Point Of Sale System

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Technology degree in Information Technology

by

Sivakumar G (37120070)
Ravikumar P (37120058)

DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600 119

MARCH – 2021

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with "A" Grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600 119.
Phone: 044 - 2450 3150 / 51 / 52 / 54 / 55 Fax: 044 - 2450 2344
www.sathyabama.ac.in

ISO 9001:2015

SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of Sivakumar G (37120070.) and Ravikumar P (37120058.) who carried out the project entitled "Point Of Sale System" under our supervision from November 2020 to March 2021.

Internal Guide

Mrs. Vimali J S, M.Tech.,(Ph.D.,)

Head of the Department

Dr. R. SUBHASHINI, M.E., Ph.D.,

Submitted for Viva voice Examination held on _____

Internal Examiner **External Examiner**

DECLARATION

We G.Sivakumar and P.Ravikumar hereby declare that the project report entitled on “Point of Sale System” done by us under the guidance of Mrs. Vimali JS, M.Tech., (Ph.D.,) at Sathyabama Institute of Science And Technology, Semmancheri, Chennai – 600 119 and is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE:

PLACE: Chennai **SIGNATURE OF THE CANDIDATES**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **Sathyabama Institute of Science and Technology** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D**, **Dean**, School of Computing, **Dr.R.Subhashini M.E., Ph.D., Head of the Department** of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my **Project Guide Mrs. Vimali J S M.Tech., (Ph.D.**,) for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

ABSTRACT

This project is about the point of sale (POS) system Implemented in Embedded Systems, which is being done for the organization LCS Controls Private Limited. Point of sale is the point at which a customer makes a payment to the merchant after buying their product. After receiving payment, the merchant may issue a receipt or bill for the transaction, which is usually printed but can also be dispensed with or sent electronically. To calculate the amount to be paid by the customer for what he/she purchased, the merchant may use various devices such as weighing scales, barcode scanners, and cash registers, to control all these devices a microcontroller is used in hardware. The microcontroller processes all the input data given to it and output's the processed data to the required output console. To process those data, a programmer should give a proper algorithm to the microcontroller through any programming language which is accepted by the microcontroller. Ex. C, Python etc. In this project a proper algorithm is given to the microcontroller through C programming language, in order to solve the point of sale problem. The problem can be solved by getting data input to the microcontroller through various input devices such as weighing scales, Keyboards and output as a bill to the billing machine. This system is an integrated weighing point of sale (POS) system, such that the weighing scale measured weight for each item is directly displayed in the output screen. POS system enables quick product return processes. When a customer cancels a purchase, the merchant can process it through POS system with just a few clicks. The number of stock that was reduced automatically increases again once the purchase has been cancelled. The Accounting details of a grocery store can be stored in well organized manner, either it may be the tax amount or the total revenue collected by the store.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION TO THE COMPANY

LCS Controls PVT. LTD. is a service oriented company who are Designers, manufacturers and “end-to-end” total solution providers in industrial weighing, batching, bagging, packing and allied automation with a full-fledged manufacturing facility in industrial hub of Chennai, India. They also provide embedded electronics, mechanical items, electrical drives, material handling equipments, sensors & pneumatic actuators, real-time software etc.

Figure 1.1 : LCS Controls Pvt Ltd. Logo

1.2 INTRODUCTION TO THE PROJECT

The point of sale (POS) is the place or time a retail transaction is performed or completed. During the point of sale, the merchant summarizes the amount owed by his/her customer, indicating that amount either through invoice or through cash register printouts and can show the options for the customers to pay those bills. It can be also called as a point where customer does payement to the merchant for any service done, any exchange of goods or things. The next step to the payment, the customer will be given with the receipt in acknowledgment to the payment customer had done.

Figure 1.2 : Sample POS System

In this project a suitable algorithm is given to the LPC1548/1549, an ARM Cortex-M3 processer to perform the task and to overcome the objective of Point of sale activity.

