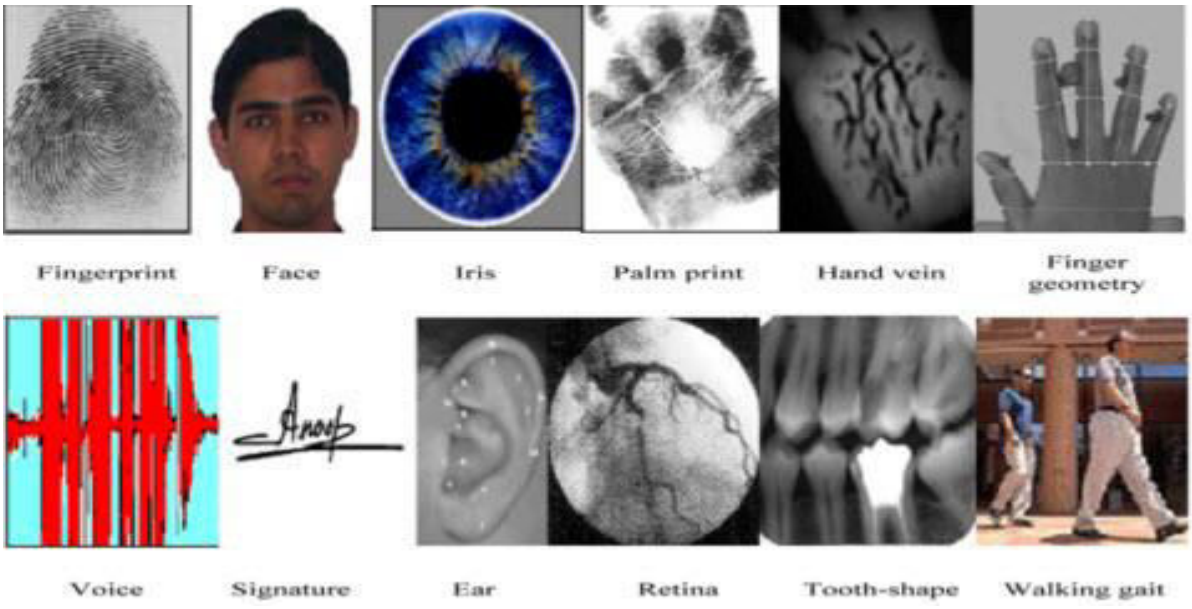


Fig. I.1. Physical and behavioural characteristics of a person [2].

I.3. Biometric Sensor [2]:

A biometric sensor is a transducer that changes a biometric treat of a person into an electrical signal. Biometric treats mainly include biometric fingerprint reader, iris, face, voice, etc. Generally, the sensor reads or measures light, temperature, speed, electrical capacity and other types of energies. Different technologies can be applied to get this conversation using sophisticated combinations, networks of sensors and digital cameras. Every biometric device requires one type of sensor. The biometrics applications mainly include: used in a high definition camera for facial recognition or in a microphone for voice capture. Some biometrics is specially designed to scan the vein patterns under your skin. Biometric sensors are an essential feature of identity technology.

I.4. Types of biometric sensor and access control [2]:

Biometric sensors or access control systems are classified into two types such as Physiological Biometrics and Behavioral Biometrics. The physiological biometrics mainly include face recognition, fingerprint, hand geometry, Iris recognition, and DNA. Whereas behavioral biometrics include keystroke, signature and voice recognition.

I.4.1. Fingerprint sensing[2]:

Fingerprint Recognition includes taking a fingerprint image of a person and records its features like arches, whorls, and loops along with the outlines of edges, minutiae, and furrows. Matching of the Fingerprint can be attained in three ways, such as minutiae, correlation, and ridge

- Minutiae based fingerprint matching stores a plane includes a set of points and the set of points are corresponding in the template and the i/p minutiae.
- Correlation-based fingerprint matching overlays two fingerprint images and the association between equivalent pixels is calculated.
- Ridge feature-based fingerprint matching is an innovative method that captures ridges, as minutiae-based fingerprint capturing of the fingerprint images is difficult in low quality.

Fig. I.2. Fingerprint sensor [2].

To capture the fingerprints, present methods employ optical sensors that use a CMOS image sensor or CCD, solid-state sensors work on the principle of transducer technology using thermal, capacitive, piezoelectric sensors or electric field, or ultrasound

sensors work on echography in which the sensor sends acoustic signals through the transmitter near the finger and captures the signals in the receiver. The scanning of the fingerprint is very stable and also reliable. It safeguards entry devices for building door locks and access of computer network are becoming more mutual. At present, a small number of banks have initiated using fingerprint readers for approval at ATMs.

I.4.2. Face sensing[2]:

A face recognition system is one type of biometric computer application that can identify or verify a person from a digital image by comparing and analyzing patterns. These biometric systems are used in security systems. Present facial recognition systems work with face prints and these systems can recognize 80 nodal points on a human face. Nodal points are nothing but endpoints used to measure variables on a person's face, which includes the length and width of the nose, cheekbone shape, and eye socket depth.

Fig. I.3. Face recognition system [2].

Face recognition systems work by capturing data for the nodal points on a digital image of a person's face and resulting data can be stored as a face print. When the conditions are favorable, these systems use face prints to identify accurately. Currently, these systems focus on smart phone applications which include personal marketing, social networking, and image tagging purposes. Social sites like **F**acebook uses software for face recognition to tag the users in photographs. This software also increases marketing personalization. For instance, billboards have been designed with integrated software that recognizes the ethnicity, gender and estimated age of onlookers to deliver targeted marketing.

I.4.3. Iris sensing [2]:

Iris recognition is one type of bio-metric method used to identify the people based on single patterns in the region of ring-shaped surrounded the pupil of the eye. Generally, the iris has a blue, brown, gray or green color with difficult patterns that are noticeable upon close inspection.

Fig. I.4. Iris recognition system [1].

I.4.4. Voice sensing [2] :

Voice recognition technology is used to produce speech patterns by combining behavioral and physiological factors that can be captured by processing speech technology. The most important properties used for speech authentication are nasal tone, fundamental frequency, inflection, cadence. Voice recognition can be separated into different categories based on the kind of authentication domain, such as a fixed text method, in the text-dependent method, the text-independent method, and conversational technique.

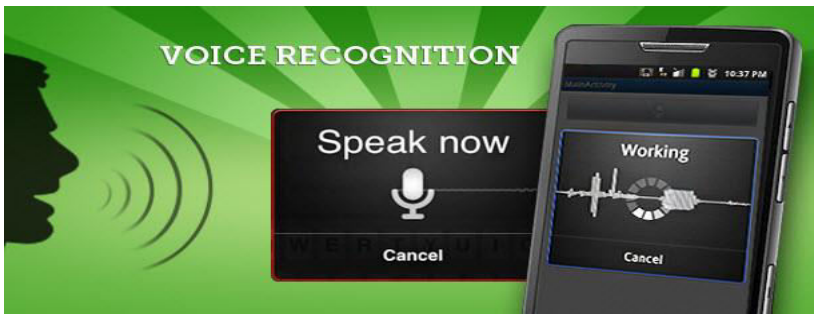


Fig. I.5. Voice recognition system [1].

I.4.5. Signature sensing [2]:

Signature recognition is one type of biometric method used to analyze and measure the physical activity of signing like the pressure applied, stroke order and speed. Some biometrics are used to compare visual images of signatures. Signature recognition can be operated in two different ways, such as static and dynamic.

Fig. I.6. Signature recognition system [1].

In static mode, consumers write their signature on paper, digitize it through a camera or an optical scanner. This system identifies the signature examining its shape.

In dynamic mode, consumers write their signature in a tablet which is digitized, which obtains the signature in real-time. Another option is gaining by means of stylus-operated PDAs. Some biometrics also operate with smart-phones with a capacitive screen, where consumers can sign using a pen or a finger. This type of recognition is also known as “on-line”.

I.5. architecture of a biometric system [1]:

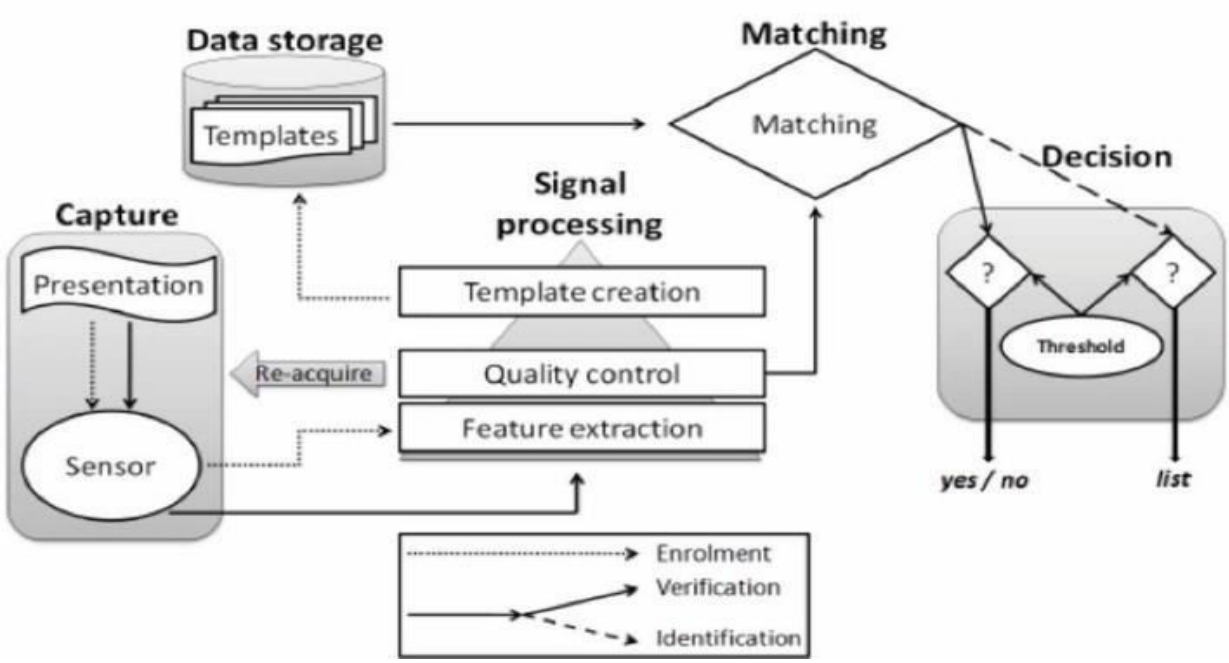


Fig. I.7. presents the architecture of a biometric system [1].

- The capture module that represents the entry point of the biometric system and consists in acquiring the biometric data in order to extract a digital representation.

- The module of signal processing makes it possible to optimize the processing time and the digital representation acquired in the enrollment phase in order to optimize the processing time of the verification phase and the identification.
- The storage module that contains the biometric templates of the system enrollees.
- The matching module that compares the data extracted by the extraction module with the data of the registered models and determines the degree of similarity between the two biometric data.
- The decision module that determines whether the similarity index returns through the matching module is sufficient to make a decision about the identity of an individual.

I.6. Difference between biometric authentication and identification[3]:

Authentication... Identification... Sometime it is very difficult to understand the difference between these two words and actions they perform.

Authentication and identification are closely connected with verification and authorization.

Biometric Identification is the automatic identification of living individuals by using their physiological and behavioral characteristics, negative identification can only be accomplished through biometric identification, if a pin or password is lost or forgotten it can be changed and reissued but a biometric identification cannot.

For example, there is a database where all the photos of users are collected. Suddenly, somebody comes to you and greets you. You want to know who it is, and put the picture of this person to the system. The system is looking for the match. When the match is found the system represents the full information about this person.

Verification means verifying person's identity. A guy comes to you and tells his name, Bill. You take the picture of Bill and put it to the system. The system finds the Bill's file and tries to match the pictures. If the result is positive the system indicates that this guy is really Bill, is negative it indicates that it is not Bill.

Authentication is the same as verification, its task to verify if the user is actually who he claims to be.

Authorization means whether the user has a right to access to the system. In practice it looks like the same as if you come to the cinema. You should buy a ticket, because you know that a person who checks the tickets will not allow you to see the movie without a ticket. There is no identification or verification process.

I.7. Different types of biometric attendance system:

I.7.1. Fingerprint recognition system [4]:

D2S, a time attendance tailored for desk and counter. The fingerprints are easily read at the top of the terminal, making your attendance more convenient. With high sensitivity sensor BioID, faster and more accurate verification, deliver a new experience. The built-in detachable battery is over long 12 days standby, 8000 times punching in on a single charge.

Fig. I.8. A time attendance tailored for desk and counter [4].

I.7.2. face recognition system[4]:

VF380 is a facial recognition terminal for time, attendance and access control which can connect with 3rd party electric lock, door sensor and exit button, etc. With the latest facial recognition algorithm and cutting-edge technology, it can save 3,000 face templates without dividing into groups. The product is able to communicate via WiFi, Ethernet, USB (host), so to ensure a smooth connection and data transfer. All the settings can be easily done on a 2.8-inch capacitive touch screen. It has been elaborately designed and finely processed to match your office perfectly.

Fig. I.9. A facial recognition terminal for time & attendance and access (VF380) [4].

I.7.3. Mobile machine (androide version)[4] :

The trend of Visible Light Facial Recognition technology have brought user experience of biometric technology to a new height. As one of the biometric industry leaders, ZKTeco launched the second-generation facial recognition terminal–Horus series, named after an Egyptian god, who has the legendary “all-seeing eye” that can clearly observe everything. Horus is one of the most advanced Access Control, time and

attendance terminal existing in the market, with incredibly compact size (almost the same size as iPhone XS max) and powerful facial recognition technology offering up to 3 meters recognition distance, ±30 degree pose angle tolerance, high anti-spoof ability, support on plentiful communication protocols (Wi-Fi, 3G, 4G, Bluetooth) and worldwide network setting, optional fingerprint and RFID card modules, up to 10,000 facial templates capacity, and is compatible with the all-in-one security, time attendance platform ZKBioSecurity and ZKBioTime.

Fig. I.10. Visible Light Facial Recognition [4].

I.7.4. Attendance with different biometric sensors[4] :

PFace202 multi-biometric time, attendance and access control terminal supports up to 600 palm templates, 1200 face templates, 2000 fingerprint templates and 10000 cards (optional).

With ZKTeco latest hardware platform and ZKFace, ZKFinger, ZKPalmVein algorithm, it offers brand new UI and user-friendly operation interface to provide smooth user experience.

Fig. I.11. PFace202 multi-biometric time & attendance and access control [4].

I.8. advantage of biometric attendance system [5]:

There are many advantage of biometric attendance system, we mention some of them:

- Avoid crowding and wasting time
- Ease of use and fast (it make seconds to put your finger, card or other thing to attend)
- The possibility of conducting studies to know the behavior and discipline of workers
- Avoid fraud(Direct registration in the machine without the presence of a person to register you and the inability to defraud the machine)
- benefit from human biometrics(More than 10,000 workers are registered)
- economic (You can use the card throughout the year instead of using papers every week or month, or using only the face or the fingerprint)

I.9. Conclusion:

The attendance recording machine is a device that records the time of the workers, attendance in a short time and is different according to the nature of the work, for example the mobile device is intended for constantly moving workers and the fixed machine is intended for workers inside an organization, also it varies according to the sensors used which are the most important part of the device, and the use of sensors varies depending on the necessity of using it, for example, if the establishment is very sensitive, sometimes several advanced sensors are used. There are 5 types of known biometric sensors, which are iris, face, fingerprint, voice and signature. The way these devices work is simple as the sensor takes a sample from the person and an algorithm processes it and extracts its features, then identifies the person by comparison or matching with the database and then records his presence, if he is not registered, then after the process of processing and extracting the features it is added to the database.

Chapter 2:

Raspberry Pi

II.1. Introduction:

We choose Raspberry Pi as a control unit to create an embedded system, that for the advantages of Raspberry Pi .

The Raspberry Pi is an ARM processor single board nanocomputer designed by professors from the computer science department of the University of Cambridge as part of the Raspberry Pi foundation [6].

Raspberry Pi is popularly used for real time Image/Video processing, IoT based applications and Robotics applications. Raspberry Pi Foundation officially provides Debian based Raspbian OS. Also, they provide NOOBS OS for Raspberry Pi. We can install several Third-Party versions of OS like Ubuntu, Archlinux, RISC OS, Windows 10 IOT Core, etc. Raspbian OS is official Operating System available for free to use. This OS is efficiently optimized to use with Raspberry Pi. Raspbian have GUI which includes tools for Browsing, Python programming, office, games, etc [7].

In this chapter we talk about general information of Raspberry Pi , we start with camera module, then we look at the power supply , as we get to know General Purpose Input/Output pins on the Raspberry Pi, for display we mention parallel display interface and how to use Raspberry Pi Touch Display , also how know the address of the peripherals ,without forgot Frequency management and thermal control , to use the Raspberry Pi we must know the different boot modes, finally we talk about Raspberry Pi 3 model B.

II.2. Camera Module [8]:

The Raspberry Pi Camera Module is an official product from the Raspberry Pi Foundation. The original 5-megapixel model was released in 2013, and an 8-megapixel Camera Module v2 was released in 2016. For both iterations, there are visible light and infrared versions.

II.3. Power Supply [8]:

The power supply requirements differ by Raspberry Pi model. All models require a 5.1V supply, but the current supplied generally increases according to model. All models up to the Raspberry Pi 3 require a microUSB power connector, whilst the Raspberry Pi 4 uses a USB-C connector.

Exactly how much current (mA) the Raspberry Pi requires is dependent on what you connect to it. The following table gives various current requirements.

Product	Recommended PSU current capacity	Maximum total USB peripheral current draw	Typical bare-board active current consumption
Raspberry Pi Model A	700mA	500mA	200mA
Raspberry Pi Model B	1.2A	500mA	500mA
Raspberry Pi Model A+	700mA	500mA	180mA
Raspberry Pi Model B+	1.8A	600mA/1.2A (switchable)	330mA
Raspberry Pi 2 Model B	1.8A	600mA/1.2A (switchable)	350mA
Raspberry Pi 3 Model B	2.5A	1.2A	400mA
Raspberry Pi 3 Model A+	2.5A	Limited by PSU, board, and connector ratings only.	350mA
Raspberry Pi 3 Model B+	2.5A	1.2A	500mA
Raspberry Pi 4 Model B	3.0A	1.2A	600mA
Raspberry Pi Zero	1.2A	Limited by PSU, board, and connector ratings only	100mA
Raspberry Pi Zero W/WH	1.2A	Limited by PSU, board, and connector ratings only.	150mA

Table. II.1.power supply of different Raspberry Pi [8].

Raspberry Pi have developed their own power supplies for use with all models. These are reliable, use heavy gauge wires and are reasonably priced.

For Raspberry Pi 0-3, we recommend our 2.5A micro USB Supply. For Raspberry Pi 4, we recommend our 3A USB-C Supply

The power requirements of the Raspberry Pi increase as you make use of the various interfaces on the Raspberry Pi. The GPIO pins can draw 50mA safely, distributed across all the pins; an individual GPIO pin can only safely draw 16mA. The HDMI port uses 50mA, the camera module requires 250mA, and keyboards and mice can take as little as 100mA or over 1000mA! Check the power rating of the devices you plan to connect to the Pi and purchase a power supply accordingly.

If you need to connect a USB device that will take the power requirements above the values specified in the table above, then you must connect it to an externally-powered USB hub.

II.4. General Purpose Input/Output pins on the Raspberry Pi:

II.4.1. GPIO pads [8]:

The GPIO connections on the BCM2835 package are sometimes referred to in the peripherals data sheet as "pads" a semiconductor design term meaning 'chip connection to outside world'.

The pads are configurable CMOS push-pull output drivers/input buffers. Register-based control settings are available for:

- Internal pull-up / pull-down enable/disable
- Output drive strength
- Input Schmitt-trigger filtering

II.4.2. Power-on states [8]:

All GPIO pins revert to general-purpose inputs on power-on reset. The default pull states are also applied, which are detailed in the alternate function table in the ARM peripherals datasheet. Most GPIOs have a default pull applied.

II.4.3. Interrupts [8]:

Each GPIO pin, when configured as a general-purpose input, can be configured as an interrupt source to the ARM. Several interrupt generation sources are configurable:

- Level-sensitive (high/low)
- Rising/falling edge
- Asynchronous rising/falling edge

Level interrupts maintain the interrupt status until the level has been cleared by system software (e.g. by servicing the attached peripheral generating the interrupt).

The normal rising/falling edge detection has a small amount of synchronization built into the detection. At the system clock frequency, the pin is sampled with the criteria for generation of an interrupt being a stable transition within a three-cycle window, i.e. a record of '1 0 0' or '0 1 1'. Asynchronous detection by passes this synchronization to enable the detection of very narrow events.

II.4.4. Alternative functions [8]:

Almost all of the GPIO pins have alternative functions. Peripheral blocks internal to BCM2835 can be selected to appear on one or more of a set of GPIO pins, for example the I2C busses can be configured to at least 3 separate locations. Pad control, such as drive strength or Schmitt filtering, still applies when the pin is configured as an alternate function.

II.5. DPI (Parallel Display Interface) [8]:

An up-to-24-bit parallel RGB interface is available on all Raspberry Pi boards with the 40 way header (A+, B+, Pi2, Pi3, Zero) and Compute Module. This interface allows parallel RGB displays to be attached to the Raspberry Pi GPIO either in RGB24 (8 bits for red, green and blue) or RGB666 (6 bits per color) or RGB565 (5 bits red, 6 green, and 5 blue).

This interface is controlled by the GPU firmware and can be programmed by a user via special config.txt parameters and by enabling the correct Linux Device Tree overlay.

II.6. Raspberry Pi Touch Display [8]:

The Raspberry Pi Touch Display is an LCD display which connects to the Raspberry Pi through the DSI connector. In some situations, it allows for the use of both the HDMI and LCD displays at the same time (this requires software support).

II.6.1. Board support [8]:

The DSI display is designed to work with all models of Raspberry Pi, however early models that do not have mounting holes (the Raspberry Pi 1 model A and B) will require additional mounting hardware to fit the HAT-dimensioned standoffs on the display PCB.

II.6.2. Physical Installation [8]:

The following image shows how to attach the Raspberry Pi to the back of the Touch Display (if required), and how to connect both the data (ribbon cable) and power (red/black wires) from the Raspberry Pi to the display. If you are not attaching the Raspberry Pi to the back of the display, take extra care when attaching the ribbon cable to ensure it is the correct way round. The black and red power wires should be attached to the GND and 5v pins respectively.

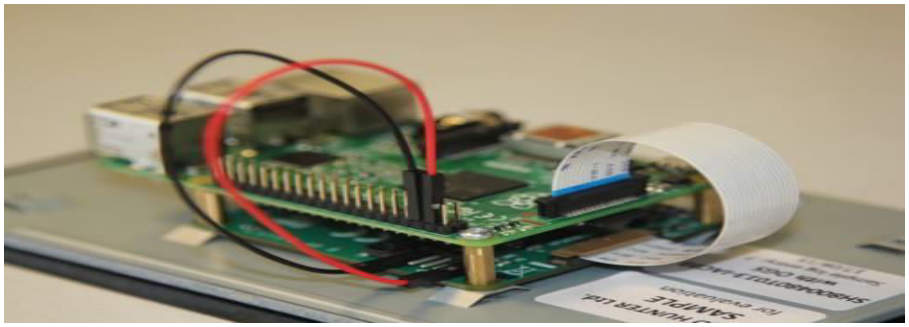


Fig. II.1. Touch Display interfacing with Rasbperry Pi [8].

II.6.3. Screen orientation [8]:

LCD displays have an optimum viewing angle, and depending on how the screen is mounted it may be necessary to change the orientation of the display to give the best results. By default, the Raspberry Pi Touch Display and Raspberry Pi are set up to work best when viewed from slightly above, for example on a desktop. If viewing from below, you can physically rotate the display, and then tell the system software to compensate by running the screen upside down.

II.6.3.1. FKMS Mode [8]:

FKMS mode is used by default on the Raspberry Pi 4B. FKMS uses the DRM/MESA libraries to provide graphics and 3D acceleration.

To set screen orientation when running the graphical desktop, select the

Screen Configuration option from the Preferences menu. Right click on the DSI display rectangle in the layout editor, select Orientation then the required option.

To set screen orientation when in console mode, you will need to edit the kernel command line to pass the required orientation to the system.

```
sudo nano /boot/cmdline.txt
```

To rotate by 90 degrees clockwise, add the following to the cmd line, making sure everything is on the same line, do not add any carriage returns. Possible rotation values are 0, 90, 180 and 270.

```
video=DSI-1:800x480@60,rotate=90
```

II.6.3.2. Legacy Graphics Mode [8]:

Legacy graphics mode is used by default on all Raspberry Pi models prior to the Raspberry Pi 4B, and can also be used on the Raspberry Pi 4B if required, by disabling FKMS mode by commenting out the FKMS line in config.txt

To flip the display, add the following line to the file /boot/config.txt :lcd_rotate=2

This will vertically flip the LCD and the touch screen, compensating for the physical orientation of the display.

You can also rotate the display by adding the following to the config.txt file.

display_lcd_rotate=x where x can be one of the following:

display_lcd_rotate	result
0	no rotation
1	rotate 90 degrees clockwise
2	rotate 180 degrees clockwise
3	rotate 270 degrees clockwise
0x10000	horizontal flip
0x20000	vertical flip

Table. II.2.types of rotate the display [8].

II.6.4. Touch screen orientation [8] :

Additionally, you have the option to change the rotation of the touch screen independently of the display itself by adding a dtoverlay instruction in config.txt, for example:

dtoverlay=rpi-ft5406,touchscreen-swapped-x-y=1,touchscreen-inverted-x=1

The options for the touch screen are:

DT parameter	Action
touchscreen-size-x	Sets X resolution (default 800)
touchscreen-size-y	Sets Y resolution (default 600)
touchscreen-inverted-x	Invert X coordinates
touchscreen-inverted-y	Invert Y coordinates
touchscreen-swapped-x-y	Swap X and Y cordinates

Table. II.3. Touch screen orientation [8].

II.6.5. Specifications [8]:

- 800×480 RGB LCD display
- 24-bit color
- Industrial quality: 140-degree viewing angle horizontal, 130-degree viewing angle vertical
- 10-point multi-touch screen
- PWM backlight control and power control over I2C interface
- Metal-framed back with mounting points for Raspberry Pi display conversion board and Raspberry Pi
- Back light life time: 20000 hours
- Operating temperature: -20 to +70 degrees centigrade
- Storage temperature: -30 to +80 degrees centigrade
- Contrast ratio: 500
- Average brightness: 250 cd/m^2
- Viewing angle (degrees):
    - Top - 50
    - Bottom - 70
    - Left - 70
    - Right - 70

II.7. Frequency management and thermal control [8]:

All Raspberry Pi models perform a degree of thermal management to avoid overheating under heavy load. The SoCs have an internal temperature sensor, which software on the GPU polls to ensure that temperatures do not exceed a predefined limit, this is 85°C on all models. It is possible to set this to a lower value, but not to a higher one. As the device approaches the limit, various frequencies and sometimes voltages used on the chip (ARM, GPU) are reduced. This reduces the amount of heat generated, keeping the temperature under control.

