IMPLEMENTATION OF NFC ON WEARABLE OBJECTS

A project report submitted in partial fulfilment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

In

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted

By

MADIRAJU ANIRUDH SAI	17011M2001
SIDDAVATAM.SALMANUDDIN AHAMAD	17011M2005
D.L.V.R SUDHEER	17011M2006
KANDALA HEMANTH	16011M2013

Under the esteemed

guidanceof

DR. B. N. BHANDARI

(ECE Department)

Jawaharlal Nehru Technological University Hyderabad

JNTUH College of Engineering, Hyderabad Department ofElectronics and Communication Engineering,

Kukatpally, Hyderabad - 500085

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY

HYDERABAD

JNTUH College of Engineering, Hyderabad

(Autonomous)(Kukatpally, Hyderabad - 500085)

Department of Electronics and Communication Engineering

DECLARATION BY THE SUPERVISOR

This is to certify that the project entitled, "IMPLEMENTATION OF NFC ON WEARABLE OBJECTS" submitted by

MADIRAJU ANIRUDH SAI	17011M2001
SIDDAVATAM.SALMANUDDIN AHAMAD	17011M2005
D.L.V.R SUDHEER	17011M2006
KANDALA HEMANTH	16011M2013

In partial fulfilment of the requirements for the award of major project in Electronics and Communication Engineering at Jawaharlal Nehru Technological University Hyderabad during the academic year 2020-2021, is an authentic work carried out by them under my supervision and guidance. To the best of my knowledge, the matter embodied in the project has not been submitted to any other University / Institute for the award of any Degree or Diploma.

PROJECT SUPERVISOR

DR. B. N. BHANDARI

Department of ECE, JNTUHCEH

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY

HYDERABAD

JNTUH College of Engineering, Hyderabad

(Autonomous)(Kukatpally, Hyderabad - 500085)

Department of Electronics and Communication Engineering

CERTIFICATE BY THE HEAD OF THE DEPARTMENT

This is to certify that the project entitled, "IMPLEMENTATION OF NFC ON WEARABLE OBJECTS" submitted by

MADIRAJU ANIRUDH SAI	17011M2001
SIDDAVATAM.SALMANUDDIN AHAMAD	17011M2005
D.L.V.R SUDHEER	17011M2006
KANDALA HEMANTH	16011M2013

In partial fulfilment of the requirements for the award of major project in Electronics and Communication Engineering at Jawaharlal Nehru Technological University Hyderabad during the academic year 2020-2021, is an authentic work carried out by them under my supervision and guidance. To the best of my knowledge, the matter embodied in the project has not been submitted to any other University / Institute for the award of any Degree or Diploma.

HEAD OF DEPARTMENT,
DR. K. ANITHA SHEELA,
PROFESSOR AND HEAD,
Department of ECE,
JNTUHCEH.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

JNTUH College of Engineering, Hyderabad

(Autonomous)(Kukatpally, Hyderabad - 500085)

Department of Electronics and Communication Engineering

DECLARATION BY THE CANDIDATES

We hereby declare that the major project entitled "IMPLEMENTATION OF NFC ON WEARABLE OBJECTS" is a bonafide record work done and submitted under the esteemed guidance of **DR.B.N.BHANDARI**, Professor and Head, Principal, JNTU, Hyderabad, in partial fulfilment of the requirements for the major project in Electronics and Communication Engineering at the Jawaharlal Nehru Technological University Hyderabad during the academic year 2020-2021.This record is a bonafide work carried out by us and the results kept in the major project have not been reproduced or copied. The results in the major project have not been submitted in any other university or institution for the award of degree or diploma.

MADIRAJU ANIRUDH SAI	17011M2001
SIDDAVATAM.SALMANUDDIN AHAMAD	17011M2005
D.L.V.R SUDHEER	17011M2006
KANDALA HEMANTH	16011M2013

ACKNOWLEDGEMENTS

We would like to express our sincere deep appreciation and indebtedness particularly to our guide DR.B.N. BHANDARI, Professor & Principal, Electronics & Communication Engineering, for his endless support, kind and understanding spirit.

The completion of this undertaking could not have been possible without the participation and assistance of so many people whose names may not all be enumerated. Their contributions are sincerely appreciated and gratefully acknowledged. I also thank my relatives, friends and others who shared their support, morally, financially or physically. Above all, I thank the Great Almighty, the author of knowledge and wisdom, for his countless love.

MADIRAJU ANIRUDH SAI	17011M2001
SIDDAVATM.SALMANUDDIN AHAMAD	17011M2005
D.L.V.R SUDHEER	17011M2006
KANDALA HEMANTH	16011M2013

TABLE OF CONTENTS

TOPIC NAME **PAGE NO:**

CHAPTER 5:

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Project Aim:

Sharing, storing private information like Aadhar card number in a ring or any customized thing using Near Field Communication (NFC) and Arduino or Raspberry Pi.

Plan:

We need NFC on the ring or any customized thing in which we want to store the information.
Next we code NFC tag using Arduino or Raspberry Pi to make it store or share the information when required using NFC reader.
Using the ring we can access to our personal information within our reach at all times.
Sharing the information will be possible through simple tap of that object on the NFC reader, NFC tag does not need any power source making it even more accessible.

Security:

Information of owner of the NFC tag will be hard coded into the object of choice to avoid information theft and no problem would arise even if the object is lost. This would be a simple solution for many things.

CHAPTER 1: INTRODUCTION

Sub technology of RFID is NFC, in RFID far field communication takes place through radio frequency reflectors and antennas.
But in NFC, communication takes place through mutual inductance.
NFC is near field communication which is a wireless short-ranged communication system where we use the concept of mutual inductance and load modulation for the short-ranged communication to work.
This requires a transmitting device and a receiving device (passive).
The information is stored inside an NFC tag from which the information is sent to devices, ***in which load modulation can be detected***.
NFC uses radio frequency for the communication to take place.
For the working of NFC, there is a standard placed which is ISO 14443-a, which basically tells us the various devices required and the usable frequency band. This can be seen in the further parts of the paper.
NFC is used between two electronic devices over a distance of 4 cm (1½ in) or less. NFC offers a low-speed connection with simple setup that can be used to bootstrap more-capable wireless connections.
Near-field communication (NFC) describes a technology which can be used for contactless exchange of data over short distances. Two NFC-capable devices are connected via a point-to-point contact over a distance of 0 to 2 cm. This connection can be used to exchange data (such as process data and maintenance and service information) between the devices. This interface can be used for parameterization of the component as well.
NFC tags are passive data stores which can be read, and under some circumstances written to, by an NFC device. NFC tags can be custom-encoded by their manufacturers or use the industry specifications.
NFC always involves an initiator and a target; the initiator actively generates an RF field that can power a passive target. This enables NFC targets to take very simple form factors such as unpowered tags, stickers, key fobs, or cards. NFC peer-to-peer communication is possible, provided both devices are powered.
NFC devices can be used in contactless payment systems, similar to those used in credit cards and electronic ticket smart cards and allow mobile payment to replace/supplement these systems. NFC can be used for social networking, for sharing contacts, text messages and forums, links to photos, videos or files and entering multiplayer mobile games.
NFC-enabled devices can act as electronic identity documents found in Passports and ID cards, and keycards for the use in Fare cards, Transit passes, Login Cards, Car key and Access badges. NFC's short range and encryption support make it more suitable than less private RFID systems.

NFC-equipped smartphones can be paired with NFC Tags or stickers that can be programmed by NFC apps. These programs can allow a change of phone settings, texting, app launching, or command execution. Such apps do not rely on a company or manufacturer, but can be utilized immediately with an NFC-equipped smartphone and an NFC tag.

Near-Field Communication (NFC) is a set of communication protocols for communication between two electronic devices over a distance of 4 cm (1 ½ in) or less. NFC offers a low-speed connection with simple setup that can be used to bootstrap more-capable wireless connections.

NFC devices can act as electronic identity documents and keycards. They are used in contactless payment systems and allow mobile payment replacing or supplementing systems such as credit cards and electronic ticket smart cards. This is sometimes called *NFC/CTLS* or *CTLS NFC*, with *contactless* abbreviated *CTLS*. NFC can be used for sharing small files such as contacts, and bootstrapping fast connections to share larger media such as photos, videos, and other files.

Similar ideas in advertising and industrial applications were not generally successful commercially, outpaced by technologies such as QR codes, barcodes and UHF RFID tags. NFC protocols established a generally supported standard. When one of the connected devices has Internet connectivity, the other can exchange data with online services.

Near-field communication (NFC) describes a technology which can be used for contactless exchange of data over short distances. Two NFC-capable devices are connected via a point-to-point contact over a distance of 0 to 2 cm. This connection can be used to exchange data (such as process data and maintenance and service information) between the devices. This interface can be used for parameterization of the component as well.

NFC-enabled portable devices can be provided with application software, for example, to read electronic tags or make payments when connected to an NFC-compliant apparatus. Earlier close-range communication used technology that was proprietary to the manufacturer for applications such as stock tickets, access control and payment readers.

Like other "proximity card" technologies, NFC is based on inductive coupling between two so-called antennas present on NFC-enabled devices—for example a smartphone and a printer—communicating in one or both directions, using a frequency of 13.56 MHz in the globally available unlicensed radio frequency ISM band using the ISO/IEC 18000-3 air interface standard at data rates ranging from 106 to 424 kbit/s.