1.3 BACKGROUND OF THE PROJECT

This Project can differentiate each item through an unique identifier called item code. Each item has its own unique item code number and its quantity is measured in twotypes, one is weight and the other is pieces. Since, the memory size is limited POS System implemented in the LPC1548/1549 microcontroller can store data of 500 Items, 1000 bills per day and it's summary (max), most recent 2 months bill summary, most recent 2 years bill summary. Each Item entered for billing are stored as a backup at the spot which can handle power cut situation so that it can retrieve all previously entered items when power is back. The user is allowed to edit/delete the items that he/she entered. The weight is measured through the weight scale and it sends the analog data to the microcontroller, the microcontroller performs ADC operations and can determine the actual weight through calibrated values. The bill can be printed through a bill printer (i.e) connected using UART protocol with the hardware. A provision has been given to print the brief summary of today's bill, Monthly's bill, yearly's bill. The list of items and shop name are loaded into the hardware through a PC based software (Windows) and an android application, detailed bill summary is retrieved from the hardware through the same software by storing it as a text file in the PC/ so that it can be viewed at any time. All the communications between the hardware and the pc will be serially at present through UART protocol.

1.4 AIM OF THE PROJECT

The main aim of the project is to make an integrated weighing point of sale (POS) system, such that the weighing scale measured weight for each item is directly processed by the hardware to get the desired output and to make the billing process more faster. The Accounting details for any store can be stored in a well organised manner, either it may be the tax amount or the total revenue collected by the store.

Figure 1.4 : Single POS System

1.5 SCOPE OF THE PROJECT

POS solutions can lower the cost of doing business while increasing productivity, improving the bottom line. Upgrading from an Cash register to a point of sale system will result in a fast return on investment (ROI), both in rupees and time spent on day-to-day operations. As the days of analog technology continue to fall further out of use behind us, so do cash registers. Today, everything is digital, and everything is faster. POS systems will be simple for employees to learn, which will result in shortening training time and help them to be more productive overall. Point of sale systems have reporting features that allow to keep a close eye on sales, profits, and expenses like Cost of Goods Sold (COGS). POS reports give data in real-time, and formatted with easy-to-read information. It also helps streamline the accounting process. Old-fashioned cash registers force accountants to sort through hundreds of receipts, but with a POS system, you can print reports and, in many cases, import data directly with a suitable software.

CHAPTER 2
LITERATURE REVIEW

2.1 AN ANALYSIS OF POINT OF SALE SYSTEM PHYSICAL CONFIGURATIONS AND SECURITY MEASURES IN ZIMBABWEAN SMES (2017)

The local client server model for POS system is referred from the research paper which helped this project to build a robust topology for communication in a network. This client server model made data communication over a network very fast. This model made connection seamless were disconnection cannot happen at all (under normal case).

2.2 POINT-OF-SALES SYSTEMS IN FOOD AND BEVERAGE INDUSTRY EFFICIENT TECHNOLOGY AND ITS USER ACCEPTANCE (2017)

This paper made the project more user responsive and completely describes the ease in usability of the POS System. The project is made trustful with use at most ease. The project is fast in usability and collects user's attitudes towards the system.

2.3 R&D LCS CONTROLS PVT. LTD.

The R&D of the company researched many technical works and made the project a good look at final. They referred non-volatile memory to store and process data any time. An external device communication via UART protocol is established, LED and LCD display is made responsive with help of the team.

CHAPTER 3
METHODOLOGY

3.1 LANGUAGES AND LIBRARIES USED

3.1.1 C Language

C is a general-purpose, procedural computer programming language supporting structured programming, lexical variable scope, and recursion, with a static type system. By design, C provides constructs that map efficiently to typical machine instructions. The main features of C language include low-level access to memory, a simple set of keywords, and clean style, these features make C language suitable for system programmings like an operating system or compiler development.

C is an imperative procedural language. It was designed to be compiled to provide low-level access to memory and language constructs that map efficiently to machine instructions, all with minimal runtime support. Despite its low-level capabilities, the language was designed to encourage cross-platform programming. A standards-compliant C program written with portability in mind can be compiled for a wide variety of computer platforms and operating systems with few changes to its source code.

C is a procedural programming language. It was initially developed by Dennis Ritchie in the year 1972. It was mainly developed as a system programming language to write an operating system. The main features of C language include low-level access to memory, a simple set of keywords, and clean style, these features make C language suitable for system programmings like an operating system or compiler development.