When the core temperature is between 80°C and 85°C, a warning icon showing a red half-filled thermometer will be displayed, and the ARM cores will be progressively throttled back. If the temperature reaches 85°C, an icon showing a fully filled thermometer will be displayed, and both the ARM cores and the GPU will be throttled back.

For Raspberry Pi 3 Model B+, the PCB technology has been changed to provide better heat dissipation and increased thermal mass. In addition, a soft temperature limit has been introduced, with the goal of maximising the time for which a device can "sprint" before reaching the hard limit at 85°C. When the soft limit is reached, the clock speed is reduced from 1.4GHz to 1.2GHz, and the operating voltage is reduced slightly. This reduces the rate of temperature increase: we trade a short period at 1.4GHz for a longer period at 1.2GHz. By default, the soft limit is 60°C, and this can be changed via the temp_soft_limit setting in “config.txt”.

The Raspberry Pi 4 Model B continues with the same PCB technology as the Raspberry Pi 3B+ to help dissipate excess heat. There is currently no soft limit defined.

II.8. Peripheral Addresses [8]:

If there is no kernel driver available, and a program needs to access a peripheral address directly with map, it needs to know where in the virtual memory map the peripheral bus segment has been placed. This varies according to which model of Raspberry Pi is being used, so there are three helper function available to provide platform independence.

`unsigned bcm_host_get_peripheral_address()` this returns the ARM-side physical address where peripherals are mapped. This is 0x20000000 on the Pi Zero, Pi Zero W, and the first generation of the Raspberry Pi and Compute Module, and 0x3f000000 on the Pi 2, Pi 3 and Compute Module 3.

`unsigned bcm_host_get_peripheral_size()` this returns the size of the peripheral's space, which is 0x01000000 for all models.

`unsigned bcm_host_get_sdram_address()` this returns the bus address of the SDRAM. This is 0x40000000 on the Pi Zero, Pi Zero W, and the first generation of the Raspberry Pi and Compute Module (GPU L2 cached), and 0xC0000000 on the Pi 2, Pi 3 and Compute Module 3 (uncached).

II.9. Raspberry Pi boot modes [8]:

The Raspberry Pi has a number of different stages of booting. This document explains how the boot modes work, and which ones are supported for Linux booting.

II.9.1. Boot sequence [8]:

USB boot defaults on the Raspberry Pi 3 will depend on which version is being used when the BCM2837 boots, it uses two different sources to determine which boot modes to enable. Firstly, the OTP (one-time programmable) memory block is checked to see which boot modes are enabled. If the GPIO boot mode setting is enabled, then the relevant GPIO lines are tested to select which of the OTP-enabled boot modes should be attempted. Note that GPIO boot mode can only be used to select boot modes that are already enabled in the OTP.

Next, the boot ROM checks each of the boot sources for a file called bootcode.bin; if it is successful it will load the code into the local 128K cache and jump to it. The overall boot mode process is as follows:

- BCM2837 boots
- Read OTP to determine which boot modes to enable
- If GPIO boot mode enabled, use GPIO boot mode to refine list of enabled boot modes
- If enabled: check primary SD for bootcode.bin on GPIO 48-53
    - Success - Boot
    - Fail - timeout (five seconds)
- If enabled: check secondary SD
    - Success - Boot

- Fail - timeout (five seconds)

- If enabled: check NAND
- If enabled: check SPI
- If enabled: check USB
  - If OTG pin == 0
    - Enable USB, wait for valid USB 2.0 devices (two seconds)
      - Device found:
        - If device type == hub
          - Recurse for each port
        - If device type == (mass storage or LAN951x)
          - Store in list of devices
    - Recurse through each MSD
      - If bootcode.bin found boot
    - Recurse through each LAN951x
      - DHCP / TFTP boot
  - else (Device mode boot)
    - Enable device mode and wait for host PC to enumerate
    - We reply to PC with VID: 0a5c PID: 0x2763 (Pi 1 or Pi 2) or 0x2764 (Pi 3)

The primary SD card boot mode is, as standard, set to be GPIOs 49-53. It is possible to boot from the secondary SD card on a second set of pins, i.e. to add a secondary SD card to the GPIO pins. However, we have not yet enabled this ability.

NAND boot and SPI boot modes do work, although they do not yet have full GPU support.

The USB device boot mode is enabled by default at the time of manufacture, but the USB host boot mode is only enabled with program_usb_boot_mode=1. Once enabled, the processor will use the value of the OTG ID pin on the processor to decide between the two modes. On a Raspberry Pi Model B, the OTG ID pin is driven to '0' and therefore will only boot via host mode once enabled (it is not possible to boot through device mode because the LAN9515 device is in the way).

The USB will boot as a USB device on the Pi Zero or Compute Module if the OTG ID pin is left floating (when plugged into a PC for example), so you can 'squirt' the bootcode.bin into the device.

II.9.2. SD card boot [8]:

The default way of using a Raspberry Pi is to boot it using an SD card: this is the recommended method for new and inexperienced users.

Beginners should start with NOOBS, which gives the user a choice of operating system from the standard distributions. The recommended distribution for normal use is Raspbian.

Alternatives are available, such as LibreELEC (Kodi media centre) or Arch Linux.

II.9.2.1. How to install NOOBS on an SD card [8]:

Once you've downloaded the NOOBS zip file, you'll need to copy the contents to a formatted SD card on your computer.

To set up a blank SD card with NOOBS:

- Format an SD card as FAT. See the instructions given below.
    - Your SD card will need to be at least 16GB for Full Raspbian, or at least 8GB for all other installs.
- Download and extract the files from the NOOBS zip file.
- Copy the extracted files onto the SD card that you just formatted, so that this file is at the root directory of the SD card. Please note that in some cases it may extract the files into a folder; if this is the case, then please copy across the files from inside the folder rather than the folder itself.
- On first boot, the "RECOVERY" FAT partition will be automatically resized to a minimum, and a list of OSes that are available to install will be displayed.

The following operating systems are currently included in NOOBS:

- Raspbian
- LibreELEC
- OSMC
- Recalbox
- Lakka
- RISC OS
- Screenly OSE
- Windows 10 IoT Core
- TLXOS

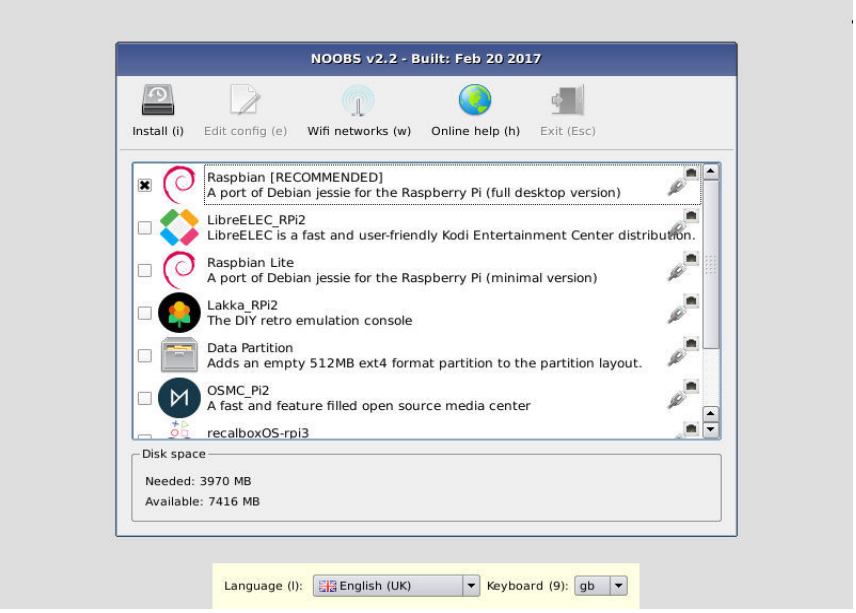


Fig. II.2. operating systems currently included in NOOBS [8].

II.9.2.2. NOOBS and NOOBS Lite [8]:

NOOBS is available in two forms: offline and network install, or network install only.

The full version has Raspbian included, so it can be installed from the SD card while offline, whereas using NOOBS Lite or installing any other operating system requires an internet connection.

Note that the operating system image on the full version can be outdated if a new version of the OS is released, but if connected to the internet you will be shown the option of downloading the latest version if there is a newer one available.

II.9.2.3. SD card size (capacity) [8]:

For installation of Raspbian with desktop and recommended software (Full) via NOOBS the minimum card size is 16GB. For the image installation of Raspbian with desktop and recommended software, the minimum card size is 8GB. For Raspbian Lite image installations we recommend a minimum of 4GB. Some distributions, for example LibreELEC and Arch, can run on much smaller cards.

II.9.2.4. SD card class [8]:

The card class determines the sustained write speed for the card, a class 4 card will be able to write at 4MB/s, whereas a class 10 should be able to attain 10 MB/s. However, it should be noted that this does not mean a class 10 card will outperform a class 4 card for general usage, because often this write speed is achieved at the cost of read speed and increased seek times.

II.9.2.5. Raspberry Pi Imager [8]:

Raspberry Pi have developed a graphical SD card writing tool that works on Mac OS, Ubuntu 18.04 and Windows, and is the easiest option for most users as it will download the image and install it automatically to the SD card.

- Download the latest version of Raspberry Pi Imager and install it.
- Connect an SD card reader with the SD card inside.
- Open Raspberry Pi Imager and choose the required OS from the list presented.
- Choose the SD card you wish to write your image to.
- Review your selections and click 'WRITE' to begin writing data to the SD card.

Note: if using the Raspberry Pi Imager on Windows 10 with Controlled Folder Access enabled, you will need to explicitly allow the Raspberry Pi Imager permission to write the SD card. If this is not done, the Raspberry Pi Imager will fail with a "failed to write" error.

II.9.3. GPIO boot mode [8]:

The Raspberry Pi can be configured to allow the boot mode to be selected at power on using hardware attached to the GPIO connector: this is GPIO boot mode. This is done by setting bits in the OTP memory of the SoC. Once the bits are set, they permanently allocate 5 GPIOs to allow this selection to be made. Once the OTP bits are set they cannot be unset so you should think carefully about enabling this, since those 5 GPIO lines will always control booting. Although you can use the GPIOs for some other function once the Pi has booted, you must set them up so that they enable the desired boot modes when the Pi boots.

To enable GPIO boot mode, add the following line to the config.txt file:

```
program_gpio_bootmode=n
```

Where n is the bank of GPIOs which you wish to use. Then reboot the Pi once to program the OTP with this setting. Bank 1 is GPIOs 22-26, bank 2 is GPIOs 39-43. Unless you have a Compute Module, you must use bank 1, the GPIOs in bank 2 are only available on the Compute Module. Because of the way the OTP bits are arranged, if you first program GPIO boot mode for bank 1, you then have the option of selecting bank 2 later. The reverse is not true: once bank 2 has been selected for GPIO boot mode, you cannot select bank 1.

Once GPIO boot mode is enabled, the Raspberry Pi will not longer boot. You must pull up at least one boot mode GPIO pin in order for the Pi to boot.

II.9.3.1. GPIO boot mode pin assignments [8]:

Raspberry Pi 3B and Compute Module 3:

Bank 1	Bank 2	boot type
22	39	SD0
23	40	SD1
24	41	NAND (no Linux support at present)
25	42	SPI (no Linux support at present)
26	43	USB

Table. II .4.GPIO boot mode pin assignments " Raspberry PI 3 B and Compute Module 3 " [8].

USB in the table above selects both USB device boot mode and USB host boot mode.in order to use a USB boot mode, it must by enabled in the OTP memory. For more information, see USB device boot and USB host boot.

Raspberry Pi 3A+, 3B+ and Compute Module 3+ :

Bank 1	Bank 2	boot type
20	37	SD0
21	38	SD1
22	39	NAND (no Linux support at present)
23	40	SPI (no Linux support at present)
24	41	USB device
25	42	USB host - mass storage device
26	43	USB host - ethernet

Table. II .5.GPIO boot mode pin assignments "- Raspberry Pi 3A+, 3B+ and Compute Module 3+ " [8].

II.9.3.2. Boot order [8]:

The various boot modes are attempted in the numerical order of the GPIO lines, i.e. SD0, then SD1, then NAND and so on.

II.9.3.3. Boot flow [8]:

SD0 is the Broadcom SD card / MMC interface. When the boot ROM within the SoC runs, it always connects SD0 to the built-in microSD card slot. On Compute Modules with an eMMC device, SD0 is connected to that; on the Compute Module Lite SD0 is available on the edge connector and connects to the microSD card slot in the CMIO carrier board. SD1 is the Arasan SD card / MMC interface which is also capable of SDIO. All Raspberry Pi models with built-in wifi use SD1 to connect to the wifi chip via SDIO.

The default pull resistance on the GPIO lines is 50K ohm, as documented on page 102 of the BCM2835 ARM peripherals datasheet. A pull resistance of 5K ohm is recommended to pull a GPIO line up: this will allow the GPIO to function but not consume too much power.

II.9.4. USB boot modes [8]:

There are two separate boot modes for USB (available only on certain models):

- USB device boot
- USB host boot with boot options:

  USB mass storage boot

  Network boot

The choice between the two boot modes is made by the firmware at boot time when it reads the OTP bits. There are two bits to control USB boot: the first enables USB device boot and is enabled by default. The second enables USB host boot; if the USB host boot mode bit is set, then the processor reads the OTG ID pin to decide whether to boot as a host (driven to zero as on the Raspberry Pi Model B) or as a device (left floating). The Pi Zero has access to this pin through the OTG ID pin on the USB connector, and the Compute Module has access to this pin on the edge connector.

There are also OTP bits that allow certain GPIO pins to be used for selecting which boot modes the Pi should attempt to use.

II.9.4.1. USB Device boot mode [8]:

The following devices can boot using USB device boot mode:

- Pi Compute Module
- Pi Compute Module 3
- Pi Zero
- Pi Zero W
- Pi A
- Pi A+
- Pi 3A+

When this boot mode is activated (usually after a failure to boot from the SD card), the Raspberry Pi puts its USB port into device mode and awaits a USB reset from the host.

The host first sends a structure to the device down control endpoint 0. This contains the size and signature for the boot (security is not enabled so no signature is required). Secondly, code is transmitted down endpoint 1 (bootcode.bin). Finally, the device will reply with a success code of:

- 0 - Success
- 0x80 - Failed

II.9.4.2. USB host boot mode [8]:

The USB host boot mode follows this sequence:

- Enable the USB port and wait for D+ line to be pulled high indicating a USB 2.0 device (we only support USB2.0)
- If the device is a hub:
  - Enable power to all downstream ports of the hub
  - For each port, loop for a maximum of two seconds (or five seconds if program_usb_boot_timeout=1 has been set)
    - Release from reset and wait for D+ to be driven high to indicate that a device is connected
    - If a device is detected:
      - Send "Get Device Descriptor"
        - If VID == SMSC && PID == 9500
          - Add device to Ethernet device list
      - If class interface == mass storage class
        - Add device to mass storage device list
- Else
  - Enumerate single device
- Go through mass storage device list
  - Boot from mass storage device
- Go through Ethernet device list
  - Boot from Ethernet

II.10. Raspberry Pi 3 model B:

II.10.1. Technical feature [9]:

Numéro du modèle de l'article	RASPBERRYPI3-MODB-1GB	**Résolution maximale d'affichage**	1080p Full HD
séries	Raspberry Pi 3 Model B	**Marque chipset graphique**	Broadcom
Couleur	vert	**GPU**	Dual Core VideoCore IV
Système d'exploitation	Linux	**Mémoire vive de la carte graphique**	1
Plate-forme du matériel informatique	Linux	**Type de mémoire vive (carte graphique)**	Shared
Description du clavier	Français	**Socket du processeur**	PLCC
Marque du processeur	Quad Core ARM cortex-a53	**Type d'alimentation**	DC
Type de processeur	ARM 7100	**Type de connectivité**	Wi-Fi

Vitesse du processeur	1.2 GHz	**Type de technologie sans fil**	802.11bgn
Nombre de coeurs	4	**Bluetooth**	Oui
Taille de la mémoire vive	1 GB	**Interface du matériel informatique**	USB 2.0
Mémoire maximale	1 GB	**Nombre de ports HDMI**	1
Interface du disque dur	USB	**Nombre de ports USB 2.0**	4
Taille de l'écran	60 pouces	**Item dimensions L x W x H**	12,2 x 7,6 x 3,4 cm
Résolution de l'écran	1920 x 1080	**Poids du produit**	45.4 grammes

Table. II.6.Raspberry Pi 3 model B Technical feature.

II.10.2. Pin mappings [10]:

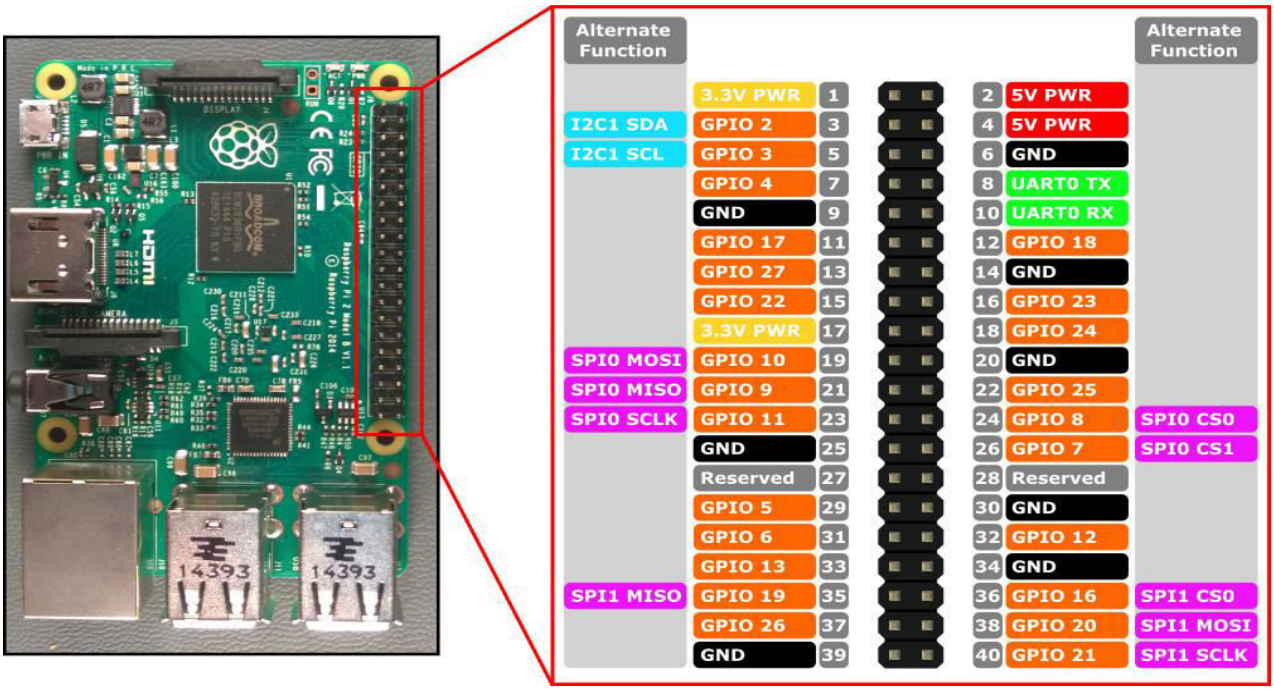


Fig. II.3. Raspberry Pi 2 & 3 Pin Mappings [10].

This figure show the hardware interfaces for the Raspberry Pi 3 is exposed through the 40-pin header **J8** on the board. Functionality includes:

- **24x** - GPIO pins
- **1x** - Serial UARTs (RPi3 only includes mini UART)
- **2x** - SPI bus
- **1x** - I2C bus
- **2x** - 5V power pins
- **2x** - 3.3V power pins
- **8x** - Ground pins

II.10.3. GPIO Pin Overview [10] :

The following GPIO pins are accessible through APIs:

GPIO	Power-on Pull	Alternate Functions	Header Pin
2	PullUp	I2C1 SDA	3
3	PullUp	I2C1 SCL	5
4	PullUp		7
5	PullUp		29
6	PullUp		31
7	PullUp	SPI0 CS1	26
8	PullUp	SPI0 CS0	24
9	PullDown	SPI0 MISO	21
10	PullDown	SPI0 MOSI	19
11	PullDown	SPI0 SCLK	23
12	PullDown		32
13	PullDown		33
16	PullDown	SPI1 CS0	36
17	PullDown		11
18	PullDown		12
19	PullDown	SPI1 MISO	35
20	PullDown	SPI1 MOSI	38
21	PullDown	SPI1 SCLK	40
22	PullDown		15
23	PullDown		16
24	PullDown		18
25	PullDown		22

26	PullDown		37
27	PullDown		13

Table. II.7.GPIO pins of Raspberry Pi 3 model B [10].

II.10.4. Alternate function [10]:

Some GPIO pins can perform multiple functions. By default, pins are configured as GPIO inputs. When you open an alternate function by calling I2cDevice.FromIdAsync() or SpiDevice.FromIdAsync(), the pins required by the function are automatically switched ("muxed") to the correct function.

When the device is closed by calling I2cDevice.Dispose() or SpiDevice.Dispose(), the pins revert back to their default function. If you try to use a pin for two different functions at once, an exception will be thrown when you try to open the conflicting function.

II.10.5. Serial UART [10]:

There is one Serial UART available on the RPi3: **UART0**

- Pin 8 - **UART0 TX**
- Pin 10 - **UART0 RX**

II.10.6. I2C Bus [10]:

There is one I2C controller I2C1 exposed on the pin header with two lines SDA and SCL. 1.8KΩ internal pull-up resistors are already installed on the board for this bus.

Signal Name	**Header Pin Number**	**Gpio Number**
SDA	3	2
SCL	5	3

Table. II.8. I2C bus [10].

II.10.7. SPI Bus [10]:

There are two SPI bus controllers available on the RPi3.

- **SPI0:**

Signal Name	**Header Pin Number**	**Gpio Number**
MOSI	19	10
MISO	21	9
SCLK	23	11
CS0	24	8
CS1	26	7

Table. II.9. SPI0 bus [10].

- **SPI1:**

Signal Name	Header Pin Number	Gpio Number
MOSI	38	20
MISO	35	19
SCLK	40	21
CS0	36	16

Table. II.10. SPI1 bus [10].

II.10.8. Raspberry Pi 3 On-chip Hardware [7]:

The On-chip hardware of Raspberry Pi 3 is as shown in below figure:

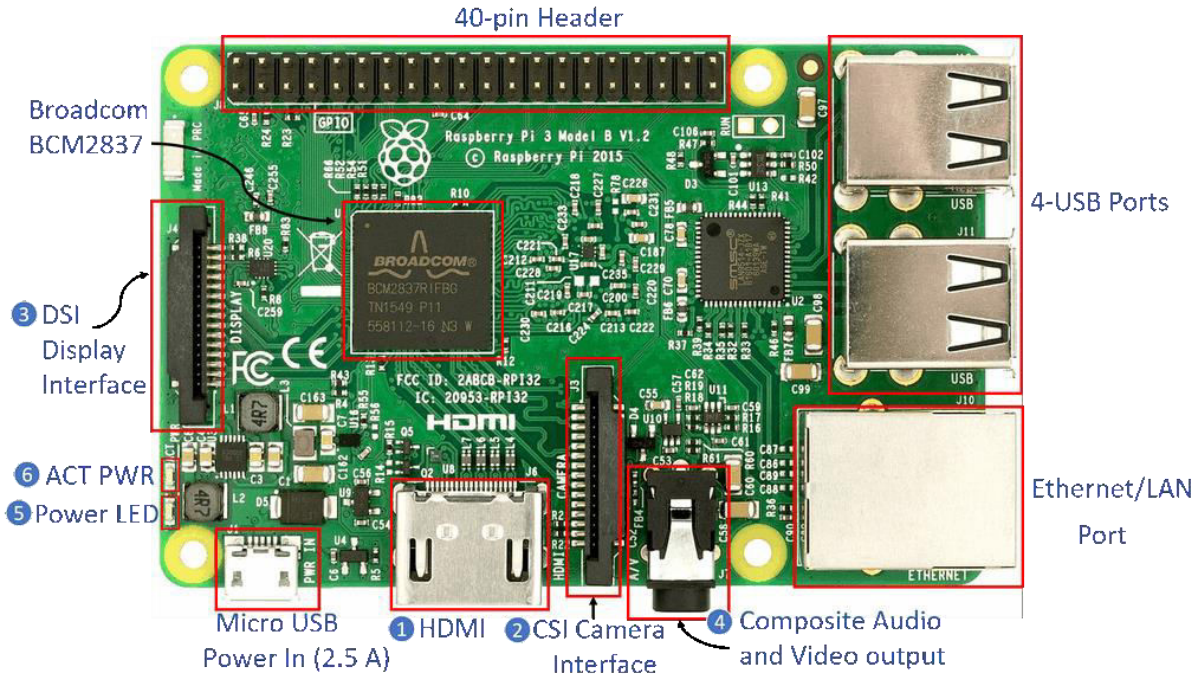


Fig. II.4. Raspberry Pi 3 Model B Hardware [7].

1- HDMI (High-Definition Multimedia Interface): It is used for transmitting uncompressed video or digital audio data to the Computer Monitor, Digital TV, etc. Generally, this HDMI port helps to connect Raspberry Pi to the Digital television.

2-CSI Camera Interface: CSI (Camera Serial Interface) interface provides a connection in between Broadcom Processor and Pi camera. This interface provides electrical connections between two devices.

3-DSI Display Interface: DSI (Display Serial Interface) Display Interface is used for connecting LCD to the Raspberry Pi using 15-pin ribbon cable. DSI provides fast High-resolution display interface specifically used for sending video data directly from GPU to the LCD display.

4-Composite Video and Audio Output: The composite Video and Audio output port carries video along with audio signal to the Audio/Video systems.

5-Power LED: It is a RED colored LED which is used for Power indication. This LED will turn ON when Power is connected to the Raspberry Pi. It is connected to 5V directly and will start blinking whenever the supply voltage drops below 4.63V.

6-ACT PWR: ACT PWR is Green LED which shows the SD card activity.