Every active NFC device can work in one or more of three modes:

NFC card emulation
Enables NFC-enabled devices such as smartphones to act like smart cards, allowing users to perform transactions such as payment or ticketing. See Host card emulation

NFC reader/writer
Enables NFC-enabled devices to read information stored on inexpensive NFC tags embedded in labels or smart posters.

NFC peer-to-peer
Enables two NFC-enabled devices to communicate with each other to exchange information in an ad hoc fashion.

NFC tags are passive data stores which can be read, and under some circumstances written to, by an NFC device. They typically contain data (as of 2015 between 96 and 8,192 bytes) and are read-only in normal use, but may be rewritable. Applications include secure personal data storage (e.g. debit or credit card information, loyalty program data, personal identification numbers (PINs), contacts). NFC tags can be custom-encoded by their manufacturers or use the industry specifications.

The standards were provided by the NFC Forum. The forum was responsible for promoting the technology and setting standards and certifies device compliance. Secure communications are available by applying encryption algorithms as is done for credit cards and if they fit the criteria for being considered a personal area network.

NFC standards cover communications protocols and data exchange formats and are based on existing radio-frequency identification (RFID) standards including ISO/IEC 14443 and FeliCa. The standards include ISO/IEC 18092 and those defined by the NFC Forum. In addition to the NFC Forum, the GSMA group defined a platform for the deployment of GSMA NFC Standards within mobile handsets. GSMA's efforts include Trusted Services Manager, Single Wire Protocol, testing/certification and secure element.

A patent licensing program for NFC is under deployment by France Brevets, a patent fund created in 2011. This program was under development by Via Licensing Corporation, an independent subsidiary of Dolby Laboratories, and was terminated in May 2012. A platform-independent free and open source NFC library, libnfc, is available under the GNU Lesser General Public License.

Present and anticipated applications include contactless transactions, data exchange and simplified setup of more complex communications such as Wi-Fi.

NFC is a set of short-range wireless technologies, typically requiring a separation of 10 cm or less. NFC operates at 13.56 MHz on ISO/IEC 18000-3 air interface and at rates ranging from 106 kbit/s to 424 kbit/s. NFC always involves an initiator and a target; the initiator actively generates an RF field that can power a passive target. This enables NFC

targets to take very simple form factors such as unpowered tags, stickers, key fobs, or cards. NFC peer-to-peer communication is possible, provided both devices are powered.

NFC tags contain data and are typically read-only, but may be writable. They can be custom-encoded by their manufacturers or use NFC Forum specifications. The tags can securely store personal data such as debit and credit card information, loyalty program data, PINs and networking contacts, among other information. The NFC Forum defines four types of tags that provide different communication speeds and capabilities in terms of configurability, memory, security, data retention and write endurance. Tags currently offer between 96 and 8,192 bytes of memory.

As with proximity card technology, NFC uses inductive coupling between two nearby loop antennas effectively forming an air-core transformer. Because the distances involved are tiny compared to the wavelength of electromagnetic radiation (radio waves) of that frequency (about 22 meters), the interaction is described as near field. Only an alternating magnetic field is involved so that almost no power is actually radiated in the form of radio waves (which are electromagnetic waves, also involving an oscillating electric field); that essentially prevents interference between such devices and any radio communications at the same frequency or with other NFC devices much beyond its intended range. They operate within the globally available and unlicensed radio frequency ISM band of 13.56 MHz. Most of the RF energy is concentrated in the ±7 kHz bandwidth allocated for that band, but the emission's spectral width can be as wide as 1.8 MHz in order to support high data rates.

Working distance with compact standard antennas and realistic power levels could be up to about 20 cm (but practically speaking, working distances never exceed 10 cm). Note that because the pickup antenna may be quenched by nearby metallic surfaces, the tags may require a minimum separation from such surfaces.

CHAPTER 2: BUILDING BLOCKS

2.1 ARDUINO

Arduino project was started at the “Interaction Design Institute Ivrea” (IDII) in Ivrea, Italy. The project goal was to create simple, low-cost tools for creating digital projects by non-engineers. The initial Arduino core team consisted of Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, and David Mellis.

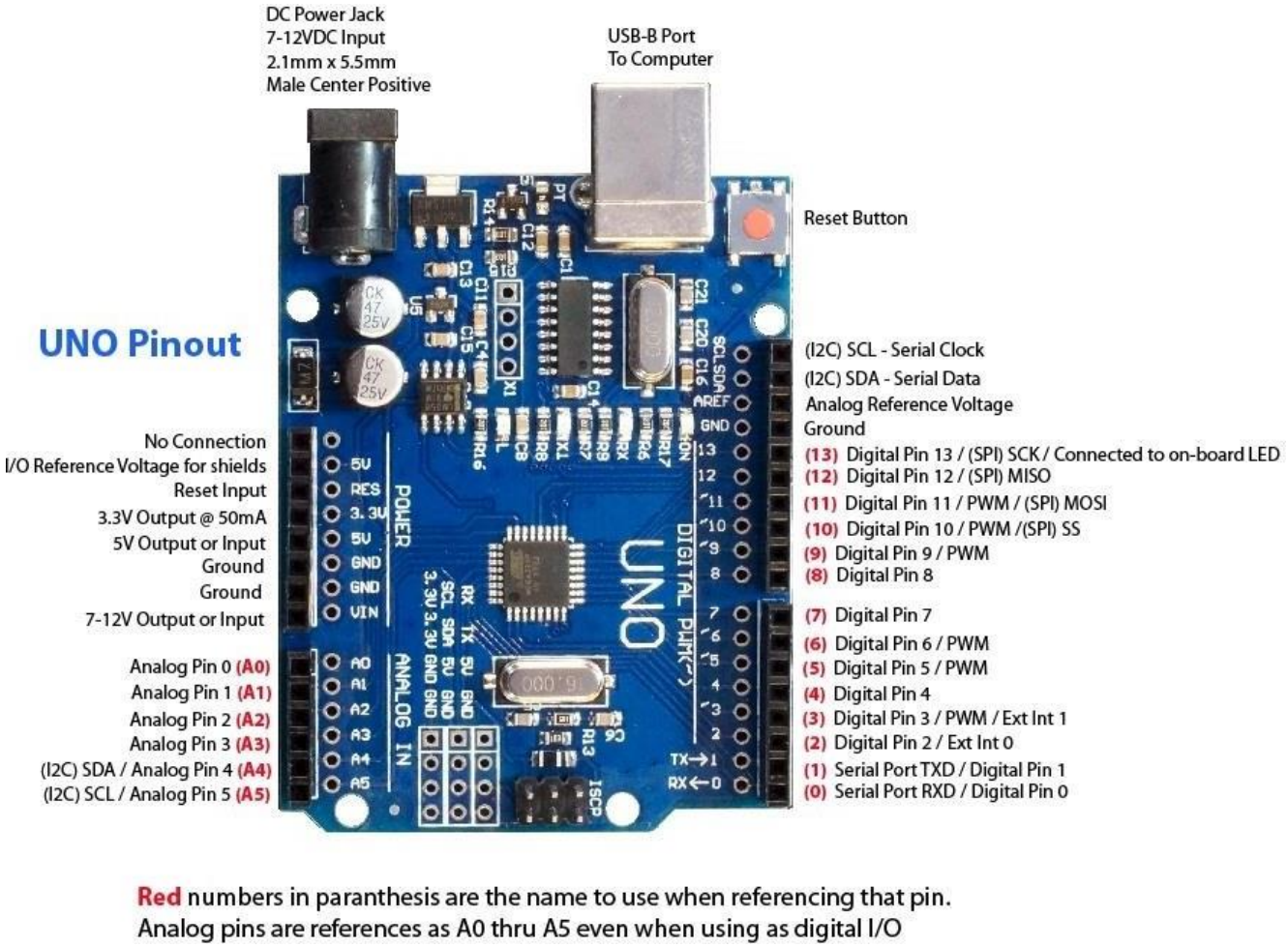
Arduino Uno is a microcontroller board based on the microchip ATmega328P. The board has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started. You can tinker with your UNO without worrying too much about doing something wrong, in the worst-case scenario you can replace the chip for a few dollars and start over again.

"UNO" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB-based Arduino boards;it and version 1.0 of the Arduino IDE were the reference versions of Arduino, which have now evolved to newer releases. The Arduino/Genuino Uno can be programmed with the (Arduino Software (IDE)). The ATmega328 on the board comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware programmer.It communicates using the original STK500 protocol You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using Arduino ISP.

The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM. Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as a recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The Uno has 6 analog inputs, labelled A0 through A5, each of which provides 10 bits of resolution (i.e. 1024 different values). By default, they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.

Table 2.1- ARDUINO UNO TECHNICAL SPECIFICATIONS

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g



(Fig 2.1 ARDUINO UNO)

2.2 LIQUID CRYSTAL DISPLAY (LCD)- 20 x 4

LCD modules are very commonly used in most embedded projects, the reason being their cheap price, availability and programmer friendly. Most of us would have come across these displays in our day to day life, either at PCO's or calculators. The appearance and the pin outs have already been visualized below now let us get a bit technical.

LCD's work based on the physics involved in when voltage is applied to a liquid crystal, this liquid crystal allows or blocks based on the orientation it is currently in. Hence, depending on orientation backlight can be passed or blocked.