Many later languages have borrowed syntax/features directly or indirectly from C language. Like syntax of Java, PHP, JavaScript, and many other languages are mainly based on C language. C++ is nearly a superset of C language (There are few programs that may compile in C, but not in C++).

Figure 3.1 : C Language

3.1.2 C#

C# is pronounced as "C-Sharp". It is an object-oriented programming language provided by Microsoft that runs on .Net Framework. By the help of C# programming language, we can develop different types of secured and robust applications:

Window applications

Web applications

Distributed applications

Web service applications

Database applications etc.

C# is approved as a standard by ECMA and ISO. C# is designed for CLI (Common Language Infrastructure). CLI is a specification that describes executable code and runtime environment. C# programming language is influenced by C++, Java, Eiffel, Modula-3, Pascal etc. languages.

C# is a modern, general-purpose programming language that can be used to perform a wide range of tasks and objectives that span over a variety of professions. C# is primarily used on the Windows .NET framework, although it can be applied to an open source platform. This highly versatile programming language is an object-oriented programming language (OOP)—which isn't very common—and fairly new to the game, yet already a reliable crowd pleaser

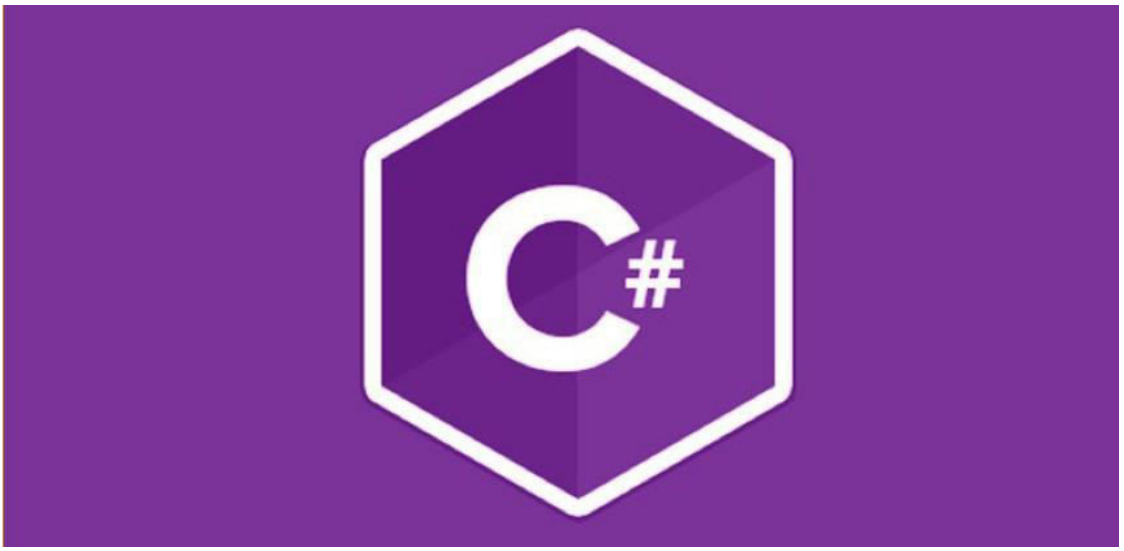


Figure 3.2 : C#

3.1.3 Java

Java is a class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let application developers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation.

Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. As of 2019, Java was one of the most popular programming languages in use according to GitHub, particularly for client-server web applications, with a reported 9 million developers.

The Java syntax is similar to C++, but is strictly an object-oriented programming language. For example, most Java programs contain classes, which are used to define objects, and methods, which are assigned to individual classes. Java is also known for being more strict than C++, meaning variables and functions must be explicitly defined. This means Java source code may produce errors or "exceptions" more easily than other languages, but it also limits other types of errors that may be caused by undefined variables or unassigned types.