II.11. Conclusion:

The Raspberry Pi is an ARM processor single board nanocomputer ,it has 6 interface ports and 40 GPIO pins which are USB, Ethernet, camera module, voice and tow display ports (HDMI and DPI) , the 40 pins has an alternative function which are :

- Logical input/output
- Interruption
- Serial UART
- SPI bus
- I2C bus
- 3.3V and 5V power
- GND pins

It runs mostly on Linux operating system and can also be run on other operating system (Raspbian, LibreELEC, OSMC, Recalbox, Lakka, RISC OS, Screenly OSE, Windows 10 IoT Core, TLXO) which are included in NOOBS. The Raspberry Pi has three boot modes (SD card boot mode, GPIO boot mode and USB boot mode)

Chapter 3:

Digital Image Processing

III.1.Introduction:

An image is a large amount of information about a thing or things (person, rigid, etc.). In the electronic field, it is a 2-dimensional signal processing, where the image is converted into a 2-dimensional table that holds many numbers. The information can be visible or invisible. Often the image is processed to extract the visual information due to the difficulty of the image processing process. Digital image processing can be defined as a transformation of a distorted image into a modified one which helps people to detect silent features without difficulty for interpretation. Digital image processing is used in many fields and disciplines because it is accurate and carries a lot of information in relation to other signs. This field of signal processing is very wide and in continuous development. Therefore, there are many methods and approaches to extract information from images. In this chapter, we will talk about two uses for processing and extracting information, namely facial recognition and fingerprint recognition. Where the user must know the basics of image processing, such as the types of basic images(binary, gray scale, color or multispectral image), also must know the image file format(vectorial or bitmap format), and to form a digital image we need to image sampling and quantization. Most of the time the image is converted to gray scale for easy processing. histogram is one of the main basics because it carries a lot of information that helps in the segmentation process and others. For facial recognition, we used the Haar classifier, which we will explain in this chapter. As for the fingerprint recognition process, an algorithm inside the fingerprint sensor module, so we will talk about the types of sensors to recognize the fingerprint, as well as the information in the fingerprint and how to extract it.

III.2. Type of digital image :

There are four type of digital image which are binary, gray-scale, color and multispectral.

III.2.1. Binary images [11]:

Binary images are the simplest type of images and can take one of two values, typically black and white, nor 0 and 1. A binary image is referred to as a 1-bit image because it takes only 1 binary digit to represent each pixel. These types of images are most frequently use in computer vision applications where the only information required for the task is general shape or outline information, for example optical character recognition (OCR).

Binary images are often created from the gray-scale images via a threshold operation, where every pixel above the threshold value is turned white (‘1’), and those below it are turned black(‘0’).

In figure bellow, we see examples of binary images.

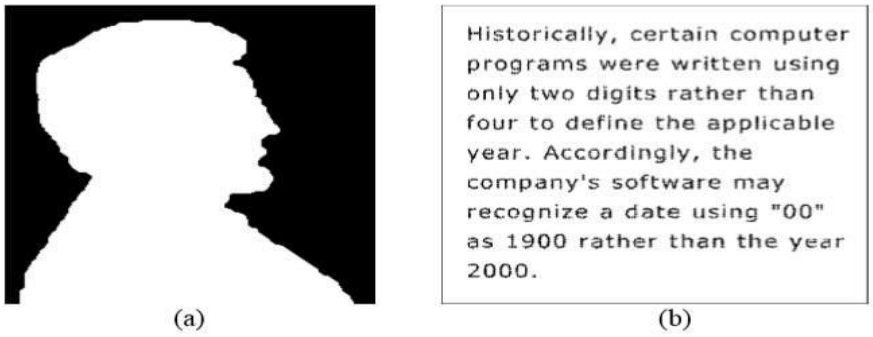


Fig.III.1. Binary images. (a) Object outline. (b)Page of text use in OCR Application [11].

III.2.2. Gray-scale images [11]:

Gray-scale images are referred to as monochrome (one-color) images. They contain gray-level information, no color information. The number of bits used for each pixel determines the number of different gray levels available. The typical gray-scale image contains 8bits/pixel data, which allows us to have 256 different gray levels. The figure below is shows examples of gray-scale images.

Fig.III.2.examples of gray-scale images [11].

In applications like medical imaging and astronomy, 12 or 16 bits/pixel images are used. These extra gray levels become useful when a small section of the image is made much larger to discern details.

III.2.3. Color image [11]:

Color image can be modeled as three band monochrome image data, where each band of data corresponds to a different color. The actual information stored in the digital image data is the gray-level information in each spectral band.

Typically color images are represented as red, green and blue (RGB images). Using the 8-bit monochrome standard as a model, the corresponding color image would have 24-bits/pixel (8-bit for each of the three color bands red, green and blue). The figure below illustrates a representation of a typical RGB color image.

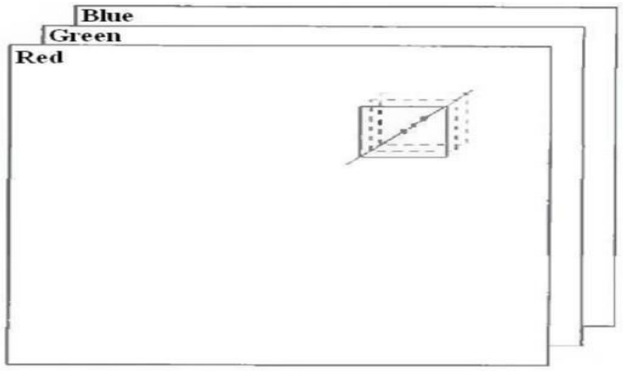


Fig.III.3. three band of color image (red, green and blue) [11].

III.2.4. Multispectral images [11]:

Multispectral image typically contain information outside the normal human perceptual range .This may include infrared, ultraviolet, X-ray, acoustic, or radar data. These are not images in the usual sense because the information represented is not directly visible by the human system. However, the information is often represented in visual form by mapping the different spectral bands to RGB components.

III.3. Digital image file format [11] :

Types of image data are divided into two primary categories: bitmap and vector image.

- Bitmap images (also called raster images) can be represented as 2-dimentional function $f(x,y)$, where they have pixel data and the corresponding gray-level values stored in some file format
- Vector images refer to methods of representing lines, curves, and shapes by storing only the key points. These key points are sufficient to define the shapes. The process of turning these into an image is called *rendering*. After the image has been rendered, it can be thought of as being in bitmap format, where each pixel has specific values associated whit it.

III.4. Image Sampling and Quantization [11]:

To convert the continuous function $f(x, y)$ to digital form, we need to sampel the function in both coordinates and in amplitudes.

- Digitizing the coordinate values is called *sampling*
- Digitizing the amplitude values is called *quantization*

In the figure below, we show how to convert the continuous image in Fig.III.4.(a) to the digital form using the sampling and quantization process . The one-dimensional function show in Fig.III.4.(b) is a plot of amplitude (gray-level) values of the continuous image along the line segment AB in Fig.III.4.(a).

To simple this function, we take equally spaced simples along line AB, as shown in Fig.III.4.(c). The simples are shown as small white squares superimposed on the function. The set of these discrete locations gives the sampled function.

In order to form a digital function, the gray-level values also must be converted (quantized) into discrete quantities. The right side of Fig.III.4.(c) shows the gray-level scale divided into eight discrete levels, ranging from black to white. The continuous gray levels are quantized simply by assigning one of the eight discrete gray levels to each sample.

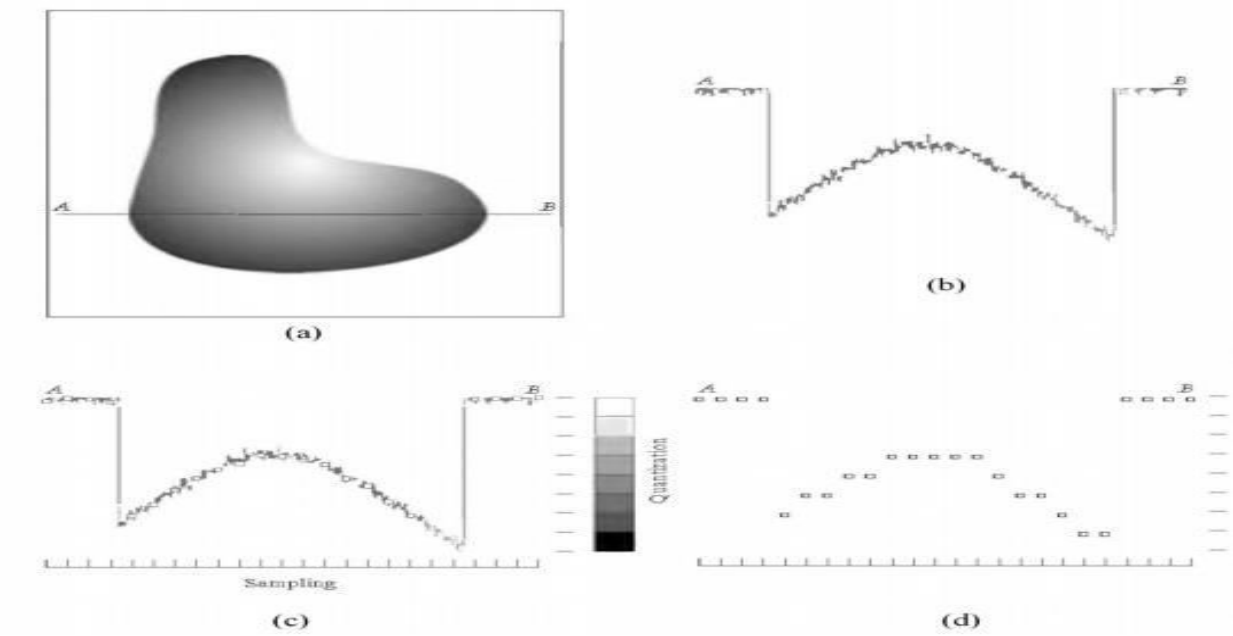


Fig.III.4.Generating a digital image .(a) Continuous image , (b) A scan line from A to B in the continuous image ,(c) Sampling and quantization ,(d) Digital scan line [11].

The digital simples resulting from both sampling and quantization are show in Fig.III.4.(d). Starting at the top of the image and carrying out this procedure line by line produces a two-dimensional digital image as shown in Fig.III.5

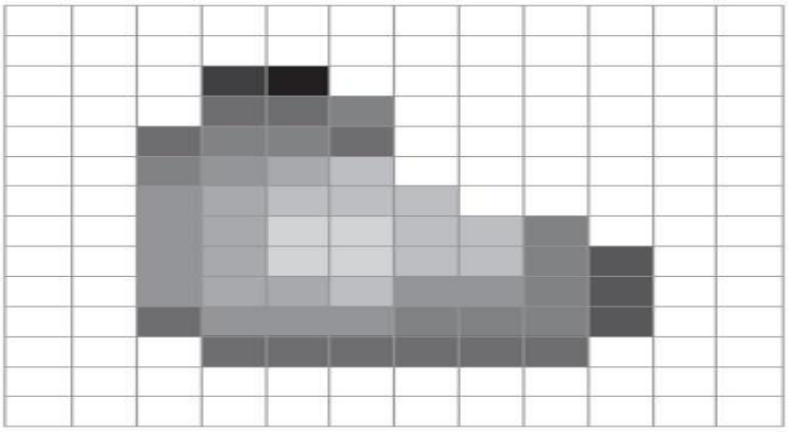


Fig.III.5.Digital image resulted from sampling and quantization [11].

III.5. Digital image processing (DIP) [12]:

Digital image processing (DIP) can be defined as a transformation of a distorted image into a modified one which helps people to detect silent features without difficulty for interpretation necessary for image analysis in different study. It is an electronic data processing on a 2-D array of numbers known as pixel which is the numeric representation of any image. The output of image processing can be an image or a set of characteristics related to the image. Digital Image Processing techniques are used to manipulate the digital images using computers. Image processing system includes treating images as 2d signal .Image processing system consists of a source of image data, a processing element and a destination for the processed results. The source of image data may be a camera, a sensor, satellite, scanner, a mathematical equation, statistical data, the Web, a SONAR system, etc.

The processing element is a computer, destination for the processed result, and the output of the processing may be a display monitor.

III.6. Major Tasks [12]:

Two major Tasks in Digital Image Processing

- Pictorial information improvement for human interpretation
- Data processing for storage, transmission and representation.

III.7. Histograms [13]:

The brightness histogram $h_f(z)$ of an image provides the frequency of the brightness value z in the image, the histogram of an image with$L$ gray-levels is represented by a one-dimensional array with$L$elements.

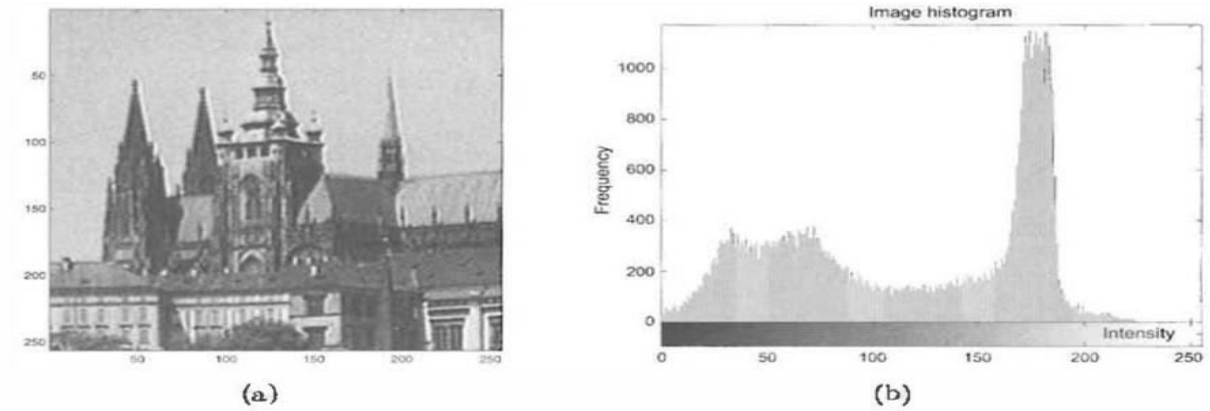


Fig.III.6.Original image (a) and its brightness histogram (b) [13].

III.8. Gray-scale transformation [13]:

Gray-scale transformations do not depend on the position of the pixel in the image. A transformation $\mathcal{T}$ of the original brightness p from scale $[p_0,p_k]$ into brightness q from a new scale $[q_0, q_k]$ is given by

$$q = \mathcal{T}(p) \quad (1)$$

The most common gray-scale transformations are shown in Fig.III.6, the piecewise linear function (a) enhances the image contrast between brightness values p_1 andp_2. The function (b) is called **brightness thresholding** and results in a black-and-white image, the straight line (c) denotes the negative transformation.

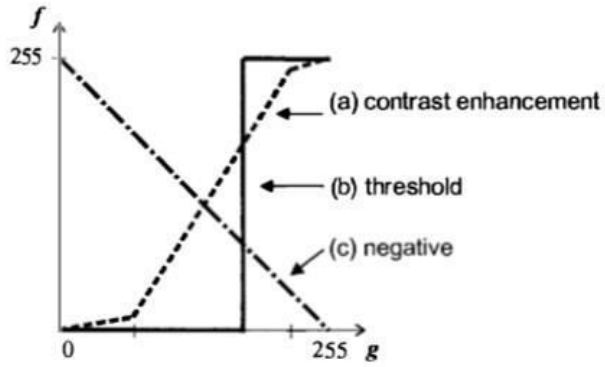


Fig.III.7. Perspective projection geometry examples [13].

Digital images have a very limited number of gray-levels, so gray-scale transformations are easy to realize both in hardware and software. Often only 256 bytes of memory (called a look-up table) are needed. The original brightness is the index to the look-up, and the table content gives the new brightness. The image signal usually passes

through a look-up table in image displays, enabling simple gray-scale transformation in real time.

The same principle can be used for color displays. A color signal consists of three components red, green, and blue; three look-up tables provide all possible color scale transformations. These tables are called the palette in personal computer terminology.

Gray-scale transformations are used mainly when an image is viewed by a human observer, and a transformed image might be more easily interpreted if the contrast is enhanced. For instance, an X-ray image can often be much clearer after transformation.

III.9. Segmentation :

This process is used to subdivide an image into its component regions or objects. Algorithms of this process are based generally on one of the following 2 basic properties of intensity values

- Discontinuity: to make partition of an image based on sharp changes in intensity (such as edges)

- Similarity: to make partition of an image into regions that are similar according to a set of predefined criteria.

III.10. Haar Classifier [14]:

The **Haar** classifier, which builds a boosted rejection cascade, it has a different format from the rest of the ML library in OpenCV because it was developed earlier as a full-fledged face-recognition application. It can also be trained to recognize faces and other solid objects.

Computer vision is a broad and fast-changing field, so the parts of OpenCV that implement a specific technique—rather than a component algorithmic piece—are more at risk of becoming out of date. The face detector that comes with OpenCV is in this “risk” category. However, face detection is such a common need that it is worth having a base-line technique that works fairly well; also, the technique is built on the well-known and often used field of statistical boosting and thus is of more general use as well. In fact, several companies have engineered the “face” detector in OpenCV to detect “mostly rigid” objects (faces, cars, bikes, human body) by training new detectors on many thousands of selected training images for each view of the object. This technique has been used to create state-of-the-art detectors, although with a different detector trained for each view or pose of the object. Thus, the **Haar** classifier is a valuable tool to keep in mind for such recognition tasks.

OpenCV implements a version of the face detection technique first developed by **Paul Viola** and **Michael Jones**, commonly known as the **Viola-Jones** detector and later extended by **Rainer Lienhart** and **Jochen Maydt**, to use diagonal features (more on this distinction to follow). OpenCV refers to this detector as the “**Haar** classifier” because it uses **Haar** features or, more precisely, Haar-like wavelets that consist of adding and subtracting rectangular image regions before thresholding the result. OpenCV ships with a set of pretrained object-recognition files, but the code also allows you to train and store

new object models for the detector. We note once again that the training (createsamples(), haartraining()) and detecting (cvHaarDetectObjects()) code works well on any objects (not just faces) that are consistently textured and mostly rigid.

The pretrained objects that come with OpenCV for this detector are in …/opencv/data/ haarcascades, where the model that works best for frontal face detection is haarcascade_ frontalface_alt2.xml. Side face views are harder to detect accurately with this technique, and those shipped models work less well.

III.11. Face detection [15]:

For face detection, **Viola Jones** algorithm is a beneficial method. In general, this algorithm is not only limited for face detection but can also be utilized for many rigid structured object detection tasks. The Viola-Jones algorithm is composed of three main concepts that make it possible to develop a real time face detector: Haar-like features, integral Image, Adaboost training and Cascading classifier. By applying these features, the system can determine the presence or the absence of a human face[15][16].

III.11.1. Haar-like features [15]:

Haar-like features is used by Haarcascade classifier for human face detection. There are three formations of Haar-like features. From the Fig.III.8, the first format is the edge feature, second type is the line feature and the last type is the four rectangle feature. Using the integral image, Haar-like principle will provide fast computation. It's called Haar-like features[15][16].

The Algorithm looks for specific **Haar** feature of a face. This detection takes the image and converts it into 24X24 window and smears each Haar feature to that window pixel by pixel. Initially, the algorithm requires a lot of positive images (images of faces) and negative images (images without faces) to train the classifier.

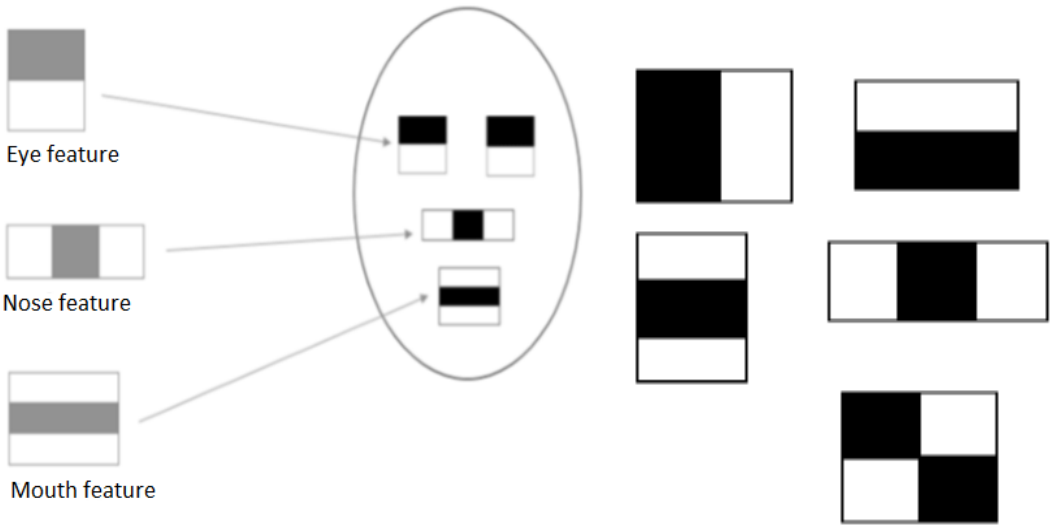


Fig.III.8. Types of Haar- like features [15].

Then, these features are extracted. Features are numerical values determined from images that are used to distinguish one image from another each feature is a single value acquired by subtracting the sum of the pixels beneath the white rectangle from the sum of the pixels beneath the black rectangle[15][17].

$$\text{Feature} = \Sigma(\text{pixels in black area}) - \Sigma(\text{pixels in white area}) \quad (2)$$

dark white

All possible sizes and locations of each kernel calculate a plenty of features. A 24×24 window results in over 160,000 features. For each feature calculation, it is necessary to find the sum of the pixels under the white and black rectangles. To solve this, the concept of integral image and adaboost algorithm is utilized, which reduces 160000 features to 6000 features [15][16].

III.11.2. Integral Image [15]:

Rectangle features can be determined rapidly via an intermediate representation of the image called the integral Image. The integral image comprises of small units representation of a given image.

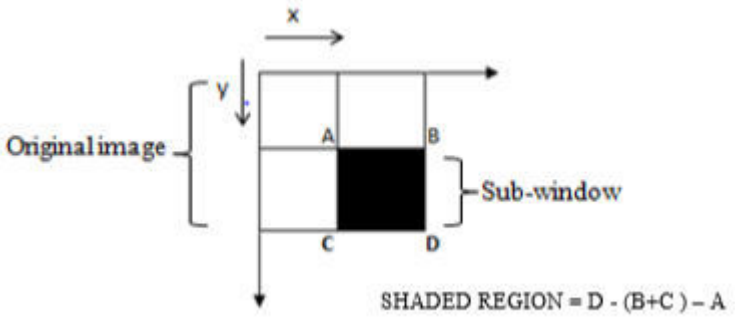


Fig.III.9. Integral image schematic diagram [15].

For example, the value of this integral image at position1 is the sum of pixels in rectangular A. The value at position 2 is A + B and so on. So, the sum total of pixels in rectangular D is:

$$S(D) = ii(4) - (ii(3) + ii(2)) + ii(1) \quad (3)$$

Where, S(D) is the sum of pixels in the rectangular D only - which is the sum of pixels in the rectangle A + B + C + D, represented by ii(4); ii(3) is the integral image of rectangle A+C ; ii(2) is the integral image of A+B and finally ii(1) is the integral image of the rectangle A (the addition is executed since the region A is subtracted twice in ii(3)and ii(2)). The integral image is outlined as:

$$ii[x, y] = i[x', y'] \quad (4)$$

Where, ii[x, y] represents integral image, and i[x', y'] represents original image.

The pixel value of integral images at any (x,y) location is the sum of all pixel values displayed before the current pixel. The integral value of an individual pixel is the sum of pixels on the top and the pixel towards the left. For example,

5	4	3	8	3
3	9	1	2	6
9	6	0	5	7
7	3	6	5	9
1	2	2	8	3

Fig.III.10. Input image [15].

5	9	12	20	23
8	21	25	35	44
17	36	40	55	71
24	46	56	76	101
25	49	61	89	117

Fig.III.11. Integral image [15].

The image is integrated in fewer pixel operations, since the traversing begins from the top left towards the bottom right. This makes the calculation of the addition to the entire pixels within any specified rectangle using only four values. In the integral image, these values are the pixels that resemble with the edges of the rectangle in the input image

III.11.3. AdaBoost Learning [15]:

AdaBoost is an adequate boosting algorithm which combines weak classifiers while reducing significantly not only the training error but also the more elusive generalized error. The main idea of Boosting lies in connecting the simple classifiers which are known as weak classifiers. Since the weak classifiers do not expect even the best classification function to classify the data well, they are called as weak classifiers. Here a classifier is combined with a single feature to easily link the **Haar** features with weak classifier. Haar-like feature is used as a threshold in AdaBoost learning algorithm by Viola and Jones. The Haar classifier is the strongest classifier since it uses the strongest features. The positive and negative samples are best separated by the feature. In order to build a strong final classifier AdaBoost is used [15][18]. It reduces the features from 160000 to 6000, thus making the computation simpler and hence it is less in computational complexity.

III.11.4. Cascade Classifier [15]:

Cascade classifier is a cascading of weak classifiers used to boost the face detection process and reduce the computational complexity. Each node in the series contains a weak classifier and filter for one **Haar** feature. AdaBoost provides weights to the nodes and the highest weighted node primarily arrives. When a filter ignores to permit image regions, that specific sub window of the image is eliminated for further processing. It is then considered as a non-face, which means that the image regions that are processed do not contain the face to be detected. This is very imperative to the performance of the classifier, since all or nearly all negative image sub-windows will be eliminated in the first stage.

On the contrary, when image regions successfully passed the filter, they go to the following stage, which contains a more complex filter. Only regions that successfully pass all filters are considered to contain a match of the face. This means that regions of the image contain the facial subject for detection. The reason behind the multi-stage classifier is to eliminate efficiently and rapidly the non-face sub- windows. The classifier is used to reject more false positives (non-face regions) of the sub-windows. The number of false positive rate is drastically reduced after several steps of processing [15][18].

III.12. Feature extraction and Comparison [15]:

After the face is detected, next step is to extract features this is done using linear binary pattern algorithm. Initial step of this algorithm is to convert the test image into gray scale. This L x M pixel size image will get divided into regions. The same pixel size is used for the regions, producing n x n regions. Each region will goes through Linear binary pattern operator.

In this process, it will compare the center pixel with its neighbor pixels. If the pixel size is greater to center pixel it is ‘1’ or it is ‘0’.