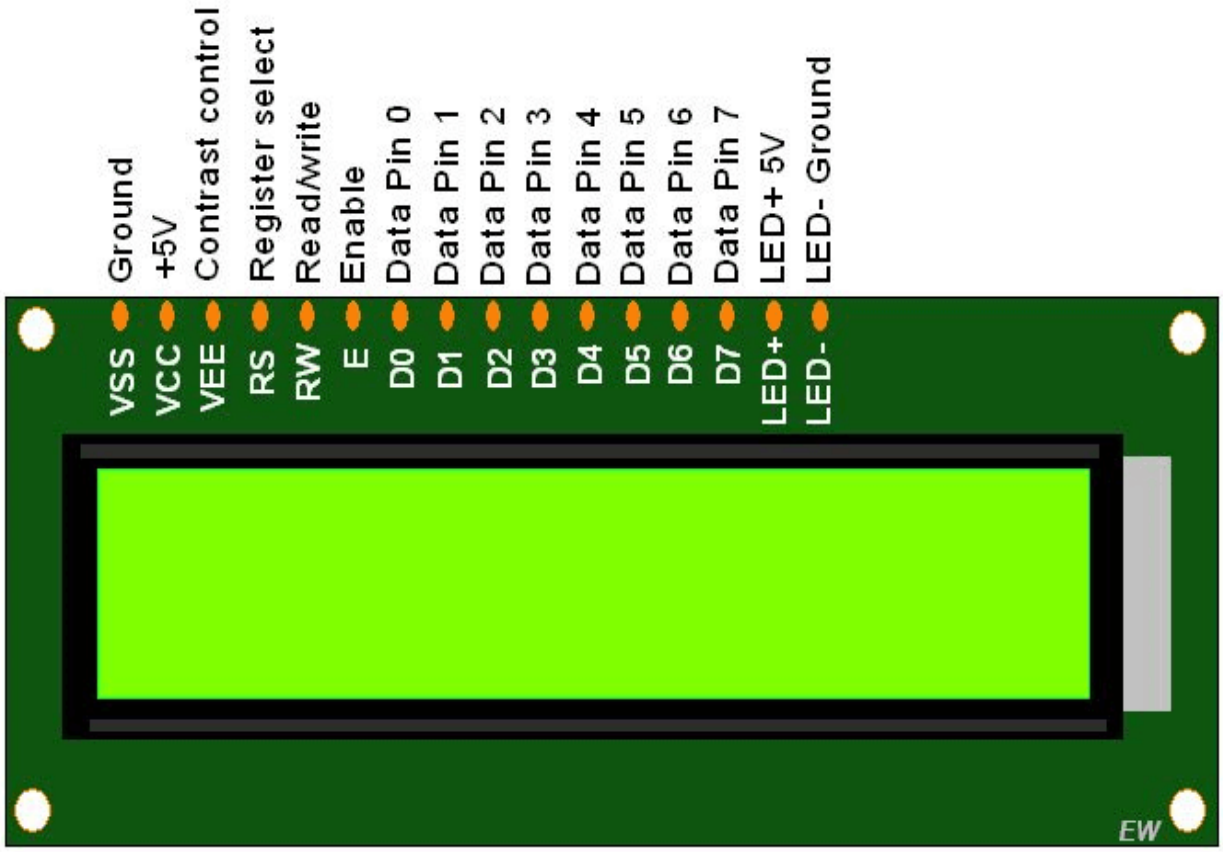


Fig 2.2 20 x 4 LCD Module: Pin-out

20×4 LCD is named so because, it has 20 Columns and 4 Rows. There are a lot of combinations available like 8×1, 8×2, 10×2, 16×1, etc. but the one we need is the 20×4 LCD. So, it will have (20x4=80) 80 characters in total and each character will be made of 5×8 Pixel Dots.

Table 2.2 – LCD Pins Configurations

Pin No:	Pin Name:	Description
1	Vss (Ground)	Ground pin connected to system ground

2	Vdd (+5V olt)	Powers the LCD with +5V (4.7V – 5.3V)
3	VE (Contrast V)	Decides the contrast level of display. Grounded to get maximum contrast.
4	Register Select	Connected to Microcontroller to shift between command/data register
5	Read/Write	Used to read or write data. Normally grounded to write data to LCD
6	Enable	Connected to Microcontroller Pin and toggled between 1 and 0 for data acknowledgement
7	Data Pin 0	Data pins 0 to 7 forms a 8-bit data line. They can be connected to Microcontroller to send 8-bit data. These LCD's can also operate on 4-bit mode in such case Data pin 4,5,6 and 7 will be left free.
8	Data Pin 1	
9	Data Pin 2	
10	Data Pin 3	
11	Data Pin 4	
12	Data Pin 5	
13	Data Pin 6	
14	Data Pin 7	
15	NC	Not Connected
16	NC	Not Connected

Now, we know that each character has (5×8=40) 40 Pixels and for 80 Characters we will have (80×40) 3200 Pixels. Further, the LCD should also be instructed about the Position of the Pixels. Hence it will be a hectic task to handle everything with the help of MCU, hence an Interface IC like HD44780is used, which is mounted on the backside of the LCD Module itself. The function of this IC is to get the Commands and Data from the MCU and process them to display meaningful information onto our LCD Screen. You can learn how to interface an LCD using the above-mentioned links. If you are an advanced programmer and would like to create your library for interfacing your Microcontroller with this LCD module then you have to understand the HD44780 IC is working and commands which can be found in its datasheet.

Features of 20*4 LCD module

- 20 Characters x 4 Lines
- 5 x 8 Dots with Cursor
- Built in Controller (HD44780 or equivalent)

- +5V Power Supply
- 1/16 Duty Circle
- RoHS Compliant

2.3 RFID RC 522 NFC READER

The MFRC522 is a highly integrated reader/writer IC for contactless communication at 13.56 MHz. The MFRC522 reader supports ISO/IEC 14443 A/MIFARE mode.
The MFRC522's internal transmitter is able to drive a reader/writer antenna designed to communicate with ISO/IEC 14443 A/MIFARE cards and transponders without additional active circuitry. The receiver module provides a robust and efficient implementation for demodulating and decoding signals from ISO/IEC 14443 A/MIFARE compatible cards and
transponders. The digital module manages the complete ISO/IEC 14443 A framing and error detection (parity and CRC) functionality.
The MFRC522 supports MF1xxS20, MF1xxS70 and MF1xxS50 products. The MFRC522
supports contactless communication and uses MIFARE higher transfer speeds up to 848 kBd in both directions.
The following host interfaces are provided:
• Serial Peripheral Interface (SPI)
• Serial UART (similar to RS232 with voltage levels dependent on pin voltage supply)
• I^2C-bus interface

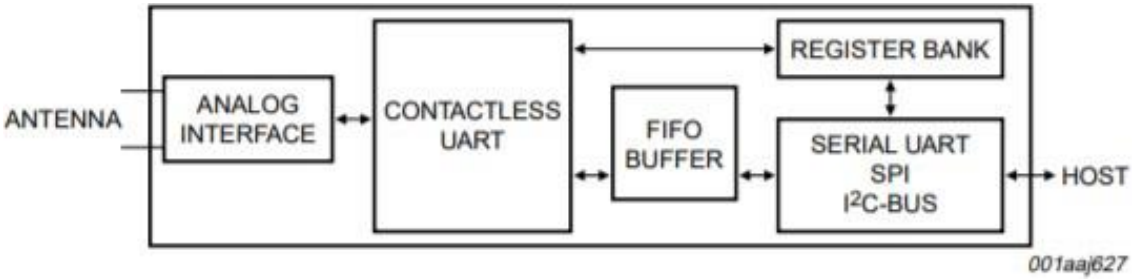


Fig 2.3.1 BLOCK DIAGRAM OF RFID RC 522

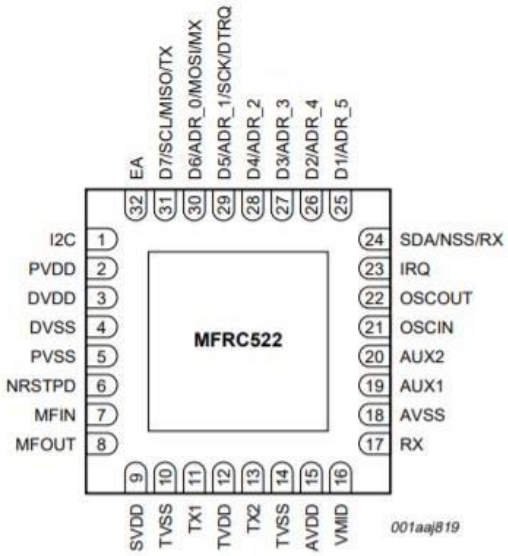


Fig 2.3.2 PIN DIAGRAM OF NFC READER

Table No. 2.3 RFID RC 522 Pin Configuration

Pin	**Symbol**	**Type [1]**	**Description**
1	I2C	I	I^2C-bus enable input
2	PVDD	P	Pin power supply
3	DVDD	P	Digital power supply
4	DVSS	G	Digital ground
5	PVSS	G	Pin power supply ground
6	NRSTPD	I	Reset and power-down input: Power-down: enabled when LOW; internal current sinks are switched off, the oscillator is inhibited and the input pins are disconnected from the outside world Reset: enabled by a positive edge
7	MFIN	I	MIFARE signal input
8	MFOUT	O	MIFARE signal output
9	SVDD	P	MFIN and MFOUT pin power supply
10	TVSS	G	Transmitter output stage 1 ground
11	TX1	O	Transmitter 1 modulated 13.56 MHz energy carrier output
12	TVDD	P	Transmitter power supply: supplies the output stage of transmitters 1 and 2
13	TX2	O	Transmitter 2 modulated 13.56 MHz energy carrier output