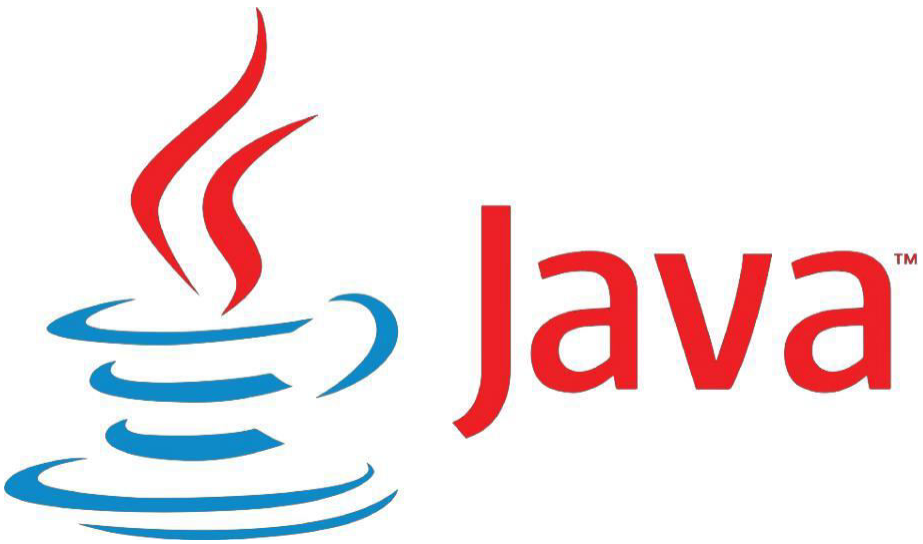


Figure 3.3 : Java

3.1.4 LPC1549 Header Files (C Language)

The LPC1549 LPCXpresso™ board with NXP®'s LPC1549 Cortex-M3 microcontroller is designed to make it as easy as possible to get started with the project. The LPC1549 header library has all the driver codes to make the code what engineer types compatible with the microcontroller.

The platform is comprised of a simplified Eclipse-based IDE and low-cost target boards which include an attached JTAG debugger. LPCXpresso is an end-to-end solution enabling embedded engineers to develop their applications from initial evaluation to final production.

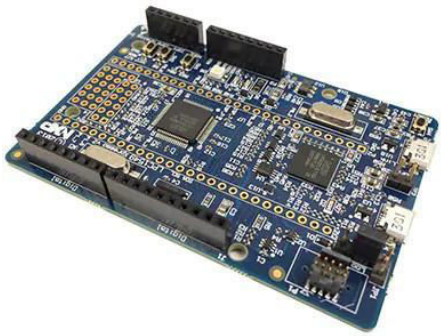


Figure 3.4 : LPC1549 LPCXpresso board

3.2 SOFTWARES USED

3.2.1 Keil

Keil MDK is the complete software development environment for a wide range of Arm Cortex-M based microcontroller devices. MDK includes the µVision IDE and debugger, Arm C/C++ compiler, and essential middleware components. It supports all silicon vendors with more than 7,500 devices and is easy to learn anduse.

The Keil µVision Debugger accurately simulates on-chip peripherals (I²C, CAN, UART, SPI, Interrupts, I/O Ports, A/D Converter, D/A Converter, and PWM Modules) of your 8051 device. Simulation helps you understand hardware configurations and avoids time wasted on setup problems. Additionally, with sWhen starting a new project, simply select the microcontroller you use from the Device Database and the µVision IDE sets all compiler, assembler, linker, and memory options for you.

Numerous example programs are included to help you get started with the most popular embedded 8051 devicesimulation, you can write and test applications before target hardware is available.

When you are ready to begin testing your software application with target hardware, use the MON51, MON390, MONADI, or FlashMON51 Target Monitors, the ISD51 In-System Debugger, or the ULINK USB-JTAG Adapter to download and test program code on your target system.

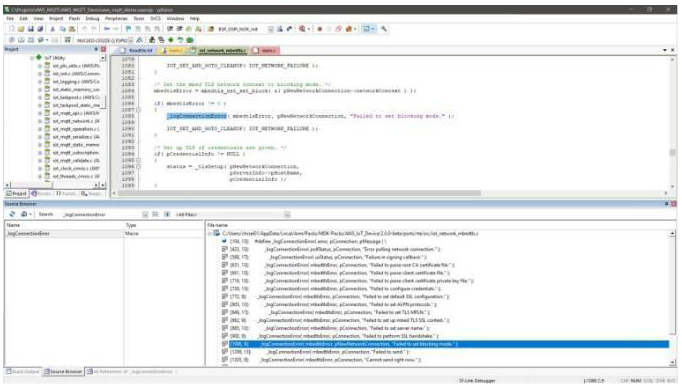


Figure 3.5 : Keil Software

3.2.2 Flash Magic Programmer

Flash Magic is a PC tool for programming flash based microcontrollers from NXP using a serial or Ethernet protocol while in the target hardware.