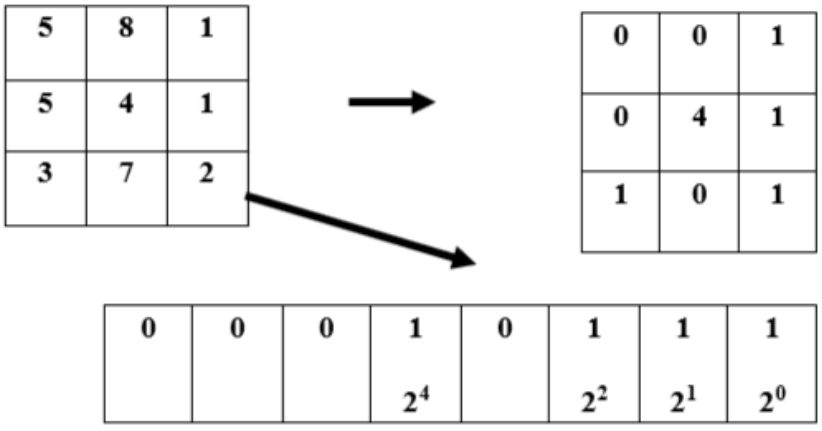


Fig.III.12. LBPH algorithm example [15].

Executing this process will result in 8 binary values. By linking the binary values it results in binary number. The LBP value is obtained by translating 8 binary numbers into a decimal number, it will be in the range of 0-255. This algorithm implementation is shown in the above Fig.III.12. The histogram for each region is drawn using the LBP Values of

each region. Each region will contain 256 cases. This implementation is shown in the below equation:

$$N_X = a_0 + \sum_{i,j} \mathrm{X}\{LBP?\left(Y(i,j)\right) = x\} \text{ , x= (0,…, 255)} \qquad (5)$$

Where, N_X is a case of value, $Y(i,j)$ is the (i,j) pixel of image and X is the conditional operator, providing '1' when it is true or '0'. After finding the histogram for each region, the sole histogram is created by uniting each region histogram. The final histogram is in the form of 256 *n *n cases and it is determined as the image feature vector [15][19]. The drawback of this algorithm is it has a fixed scale (3 x 3 scale). To overcome this, there is an extension of original LBP implementation to handle multiple neighborhoods. There are two parameters: first is 'p' which is the number of points in the symmetric circle neighborhood, second is 'r' the circle radius.

There is an important concept called LBP uniformity. A LBP is uniform if it has at most two 1-0 or 0-1 transitions, for example: consider pattern 10000000(1 transition) when 00100000(2 transitions) they both are uniform, the pattern 00100100(4 transitions) is considered as uniform. LBP uniformity completely depends on the 'p' value. When p increases resulting histogram dimensionality increases[15][20].

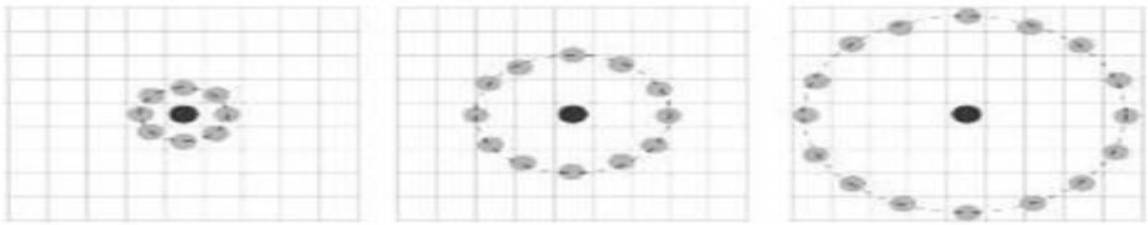


Fig.III.13. Represents varying p and r to form a Local Binary pattern [15].

III.13. Live-Scan Fingerprint Sensing [21]:

The most important part of a live-scan fingerprint scanner is the sensor (or sensing element), which is the component where the fingerprint image is formed. Almost all the existing sensors belong to one of the following three families: optical, solid-state, and ultrasound.

III.13.1. Optical sensors [21]:

Frustrated Total Internal Reflection (FTIR), this is the oldest and most commonly used live-scan acquisition technique today (**Hase** and **Shimisu** (1984), **Bahuguna** and **Corboline** (1996)). As the finger touches the top side of a glass/plastic prism, the ridges are in optical contact with the prism surface, but the valleys remain at a certain distance (see Fig.III.14). The left side of the prism is typically illuminated through a diffused light (a bank of light-emitting diodes [LEDs] or a film of planar light). The light entering the prism is reflected at the valleys, and randomly scattered (absorbed) at the ridges. The lack of reflection allows the ridges (which appear dark in the image) to be discriminated from the valleys (appearing bright). The light rays exit from the right side of the prism and are focused through a lens onto a CCD or CMOS image sensor. Because FTIR devices sense a

three-dimensional finger surface, they cannot be easily deceived by presentation of a photograph or printed image of a fingerprint.

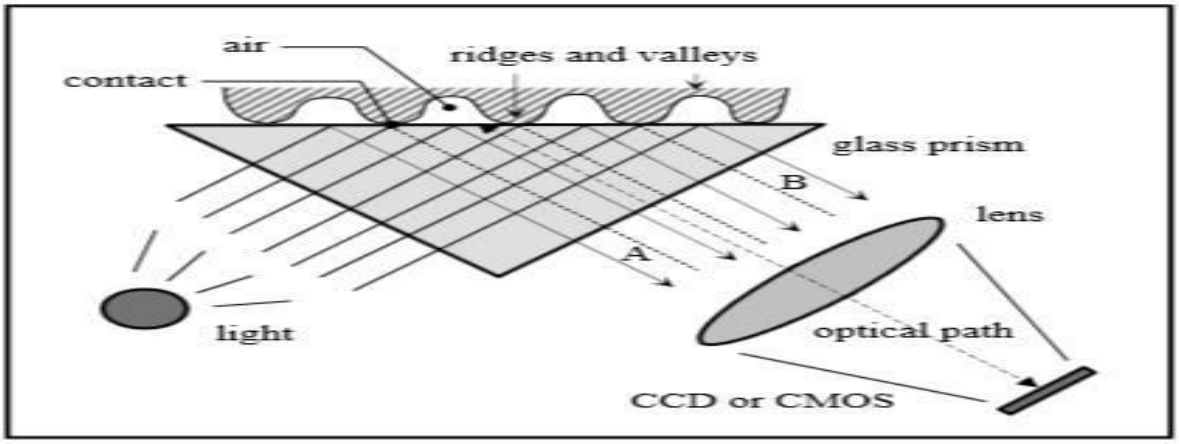


Fig.III.14. FTIR-based fingerprint sensor operation [21].

III.13.2. Solid-state sensors [21]:

Although solid-state sensors (also known as silicon sensors) have been proposed in patent literature since the 1980s, it was not until the middle 1990s that they became commercially viable (**Xia** and **O'Gorman** (2003)). Solid-state sensors were designed to overcome the size and cost problems which, at the time seemed to be a barrier against the wide spread deployment of fingerprint recognition systems in various consumer applications. All silicon-based sensors consist of an array of pixels, each pixel being a tiny sensor itself. The user directly touches the surface of the silicon: neither optical components nor external CCD/CMOS image sensors are needed. Four main technologies have been proposed to convert the fingerprint pattern into electrical signals: capacitive, thermal, electric field, and piezoelectric.

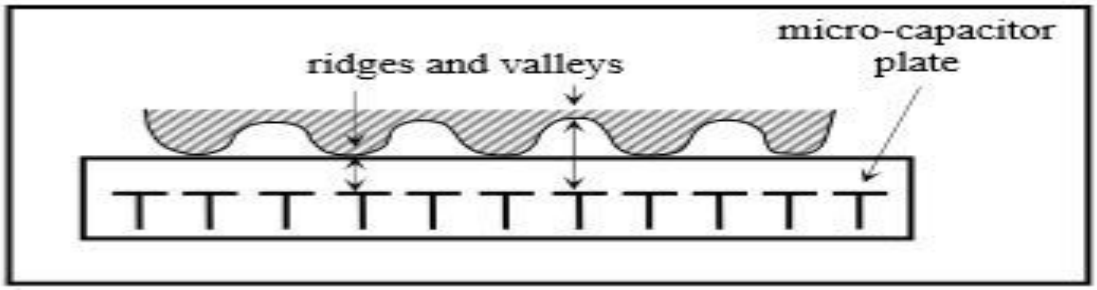


Fig.III.15. Capacitive sensing [21].

III.13.3. Ultrasound sensors [21]:

Ultrasound sensing may be viewed as a kind of echography. It is based on sending acoustic signals toward the fingertip and capturing the echo signal (see Fig.III.16). The echo signal is used to compute the range (depth) image of the fingerprint and, subsequently, the ridge structure itself.

The ultrasound sensor has two main components: a transmitter, which generates short acoustic pulses, and a receiver, which detects the responses obtained when these

pulses bounce off the fingerprint surface (Schneider and **Wobschall** (1991), **Bicz** et al. (1999), **Schneider** (2007)). This method images the subsurface of the finger skin (even through thin gloves); therefore, it is resilient to dirt and oil accumulations on the finger. While good quality images may be obtained by this technology, current ultrasound scanners are bulky with mechanical parts and quite expensive (several hundred dollars). Moreover, it takes a few seconds to acquire an image. Hence, this technology is not yet mature enough for large-scale deployment.

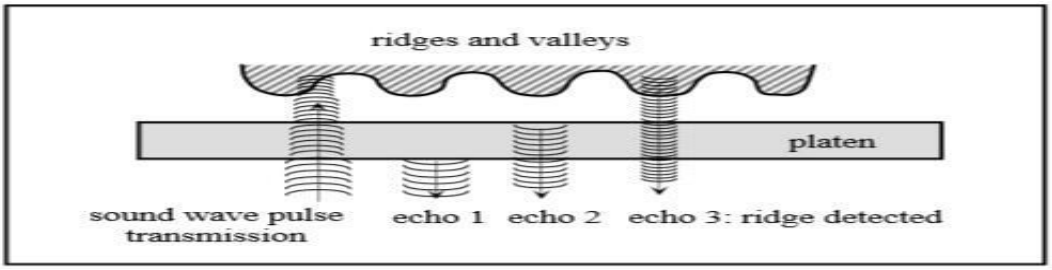


Fig.III.16. The basic principle of the ultrasound technique [21].

The basic principle of the ultrasound technique, a characteristic of sound waves is their ability to penetrate materials, giving a partial echo at each impedance change.

III.14. Singular regions and minutiae [21]:

At the global level (Level 1), ridges often run smoothly in parallel but exhibit one or more regions where they assume distinctive shapes (characterized by high curvature, frequent ridge terminations, etc.). These regions, called singularities or singular regions, may be broadly classified into three typologies: loop, delta, and whorl (see Fig.III.17). Singular regions belonging to loop, delta, and whorl types are typically characterized by ∩, Δ, and O shapes, respectively. Sometimes whorl singularities are not explicitly introduced because a whorl type can be described in terms of two loop singularities facing each other.

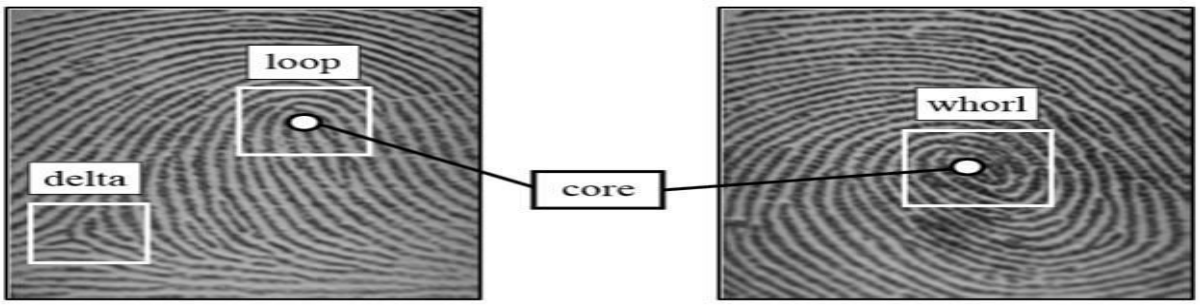


Fig.III.17. Singular regions delta (white boxes) and core points (small circles) in fingerprint images [21].

At the local level (Level 2), other important features, called minutiae can be found in the fingerprint patterns. Minutia means small detail, in the context of fingerprints, it refers to various ways that the ridges can be discontinuous (see Fig.III.18). For example, a ridge can suddenly come to an end (ridge ending), or can divide into two ridges (bifurcation).

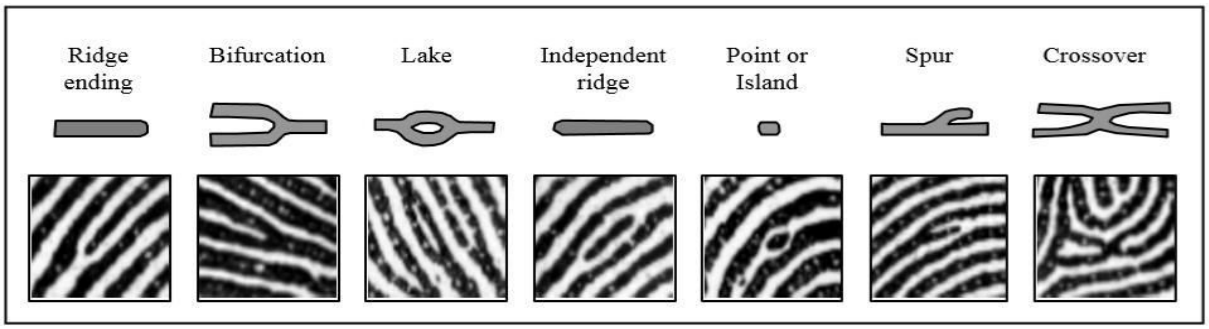


Fig.III.18. Seven most common minutiae types [21].

III.15. Minutiae Detection [21]:

Most of the proposed methods require the fingerprint gray-scale image to be converted into a binary image. Some binarization processes greatly benefit from an a priori enhancement, on the other hand, some enhancement algorithms directly produce a binary output, and therefore the distinction between enhancement and binarization is sometimes faded. The binary images are usually submitted to a thinning stage which allows for the ridge line thickness to be reduced to one pixel, resulting in a skeleton image (Fig.III.22). A simple image scan then allows the detection of pixels that correspond to minutiae

Some authors have proposed minutiae extraction approaches that work directly on the gray-scale images without binarization and thinning. This choice is motivated by the following considerations:

• A significant amount of information may be lost during the binarization process.

• Binarization and thinning are time consuming; thinning may introduce a large number of spurious minutiae.

• In the absence of an a priori enhancement step, most of the binarization techniques do not provide satisfactory results when applied to low-quality images.

III.15.1. Direct gray-scale extraction [21]:

Given a starting point [xc, yc] and a starting direction θc, the ridge line following algorithm (see Fig.III.19) computes a new point [xt, yt] at each step by moving μ pixels from the current point [xc ,yc] along direction θc. Then it computes the section set Ω as the set of points belonging to the section segment lying on the xy-plane with a median point [xt, yt], direction orthogonal to θc and length 2σ+1. A new point [xn, yn], belonging to the ridge line, is chosen among the local maxima of an enhanced version of the set Ω. The point [xn, yn] becomes the current point [xc,yc] and a new direction θc is computed (Fig.III.19).

The optimal value of the parameters μ and σ can be determined according to the average thickness of the ridge lines. The algorithm runs until one of the four stopping criteria becomes true. In particular, when a ridge line terminates or intersects another ridge line (location of a minutia), the algorithm stops and returns the characteristics (type, coordinates and direction) of the detected minutia. The ridge line following algorithm extracts a ridge line, given a starting point and a direction. By exploiting such an algorithm, it is possible to define a schema for extracting all the ridge lines in an image and, consequently, detect all the minutiae. The main problems arise from the difficulty of examining each ridge line only once and locating the intersections with the ridge lines already extracted. For this purpose, an auxiliary image T of the same dimension as I is used. T is initialized by setting its pixel values to 0; each time a new ridge line is extracted from I, the pixels of T corresponding to the ridge line are labeled. The pixels of T corresponding to a ridge line are the pixels belonging to the ε-pixel thick polygonal chain, which links the consecutive maximum points [xn, yn] located by the ridge line following algorithm (Fig.III.20).

Let G be a regular square-meshed grid superimposed on the image I. For each node of G, the minutiae detection algorithm searches the nearest ridge line and tracks it by means of the ridge line following routine. Because the initial point can be anywhere in the middle of a ridge line, the tracking is executed alternately in both directions. The auxiliary image T, which is updated after each ridge line following, provides a simple and effective way to discover ridge line intersections and to avoid multiple trackings.

Fig.III.21 shows the results obtained by applying the minutiae detection algorithm to a sample fingerprint. **Maio** and **Maltoni** (1997) compared their method with four binarization thinning-based approaches and concluded that direct gray-scale extraction can significantly reduce processing time as well as the number of spurious minutiae resulting from thinning algorithms.

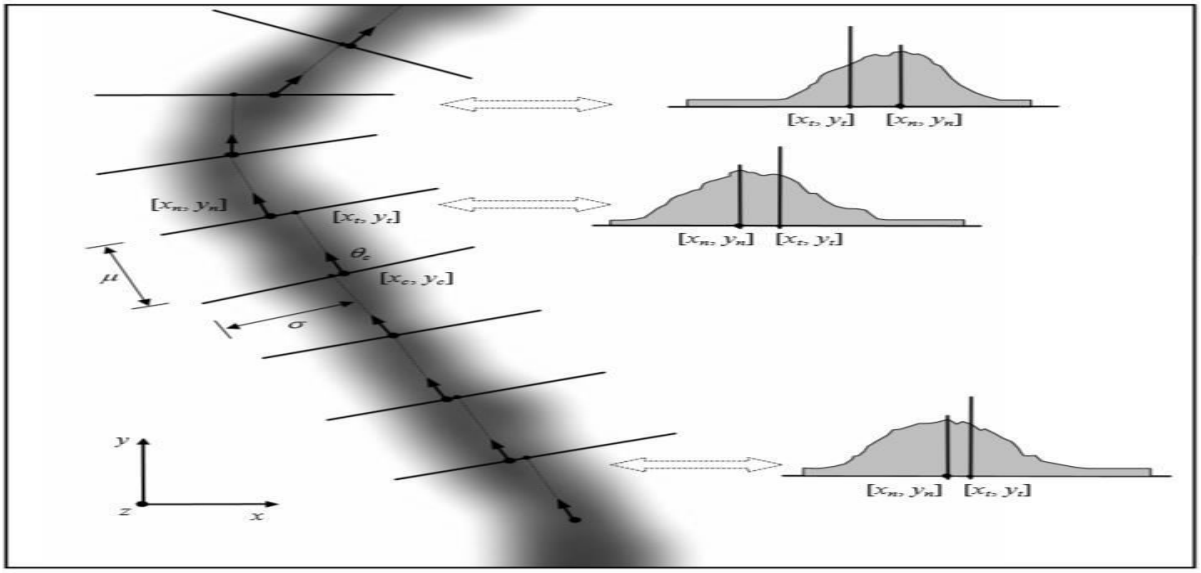


Fig.III.19. Some ridge line following steps (Maio and Maltoni (1997)). On the right, some sections of the ridge line are shown [21].

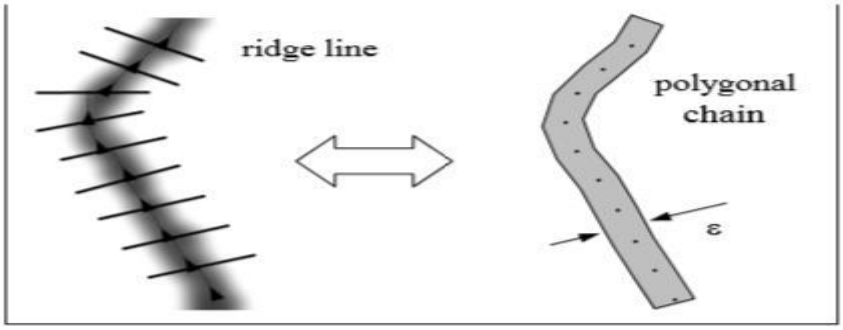


Fig.III.20. A ridge line and the corresponding ε -pixel thick polygonal chain (Maio and Maltoni (1997)) [21].

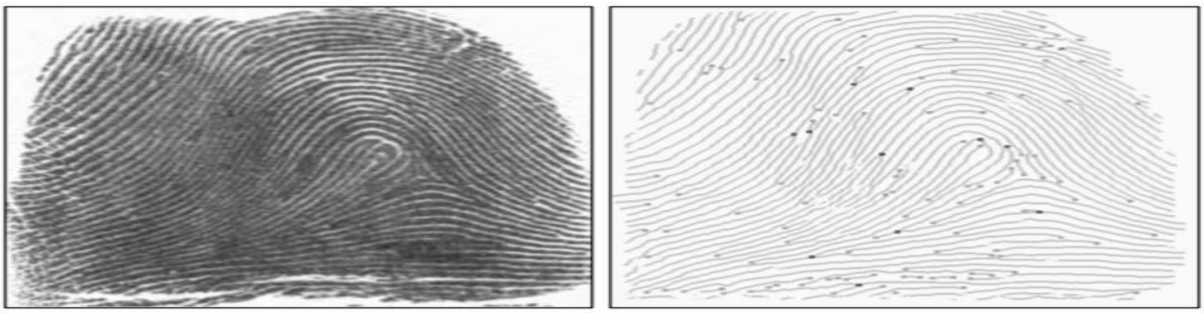


Fig.III.21. Minutiae detection on a sample fingerprint by using the Maio and Maltoni (1997) method [21].

Contextual filtering is performed on "touched" pixels during ridge line following. The ridge lines are represented through the corresponding polylines of T. Ridge ending minutiae are denoted by gray boxes and bifurcation minutiae are denoted by black boxes.

III.15.2. Binarization-based methods [21]:

The general problem of image binarization has been widely studied in the fields of image processing and pattern recognition (**Trier** and **Jain** (1995)). The simplest approach uses a global threshold t and works by setting the pixels whose gray-level is lower than t to 0 and the remaining pixels to 1.

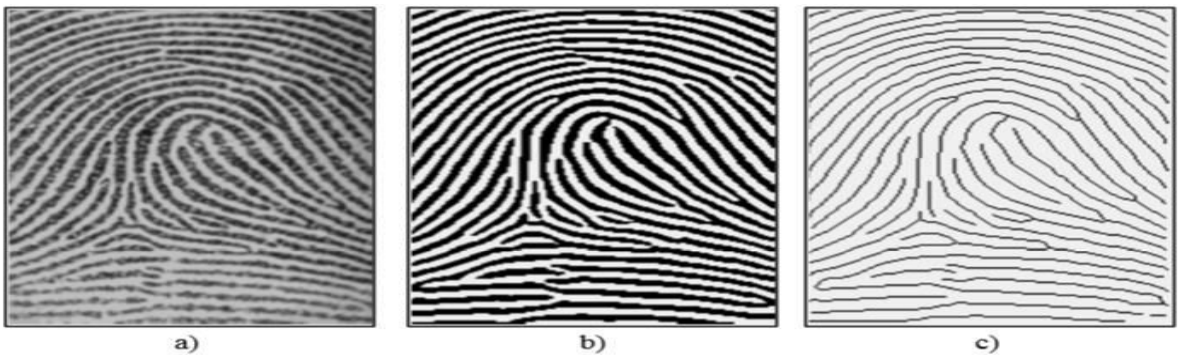


Fig.III.22. a) A fingerprint gray-scale image; b) the image obtained after binarization of the image in a); c) skeleton image obtained after a thinning of the image in b). Reprinted with permission from Maio and Maltoni (1997) [21].

In general, different portions of an image may be characterized by different contrast and intensity and, consequently, a single threshold for the entire image is not sufficient for a correct binarization. For this reason, the local threshold technique changes t locally, by adapting its value to the average local intensity. In the specific case of fingerprint images, which are sometimes of very poor quality, a local threshold method cannot always guarantee acceptable results and more effective fingerprint-specific solutions are necessary.

Once a binary skeleton has been obtained, a simple image scan allows the pixels corresponding to minutiae to be detected according to the ANSI/NIST-ITL 1 (2007) coordinate models: in fact the pixels corresponding to minutiae are characterized by a crossing number different from 2. The crossing number cn(p) of a pixel p in a binary image is defined (**Arcelli** and **Baja** (1984)) as half the sum of the differences between pairs of adjacent pixels in the 8-neighborhood of p:

$$\text{cn(p)} = \frac{1}{2}\sum_{i=1\ldots8} |val(p_{imod\,8}) - val(p_{i-1})|\} \qquad (6)$$

where p0, p1, ...p7 are the pixels belonging to an ordered sequence of pixels defining the eight- neighborhood of p and val(p) ∈{0,1} is the pixel value. It is simple to note (Fig.III.23) that a pixel p with val(p) = 1:

- Is an intermediate ridge point if cn(p) = 2.
- Corresponds to a ridge ending minutia if cn(p) = 1.
- Corresponds to a bifurcation minutia if cn(p) = 3.
- Defines a more complex minutia (e.g., crossover) if cn(p) > 3.

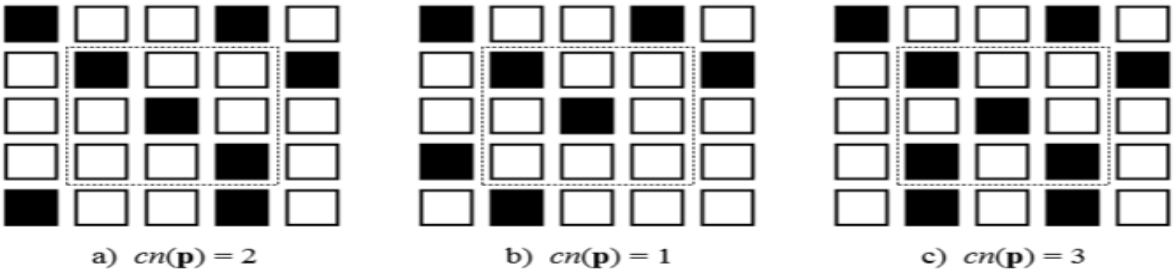


Fig.III.23. a) intra-ridge pixel; b) ridge ending minutia; c) bifurcation minutia [21].

III.16. Conclusion :

In this chapter we found that the Digital image processing is important ,because the image have a lot of information and we can replace many sensors with camera module, you can found all information of all sensors in one image ,this process reduce time processing and lower cost. All processing operations use the same basic elements in digital image processing like gray-scale transformation, histogram, segmentation, etc. in face recognition process we use haar-like cascade algorithm which use haar-like feature (line, edge and four rectangle) to detect human face, this algorithm use cascade classifier which is a cascading of weak classifiers used to boost the face detection process and reduce the computational complexity. Each node in the series contains a weak classifier and filter for one **Haar** feature. After the face is detected, next step is to extract features this is done using linear binary pattern algorithm. In fingerprint recognition process there are many algorithm, there is same algorithm extract fingerprint feature directly from gray-scale image, the other way use binarization process.