14	TVSS	G	Transmitter output stage 2 ground
15	AVDD	P	Analog power supply
16	VMID	P	Internal reference voltage
17	RX	I	RF signal input
18	AVSS	G	Analog ground
19	AUX1	O	Auxiliary outputs for test purposes
20	AUX2	O	Auxiliary outputs for test purposes
21	OSCIN	I	Crystal oscillator inverting amplifier input; also the input for an externally generated cock (f_{clk}=27.12 MHz)
22	OSCOUT	O	Crystal oscillator inverting amplifier output
23	IRQ	O	Interrupt request output: indicates an interrupt event
24	SDA NSS RX	I/O I I	I^2C-bus serial data line input/output SPI signal input UART address input
25	D1 ADR_5	I/O I/O	Test port I^2C-bus address 5 input
26	D2 ADR_4	I/O I	Test port I^2C-bus address 4 input
27	D3 ADR_3	I/O I	Test port I^2C-bus address 3 input
28	D4 ADR_2	I/O I	Test port I^2C-bus address 2 input
29	D5 ADR_1 SCK DTRQ	I/O I I O	Test port I^2C-bus address 1 input SPI serial clock input UART request to send output to microcontroller
30	D6 ADR_0 MOSI MX	I/O I I/O O	Test port I^2C-bus address 0 input SPI master out, slave in UART output to microcontroller
31	D7 SCL MISO TX	I/O I/O I/O O	Test port I^2C-bus clock input/output SPI master in, slave out UART data output to microcontroller
32	EA	I	External address input for coding I^2C-bus address

Table 2.3 Pin Configuration of NFC Reader

A serial peripheral interface (SPI compatible) is supported to enable high-speed communication to the host. The interface can handle data speeds up to 10 Mbit/s. When communicating with a host, the MFRC522 acts as a slave, receiving data from the external host for register settings, sending and receiving data relevant for RF interface communication. An interface compatible with SPI enables high-speed serial communication between the MFRC522 and a microcontroller. The implemented interface is in accordance with the SPI standard.

An I^2C-bus (Inter-IC) interface is supported to enable a low-cost, low pin count serial bus interface to the host. The interface can only act in Slave mode. Therefore the MFRC522 does not implement clock generation or access arbitration.

Data on the SDA line must be stable during the HIGH clock period. The HIGH or LOW state of the data line must only change when the clock signal on SCL is LOW.

The MFRC522 is divided into a digital circuit block and an analog circuit block. The digital block contains state machines, encoder and decoder logic and so on. The analog block contains the modulator and antenna drivers, receiver and amplifiers. The interface between these two blocks can be configured so that the interfacing signals can be routed to pins MFIN and MFOUT.

The most important use of pins MFIN and MFOUT is found in the active antenna concept. An external active antenna circuit can be connected to the MFRC522's digital block.

An 8 × 64 bit FIFO buffer is used in the MFRC522. It buffers the input and output data stream between the host and the MFRC522's internal state machine. This makes it possible to manage data streams up to 64 bytes long without the need to take timing constraints into account.

The MFRC522 indicates certain events by setting the Status1Reg register's IRq bit and, if activated, by pin IRQ. The signal on pin IRQ can be used to interrupt the host using its interrupt handling capabilities. This allows the implementation of efficient host software. The timer unit can be used to measure the time interval between two events or to indicate that a specific event occurred after a specific time. The timer does not influence any internal events, for example, a time-out during data reception does not automatically influence the reception process. Furthermore, several timer-related bits can be used to generate an interrupt.

2.4 NFC Sticker NTAG 213

NTAG213, NTAG215 and NTAG216 have been developed by NXP Semiconductors as standard NFC tag ICs to be used in mass market applications such as retail, gaming and

consumer electronics, in combination with NFC devices or NFC compliant Proximity Coupling Devices. Target applications include Out-of-Home and print media smart advertisement, SoLoMo applications, product authentication, NFC shelf labels, mobile companion tags. Thanks to the high input capacitance, NTAG21x tag ICs are particularly tailored for applications requiring small footprints, without compromise on performance. Small NFC tags can be more easily embedded into e.g. product labels or electronic devices. The mechanical and electrical specifications of NTAG21x are tailored to meet the requirements of inlay and tag manufacturers. Communication to NTAG21x can be established only when the IC is connected to an antenna. When NTAG21x is positioned in the RF field, the high speed RF communication interface allows the transmission of the data with a baud rate of 106 kbit/s.

Security:

• Manufacturer programmed 7-byte UID for each device
• Pre-programmed Capability container with one time programmable bits
• Field programmable read-only locking function
• ECC based originality signature
• 32-bit password protection to prevent unauthorized memory operations

Anticollision:

An intelligent anticollision function allows to operate more than one tag in the field simultaneously. The anticollision algorithm selects each tag individually and ensures that the execution of a transaction with a selected tag is performed correctly without interference from another tag in the field.

Features and benefits:

- Contactless transmission of data and supply energy
- Operating frequency of 13.56 MHz
- Data transfer of 106 kbit/s
- Data integrity of 16-bit CRC, parity, bit coding, bit counting
- Operating distance up to 100 mm (depending on various parameters as e.g. field
- strength and antenna geometry)
- 7-byte serial number (cascade level 2 according to ISO/IEC 14443-3)
- UID ASCII mirror for automatic serialization of NDEF messages
- Automatic NFC counter triggered at read command

- NFC counter ASCII mirror for automatic adding the NFC counter value to the NDEF message
- ECC based originality signature
- Fast read command
- True anticollision
- 50 pF input capacitance

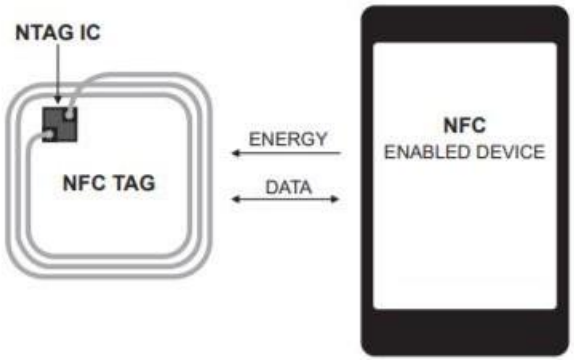


Fig 2.4.1 schematic of scenario

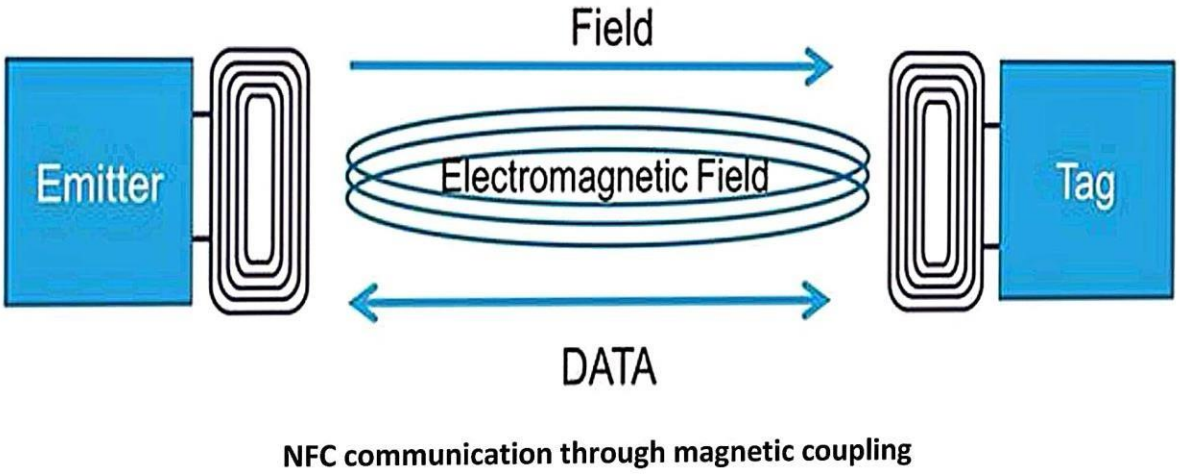


Fig 2.4.2 concept of working

2.5 ISO 14443 A

The ISO/IEC 14443 standard is a four-part international standard for contact-less smart cards operating at 13.56 MHz in close proximity (~10cm) with a reader antenna. This ISO standard describes the modulation and transmission protocols between card and reader to create interoperability for contact-less smart card products. There are two main communication protocols supported by the ISO/IEC 14443 standard, they are addressed as Type A and Type B. Most High Frequency (13.56 MHz) proximity tags are using these guidelines for their communication.

Near Field Communication devices implement native support for ISO14443-A tags. The NFC Forum refers to these tags as Type 1, Type 2 and Type 4 tags. FIXME integrate this The Anti-Collision example describes the initialization messages used to set up a communication channel and to retrieve the identifier and supported features from a tag. During the anti-collision phase, three or four different frames are received from a tag (ATQA, UID, SAK and optional ATS).

The *ATQA*, *SAK* and *ATS* values can be used to identify the manufacturer, tag type and application. However, it's not recommended to rely on *ATQA* due to potential collision when more than one target are in the field.

Type 1 Tag is based on ISO/IEC 14443A. Tags are read and re-write capable; users can configure the tag to become read-only. Memory availability is 96 bytes and expandable to 2 kbyte.

Type 2 Tag is based on ISO/IEC 14443A. Tags are read and re-write capable; users can configure the tag to become read-only. Memory availability is 48 bytes and expandable to 2 kbyte.