Flash Magic is a PC burner tool for programming flash memory based microcontroller using serial or Ethernet protocol built by NXP. This tool helps the developer to easily burn the hex file generated by the embedded software like Keil µvision for 8051 and ARM microcontrollers or MPLAB for PIC microcontrollers. It is available free and easy to install in Windows or Mac OS based PC.

Features : Automatically program checksums. Using the supplied checksum calculation routine your firmware can easily verify the integrity of a Flash block, ensuring no unauthorized or corrupted code can ever be executed. Check which Flash blocks are blank or in use with the ability to easily erase all blocks in use. Reprogram the Boot Vector and Status Byte with the help of confirmation features that prevent accidentally programming incorrect values. Powerful, flexible Just In Time Code feature. Write your own JIT Modules to generate last minute code for programming, for example serial number generation. Check which Flash blocks are blank or in use with the ability to easily erase all blocks in use. Control the DTR and RTS RS232 signals to place the device into BootROM and Execute modes automatically (requires hardware support). Support programming certain LPC1xxx/LPC2xxx devices via Ethernet. Build your own Flash Magic based applications using the DLLs for C, C++, Python.

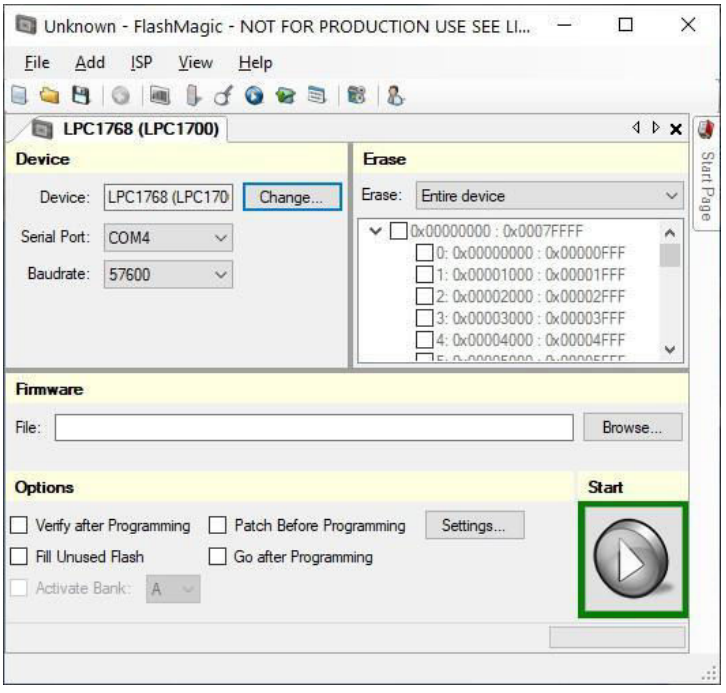


Figure 3.6 : Flash Magic Programmer

3.2.3 Hercules Setup Utility

Hercules SETUP utility is useful serial port terminal (RS-485 or RS-232 terminal), UDP/IP terminal and TCP/IP Client Server terminal. It was created for HW group internal use only, but today it's includes many functions in one utility and it's Freeware! With our original devices (Serial/Ethernet Converter, RS-232/Ethernet Buffer or I/O Controller) it can be used for the UDP Config.

Hercules SETUP provides the users for setting up environment the UDP and TCP serial terminal ports. You don't have to install this software on your computer to start the operations instead it comes with an execution-able file that can allow you to start working as soon as you get it.

You can also view file formats, enable macros, transfer files, and use the debugging features of the serial port terminals through this application. Furthermore, it is easy to use and have simple buttons for implementing the changes.

One of the drawbacks of this application is that it is not supported by many devices, which can be a bit problematic if you are expecting to use this application with many devices.

You can setup UDP and TCP serial terminal ports with Hercules SETUP. However, its restrictions include the limited device support.