Chapter 4:

Conception And Realization

IV.1. Introduction:

In this chapter, we will talk about designing an attendance management system using Raspberry Pi as a controller, two sensors modules (fingerprint and webcam), an LCD and 4X4 matrix keypad, the system contain tow part hard and soft part, in hard part we will talk about how we connect the tow sensor modules, 4X4 matrix keypad and the LCD with Raspberry Pi and how does they work, in soft part we will explain our application and How to create a report by tow application. The device recognizes employers using three different methods face recognition, fingerprint recognition and ACN code recognition.

IV.2. Hard part:

IV.2.1. system block diagram:

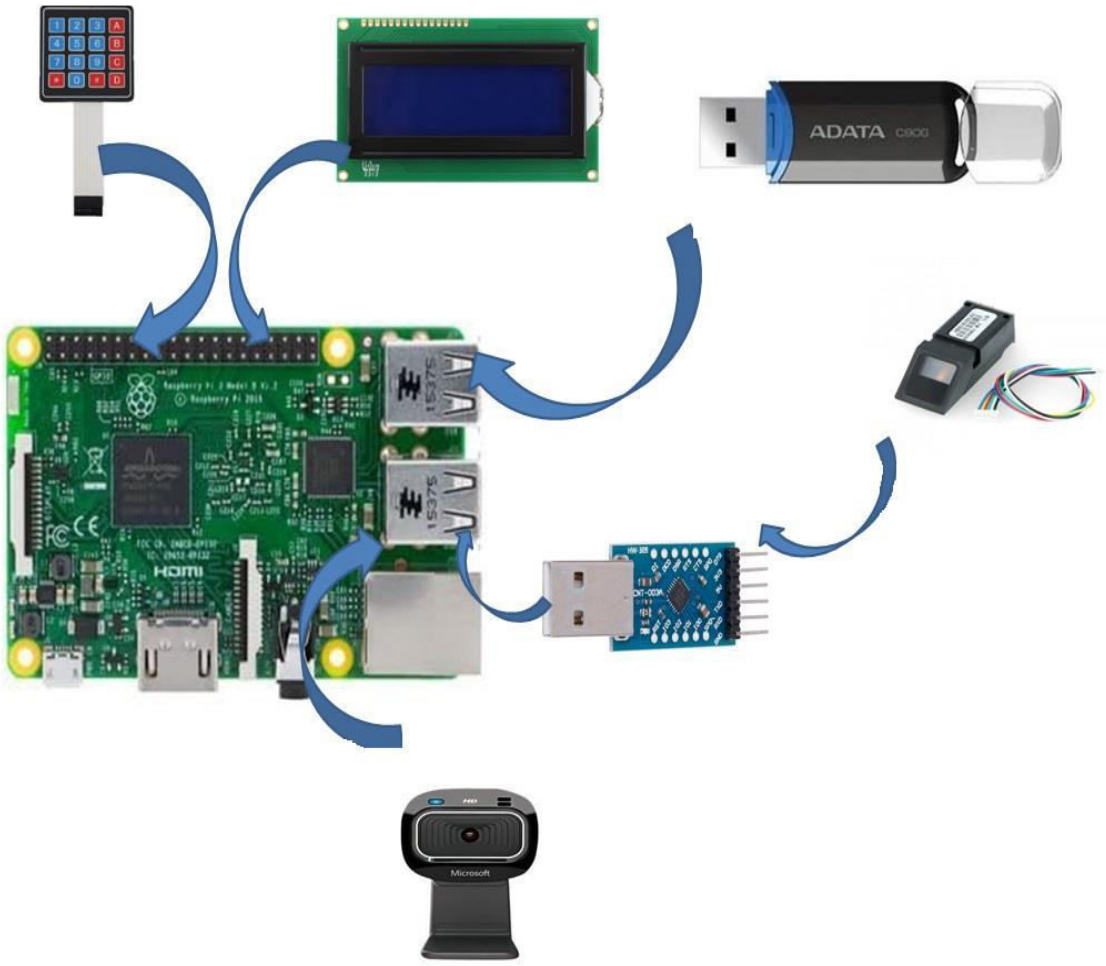


Fig.IV.1.System block diagram.

System block diagram shows us the general shape of the hard part, it consisting of a Raspberry Pi as control unit, a 4X4 matrix keypad, a LCD, a web camera, a fingerprint sensor, and a USB to UART converter.

The LCD and the 4X4 matrix keypad are connected to the GPIO pins of the Raspberry Pi, we will talk later on how to connect them, LCD is used as a display screen to display the available options such as recording attendance by face or by using an ACN code or taking an attendance registration file or menu, which contains the addition of an user or delete an employer or downloading employers file. The keypad is used to select one of the available options and it is also used to add a new employer or enter the number of the employer to be removed. USB flash memory is used to upload files, attendance records file and employers file. And to record via facial recognition, we used a web cam, so that it communicates with a Raspberry Pi via the USB port. To identify the fingerprint we used the fingerprint sensor, the connection is made between the fingerprint sensor and the control unit via a bridge, which is USB to UART converter, so that the RX and TX pins of the sensor are linked with the RX and TX of the converter, and the converter communicates with the control unit through the USB port, and this is to protect the sensor because Raspberry Pi pins dissolve high electric current. The good thing about the Raspberry Pi is that there are a lot of connection pins and several USB ports, and it is so easy to use that it can be considered as a mini computer.

The way the Hard part works is as follows: Attendance registration is carried out using 3 methods, by placing a fingerprint or interviewing the camera using the face or by pressing the button 1 and then entering the access number code. as shown in the figure below that show the available options, to create a report, an attendance record file must be taken via USB flash memory or Ethernet, so that USB flash memory is connected to the USB port, then press the button 3 on the keyboard, after pressing the button, two files are transferred, the first for the application that we designed and the second for another application. The fourth option menu contains two options, in order to add or delete a employer, when adding the employers file must be downloaded via USB flash memory so that after the connection first, when selecting the first option "Add", the first thing it will download is the employers file, then choose one of the new employer by pressing the "A" (top) or "D" (bottom) button. If the employer chooses, then press the OK button, but the employer must be present in front of the machine to take a fingerprint and pictures of his face to complete the addition process. To delete an employer, we press the second option button, then enter his number and press OK

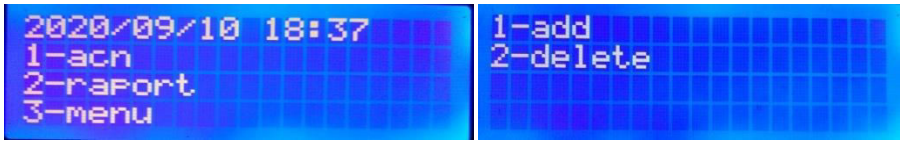


Fig.IV.2. The available options.

IV.2.2. Fingerprint sensor module:

Fingerprint sensor modules, like the one in the following figure, made fingerprint recognition more accessible and easy to add to your projects. This means that is super easy to make fingerprint collection, registration, comparison and search.

Fig.IV.3. Fingerprint sensor module.

The fingerprint sensor we are using is an Optical Type. The way this optical fingerprint sensor works is that it captures a photo of our finger ridges, and then it uses certain algorithm to match it with stored data and displays result of the same. These modules come with FLASH memory to store the fingerprints and work with any microcontroller or system with TTL serial. These modules can be added to security systems, door locks, time attendance systems, and much more.

Here's the specifications of the fingerprint sensor module we're using :

- Supply voltage: 3.6 - 6.0V DC
- Operating current: 120mA max
- Peak current: 150mA max
- Backlight color: green
- Fingerprint imaging time: <1.0 seconds
- Window area: 14mm x 18mm
- Template file: 512 bytes
- Storage capacity: 1000 templates
- Safety ratings (1-5 low to high safety)
- False Acceptance Rate: <0.001% (Security level 3)
- False Reject Rate: <1.0% (Security level 3)
- Interface: TTL Serial
- Baud rate: 9600, 19200, 28800, 38400, 57600 (default is 57600)
- Working temperature rating: -20C to +50C
- Working humidy: 40%-85% RH
- Full Dimensions: 56 x 20 x 21.5mm
- Exposed Dimensions (when placed in box): 21mm x 21mm x 21mm triangular
- Weight: 20 grams

Fingerprint sensor module has 6 pins:

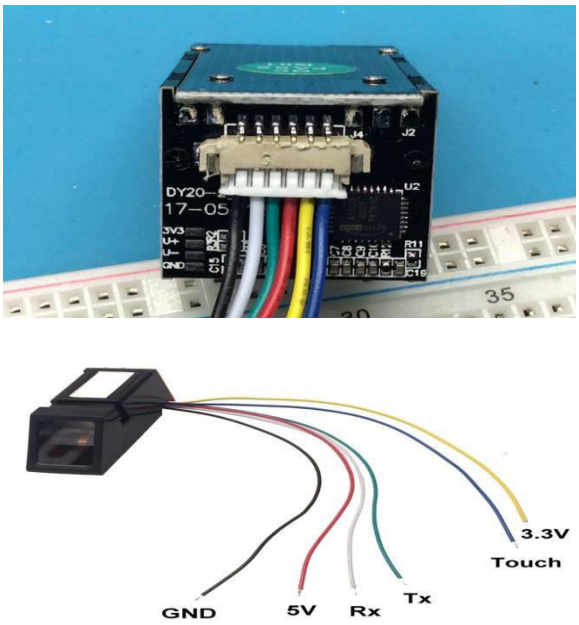


Fig.IV.4. Fingerprint sensor module mappings .

Pin	Pin Name	Type	Details
1	5V	Out	Regulated 5V DC (Read)
2	GND	---	Common Ground (Black)
3	TXD	Out	Data output - Connect to MCU RX (Green)
4	RXD	In	Data Input - Connect to MCU TX (White)
5	TOUCH	In	Active Low output when there is touch on sensor by finger(Blue)
6	3.3V	Out	Use this wire to give 3.3V to sensor instead of 5V (Yellow)

Table.IV.1.fingerprint sensor module pins.

IV.2.2.1. Working Principle :

Fingerprint processing includes two parts, fingerprint enrollment and fingerprint matching (the matching can be 1:1 or 1:N). When enrolling, user needs to enter the finger two times. The system will process the two time finger images, generate a template of the finger based on processing results and store the template. When matching, user enters the finger through optical sensor and system will generate a template of the finger and compare it with templates of the finger library.

For matching, system will compare the live finger with specific template designated in the Module; for 1:N matching, or searching, system will search the whole finger library for the matching finger. In both circumstances, system will return the matching result, success or failure.

IV.2.2.2. communication protocol :

Both UART and USB interfaces use a common serial communication protocol based on a packet format (the manual refers packets as "packages"). All data and commands are to be sent as data packets and all responses from the module will also be packets. So we need to frame data and commands as packets before sending out, and must extract data from response packets. The UART frame format is 10 bit with 1 start bit, 1 stop bit and 8 data bits.

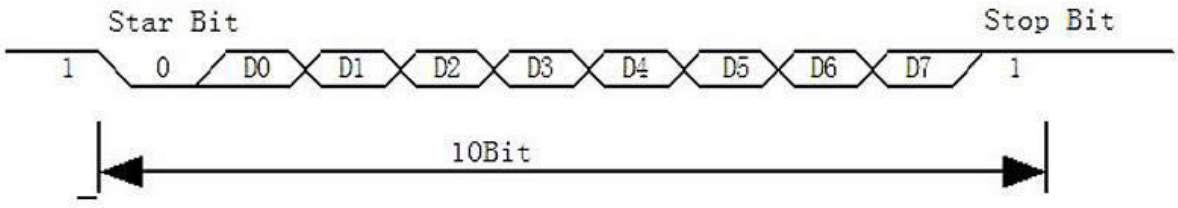


Fig. IV.5. R307 UART frame format.

The packet format is as follows (length in bytes is shown in brackets) :

Header (2)	Address (4)	Packet Identifier (1)	PacketLength (2)	Packet Content (Instruction/Data /Parameter)	Checksum (2)

Fig. IV.6.The packet format.

The UART frame is how a byte of data is transferred via the UART interface. A packet is a group of many such bytes (or frames).

1. **Header**: This indicates the start of a packet. It has to be the fixed value 0xEF01. It is 2 bytes long and the high byte is always transferred first.
2. **Address**: This is the 32-bit address of the scanner module. The module will accept instructions only if the address we are sending matches the address stored in the module. The default address is 0xFFFFFFFF and can be modifie.
3. **Packet Identifier**: This determines what type of packet we're sending or receiving. It is 1 byte long and depending on the value the packet types can be,
    - 0x01: The packet contains a command.
    - 0x02: Data packet. A data packet must be followed by a command packet or acknowledge packet.
    - 0x07: Acknowledge packet. It is sent by the module in response to a command.

- 0x08: End of data transfer packet. When we send large volume data such as an image, the data transfer will be terminated by this packet.

4. **Packet Length**: This is the total length of *Packet Content* and *Checksum* in bytes. Maximum length is 256 bytes and high byte is transferred first.
5. **Packet Content**: This can be data/command/parameters etc. of varying length. The Packet Length is the value that specifies the length of the data here in bytes.
6. **Checksum**: This is the arithmetic sum of all bytes in *Packet Identifier*, *Packet Length* and *Packet Content*. Overflowing bits are ignored. High byte is always transferred first.

IV.2.2.3. Hardware interface:

The module itself does all complex tasks behind reading and identifying the fingerprints with an on-board optical sensor and fingerprint algorithm. All you need to do is send it simple commands, and the fingerprint scanner can store different fingerprints.

The database of prints can even be downloaded from the unit and distributed to other modules. As well as the fingerprint template, the analyzed version of the print, you can also retrieve the image of a fingerprint and even pull raw images from the optical sensor.

Fig. IV.7. Fingerprint sensor module hardware interfacing.

Although a number of fingerprint sensor modules with slight variations are available now, most have a 4-pin or 6-pin external connection interface. By way of the serial interface, fingerprint sensor module can communicate with a Raspberry Pi runs on of 3.3V or 5V power supply. TX/TD pin of the module connects with RXD (RX pin of CP2104 HW-309), and RX/RD pin connects with TXD (TX pin of CP2104 HW-309).

IV.2.3. USB to UART converter:

This is an USB2.0 to TTL UART Converter module which is based on CP2104 Bridge by SiLabs. This module creates a virtual COM port using USB which can support various standard Baud Rates for serial communication. The CP2104 UART interface implements all RS-232/RS-485 signals. The feature which makes it more convenient is the TTL level data i/o. The Rx and Tx pin can be connected directly to the MCUs pins (assuming 5v i/o).

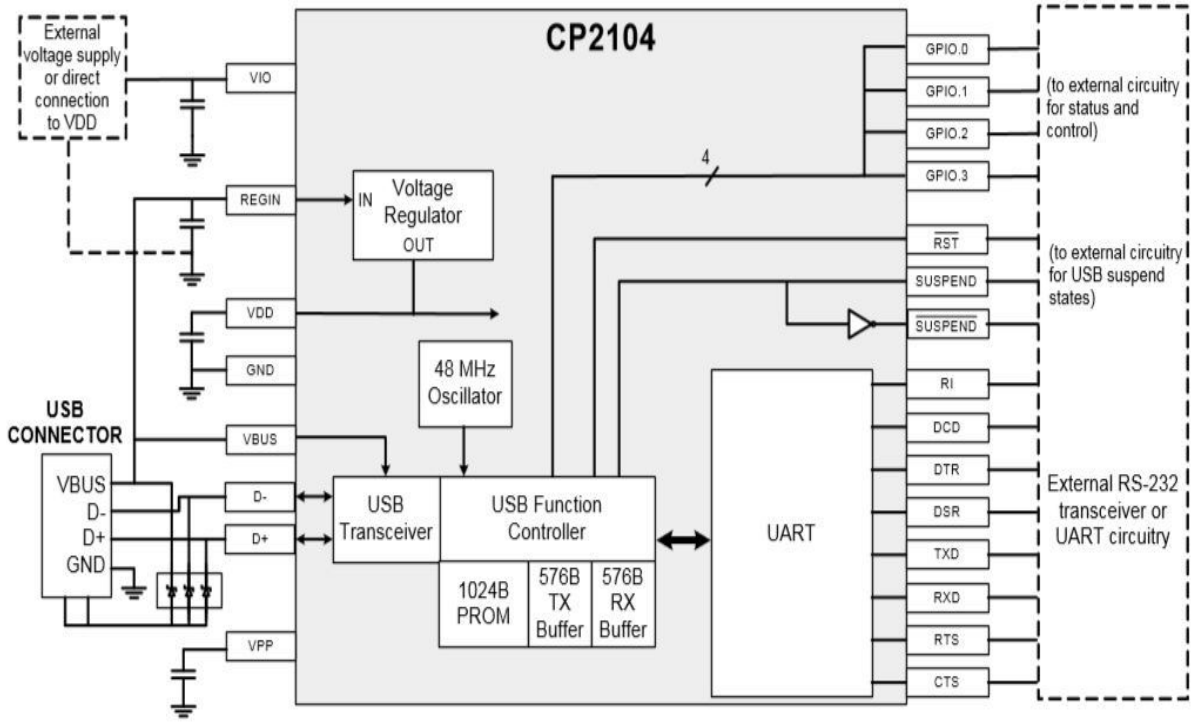


Fig. IV.8.CP2104 system diagram.

Pin	Type	Function
VDD	POWER	3.45 V voltage regulator output
VBUS	POWER	USB bus voltage (5 V)
GND	POWER	Ground
$\overline{\text{RST}}$	In	Device reset
TX	Out	Asynchronous serial data transmit
RX	In	Asynchronous serial data receive
CTS	In	“Clear to send” control input(often used with RTS)
RTS	Out	“Ready to send” control output(often used with CTS)

DSR	In	"Data set ready" control input (active low)(often used with DTR)
DTR	Out	"Data terminal ready" control output (active low)(often used with DSR)
DCD	In	"Data carrier detect" control input (active low)
RI	In	"Ring indicator" control input (active low)
SUSPEND	Out	Driven high when in USB suspend state
$\overline{\text{SUSPEND}}$	Out	Driven low when in USB suspend state(connected to read LED)
GPIO.00	I/O	User-configurable inputs or outputs(one-time programmable)
GPIO.01		
GPIO.02		
GPIO.03		

Table. IV.2. HW-309 CP2104 Module pins.

Fig. IV.9. HW-309 CP2104 Module.

This module has 6 pin breakout which includes :

- TXD = Transmit Output - Connect to Receive Pin (RXD) of Fingerprint sensor. This pin is TX pin of CP2104 on board.
- RXD = Receive Input - Connect to Transmit Pin (TXD) of Fingerprint sensor. This pin is RX pin of CP2104 on board.
- GND = Should be common to Fingerprint sensor ground.
- 3V3 = Optional output to power external circuit up to 50mA.
- 5V = Optional output to power external circuit up to 500mA
- DTR/RST = Optional output pin to reset external microcontrollers.

IV.2.4. Keypad:

There are numerous ways for users to input data on a Raspberry Pi. One of them is to use a 16-button keypad that contains the numbers from zero to nine as well as a few extra buttons. it have four rows and four columns.

To detect which button is pressed, the Raspberry Pi has to send a pulse to each of the four rows of the keyboard. When a user presses a button that's connected to the line which is currently pulled high, the corresponding column is also pulled high.

By decoding the combination of line and column, you can determine which button got pressed.

If a user, for example, presses the B button located on the second row in the fourth column, the Raspberry Pi detects this button press when it sends a pulse to the second line and then checks which of the four columns was pulled high.

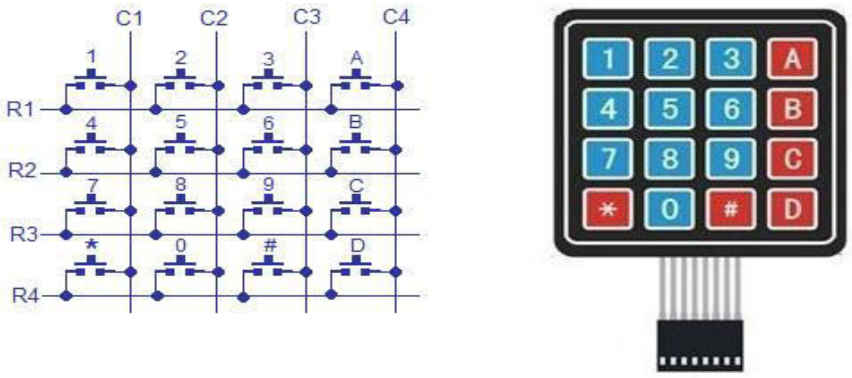


Fig. IV.10. 4X4 Matrix keypad circuit diagram.

We use 4X4 matrix keypad for :

- Select chooses (acn, send the attendance files or menu which has add or delete)
- Enter registration number
- Enter ACN code

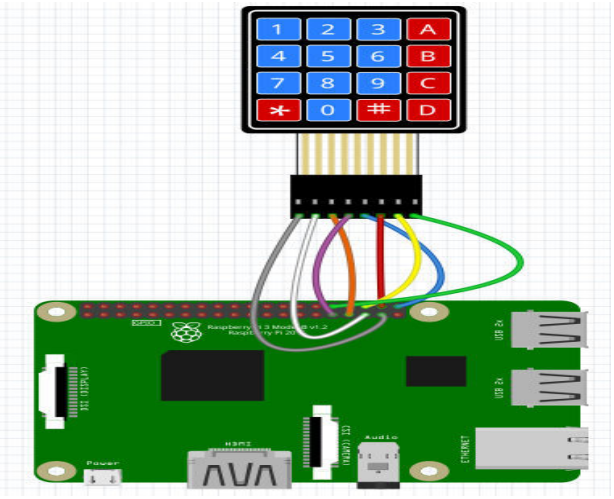


Fig. IV.11.The circuit connecting the 4X4 matrix keypad with Raspberry Pi 3B.

IV.2.5. LCD:

20×4 LCD modules are relatively easy and cheap to obtain. They have the same 16 pin interface as the 16×2 modules but still only require 6 GPIO pins on your Pi (an extra pin is required for the backlight switch).

Usually the device requires 8 data lines to provide data to Bits 0-7. However this LCD can be configured to use a "4 bit" mode which allows you to send data in two chunks (or nibbles) of 4 bits. This reduces the number of GPIO connections we need when interfacing with Raspberry Pi :

LCD Pin	Function	Pi Function	Pi Pin
01	GND	GND	P3-06
02	+5V	+5V	P3-02
03	Contrast		
04	RS	GPIO27	P3-13
05	RW	GND	P3-06
06	E	GPIO22	P3-15
07	Data 0		
08	Data 1		
09	Data 2		
10	Data 3		
11	Data 4	GPIO25	P3-22
12	Data 5	GPIO24	P3-18
13	Data 6	GPIO23	P3-16
14	Data 7	GPIO18	P3-12
15	+5V via 560 ohm		
16	GND		P3-06

Table. IV.3. LCD interfacing with Raspberry Pi .

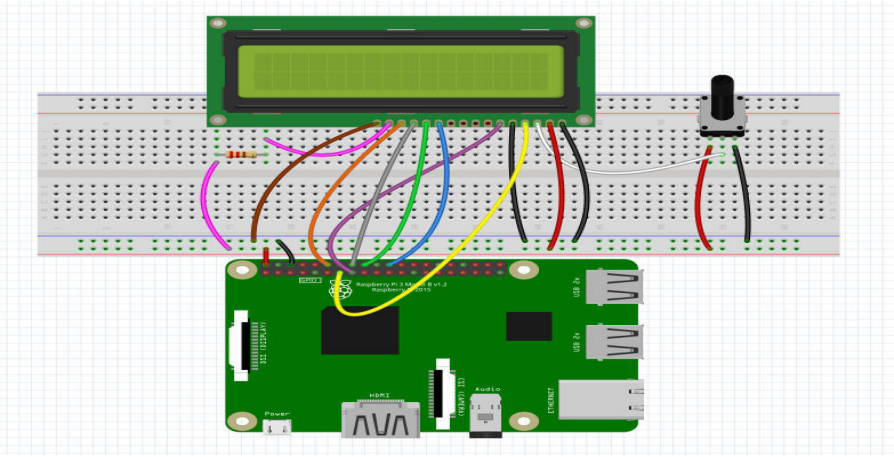


Fig. IV.12.The circuit connecting the LCD with Raspberry Pi 3B.

IV.2.6. Camera module :

We use Microsoft LifeCam HD-3000 Web Camera, 720p HD Video, Built-in Microphone, TrueColor Technology. The Microsoft LifeCam HD-3000 Web Camera allows you to post some clear pictures and videos as it uses CMOS sensor to give you 720p HD pictures. This full-featured webcam is capable of 720p video (1280 x 720 resolution) at 30 frames per second. Installing this webcam is so easy, simply use its USB 2.0 interface

Fig. IV.13. Microsoft LifeCam HD-3000 Web Camera.