Type 3 Tag is based on the Japanese Industrial Standard (JIS) X 6319-4, also known as FeliCa. Tags are pre-configured at manufacture to be either read and re-writable, or read-only. Memory availability is variable, theoretical memory limit is 1MByte per service.

Type 4 Tag is fully compatible with the ISO/IEC 14443 standard series. Tags are pre-configured at manufacture to be either read and re-writable, or read-only. The memory availability is variable, up to 32 KBytes per service; the communication interface is either Type A or Type B compliant.

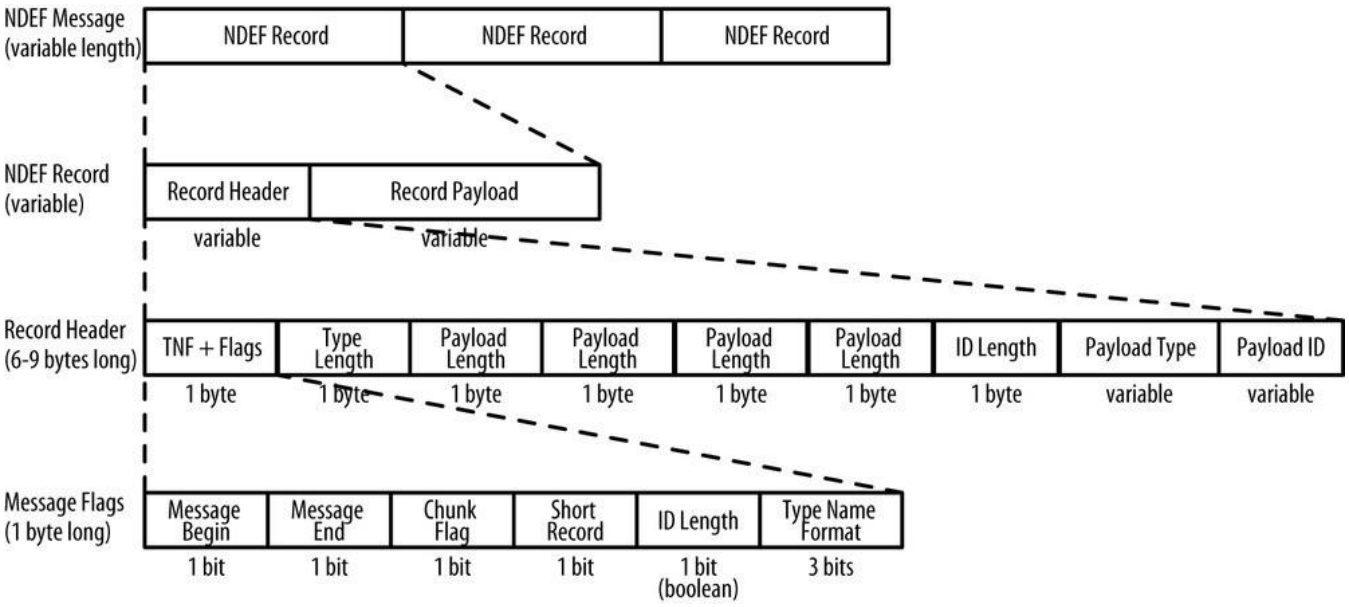


Fig 2.5 Frame format

2.6 PUSH TYPE SWITCHES

A **push switch (button)** is a momentary or non-latching switch which causes a temporary change in the state of an electrical circuit only while the switch is physically actuated. An automatic mechanism (i.e. a spring) returns the switch to its default position immediately afterwards, restoring the initial circuit condition. There are two types:[1]

A 'push to make' switch allows electricity to flow between its two contacts when held in. When the button is released, the circuit is broken. This type of switch is also known as a Normally Open (NO) Switch. (Examples: doorbell, computer case power switch, calculator buttons, individual keys on a keyboard)

A 'push to break' switch does the opposite, i.e. when the button is not pressed, electricity can flow, but when it is pressed the circuit is broken. This type of switch is also known as a Normally Closed (NC) Switch. (Examples: Fridge Light Switch, Alarm Switches in Fail-Safe circuits)

Many Push switches are designed to function as both 'push to make' and 'push to break' switches. For these switches, the wiring of the switch determines whether the switch functions as a 'push to make' or as a 'push to break' switch.

Integral Series 60 Push Button Switches are designed to incorporate all the advantages of the time tested Toggle switches. The actuation is Push to On / Push to Off and the cap

size and colour can be selected to match the front panel requirements. The switches are also available for direct P.C.B. Mounting and provide an elegant and robust way to switch-on equipments.

Fig 2.6 PUSH BUTTON

2.7 ARDUINO - IDE

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

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows and Linux. Arduino is a key tool to learn new things. Teachers and students use it to build low-cost scientific instruments, to prove the chemistry and physical principles, or to get started with programming and robotics. Arduino boards are relatively inexpensive.

Arduino IDE where IDE stands for Integrated Development Environment. It is mainly used for writing, compiling and uploading the code to the Arduino Board. The Arduino IDE supports C and C++ languages using special rules of code structuring. The Arduino IDE supplies a software library for wiring projects, which provides many common input and output procedures.

Programs written using Arduino IDE called sketches. These sketches are written in a text editor and are saved with a file extension.ino. The editor has features for cutting/pasting and for searching/replacing the text. The message area gives feedback while saving, exporting and displaying errors.

The console displays text output by Arduino software (IDE), including complete error messages and some other information. The bottom right-hand corner of windows displays the configured board and serial port. The toolbar button allows you to verify and upload programs, create, open and save sketches and open the serial monitor.

This software is provided by Arduino developers and is mostly based on embedded C language, C language, C++. It also provides graphical representation of received data on serial plotter. The data read/ processed by Arduino is displayed on serial monitor as well as serial plotter. Inputs are given to Arduino serially by taking input from serial monitor.

The baud rate, i.e, number of bits per second is by default set to 9600, but it can be changed in program in the function serial.begin.

This program is complied by using built-in complier but personal compliers can be use, the binary data can be exported or imported in this IDE.

To send data to Arduino i.e, dumping program into the micro controller present in Arduino is done through bootloader and when done, the program can be run directly on Arduino without further help from the programmer.

This software also provides a way to handle interrupts easily, because of which this IDE is widely utilized by programmers of micro controllers.

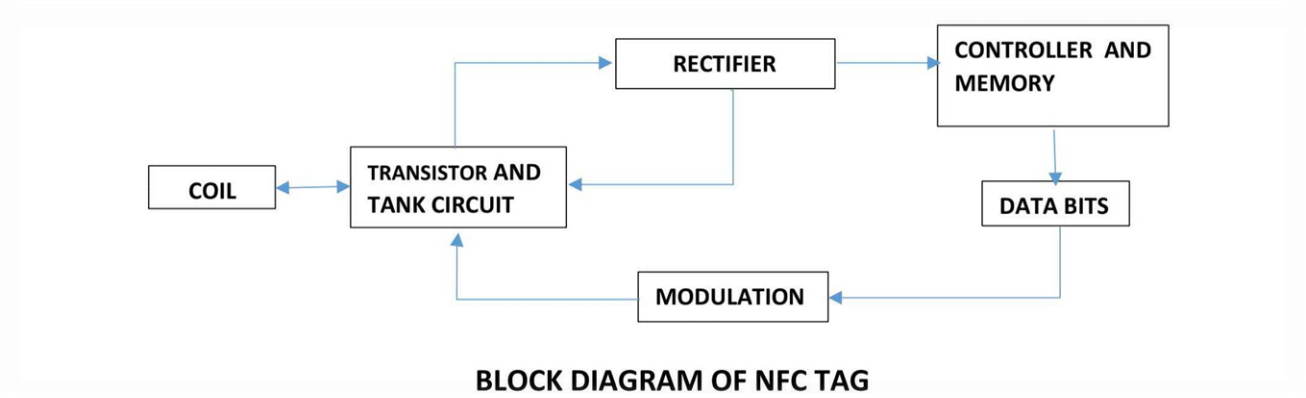
The programing of this project is done in Arduino IDE version 1.1.6 through drivers provided by Arduino website. The program is displayed at further section of this paper and can be used by others as well, it will work on any Arduino device capable with the hardware used in this project.

CHAPTER 3: BUILDING MODEL

3.1 BLOCKDIAGRAM

A **block diagram** is a diagram of a system in which the principal parts or functions are represented by blocks connected by lines that show the relationships of the blocks They are heavily used in engineering in hardware design, electronic design, software design, and process flow diagrams.

Block diagrams are typically used for higher level, less detailed descriptions that are intended to clarify overall concepts without concern for the details of implementation. Contrast this with the schematic diagrams and layout diagrams used in electrical engineering, which show the implementation details of electrical components and physical construction.



BLOCK DIAGRAM OF NFC TAG

This above figure is the Block diagram of the NFC tag that is required and we are using it here. This explains the internal working of the NFC tag. First the incoming signal is tuned using the tank circuit and is then converted to DC using the rectifier, as the controller needs DC input and now the data bits will be modulated and returned to the primary coil, which is the NFC reader.

3.2 CIRCUITDIAGRAM

The below shown circuit shows the inductive coupling between the two coils used for the information transfer. Good coupling is needed for proper data transfer using NFC. If

coupling is weak then energy transfer to secondary coil doesn't take place.