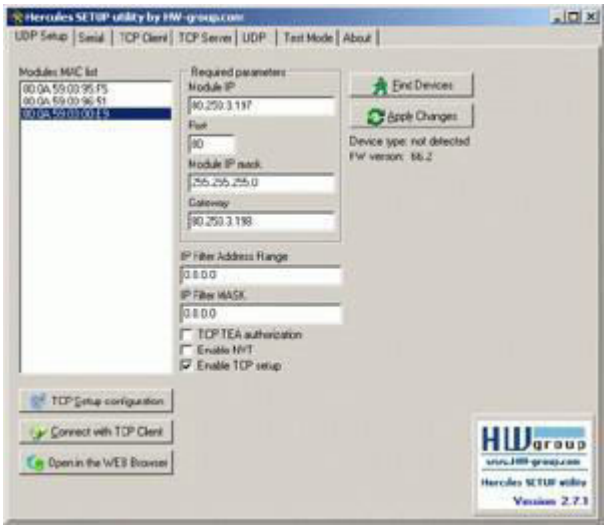


Figure 3.7 : Hercules Setup Utility

3.2.4 Visual Studios 2010

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that expand the functionality at almost every level—including adding support for source control systems (like Subversion and Git) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Azure DevOps client: Team Explorer).

Visual Studio supports 36 different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, C++/CLI, Visual Basic .NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML, and CSS. Support for other languages such as Python, Ruby, Node.js, and M among others is available via plug-ins. Java (and J#) were supported in the past.

Figure 3.8 : Visual Studios 2010

3.2.5 Android Studios

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains, IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems or as a subscription-based service in 2020. It is a replacement for the Eclipse Android Development Tools (E-ADT) as the primary IDE for native Android application development.

Android Studio was announced on May 16, 2013 at the Google I/O conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014.The first stable build was released in December 2014, starting from version 1.0.

On May 7, 2019, Kotlin replaced Java as Google's preferred language for Android app development.Java is still supported, as is C++.

Android Studio supports all the same programming languages of IntelliJ (and CLion) e.g. Java, C++, and more with extensions, such as Go; and Android Studio 3.0 or later supports Kotlin and "all Java 7 language features and a subset of Java 8 language features that vary by platform version."External projects backport some Java 9 features. While IntelliJ states that Android Studio supports all released Java versions, and Java 12, it's not clear to what level Android Studio supports Java versions up to Java 12 (the documentation mentions partial Java 8 support). At least some new language features up to Java 12 are usable in Android.

Figure 3.9 : Android Studio

3.3 MATERIALS/HARDWARE USED

3.3.1 RS232 Connector & Bill Printer (32 char)

RS232 is a standard protocol used for serial communication, it is used for connecting computer and its peripheral devices to allow serial data exchange between them. As it obtains the voltage for the path used for the data exchange between the devices. Bill printer of 32 characters per line.

RS232 connector is a port used for data exchange between equipments. It was designed for data exchange between DTE (Data Terminal Equipment) or PC and DCE (Data Communication Equipment) or MODEM. The need for RS232 came from limitations raised by parallel data exchange. RS232 uses serial communication protocol where data exchange is done bit by bit. Although RS232 is later replaced by faster USB (Universal Serial Bus) it is still popular in some areas. RS232 used to have 25 pin, now it is shrunk to just 9 pin.

New RS232 has nine pins as mentioned earlier. These nine pins are arranged in the port as shown in RS232 Connector Pinout. The DCE and DTE ports are exactly similar except for the direction of data flow.

When you want a simple communication interface between two units. A two pin full duplex communication can be establishes easily on RS232 port.

RS232 is used in systems where clock sharing is difficult. RS232 is ASYNCHRONOUS so there will be no clock sharing between systems. All you need to do is set data bit rate for each unit. Once baud rate is set the units will sample the data according to set baud rate.

Figure 3.10 : RS232 Connector & Bill Printer

3.3.2 Weighing Scale

A weighing scale (or weighing balance) is a device to measure weight or mass. One plate holds an object of unknown mass (or weight), while known masses are added to the other plate until static equilibrium is achieved and the plates level off, which happens when the masses on the two plates are equal. A spring scale will make use of a spring of known stiffness to determine mass (or weight). Suspending a certain mass wil extend the spring by a certain amount depending on the spring's stiffness (or spring constant). The heavier the object, the more the spring stretches, as described in Hooke's law.