IV.2.7. Electronic device:

The figure below show us the frontal and inside of the electronic device

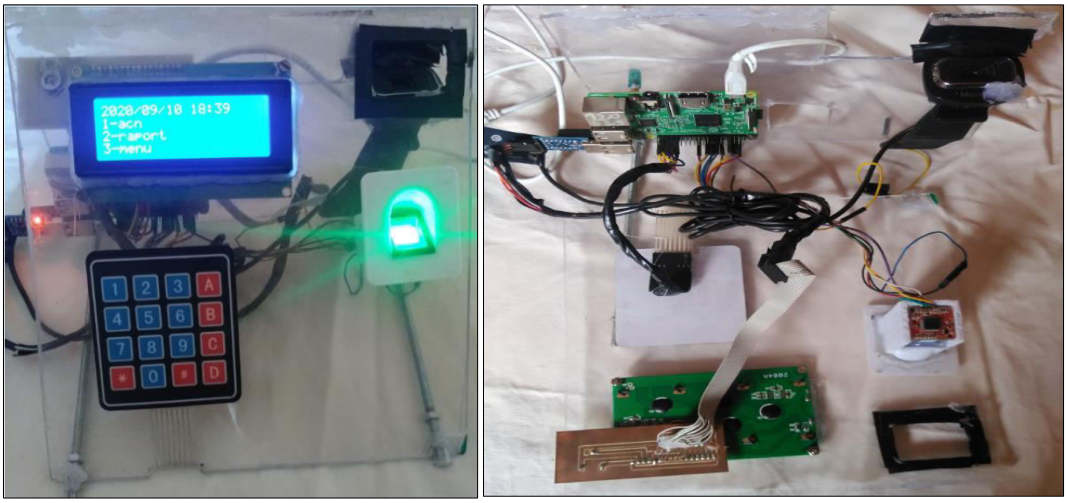
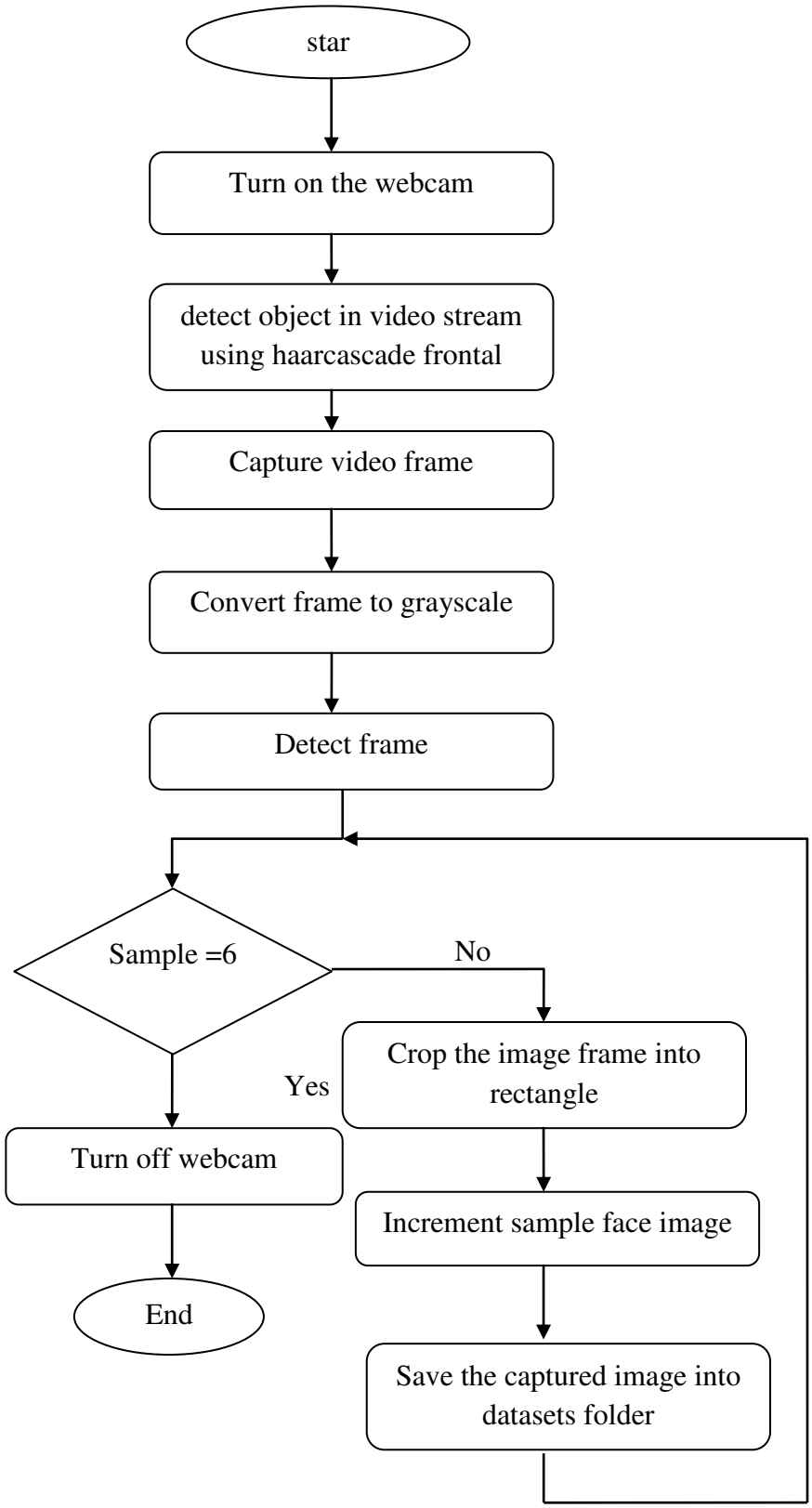


Fig.IV.14. The electronic device (frontal and inside the device).

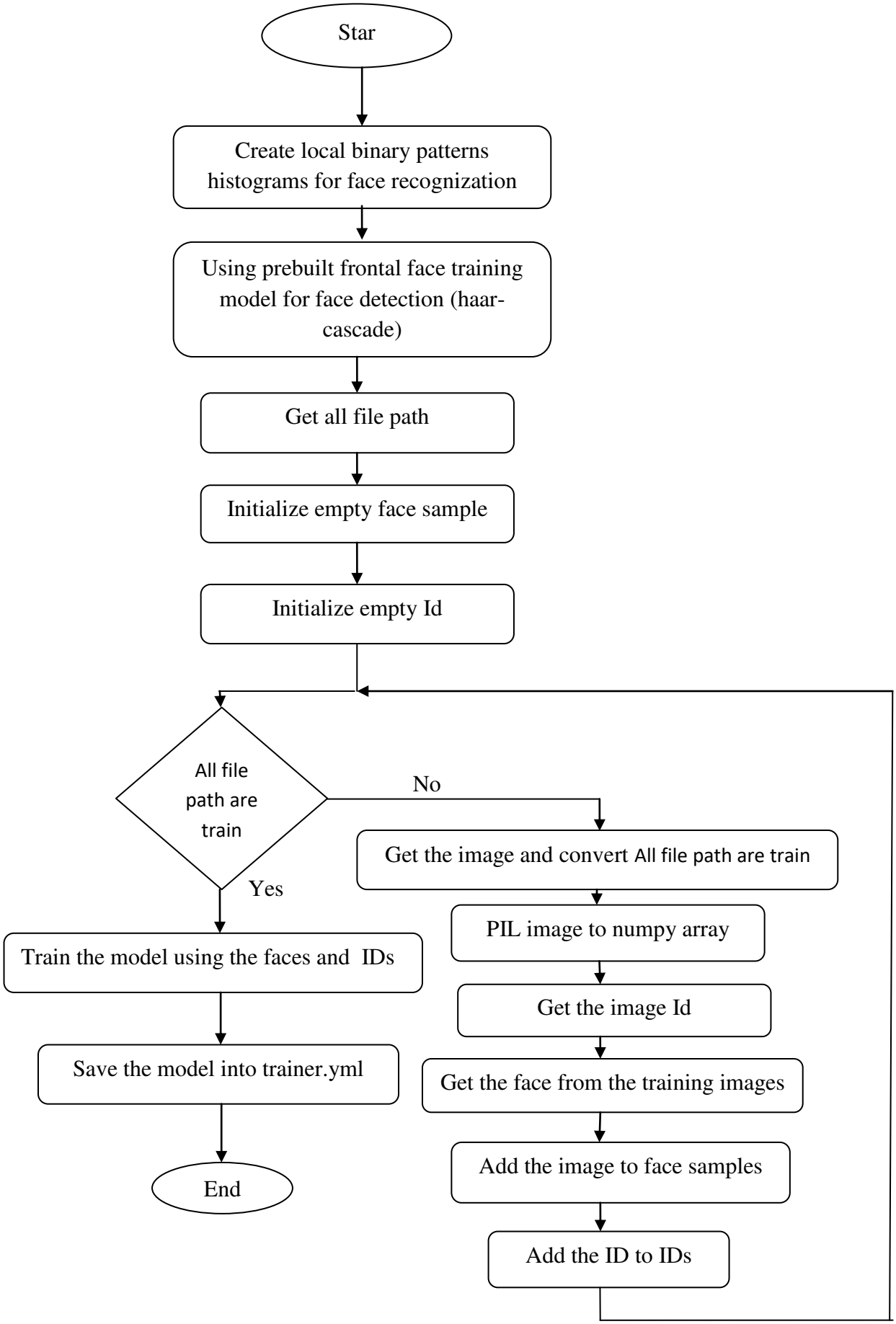
IV.3.Soft part :

We use python language for make a program on Raspberry Pi to control our system. We will explain how does the program work through the organigrams.

IV.3.1. Organigram of dataset (get pictures and save it):

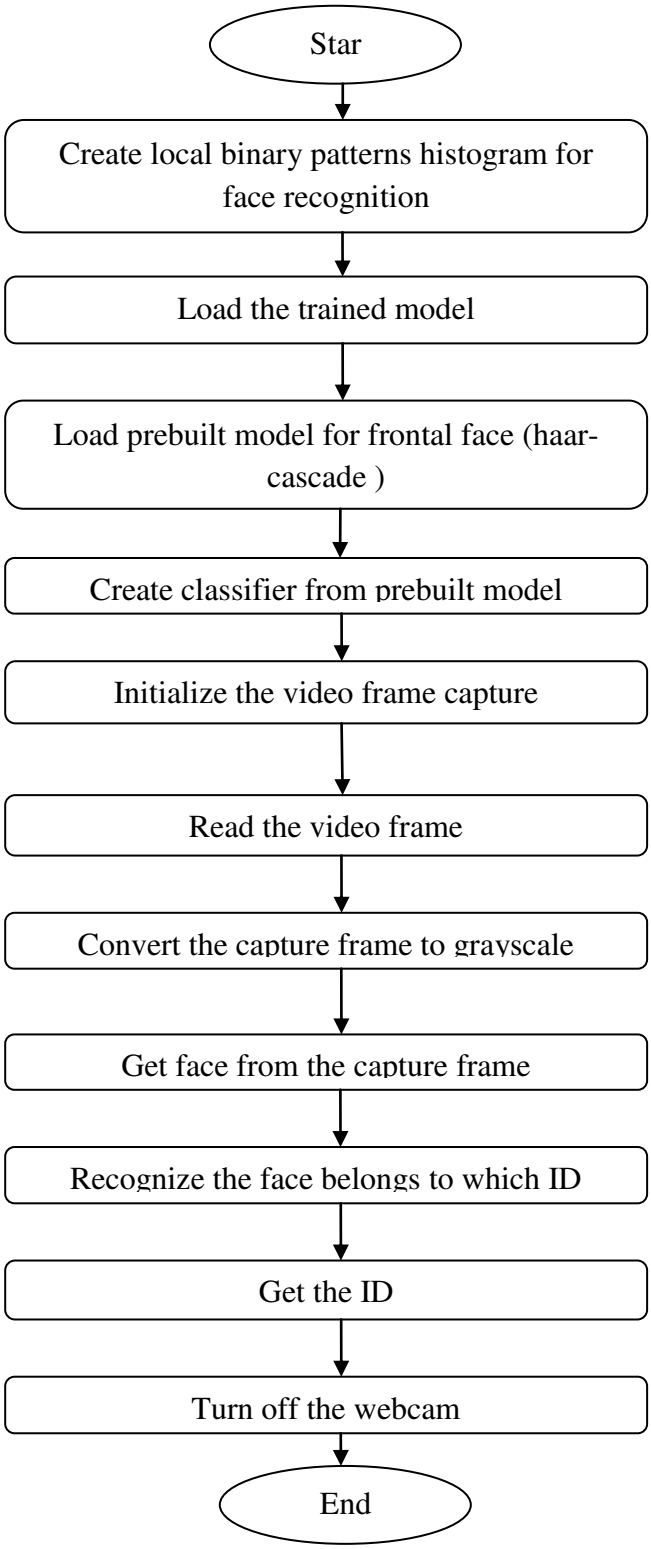


IV.3.2. Organigram of training for face recognition :

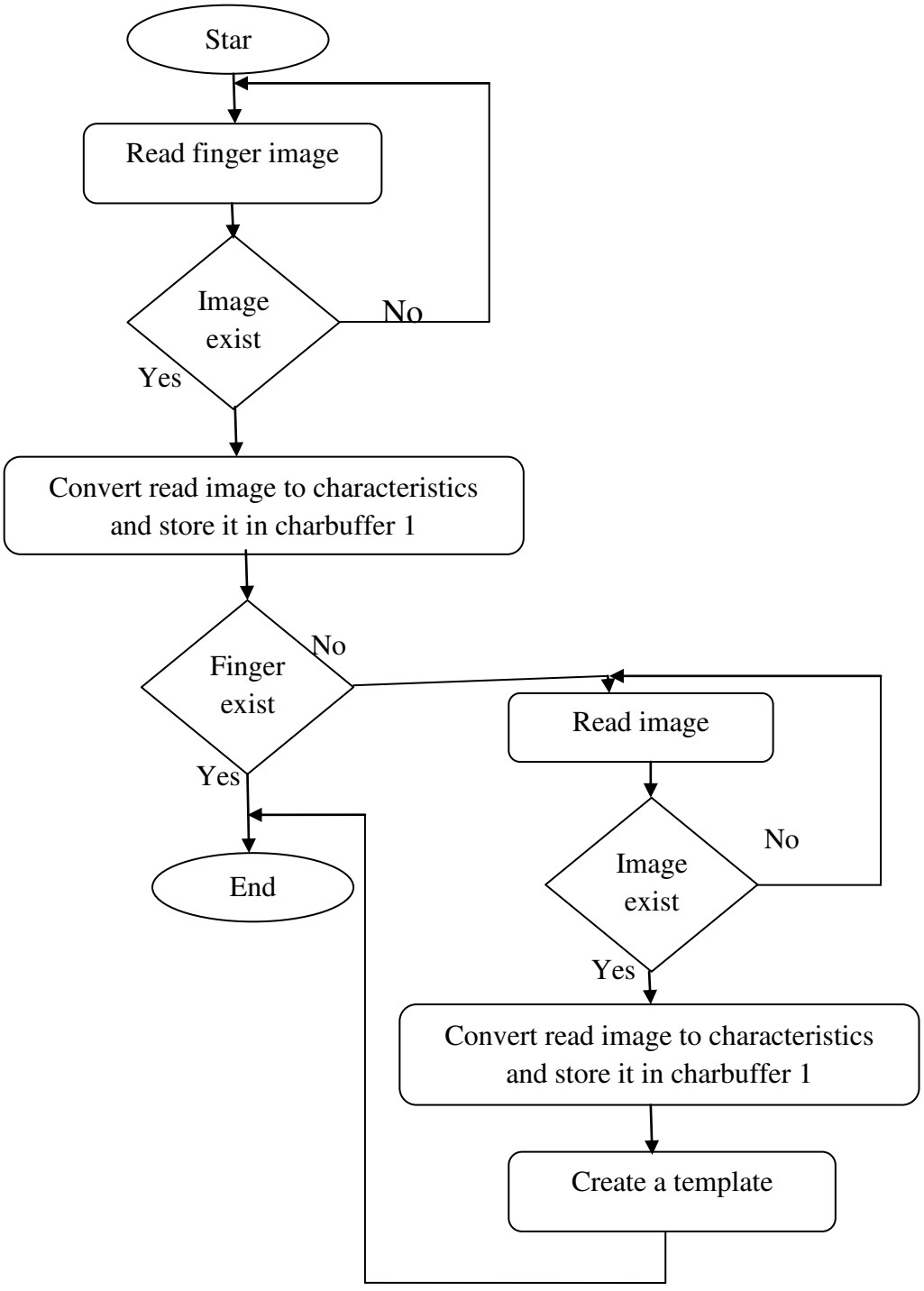


IV.3.3. Organigram of face recognition:

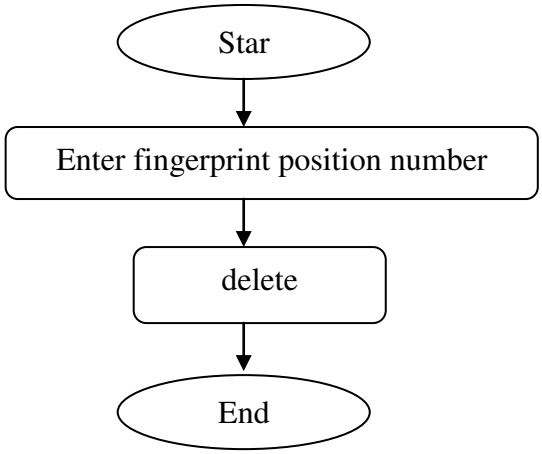
The human face is detected after converting the image frame into gray scale, then use the haarcascade classifier, the haarcascade classifier use three formations of Haar-like features. The algorithm of face recognition is composed of three main concepts: Haar-like features, integral Image, Adaboost training and Cascading classifier.



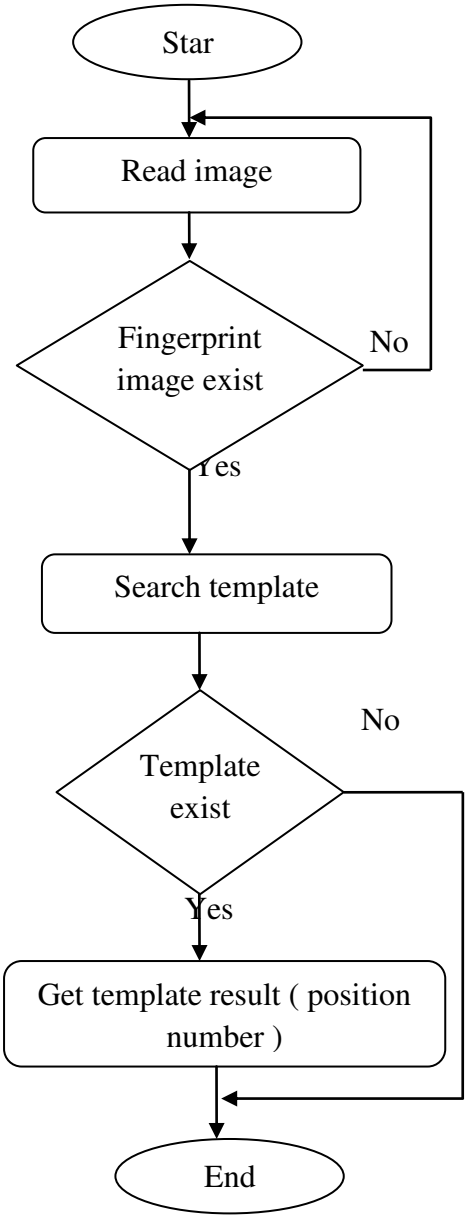
IV.3.4. Organigram of add fingerprint to fingerprint sensor module :



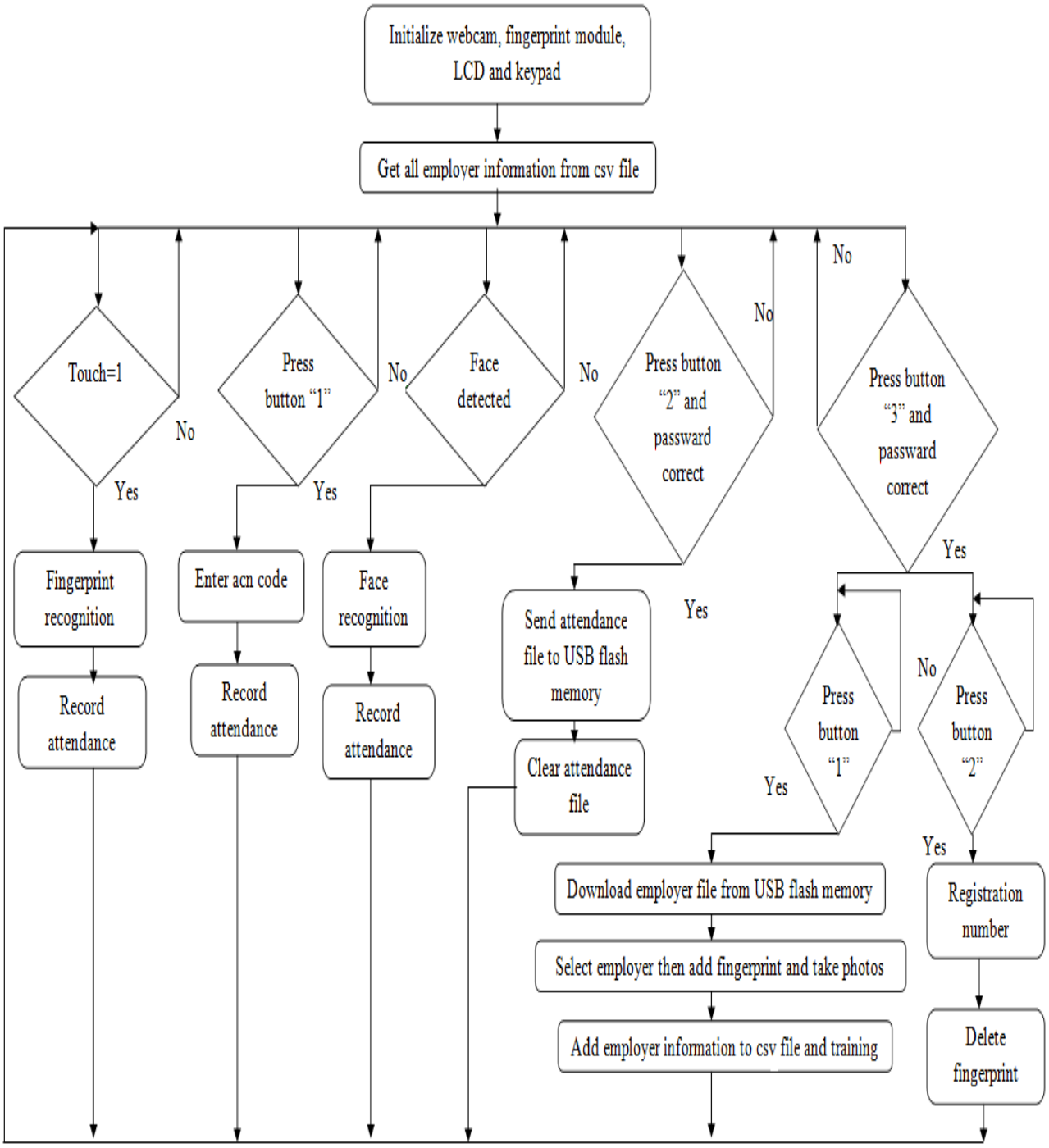
IV.3.5. Organigram of delete fingerprint from fingerprint sensor module :



IV.3.6. Organigram of fingerprint recognition :



IV.3.7. Organigram of all system :



IV.3.8. Attendance management application:

The Attendance management application is made by python language, and its function organize the registration times for each employer and make a report, the registration time is taking from attendance file. This application also enters the information of employers (name, ACN code, registration code, gender, address) and stores them in the data base that is in a local host, as for the information related to fingerprint and face, it is entered through a hard part. The communication with hard part is done using two methods, the first using USB flash memory and the second using Ethernet. The application contains three parts:

- Information part
- Report part
- Connection Part

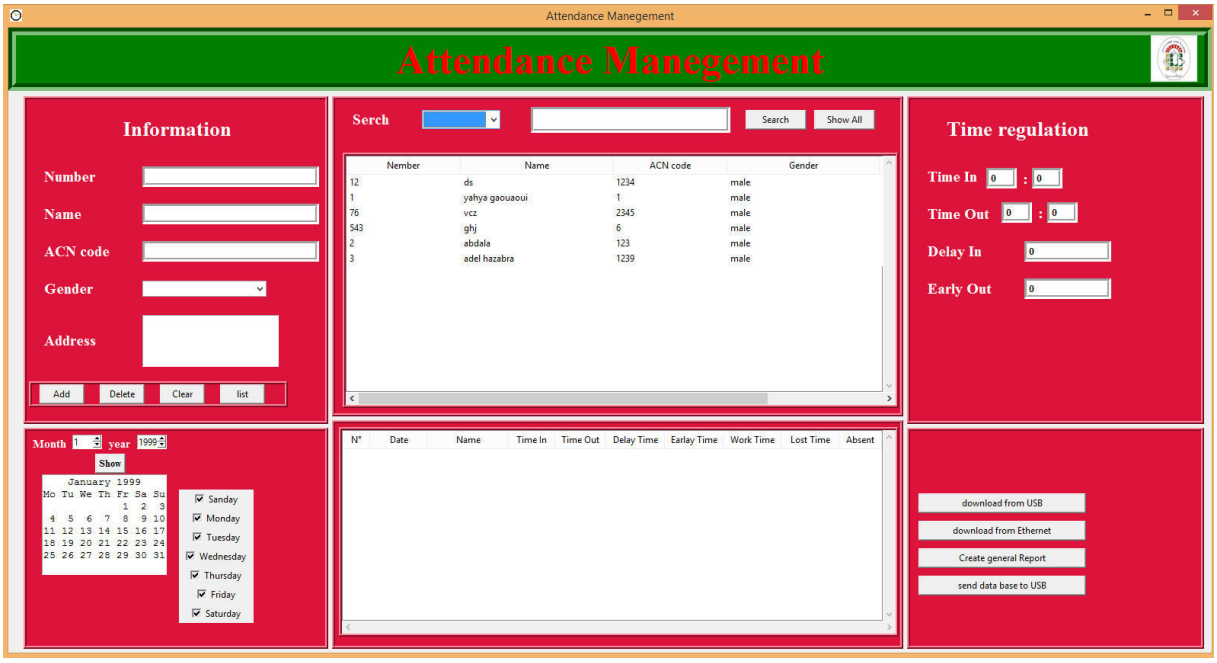


Fig.IV.15. Attendance management application.

IV.3.8.1. The information part:

Information about employers is entered through the input boxes, there are also 4 buttons (add, delete, clear and list), so that the “add” button stores the information entered in the data base and adds it to the information display table, and a “clear” button that cleans the input boxes, “delete” button deletes the information related to the employer to be deleted by clicking on the information box to be deleted in the information display table and then clicking the delete button, “list” button to create a Word file containing a list of all employers. As for the two buttons above the information display table (Search and Show All), the “Show All” button displays all the information stored in the data base, and a “search” button that searches on information related to a employer using one of the three information (name, number or ACN).

IV.3.8.2. The report part:

To generate a report, it is necessary to enter some information. So that the time of entry and exit, as well as the time of delay and early allowed, is entered through the input boxes(time in, time out, delay in, early out), and the month and year must also be entered and select working days. “Show” button displays the days of the month entered in the small schedule, as there is a table for organizing the recording times, the recording times file is taken through USB flash memory or Ethernet, by pressing the button download from USB or download from Ethernet, then the program calculates the work time and the lost time and records the presence or absence of the employer, presence is recorded when there is a time to register to enter and leave for that day, and absence is recorded when there are no registration times or one of the times (entry or leave) for that day. And the process of creating a general report is after the organization process is done by pressing the Create general Report button, so that it creates a excel file that contains all the information related to the attendance of employers in that month.

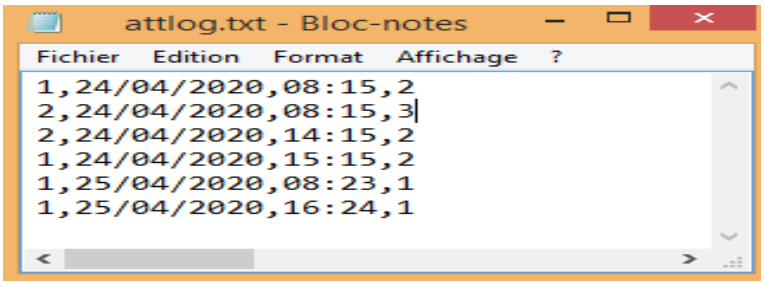


Fig.IV.16.example of attendance file.