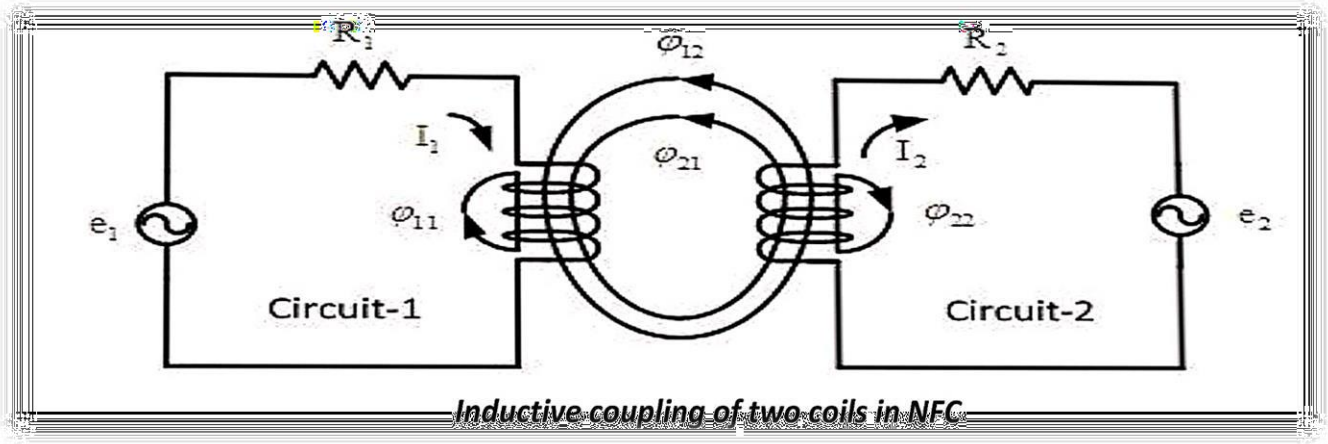
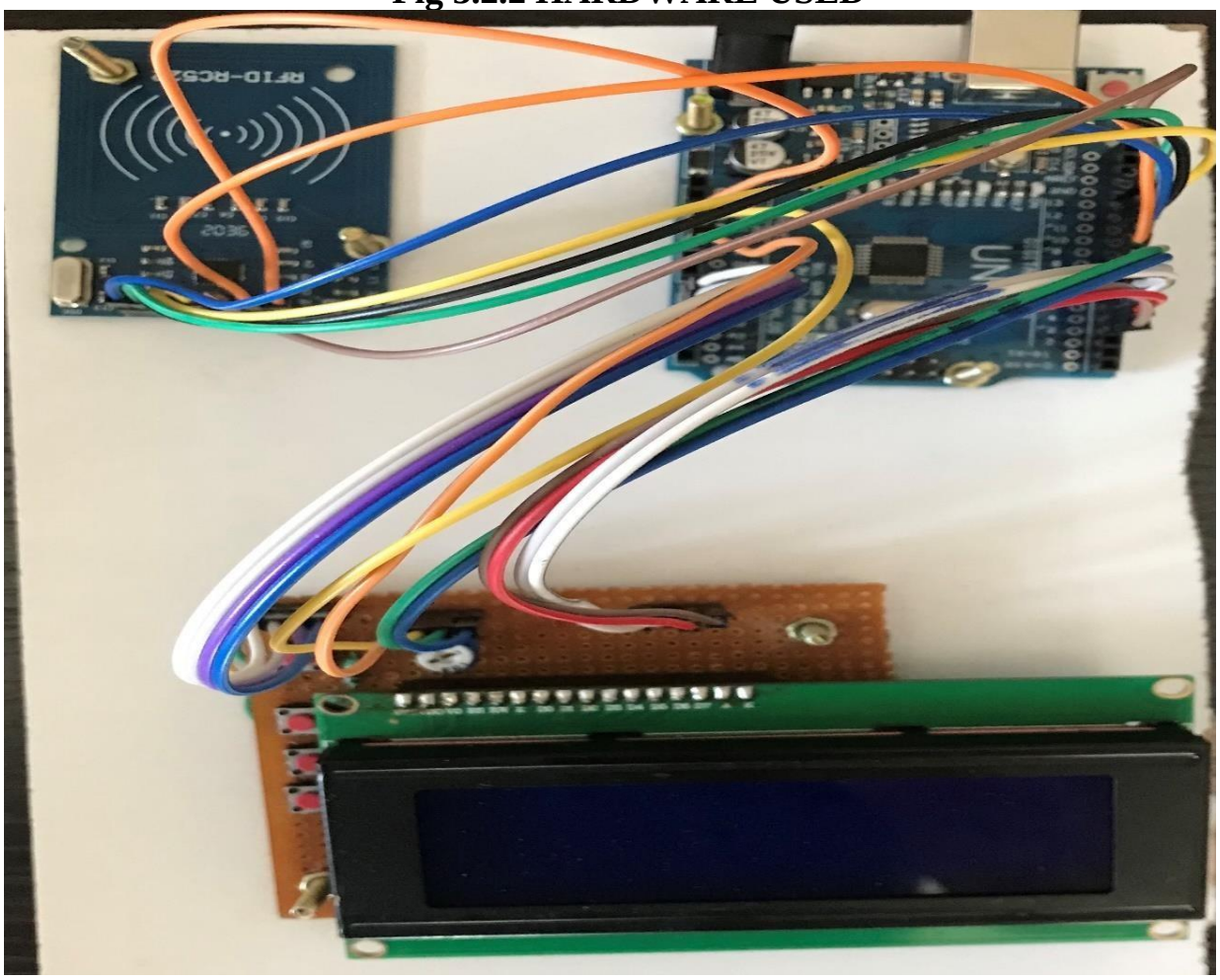


Fig 3.2.1 mutual inductance.

Let us look at the actual hardware that we used in the below picture :

Fig 3.2.2 HARDWARE USED



3.3 Simulation of NFC tag

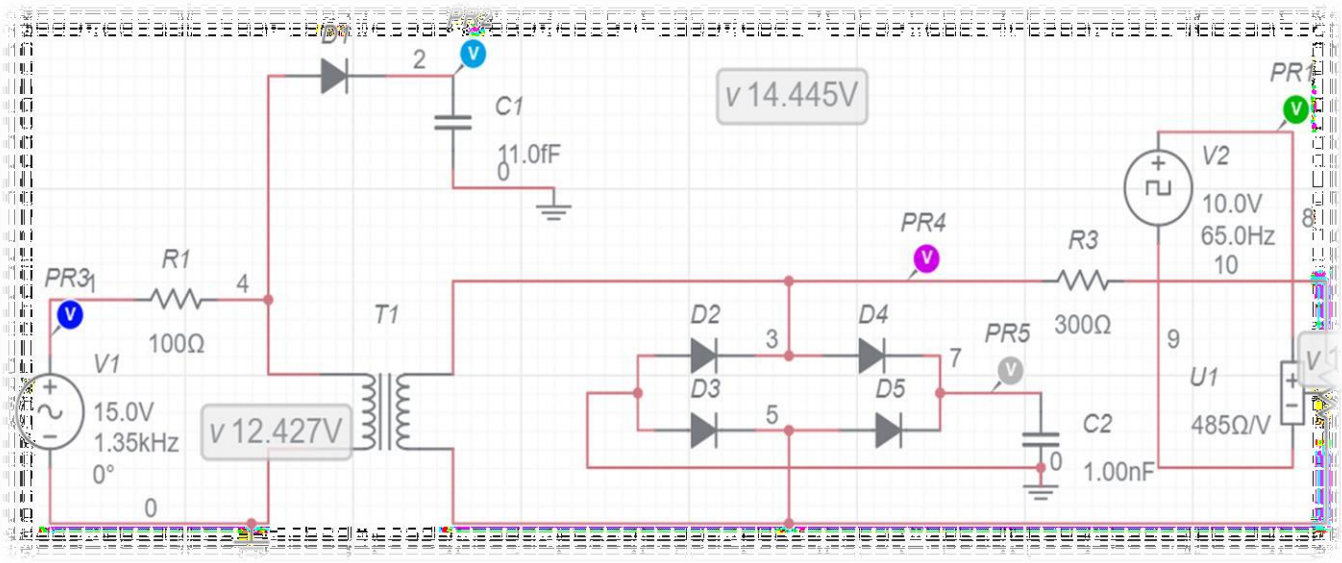


Fig 3.3.1 circuit used for simulation in MULTISM.

This is the circuit used to simulate how the NFC tag works, at the primary coil we send in Alternating Current for the coupling to take place and the current is transferred to the secondary coil through Mutual Inductance. This AC is converted to DC using bridge rectifier which is used as power source by the chip (here chip is not used, so connection is left as it is), then the memory is accessed by addressing and data present on that address is used to modulate resistance on the secondary coil, as the resistance changes the power induced changes hence resulting in change of current drawn from primary coil. Here instead of data, square wave is used to represent data. As resistance is varied according to the data , this scheme of modulation is nick named as load modulation.

Data transmission in NFC is based on load modulation and the resulting wave form will be amplitude modulated as shown in the figure below, the data is recovered through envelop detector which gives the trace of positive side of wave form.

All of the simulated wave forms are shown below, these wave forms are colored and represent the signal at the particular point in the circuit.