A scale or balance is a device to measure weight or mass. These are also known as mass scales, weight scales, mass balances, weight balances.

The traditional scale consists of two plates or bowls suspended at equal distances different physical principles also exist.from a fulcrum. One plate holds an object of unknown mass (or weight), while knownmasses are added to the other plate until static equilibrium is achieved and the plates level off, which happens when the masses on the two plates are equal. The perfect scale rests at neutral. A spring scale will make use of a spring of known stiffness to determine mass (or weight). Suspending a certain mass will extend the spring by a certain Amount depending on the spring's stiffness (or spring constant). The heavier the object, the More the spring stretches, as described in Hooke's law. Other types of scales making use of Some scales can be calibrated to read in units of force (weight) such as newtons instead of units of mass such as kilograms. Scales and balances are widely used in commerce, as many products are sold and packaged by mass.

Figure 3.11 : Weighing Scale

3.3.3 LPC1548/1549 MICROCONTROLLER

The LPC15xx are ARM Cortex-M3 based microcontrollers for embedded applications featuring a rich peripheral set with very low power consumption. The ARM Cortex-M3 is a next generation core that offers system enhancements such as enhanced debug features and a higher level of support block integration.

The LPC15xx operate at CPU frequencies of up to 72 MHz. The ARM Cortex-M3 CPU incorporates a 3-stage pipeline and uses a Harvard architecture with separate local instruction and data buses as well as a third bus for peripherals. The ARM Cortex-M3 CPU also includes an internal prefetch unit that supports speculative branching.

The LPC15xx includes up to 256 kB of flash memory, 32 kB of ROM, a 4 kB EEPROM, and up to 36 kB of SRAM. The peripheral complement includes one full-speed USB 2.0 device, two SPI interfaces, three USARTs, one Fast-mode Plus I2C-bus interface, one C_CAN module, PWM/timer subsystem with four configurable, multi- purpose State Configurable Timers (SCTimer/PWM) with input pre-processing unit, a Real-time clock module with independent power supply and a dedicated oscillator,
two 12-channel/12-bit, 2 Msamples/s ADCs, one 12-bit, 500 kSamples/s DAC, four voltage comparators with internal voltage reference, and a temperature sensor. A DMA engine can service most peripherals.