The first parametre is the registration number then the date of attendance and the time , the number befor that is for the process of attendance (using face, fingerprint or ACN code).

Daily attendance																																		
Name	N°	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	work time	lost time
yahya gaouaoui	1	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	P	P	A	A	A	A		15:1	231:38
abdala	2	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	P	A	A	A	A	A		6:0	239:30
ds	12	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A		0	255:0
vcz	76	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A		0	255:0
ghj	543	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A		0	255:0
adel hazabra	3	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A		0	255:0

Fig.IV.17.example of report

IV.3.8.3. The connection Part :

The connection between Soft part and Hard Part is made to make report or add new employers.

- To take the attendance file, we can use USB flash memory or Ethernet connection.
- To add new employers, we send employers file through only the USB flash memory.

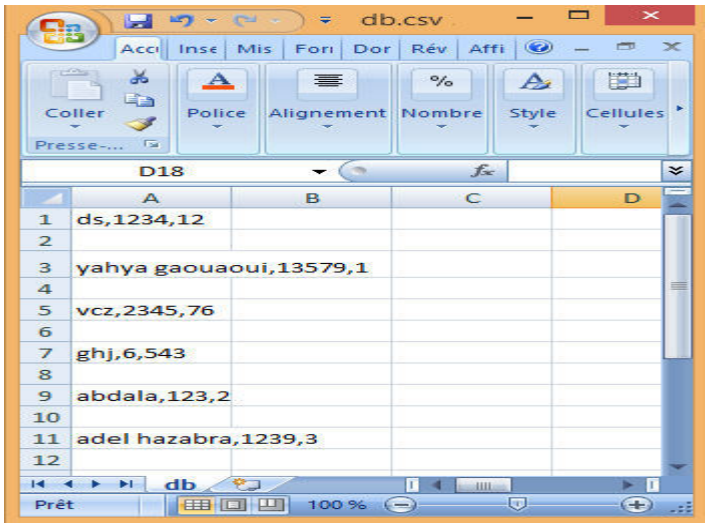


Fig.IV.18.example of employers file .

This scv file containe all employers, when we need to add emlployer or more it's necessary to transfer this file to hard part via USB flash memory, the first parametre is the name then acn code then the registration number.

IV.3.9. Create report by ZKTeco application :

we can also create report by an other application (ZKTeco), for that reason we create other attendance file for this application, ZKTeco application is more accurate.

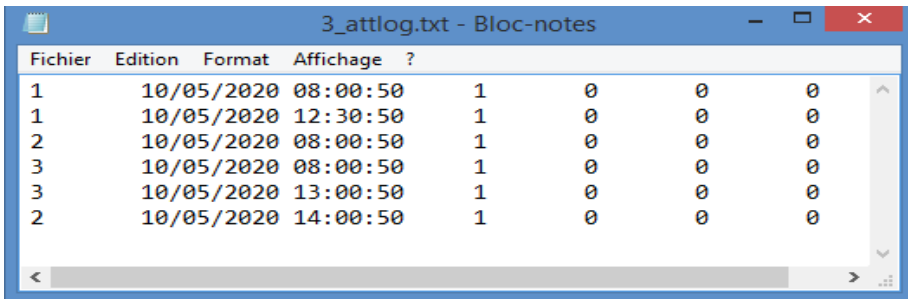


Fig. IV.19. Example of attendance file for ZKTeco application.

The first parametre is the user number then the date of attendance and the time, the number befor that is for the process of attendance (using face, fingerprint or ACN code) and there is three zeros at the end.

To create rapport it is take 6 steps :

1- Get attendance file from usb flash memory

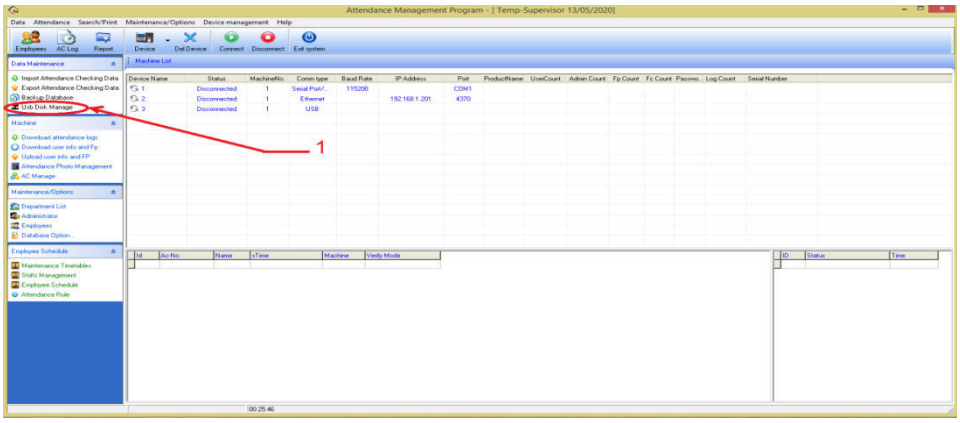


Fig. IV.20. Step 1 to create a report by ZKTeco application.

2- Select first item (5 Code) from new window then click on OK

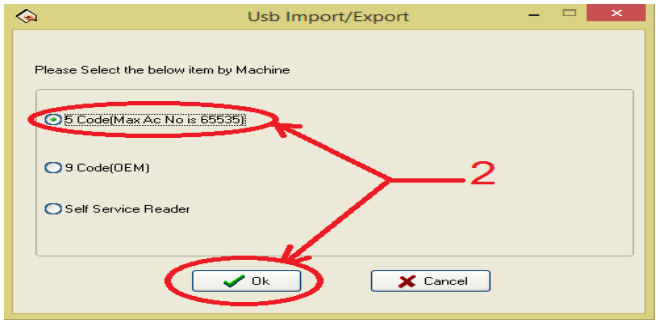


Fig. IV.21. Step 2 to create a report by ZKTeco application.

3- Select “Record data import”

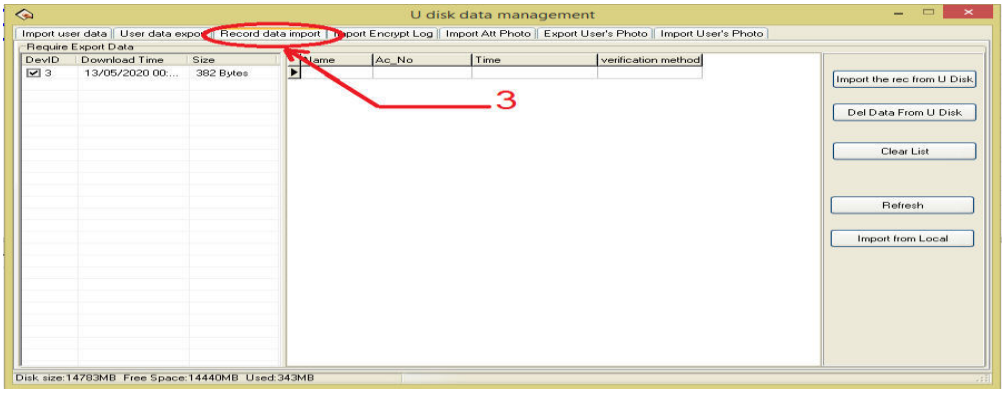


Fig. IV.22. Step 3 to create a report by ZKTeco application.

4- Then click on “emport the rec from U disk” and click on OK

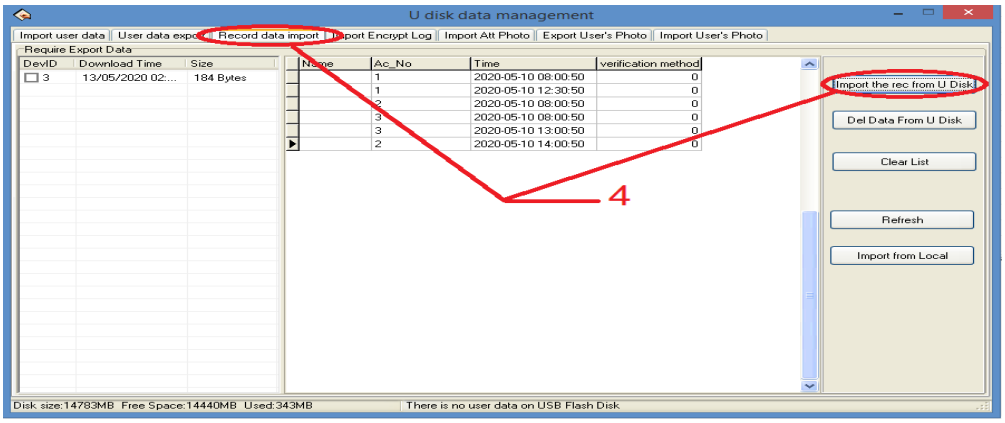


Fig. IV.23. Step 4 to create a report by ZKTeco application.

5- Click on “Report”

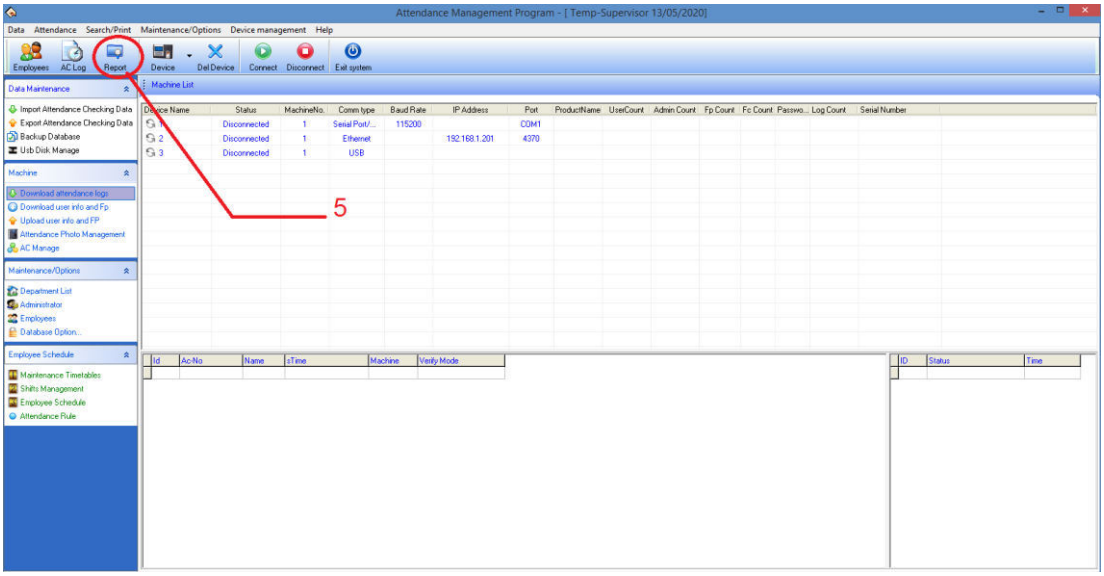


Fig. IV.24. Step 5 to create a report by ZKTeco application.

6- Finaly click on “Calculate”

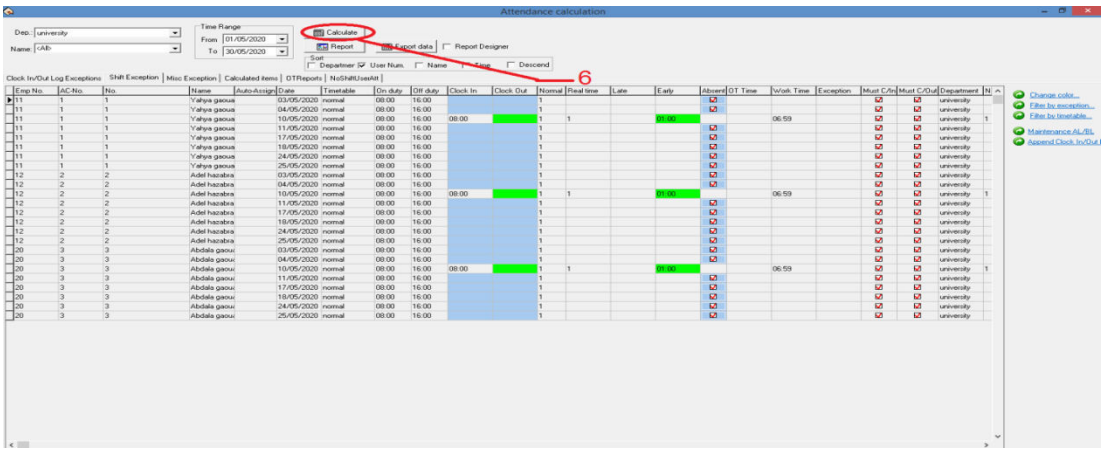


Fig. IV.25. Step 6 to create a report by ZKTeco application.

IV.4. Conclusion:

Attendance management system is a device that recognizes employers and records their entry and exit times, it consists of two parts, a hard part and a soft part. In our project we use for employers recognition three different ways face recognition, fingerprint recognition and ACN code, time is recorded by getting to know the employer in one of these three ways. The hard part do the recognition and record attendance time process, the soft part arrange the attendance times and make a report for each employer. The hard part contain tow sensor modules, 4X4 matrix keypad and LCD for display, they are controlled by Raspberry Pi, the face recognition process done by take a picture by camera module (webcam) then an algorithm of face recognition use for recognize the employer. The fingerprint sensor module recognize the fingerprint of employer by getting an image of his finger then extract its features and recognize it by an algorithm inside the module , each registered employer has an identifier number, for recognize employer and get his number we must send a serial UART command by Raspberry Pi. The ACN code entre by 4X4 matrix keypad. The soft part contain software application that arrange times and make a report by connecting with hard part by get report file from hard part to soft part using USB flash memory.

GENERAL CONCLUSION

Biometrics is increasingly becoming part of our daily lives, is becoming the new solution for companies and organizations to fight fraud in relation to arrival and departure times in addition to the time spent working within the companies.

This work represents a study of the attendance management system, which records the attendance of employee after the identification process. We designed a system using the Raspberry Pi control unit which is a minicomputer, this system identifies the employers through three ways, the first method is via fingerprint so that the fingerprint sensor module we used can carry 1000 Fingerprint, the second method is to recognize the face using a Haar classifier and it is possible to download more than 1000 pictures depending on the size of the SD card used, the third method is by using ACN code. This means that the system can carry 1000 employers.

The results of the recognition of the employers were very good, the rate of recognition of the employers through the fingerprint is more than 95%, the percentage of face recognition exceeds 90% and in every process of adding an employee the percentage increases because in every process of increase there is a training process.

Finally, we hope that this project will be developed in the future by adding an iris recognition pointing system.

Bibliography

[1]. Souhail Guennouni, Anass Mansouri and Ali Ahaitouf (March 1st 2019). Biometric Systems and Their Applications, Visual Impairment and Blindness - What We Know and What We Have to Know, Giuseppe Lo Giudice and Angel Catalá, IntechOpen, DOI: 10.5772/intechopen.84845. Available from: https://www.intechopen.com/books/visual-impairment-and-blindness-what-we-know-and-what-we-have-to-know/biometric-systems-and-their-applications

[2]. https://www.elprocus.com/different-types-biometric-sensors/

[3]. Aleksandra, Babich. "Biometric Authentication. Types of biometric identifiers Bachelor's Thesis: Degree Programme in Business Information Technology. Finland: HAAGA-HELIA University of Applied Sciences. 2012, 53 p

[4]. ZKTeco [**online**]. available from: https://www.zkteco.com

[5]. Sayah, Manel. "Système de pointage par empreinte digitale". Master thesis: embedded system. Biskra: Mohamed Khider University of Biskra, 2019, 98 p

[6]. Wikipedia [**online**]. modified 13 août 2020. Available from :

https://fr.wikipedia.org/wiki/Raspberry_Pi

[7]. Electronicwings [**online**]. Available from:

https://www.electronicwings.com/raspberry-pi/raspberry-pi-introduction

[8].Raspberry Pi [**online**]. Available from:

https://www.raspberrypi.org/documentation/hardware/

[9]. Amazon [**online**]. Available from:

https://www.amazon.fr/Raspberry-Pi-Carte-M%C3%A8re-Model/dp/B01CD5VC92

[10].Microsoft [**online**]. 08/28/2017. available from :

https://docs.microsoft.com/en-us/windows/iot-core/learn-about-hardware/pinmappings/pinmappingsrpi

[11]. Wasseem Nahy Ibrahem. "Image Processing [lecture 2]". Iraq: university of technology of Iraq. 2017, 13 p. available from: http://uotechnology.edu.iq/ce/Lectures/Image_Processing_4th/DIP_Lecture2.pdf

[12].Shammi, S. Muhammad, S. & Suraiya, B. "FUNDAMENTALS OF DIGITAL IMAGE PROCESSING AND BASIC CONCEPT OF CLASSIFICATION". International Journal of Chemical and Process Engineering Research. 2014, Vol. 1, No. 6, p. 98-108.

[13]. Milan, S. Vaclav, H., & Roger B. "Image Processing , Analysis, and Machine Vision" . 3rd ed. United States : Thomson Learning . 2008, p. 24-115

[14]. Gary, B. & Adrian K. "Learning OpenCV Computer Vision with the OpenCV Library". 1st ed. United States: O'Reilly Media. 2008, p. 506-507

[15]. Senthamizh Selvi.R , D.Sivakumar, et al."Face Recognition Using Haar - Cascade Classifier for Criminal Identification". International Journal of Recent Technology and Engineering. April 2019, Vol. 7, Issue-6S5, p.1871-1876

[16]. Willberger[**online**]. Jan 13, 2018. "Deep learning Haar – cascade explained". available from : www.willberger.org/cascade-haar-explained

[17]. Ayman A. Wazwaz, Amir O. Herbawi, Mohammad J.Teeti, Sajed Y.Hmeed. “Raspberry-Pi and Computers- Based Face Detection and Recognition System”, 4th International Conference on Computer and Technology Applications, pp. 01-03, 2018.

[18]. Souhail Guennouni, Ali Ahaitouf, Anass Mansouri . “Face Detection: Comparing Haar-like combined with Cascade Classifiers and Edge Orientation Matching”.

[19]. S L Suma, Sarika Raga. “Real Time Face Recognition of Human Faces by using LBPH and Viola Jones Algorithm”. International Journal of Scientific Research in Computer Science and Engineering. Oct. 2018, Vol.6, Issue.5, pp.01- 03.

[20]. Rabab M. Ramadan and Rehab F. Abdel - Kader. “Face Recognition Using Particle Swarm Optimization-Based Selected Features”. International Journal of Signal Processing, Image Processing and Pattern Recognition. Jun. 2009, Vol.6, No.2.

[21].Davide, M. Dario, M. Anil, K.J. & Salil, P. “Handbook of Fingerprint Recognition”. 2nd ed. London: Springer. 2009, p. 63-154.

Appendix:

Fingerprint Sensor Module

Fingerprint Identification Module

User Manual

I Introduction

Operation Principle

Fingerprint processing includes two parts: fingerprint enrollment and fingerprint matching (the matching can be 1:1 or 1: N).

When enrolling, user needs to enter the finger two times. The system will process the two time finger images, generate a template of the finger based on processing results and store the template. When matching, user enters the finger through optical sensor and system will generate a template of the finger and compare it with templates of the finger library. For 1:1 matching, system will compare the live finger with specific template designated in the Module; for 1: N matching, or searching, system will search the whole finger library for the matching finger. In both circumstances, system will return the matching result, success or failure.

II Main Parameters

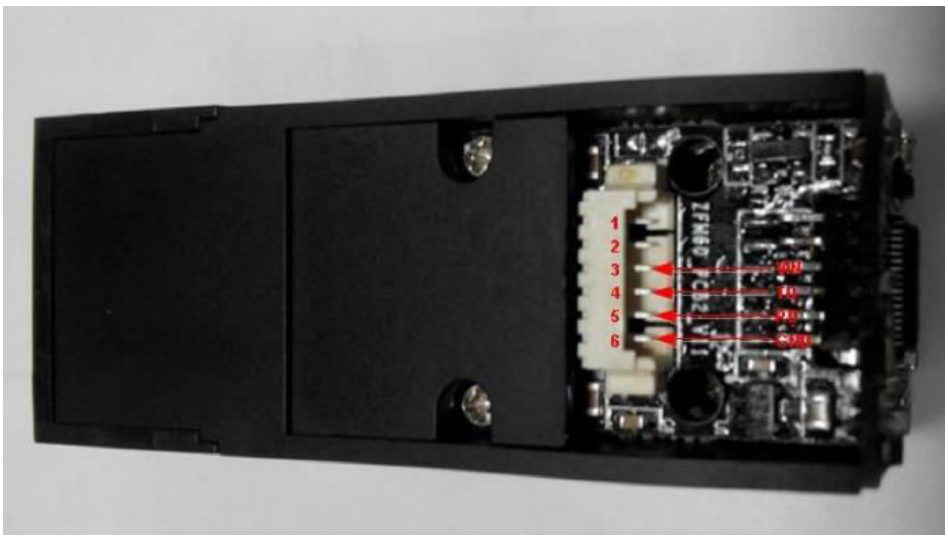
Power	DC 3.8V-7.0V	**Interface**	UART(TTL logical level)
Working current	Typical: ＜65mA Peak: ＜95mA	**Matching Mode**	1:1 and 1:N
Baud rate	(9600*N)bps, N=1～12 (default N=6）	**Character file size**	256 bytes
Image acquiring time	<1s	**Template size**	512 bytes
Storage capacity	1000	**Security level**	5 (1, 2, 3, 4, 5(highest))
FAR	<0.001%	**FRR**	<1.0%
Average searching time	< 1s (1:500)	**Window dimension**	14.5mm*19.4mm
Working environment	Temp: -20℃- +60℃	**Storage environment**	Temp: -40℃- +85℃
	RH: 40%-85%		RH: <85%
Outline Dimention	Integral type	54*20*20.5mm	

III Hardware Interface

3.1 Connecting with upper computer

3.1.1 Serial Communication

Pin Nmuber	Name	Type	Function Description
1	Vtouch	in	Touch induction power input（cable color: blue）
2	Sout	out	Induction signal output（cable color:yellow）
3	Vin	in	Power input（cable color: red）
4	TD	out	Data output. TTL logical level (cable color: green）
5	RD	in	Data input. TTL logical level (cable color: white)
6	GND	—	Signal ground. Connected to power ground (cable color: black）



3.1.1.1 Hardware connection

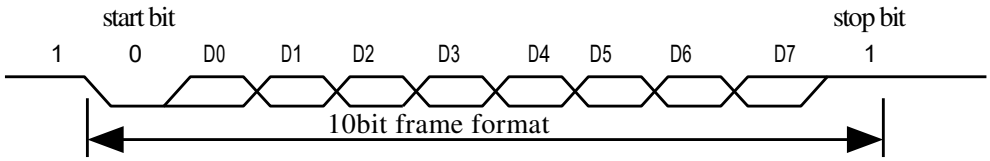
Via serial interface, the Module may communicate with MCU of 5V power: TD connects with RXD (receiving pin of MCU), RD connects with TXD (transferring pin of MCU).

Should the upper computer (PC) be in RS-232 mode, please add level converting circuit, like MAX232, between the Module and PC.

3.1.1.2 Serial communication protocol

The mode is semiduplex synchronism serial communication. And the default baud rate is 57600bps. User may set the baud rate in 9600～115200bps.

Transferring frame format is 10 bit: the low-level starting bit, 8-bit data with the LSB first, and an ending bit. There is no check bit.



3.1.1.3 Reset time

At power on, it takes about 300ms for initialization. During this period, the Module can't accept commands for upper computer.

Module initialized immediately after sending a byte (0x55) to the host computer, said module can already work normally and the receiving host computer instruction.

3.1.1.4 Electrical parameter (All electrical level takes GND as reference)

1. Power supply

Item	Parameter			Unit	Note
	Min	Typ	Max		
Power Voltage (Vin)	3.8		7.0	V	Normal working value.
Maximum Voltage (Vin_{max})	−0.3		9.0	V	**Exceeding the Maximum rating may cause permant harm to the Module.**
Operation Current (Icc)	90	110	130	mA	
Peak Current (Ipeak)			130	mA	

2. TD（output, TTL logic level）

Item	Condition	Parameter			Unit	Note
		Min	Typ	Max		
V_{OL}	I_{OL}=−4mA			0.4	V	Logic 0
V_{OH}	I_{OH}= 4mA	2.4		3.3	V	Logic 1

3. RD（input, TTL logic level）

Item	Condition	Parameter			Unit	Note
		Min	Typ	Max		
V_{IL}				0.6	V	Loigc 0
V_{IH}		2.4			V	Logic 1
I_{IH}	V_{IH} =5V		1		mA	
	V_{IH} =3.3V		30		uA	
V_{Imax}		−0.3		5.5	V	**Maximum input voltage**

IV System Resources

To address demands of different customer, Module system provides abundant resources at users use.

4.1 Notepad

The system sets aside a 512-bytes memory (16 pages* 32 bytes) for user's notepad, where data requiring power-off protection can be stored. The host can access the page by instructions of PS_WriteNotepad and PS_ReadNotepad.

Note: when write on one page of the pad, the entire 32 bytes will be written in wholly covering the original contents.

4.2 Buffer

There are an image buffer and two 512-byte-character-file buffer within the RAM space of the module. Users can read & write any of the buffers by instructions.

Note: Contents of the above buffers will be lost at power-off.

4.2.1 Image buffer

ImageBuffer serves for image storage and the image format is 256*288 pixels.

When transferring through UART, to quicken speed, only the upper 4 bits of the pixel is transferred (that is 16 grey degrees). And two adjacent pixels of the same row will form a byte before the transferring. When uploaded to PC, the 16-grey-degree image will be extended to 256-grey-degree format. That's 8-bit BMP format.

4.2.2 Character file buffer

Character file buffer, CharBuffer1, CharBuffer2, can be used to store both character file and template file.

4.3 Fingerprint Library

System sets aside a certain space within Flash for fingerprint template storage, that's fingerprint library. Contents of the library remain at power off.

Capacity of the library changes with the capacity of Flash, system will recognize the latter automatically. Fingerprint template's storage in Flash is in sequential order. Assume the fingerprint capacity N, then the serial number of template in library is 0, 1, 2, 3……N-2, N-1. User can only access library by template number.

4.4 System Configuration Parameter

To facilitate users developing, Module opens part system parameters for use. And the basic instructions are SetSysPara & ReadSysPara. Both instructions take Parameter Number as parameter.