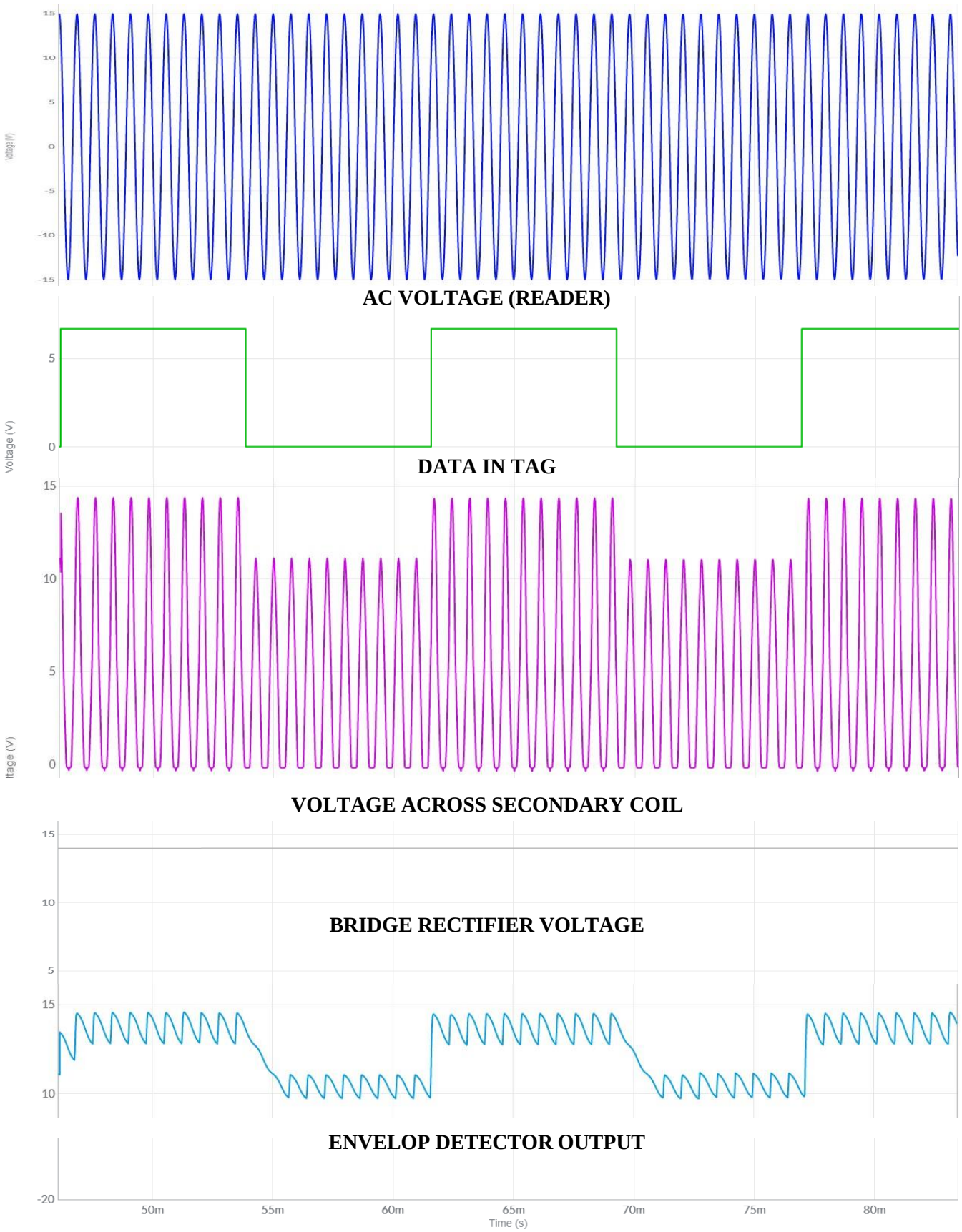
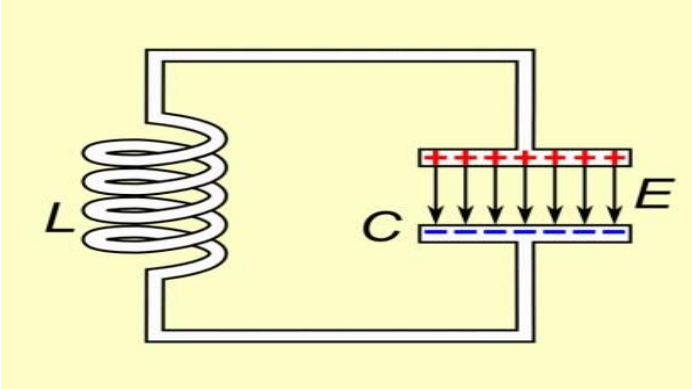


Fig 3.3.2 WAVE FORMS OBTAINED FROM MULTISM

FREQUENCY SELECTION:

These are the various signals obtained at various points of the NFC circuit. It shows how the waveforms transform at different stages. Now let us look at the tuning circuit and how it can be used.



$$F = \frac{1}{2\pi\sqrt{LC}}$$

Fig 3.3.3 tuning circuit

This is the circuit used to tune circuits to certain frequency which will be needed in various circuits. In NFC the optimum frequency needed is 13.56MHz. Here the Inductance and Capacitance will be calculated for the required frequency using the formula mentioned above. Depending on the availability the inductor, capacitor is selected.

As we need to implement NFC in wearable objects, the shape of the coil will not be the same in different objects. A circular, rectangular, hexagonal, octagonal, … shapes may be present, so the coil's shape depends on object shape. But inductor value changes with number of turns in coil as well as the shape of the coil because different shapes will have different distances when measured from object's centre to end point on a coil. Because of this calculating inductor value is necessary and after calculating inductance, capacitor can be chosen by finding 'C' in the above formula.

The inductances of different shapes is shown in the reference paper [3].

CHAPTER 4: RESULTS

We were able to store the required data, which is:

- Aadhar Card Number
- PAN Card number
- College ID Card
- Driving License

These details were stored in the NFC chip through writing code shown above and we have been able to read this data successfully through the reading code shown above . This data is displayed in the LCD that we used.

DISCUSSION

Sometimes displaying complete data stored in tag is not favourable, so to overcome this problem switches are used in this project but these switches can be replaced with unique ID's and only if these unique id's are present the data will be displayed or else it will not be displayed. Creation of unique id's will be further extension to this project so these are replaced with switches in this project.

There are buttons present in our design which allows us to select which data is needed to be displayed on the LCD. This is done in order to separate all the various data and it now can be displayed separately instead of showing all the data at once.

The above shown data is just only a part of what can be stored. This can also be used to store payment information as well and can be used to pay for goods, but this will require encryption of data so that relaying attacks can be avoided. Encryption is further extension of this project and in the final outcome will be complete storage of personal information in wearable objects.

Instead of manufacturing an object capable of NFC, we have used NFC stickers, which can be attached to personal objects, we chose this as the manufacturing of objects will be out of our scope, so the alternative has been selected.

Through this we have achieved a transmission distance of upto 10 cm in an open environment and upto 6 cm in environment where metal objects are present in the near vicinity of NFC tag. Even more distances can be achieved by increasing the magnetic field strength of NFC reader.

CHAPTER 5: CONCLUSION

NFC is a new technology which makes lives easier and has a wide range of applications. In our project we have used NFC for the purpose of showing personal identification to others when required. By using chips, which have been embedded in a small sticker, we can stick it on to any object and use it to show the data present. This makes it easier for anyone to have the data and just put it on an NFC reader and this reader reads the required data.

We can reduce the number of cards we carry in a wallet everyday and instead use NFC in order to store all the card information in a personal object. This hence gives us an easier way to show our personal identity whenever it is required.

What we have done is just a small part which can be further developed and can be implemented in various places where we may need to show our personal identity and can be used for payment interfaces as well.

New technologies take time for full utilization by the public and this project displays its potential in this real world. In India many don't know how to use a mobile phone so to provide personal info such as aadhar, pan… they use papers and these cause damage to our environment, by implementing this at places where it is required most will help reduce the paper consumption and it is actually cheaper than a paper in the long run, because storing in tag needs to be done only once and this tag can be used for all the above said purposes.

NFC in personal objects is passive, so recharging this tag is not necessary and this also follows the "plug and play" objective in the real world.

FUTURE SCOPE

Our project gives us the use of NFC to send personal identification using NFC Tag and Mutual Inductance. By using more sophisticated equipment, a more efficient way to transfer data can be done.

- By integrating it with mobile phones we can no longer need the use of wallets to show our personal identity.

- Developing NFC to transfer data at higher rates will help in using this technology for internet surfing, and the host can carry a passive device and when ever he/she needs internet, they can be near a NFC transmitter and they'll be able to connect to internet with a passive device!

- Hash codes can be used to identify a person, as hash codes are unique they can be given to a particular person and that person can use it through NFC to prove his/her identity in working environment.

- Data can be encrypted in NFC tag which will provide safety to the information stored, if at all host loses personal object.

This can be developed further to provide smart home benefits to authorized user only. As simple tap transfers data from object to reader, it can be used to customize surroundings according to the particular person and if that person moves away everything will be reset to defaults.