When upper computer sends command to modify parameter, Module first responses with original configurations, then performs the parameter modification and writes configuration record into Flash. At the next startup, system will run with the new configurations.

4.4.1 Baud rate control (Parameter Number: 4)

The Parameter controls the UART communication speed of the Module. Its value is an integer N, N= [1, 12]. Corresponding baud rate is 9600*N bps.

4.4.2 Security Level (Parameter Number: 5)

The Parameter controls the matching threshold value of fingerprint searching and matching. Security level is divided into 5 grades, and corresponding value is 1, 2, 3, 4 and 5. At level 1, FAR is the highest and FRR is the lowest; however at level 5, FAR is the lowest and FRR is the highest.

4.4.3 Data package length (Parameter Number: 6)

The parameter decides the max length of the transferring data package when communicating with upper computer. Its value is 0, 1, 2, 3, corresponding to 32 bytes, 64 bytes, 128 bytes, 256 bytes respectively.

4.5 System status register

System status register indicates the current operation status of the Module. Its length is 1 word, and can be read via instruction *ReadSysPara.* Definition of the register is as follows:

Bit Num	15 4	3	2	1	0
Description	Reserved	ImgBufStat	PWD	Pass	Busy

Note:

- Busy：1 bit. 1: system is executing commands; 0: system is free;
- Pass：1 bit. 1: find the matching finger; 0: wrong finger;
- PWD：1 bit. 1: Verified devices handshaking password.
- ImgBufStat：1 bit. 1: image buffer contains valid image.

4.6 Module password

The default is 0x00000000，If the default password is not modified；If be modified through UART communication or password, the first instruction is the host computer and the communication module must be verify the password, only the password verification through, module can enter the normal working state, receiving other instructions（That is, the serial communication must first perform a handshake signal processing）. Password modification, new password stored in Flash, power cuts are still preserved（The modified password cannot be acquired through communication instructions, such as accidentally forgetting the modules cannot communicate, please kindly with）.

4.7 Module address

Each module has an identifying address. When communicating with upper computer, each instruction/data is transferred in data package form, which contains the address item. Module system only responds to data package whose address item value is the same with its identifying address.

The address length is 4 bytes, and its default factory value is 0xFFFFFFFF. User may modify the address via instruction *SetAdder*. The new modified address remains at power off.

V Communication Protocol

5.1 Data package format

When communicating, the transferring and receiving of command/data/result are all wrapped in data package format.

Data package format

Header	Adder	Package identifier	Package length	Package content (instruction/data/Parameter)	Checksum

Definition of Data package

Name	Symbol	Length	Description	
Header	START	2 bytes	Fixed value of EF01H; High byte transferred first.	
Adder	ADDR	4 bytes	Default value is 0xFFFFFFFF, which can be modified by command. High byte transferred first and at wrong adder value, module will reject to transfer.	
Package identifier	PID	1 byte	01H	Command packet;
			02H	Data packet; Data packet shall not appear alone in executing process, must follow command packet or acknowledge packet.
			07H	Acknowledge packet;
			08H	End of Data packet.
Package length	LENGTH	2 bytes	Refers to the length of package content (command packets and data packets) plus the length of Checksum (2 bytes). Unit is byte. Max length is 256 bytes. And high byte is transferred first.	
Package contents	DATA	—	It can be commands , data , command's parameters, acknowledge result, etc. (fingerprint character value, template are all deemed as data);	
Checksum	SUM	2 bytes	The arithmetic sum of package identifier, package length and all package contents. Overflowing bits are omitted. High byte is transferred first.	

VI Module Instruction System

6.1 System-related instructions

6.1.1 Verify password :VfyPwd

Description: The password verification module（Serial communication must be shake）

Input Parameter: control code 0

Return Parameter: confirmation code;

Instruction code: 13H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	4byte	2 bytes
Header	Chip address	Package identifier	Package length	Instruction code	Control code	Checksum
EF01H	xxxx	01H	0007H	13H	0	001BH

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Chip address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: Port operation complete;

Confirmation code=01H: error when receiving package;

Confirmation code=13H: fail to operate the communication port;

6.1.2 Set password : SetPwd

Description: Module password settings. (Refer to 4.6 for more information)

Input Parameter: PassWord;

Return Parameter: Confirmation code

Instruction code: 12H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	4byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	PassWord	Checksum
EF01H	xxxx	01H	0007H	12H	PassWord	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1	2 bytes
Header	Module address	Package identifier	Package length	Confirmation	Checksum
EF01H	xxxx	07H	0003H	xx	Sum

Note: Confirmation code=00H: OK;

Confirmation code=01H: error when receiving package;

6.1.3 Set Module address: SetAdder

Description: Set Module address.

Input Parameter: New Module address

Return Parameter: Confirmation code (1 byte)

Instruction code: 15H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	4 bytes	2 bytes
Header	Original Module address	Package identifier	Package length	Instruction code	New Module address	Checksum
EF01H	xxxx	01H	0007H	15H	xxxx	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	New Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	Sum

Note: Confirmation code=00H: address setting complete;

Confirmation code=01H: error when receiving package;

6.1.4 Set module system's basic parameter: SetSysPara

Description: Operation parameter settings. (Refer to 4.4 for more information)

Input Parameter: Parameter number+ Contents;

Return Parameter: Confirmation code (1 byte)

Instruction code: 0eH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1byte	1byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Parameter number	Contents	Checksum
EF01H	xxxx	01H	0005H	0eH	4/5/6	xx	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	Sum

Note: Confirmation code=00H: parameter setting complete;

Confirmation code=01H: error when receiving package;

Confirmation code=1aH: wrong register number;

Table6.1 Parameter number+ Contents

Name	Parameter number	Contents
Baud rate	4	N(N=1~12, Corresponding baud rate is 9600*N bps.)
Security level	5	N(1、2、3、4、5)
Data package length	6	N(0、1、2、3，corresponding length is 32、64、128、256（bytes）)

6.1.5 Read system Parameter: ReadSysPara

Description: Read Module's status register and system basic configuration parameters;（
Refer to 4.4 for system configuration parameter and 4.5 for system status register）.

Input Parameter：none

Return Parameter：Confirmation code (1 byte) + basic parameter（16bytes）

Instruction code: 0Fh

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	0fH	0013H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	16 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Basic parameter list	Checksum
EF01H	xxxx	07H	0013H	xxH	See following Table6.2	sum

Note: Confirmation code=00H: read complete;
Confirmation code=01H: error when receiving package;

Table6.2 system basic parameters

Name	Description	Offset (word)	Size (word)
Status register	Contents of system status register	0	1
System identifier code	Fixed value: 0x0000	1	1
Finger library size	Finger library size	2	1
Security level	Security level (1, 2, 3, 4, 5)	3	1
Device address	32-bit device address	4	2
Data packet size	Size code (0, 1, 2, 3)	6	1
Baud settings	N (baud = 9600*N bps)	7	1

6.1.6 Read the fingerprint template index table: ReadConList

Description: Reading fingerprint template index table and each time the most read 256 fingerprint template.
Input Parameter: Index page=0~3.
Index page 0 representative read 0 ~ 255 fingerprint template index table
Index page 1 representative read 256 ~ 511 fingerprint template index table
Index page 2 representative read 512 ~ 767 fingerprint template index table
Index page 3 representative read 768 ~ 1024 fingerprint template index table

Return Parameter: Confirmation code (1 byte) + template index table

Instruction code: 1fH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Index page	Checksum
EF01H	xxxx	01H	0004H	1FH	0/1/2/3	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	32 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Index table	Checksum
EF01H	xxxx	07H	0023H	xxH	See following Table6.3	sum

1：Confirmation code =0x00: Read index table success；
Confirmation code =0x01: error when receiving package;
2：Every time the most read 256 fingerprint template index data, the data is insufficient 256 bit "0".
3：Index table data structure: each of the 8 as a group, and each group consists of beginning a high output.

Table6.3 Index table data structure

Order of transmission	From low to high byte output, and each byte by beginning a high output.								
least significant byte	Template Number	7	6	5	4	3	2	1	0
	Template index table data	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
Template Number	Template Number	15	14	13	12	11	10	9	8
	Template index table data	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
…	...	…							
Most significant byte	Template Number	255	254	253	252	251	250	249	248
	Template index table data	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
Note: the index table data "0" on behalf of the corresponding position without a valid template; "1" represents the corresponding to the location of the effective template.									

6.1.7 Read valid template number: TempleteNum

Description: read the current valid template number of the Module
Input Parameter: none
Return Parameter: Confirmation code (1 byte)，template number N
Instruction code: 1dH
Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	1DH	0021H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Template number	Checksum
EF01H	xxxx	07H	0005	xxH	N	sum

Note: Confirmation code=00H: read complete;
Confirmation code=01H: error when receiving package;

6.2 Fingerprint-processing instructions

6.2.1 To collect finger image: GenImg

Description: detecting finger and store the detected finger image in ImageBuffer while returning successful confirmation code; If there is no finger, returned confirmation code would be" can't detect finger".
Input Parameter: none
Return Parameter: Confirmation code (1 byte)
Instruction code: 01H
Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	01H	0005H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	Sum

Note: Confirmation code=00H: finger collection success;
Confirmation code=01H: error when receiving package;
Confirmation code=02H: can't detect finger;
Confirmation code=03H: fail to collect finger;

6.2.2 Open the fingerprint lighting background LED : OpenLED

Description: Open the fingerprint lighting background LED
Input Parameter: none
Instruction code: 50H
Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	50H	0054H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Confirmation code=00H: operation success;
Confirmation code =others: operation failed

6.2.3 Close the fingerprint lighting background LED : CloseLED

Description: Close the fingerprint lighting background LED
Input Parameter: none
Instruction code: 51H
Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	51H	0055H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Confirmation code=00H: operation success;
Confirmation code =others: operation failed

6.2.4 Fingerprint get image free lighting : GetmageFree

Description: Fingerprint get image free lighting
Input Parameter: none
Instruction code: 52H
Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	52H	0056H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Confirmation code =00H: finger collection success;
Confirmation code =01H: error when receiving package;
Confirmation code =02H: sensor has no finger;
Confirmation code =03H: fail to collect finger

6.2.5 Handshake : GetEcho

Description: Instructions to the module to send to shake hands, if the module is working correctly, will return to the confirmation code 0x55, the computer can continue to send instructions to the module; if the confirmation code for other or no response, said equipment abnormal.
Input Parameter: none
Instruction code: 53H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	53H	0057H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Confirmation code=55H: The equipment is normal, can receive the command;
Confirmation code = Other or no response, said equipment abnormal.
In addition, module after power on automatically send 0x55 as handshake marks, MCU detected after 0x55, can immediately send the command to enter the working state.

6.2.6 Auto-login : AutoLogin

Description: Send the instruction, can make the module automatically complete the image acquisition, generation characteristics, synthetic template and the stored template work, To the four command line" To collect finger image (GenImg)", "To generate character file from image (Img2Tz)", "To generate template (RegModel) ", "To store template (Store)" into an instruction to complete.
Input Parameter: Fingerprint wait time + Number of times for pressing the fingerprint +Stored sequence number
Instruction code: 54H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Fingerprint wait time	Number of times for pressing the fingerprint	Stored sequence number	Repeated registration mark	Checksum
EF01H	xxxx	01H	0003H	54H	xxH	2/3	xxxx	0/1	sum

1. The fingerprint wait time is to wait the longest finger presses each image acquisition, if there is no finger presses on this parameter setting time, is that there is no finger. The domain values range from 1 to 255, the higher the value, the more time. The 70 series, usually taken as 54 (36H), corresponding to a time of 3.5 seconds, the other time intervals are listed in the following table:

Fingerprint wait time	Real time interval（s）	Fingerprint wait time	Real time interval（s）
31(1fH)	2	62(3eH)	4
38(26H)	2.5	69(45H)	4.5
46(2eH)	3	77(4dH)	5
54(36H)	3.5	85(55H)	5.5

2. Number of times for pressing the fingerprint is to Press the number of fingers to confirm the registration fingerprint, only the value of 2 or 3. Value of 2 represents the two press fingerprint recognition, value of 3 represents the 3 press fingerprint recognition.

3. Number of times for pressing the fingerprint is 2, the command will gather two fingerprint to register as a template, if collect the fingerprint success for the first time will sending a response code 56H (PS_AUTOLOGIN_OK1), then continue the second fingerprint process. Number of times for pressing the fingerprint is 3, the command will gather three fingerprint to register as a template, if collect the fingerprint success for the first time will sending a response code 56H (PS_AUTOLOGIN_OK1), and if collect the fingerprint success for the second time will sending a response code 57H (PS_AUTOLOGIN_OK2), then continue the third fingerprint process.

4. Repeated registration mark is set whether to allow repeated registration. 0 representative does not allow duplicate registration, i.e. if the current registration finger has been registered in the fingerprint database, then this will no longer register. 1 representative allows repeated registration, namely the current registration finger regardless of the fingerprint database whether registered, the registration of all.

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Confirmation code =00H: auto Login success

Confirmation code =02H: sensor has no finger;

Confirmation code=06H: fail to generate character file due to the over-disorderly fingerprint image;

Confirmation code=07H: fail to generate character file due to lackness of character point or over-smallness of fingerprint image;

Confirmation code =0aH: fail to combine the character files. That's, the character files don't belong to one finger;

Confirmation code =0bH: the stored sequence number exceeds the effective range;

Confirmation code =56H: the first finger collection success;

Confirmation code =57H: the Second finger collection success;

Confirmation code =24H: failure due to repeated registration（That is, the current registered fingerprint in fingerprint database already exists）

6.2.7 Auto-Search ：AutoSearch

Description：Send the instruction, can make the module automatically complete the image acquisition, generation characteristics and search fingerprint in the fingerprint template library work. To the three command line " To collect finger image (GenImg)", "To generate character file from image (Img2Tz)", " To search finger library (Search) " into an instruction to complete.

Input Parameter：Fingerprint wait time + Start page number + Search number.

Instruction code：55H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Fingerprint wait time	Start page number	Search number	Checksum
EF01H	xxxx	01H	0008H	55H	xxH	xxxx	xxxx	sum

The fingerprint wait time is to wait the longest finger presses each image acquisition, if there is no finger presses on this parameter setting time, is that there is no finger. The domain values range from 1 to 255, the higher the value, the more time. The 70 series, usually taken as 54 (36H), corresponding to a time of 3.5 seconds, the other time intervals are listed in the following table:

Fingerprint wait time	Real time interval（s）	Fingerprint wait time	Real time interval（s）
31(1fH)	2	62(3eH)	4
38(26H)	2.5	69(45H)	4.5
46(2eH)	3	77(4dH)	5
54(36H)	3.5	85(55H)	5.5

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Number	score	Checksum
EF01H	xxxx	07H	0007H	xxH	xxxx	xxxx	sum

Confirmation code =00H: search success;

Confirmation code =09H: search failed;

Confirmation code =02H: sensor has no finger;

Confirmation code=06H: fail to generate character file due to the over-disorderly fingerprint image;

Confirmation code=07H: fail to generate character file due to lackness of character point or over-smallness of fingerprint image;

Confirmation code =22H: residual fingerprint;

Confirmation code =23H: The specified interval does not exist an effective fingerprint template

6.2.8 Search fingerprints（With residual judgment）: SearchResBack

Description：to search the whole or part finger library for the template that matches the one in CharBuffer1 or CharBuffer2. When found, PageID will be returned. This command with Search (Instruction code = 04H) difference is for the remaining fingerprint return code is different, SearchResBack detected residual return code 22H, and the Search command detection residual return code 09H

Input Parameter：BufferID + StartPage + PageNum

Return Parameter：Confirmation code + Number（Match the fingerprint template）

Instruction code：56H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Buffer number	StartPage	Page number	Checksum
EF01H	xxxx	01H	0008H	56H	BufferID	StartPage	PageNum	sum

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively.

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Page number	Score	Checksum
EF01H	xxxx	07H	0007H	xxH	PageID	MatchScore	sum

Confirmation code =00H，search success；
Confirmation code =01H，error when receiving package;
Confirmation code =09H，search failed;
Confirmation code =22H，residual fingerprint

6.2.9 Upload image: UpImage

Description: to upload the image in ImageBuffer to upper computer.

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instruction code: 0aH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	0AH	000EH

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Data package format（have subsequent packet）:

2 bytes	4bytes	1 byte	2 bytes	N bytes	2 bytes
Header	Module address	Package identifier	Package length	Contents	Checksum
EF01H	xxxx	02H	N+2	Image data	Sum

End package format（have not subsequent packets）:

2 bytes	4bytes	1 byte	2 bytes	N bytes	2 bytes
Header	Module address	Package identifier	Package length	Contents	Checksum
EF01H	xxxx	08H	N+2	Image data	sum

Note 1： Confirmation code=00H: ready to transfer the following data packet;
Confirmation code=01H: error when receiving package;
Confirmation code=0fH: fail to transfer the following data packet;

2：Module shall transfer the following data packet and end packet after responding to the upper computer. And data packet and end packet no reply packet.
3.The value of N(number of bytes of the packet content) is determined by the length of the packet content, factory package content length is set to 128 bytes.

6.2.10 Download the image: DownImage

Description: to download image from upper computer to ImageBuffer. The image must be 256*288 size in BMP format.

Input Parameter: none

Return Parameter: Confirmation code (1 byte)

Instruction code: 0bH

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	0bH	000FH

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Data package format（have subsequent packet）:

2 bytes	4bytes	1 byte	2 bytes	N bytes	2 bytes
Header	Module address	Package identifier	Package length	Contents	Checksum
EF01H	xxxx	02H	N+2	Image data	Sum

End package format（have not subsequent packets）:

2 bytes	4bytes	1 byte	2 bytes	N bytes	2 bytes
Header	Module address	Package identifier	Package length	Contents	Checksum
EF01H	xxxx	08H	N+2	Image data	sum

Note: 1：Confirmation code=00H: ready to transfer the following data packet;

Confirmation code=01H: error when receiving package;

Confirmation code=0eH: fail to transfer the following data packet;

2：Module shall transfer the following data packet and end packet after responding to the upper computer.

3：The value of N (number of bytes of the packet content) is determined by the length of the packet content, factory package content length is set to 128 bytes.

6.2.11 To generate character file from image: Img2Tz

Description: to generate character file from the original finger image in ImageBuffer and store the file in CharBuffer1 or CharBuffer2.

Input Parameter: BufferID (character file buffer number)

Return Parameter: Confirmation code (1 byte)

Instruction code: 02H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Buffer number	Checksum
EF01H	xxxx	01H	0004H	02H	BufferID	sum

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively. Other values (except 1h, 2h) would be processed as CharBuffer2.

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: generate character file complete;
Confirmation code=01H: error when receiving package;
Confirmation code=06H: fail to generate character file due to the over-disorderly fingerprint image;
Confirmation code=07H: fail to generate character file due to lackness of character point or over-smallness of fingerprint image;
Confirmation code=15H: fail to generate the image for the lackness of valid primary image;

6.2.12 To generate template: RegModel

Description: To combine information of character files from CharBuffer1 and CharBuffer2 and generate a template which is stored back in both CharBuffer1 and CharBuffer2.
Input Parameter：none
Return Parameter：Confirmation code (1 byte)
Instruction code: 05H
Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	05H	09H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: operation success; Confirmation code=01H: error when receiving package;
Confirmation code=0aH: fail to combine the character files. That's, the character files don't belong to one finger.

6.2.13 To upload character or template: UpChar

Description: to upload the character file or template of CharBuffer1/CharBuffer2 to upper computer;
Input Parameter: BufferID (Buffer number)
Return Parameter: Confirmation code (1 byte)
Instruction code: 08H
Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Buffer number	Checksum
EF01H	xxxx	01H	0004H	08H	BufferID	sum

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively.

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Data package format（have subsequent packet）:

2 bytes	4bytes	1 byte	2 bytes	N bytes	2 bytes
Header	Module address	Package identifier	Package length	Contents	Checksum
EF01H	xxxx	02H	N+2	Template data	Sum

End package format（have not subsequent packets）:

2 bytes	4bytes	1 byte	2 bytes	N bytes	2 bytes
Header	Module address	Package identifier	Package length	Contents	Checksum
EF01H	xxxx	08H	N+2	Template data	sum

Note 1: Confirmation code=00H: ready to transfer the following data packet;
Confirmation code=01H: error when receiving package; Confirmation code=0dH: error when uploading template;

2: Module shall transfer following data packet after responding to the upper computer.;
3.The value of N(number of bytes of the packet content) is determined by the length of the packet content, factory package content length is set to 128 bytes.

4: The instruction doesn't affect buffer contents.

6.2.14 To download character file or template: DownChar

Description: to download character file or template from upper computer to the specified buffer of Module;

Input Parameter: BufferID (buffer number)

Return Parameter: Confirmation code (1 byte)

Instruction code: 09H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	buffer number	Checksum
EF01H	xxxx	01H	0004H	09H	BufferID	sum

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively.

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Data package format（have subsequent packet）:

2 bytes	4bytes	1 byte	2 bytes	N bytes	2 bytes
Header	Module address	Package identifier	Package length	Contents	Checksum
EF01H	xxxx	02H	N+2	Template data	Sum

End package format（have not subsequent packets）:

2 bytes	4bytes	1 byte	2 bytes	N bytes	2 bytes
Header	Module address	Package identifier	Package length	Contents	Checksum
EF01H	xxxx	08H	N+2	Template data	sum

Note 1:Confirmation code=00H: ready to transfer the following data packet;
Confirmation code=01H: error when receiving package;
Confirmation code=0eH: fail to receive the following data packages.
2: Module shall transfer the following data packet after responding to the upper computer.
3.The value of N(number of bytes of the packet content) is determined by the length of the packet content, factory package content length is set to 128 bytes.

6.2.15 To store template: Store

Description: to store the template of specified buffer (Buffer1/Buffer2) at the designated location of Flash library.

Input Parameter: BufferID(buffer number)+PageID（Flash location of the template, two bytes with high byte front and low byte behind）

Return Parameter: Confirmation code (1 byte)

Instruction code: 06H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	buffer number	Location number	Checksum
EF01H	xxxx	01H	06H	0006H	BufferID	PageID	sum

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively.

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: storage success;
Confirmation code=01H: error when receiving package;
Confirmation code=0bH: addressing PageID is beyond the finger library;
Confirmation code=18H: error when writing Flash.

6.2.16 To read template from Flash library: LoadChar

Description: to load template at the specified location (PageID) of Flash library to template buffer CharBuffer1/CharBuffer2

Input Parameter: BufferID(buffer number)+PageID (Flash location of the template, two bytes with high byte front and low byte behind)。

Return Parameter: Confirmation code (1 byte)Instruction code: 07H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	buffer number	Page number	Checksum
EF01H	xxxx	01H	0006H	07H	BufferID	PageID	sum

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively.

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: load success;
Confirmation code=01H: error when receiving package;
Confirmation code=0cH: error when reading template from library or the readout template is invalid;
Confirmation code=0BH: addressing PageID is beyond the finger library;

6.2.17 To delete template: DeletChar

Description: to delete a segment (N) of templates of Flash library started from the specified location (or PageID);
Input Parameter: PageID (template number in Flash)+ N (number of templates to be deleted)
Return Parameter: Confirmation code (1 byte)
Instruction code: 0cH
Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes	2bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Page number	number of templates to be deleted	Checksum
EF01H	xxxx	01H	0007H	0cH	PageID	N	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: delete success;
Confirmation code=01H: error when receiving package;
Confirmation code=10H: failed to delete templates;

6.2.18 To empty finger library: Empty

Description: to delete all the templates in the Flash library
Input Parameter: none
Return Parameter: Confirmation code (1 byte)
Instruction code: 0dH
Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	0dH	0011H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum

EF01H	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: empty success;
Confirmation code=01H: error when receiving package;
Confirmation code=11H: fail to clear finger library;

6.2.19 To carry out precise matching of two finger templates: Match

Description: to carry out precise matching of templates from CharBuffer1 and CharBuffer2, providing matching results.

Input Parameter: none
Return Parameter: Confirmation code (1 byte) + matching score.

Instruction code: 03H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Checksum
EF01H	xxxx	01H	0003H	03H	0007H

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Matching score	Checksum
EF01H	xxxx	07H	0005H	xxH	xxH	sum

Note 1: Confirmation code=00H: templates of the two buffers are matching;
Confirmation code=01H: error when receiving package;
Confirmation code=08H: templates of the two buffers aren't matching;
2: The instruction doesn't affect the contents of the buffers.

6.2.20 To search finger library: Search

Description: to search the whole or part finger library for the template that matches the one in CharBuffer1 or CharBuffer2. When found, PageID will be returned.

Input Parameter: BufferID+StartPage (searching start address)+ PageNum（searching numbers）

Return Parameter: Confirmation code (1 byte)+PageID (matching templates location）

Instruction code: 04H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1 byte	2 bytes	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instructio n code	buffer number	Start page number	Number	Checksum
EF01H	xxxx	01H	0008H	04H	BufferID	StartPage	PageNum	sum

Note: BufferID of CharBuffer1 and CharBuffer2 are 1h and 2h respectively.

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes	2 bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Page	Score	Checksum
EF01H	xxxx	07H	0007H	xxH	PageID	MatchScore	sum

Note 1: Confirmation code=00H: found the matching finer;
Confirmation code=01H: error when receiving package;

Confirmation code=09H: No matching in the library (both the PageID and matching score are 0);

2: The instruction doesn't affect the contents of the buffers.

6.3 Other instructions

6.3.1 To write note pad: WriteNotepad

Description: for upper computer to write data to the specified Flash page;

Input Parameter: NotePageNum, user content (or data content)

Return Parameter: Confirmation code (1byte)

Instruction code: 18H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1byte	32 bytes	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Page number	Data content	Checksum
EF01H	xxxx	01H	0024H	18H	0~15	content	sum

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	Checksum
EF01H	xxxx	07H	0003H	xxH	sum

Note: Confirmation code=00H: write success;

Confirmation code=01H: error when receiving package;

6.3.2 To read note pad: ReadNotepad

Description: to read the specified page's data content;.

Input Parameter: none

Return Parameter: Confirmation code (1 byte) + data content

Instruction code: 19H

Command (or instruction) package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	1byte	2 bytes
Header	Module address	Package identifier	Package length	Instruction code	Page number	Checksum
EF01H	xxxx	01H	0004H	19H	0~15	xxH

Acknowledge package format:

2 bytes	4bytes	1 byte	2 bytes	1 byte	32bytes	2 bytes
Header	Module address	Package identifier	Package length	Confirmation code	User content	Checksum
EF01H	xxxx	07H	0023H	xxH	content	sum

Note: Confirmation code=00H: read success;

Confirmation code=01H: error when receiving package;

Dimentions

Dimentions of integral type Module（unit：mm）