REFERENCES

[1] https://www.arduino.cc/datasheets

[2] https://www.arduino.cc/hardwarespecs

[3] https://www.augmented.life.com/an2866

[4] https://arduino.cc/sofware/package

[5] www.futurlec.com/LED/LCD20X4.shtml

[6] www.nxp.com/docs/en/data-sheet/MFRC522.pdf

[7] pdf1.alldatasheet.com/datasheet-pdf/view/531146/NXP/NTAG213.html

[8] https://www.github.com/NFC

[9] https://www.ISO.org/NFC

[10] www.ieeexplore.com/NFCreader

APPENDIX

Write code:

```
/* RFID-RC522 (SPI connection)
 *
 * CARD RC522 Arduino (UNO)
 * SDA ----------- 10 (Configurable, see SS_PIN constant)
 * SCK ----------- 13
 * MOSI------------11
 * MISO------------12
 * IRQ _____
 * GND------------GND
 * RST------------9 (configurable, see RST_PIN constant)
 * 3.3V -------- -- 3.3V
 *
 */
 #include <SPI.h>
 #include <MFRC522.h>
 #define SS_PIN 10
 #define RST_PIN 9
 MFRC522 mfrc522(SS_PIN, RST_PIN);                // Create MFRC522 instance
 MFRC522::StatusCode status;                      //variable to get card status
 byte buffer[70]; //data transfer buffer (16+2 bytes data+CRC)
 byte buffer1[70];
 byte buffer2[70];
 byte buffer3[70];
 byte size = sizeof(buffer);
 uint8_t pageAddr = 0x06;                              //In this example we will write/read
                                                              //16 bytes (page 6,7,8 and 9).
                                                    //Ultraligth mem = 16 pages. 4 bytes per page.
                                                          //Pages 0 to 4 are for special functions.
 uint8_t page1Addr = 0x0a;
uint8_t page2Addr = 0x0e;
uint8_t page3Addr = 0x13;
void setup() {
 Serial.begin(9600);                        // Initialize serial communications with the PC
 SPI.begin();                               // Init SPI bus
 mfrc522.PCD_Init();                        // Init MFRC522 card
```

```
 Serial.println(F("Sketch has been started!"));
 memcpy(buffer,"AD 233270195486",15);
 memcpy(buffer1,"PN DUOPM0337R ",13);
 memcpy(buffer2,"TS10820200016822",16);
 memcpy(buffer3,"CID 17011M2001",14);
}
void loop() {
                                                        // Look for new cards
 if ( ! mfrc522.PICC_IsNewCardPresent())
 return;
                                                       // Select one of the cards
 if ( ! mfrc522.PICC_ReadCardSerial())
 return;

 // Write data **********************************************

 for (int i=0; i < 4; i++) {
status = (MFRC522::StatusCode) mfrc522.MIFARE_Ultralight_Write(pageAddr+i,
&buffer[i*4], 4);                                    //data is writen in blocks of 4 bytes (4
                                                     //bytes per page)

 if (status != MFRC522::STATUS_OK) {
 Serial.print(F("MIFARE_Read() failed: "));
 Serial.println(mfrc522.GetStatusCodeName(status));
 return;
 }
 }
 for (int i=0; i < 4; i++) {

 //data is writen in blocks of 4 bytes (4 bytes per page)

 status = (MFRC522::StatusCode) mfrc522.MIFARE_Ultralight_Write(page1Addr+i,
&buffer1[i*4], 4);
 if (status != MFRC522::STATUS_OK) {
 Serial.print(F("MIFARE_Read() failed: "));
 Serial.println(mfrc522.GetStatusCodeName(status));
 return;
 }
 }
 for (int i=0; i < 4; i++) {
```

```
 //data is writen in blocks of 4 bytes (4 bytes per page)
 status = (MFRC522::StatusCode) mfrc522.MIFARE_Ultralight_Write(page2Addr+i,
&buffer2[i*4], 4);
 if (status != MFRC522::STATUS_OK) {
 Serial.print(F("MIFARE_Read() failed: "));
 Serial.println(mfrc522.GetStatusCodeName(status));
 return;
 }
 }
 for (int i=0; i < 4; i++) {
 //data is written in blocks of 4 bytes (4 bytes per page)
 status = (MFRC522::StatusCode) mfrc522.MIFARE_Ultralight_Write(page3Addr+i,
&buffer3[i*4], 4);
 if (status != MFRC522::STATUS_OK) {
 Serial.print(F("MIFARE_Read() failed: "));
 Serial.println(mfrc522.GetStatusCodeName(status));
 return;
 }
 }
 Serial.println(F("MIFARE_Ultralight_Write() OK "));
 Serial.println();
 delay(10000);
}
```

Read code:

```
#include <SPI.h>
 #include <MFRC522.h>
 #include <LiquidCrystal.h>
 #define SS_PIN 10
 #define RST_PIN 9

uint8_t pageAddr = 0x06;
uint8_t page1Addr = 0x0A;
uint8_t page2Addr = 0x0E;
uint8_t page3Addr = 0x13;
const int sw1 = 14;
const int sw2 = 15;
```

```
const int sw3 = 16;
const int sw4 = 17;
const int rs = 8, en = 7, d4 = 6, d5 = 5, d6 = 4, d7 = 3;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
int button1=0;
int button2=0;
int button3=0;
int button4=0;
int v1=0;
 MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance
 MFRC522::StatusCode status; //variable to get card status
 byte buffer[70]; //data transfer buffer (16+2 bytes data+CRC)
 byte size = sizeof(buffer);
 void setup() {
 Serial.begin(9600); // Initialize serial communications with the PC
 SPI.begin(); // Init SPI bus
 mfrc522.PCD_Init(); // Init MFRC522 card
 Serial.println(F("Sketch has been started!"));
 lcd.begin(20, 4);
 lcd.print("place object near reader ");
 }
 void loop() {
 // Look for new cards
 if ( ! mfrc522.PICC_IsNewCardPresent())
 return;
 // Select one of the cards
 if ( ! mfrc522.PICC_ReadCardSerial())
 return;
 Serial.print("UID tag :");
 String content= "";
 byte letter;
 for (byte i = 0; i < mfrc522.uid.size; i++)
 {
 Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
 Serial.print(mfrc522.uid.uidByte[i], HEX);
 content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
 content.concat(String(mfrc522.uid.uidByte[i], HEX));
 }
 Serial.println();
 Serial.print("Message : ");
```

```
content.toUpperCase();
if (content.substring(1) == "53 CD 98 8E 02 FD 00"||content.substring(1) == "53 84 A9
8C 02 2A
00"||content.substring(1) == "53 74 DC A5 02 C6 00"||content.substring(1) == "53 14 8D
87 02 81 00")
{
lcd.clear();
lcd.print("Authorized access");
for(v1=0;v1<60000;v1++)
{
button1 = digitalRead(sw1);
if(button1==1)
{

lcd.clear();
Serial.println(F("Reading data ... "));
//data in 4 block is readed at once.
status = (MFRC522::StatusCode) mfrc522.MIFARE_Read(pageAddr, buffer, &size);
if (status != MFRC522::STATUS_OK)
{
Serial.print(F("MIFARE_Read() failed: "));
Serial.println(mfrc522.GetStatusCodeName(status));
}
lcd.setCursor(0, 1);
lcd.print("AADHAR NUMBER");
lcd.setCursor(0, 2);
Serial.print(F("Readed data: "));
//Dump a byte array to Serial
for (byte i = 0; i < 12; i++)
{
Serial.write(buffer[i]);
lcd.write(buffer[i]);
}
Serial.println();
}

button2= digitalRead(sw2);
if(button2==1)
{
lcd.clear();
```

```
Serial.println(F("Reading data ... "));
//data in 4 block is readed at once.
status = (MFRC522::StatusCode) mfrc522.MIFARE_Read(page1Addr, buffer, &size);
if (status != MFRC522::STATUS_OK)
{
Serial.print(F("MIFARE_Read() failed: "));
Serial.println(mfrc522.GetStatusCodeName(status));
}
lcd.setCursor(0, 1);
lcd.print("PAN NUMBER");
lcd.setCursor(0, 2);
Serial.print(F("Readed data: "));
//Dump a byte array to Serial
for (byte i = 0; i <10; i++) {
Serial.write(buffer[i]);
lcd.write(buffer[i]);
}
Serial.println();
}
button3= digitalRead(sw3);
if(button3==1)
{
lcd.clear();
Serial.println(F("Reading data ... "));
//data in 4 block is readed at once.
status = (MFRC522::StatusCode) mfrc522.MIFARE_Read(page2Addr, buffer, &size);
if (status != MFRC522::STATUS_OK) {
Serial.print(F("MIFARE_Read() failed: "));
Serial.println(mfrc522.GetStatusCodeName(status));

}
lcd.setCursor(0, 1);
lcd.print("DRIVING LICENSE");
lcd.setCursor(0, 2);
Serial.print(F("Readed data: "));
//Dump a byte array to Serial
for (byte i = 0; i < 16; i++) {
Serial.write(buffer[i]);
lcd.write(buffer[i]);
}
```

```
Serial.println();
  }
  button4
  =
  digitalR
  ead(sw
  4);
  if(butto
  n4==1)
  {
  lcd.clear();
  Serial.println(F("
  Reading data ...
  "));
  //data in 4 block is readed at once.
  status = (MFRC522::StatusCode)
  mfrc522.MIFARE_Read(page3Addr, buffer, &size); if (status !=
  MFRC522::STATUS_OK) {
  Serial.print(F("MIFARE_Read() failed: "));
  Serial.println(mfrc522.GetStatusCodeName(status));
  }
  lcd.setCurso
  r(0, 1);
  lcd.print("C
  OLLEGE
  ID");
  lcd.setCurso
  r(0, 2);
  Serial.print(
  F("Readed
  data: "));
  //Dump a
  byte
  array to
  Serial for
  (byte i =
  0; i < 10;
  i++) {
  Serial.wr
  ite(buffer
  [i]);
```

```
cd.write(
buffer[i])
;
}
Serial.println();
}
}
}
else
{
Serial.println("Ac
cess denied");
lcd.clear();
lcd.print("UNRE
ADABLE
CARD");
```