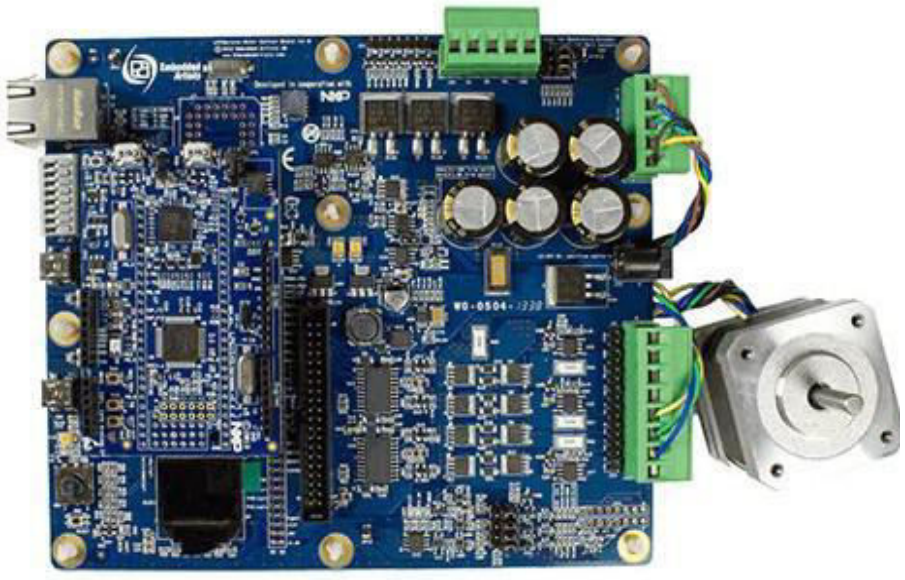


Figure 3.12 : LPC1548/1549 MICROCONTROLLER

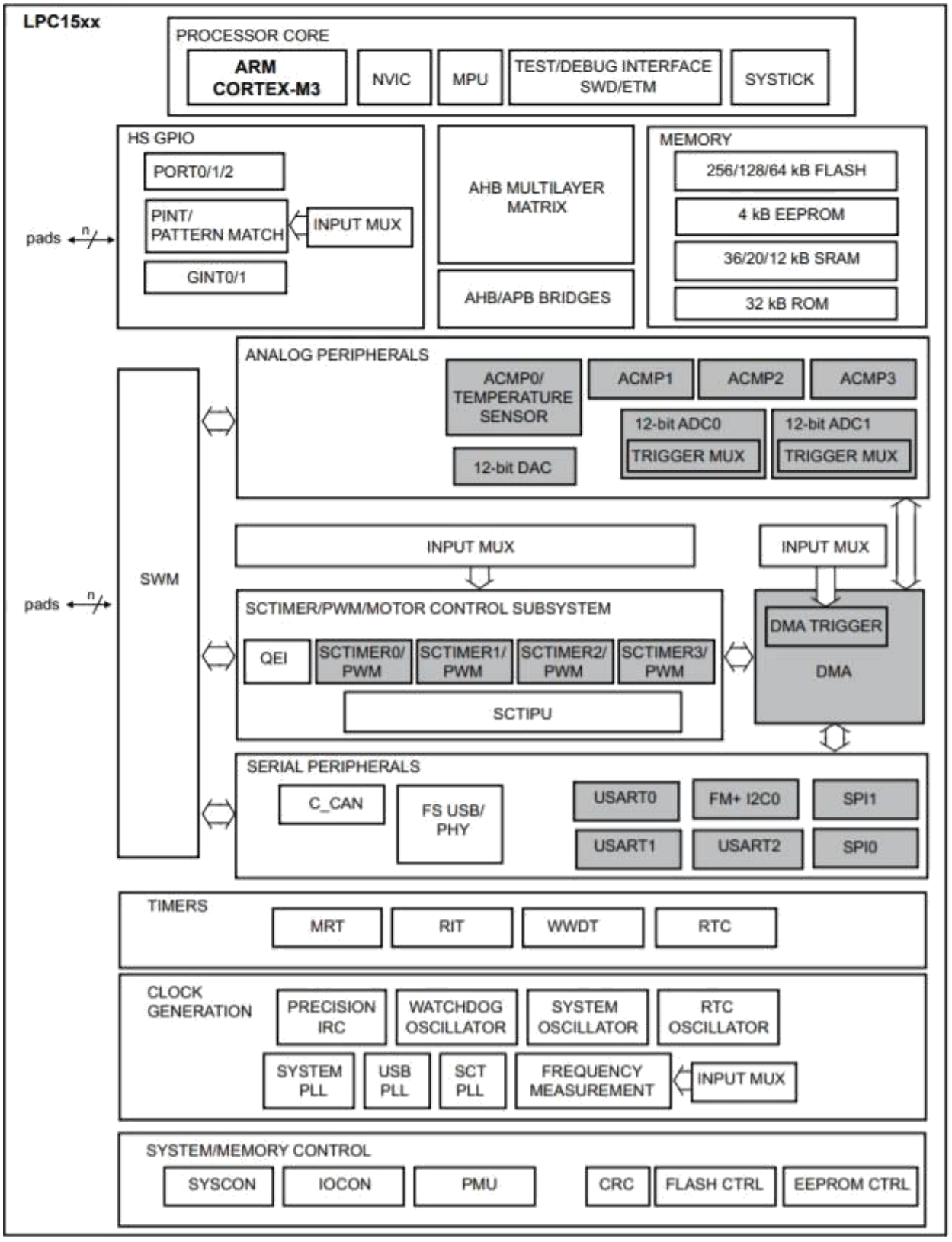


Fig 3.13：LPC15XX Block Diagram

3.4 METHODS AND ALGORITHM

3.4.1 Algorithm StartBilling()

STEP 1: Check if there is any previously entered items without billed, stored in E2 Prom.

STEP 1.1: If there Items exist read it and continue billing

STEP 1.2: If there is no item in the E2 Prom start billing newly.

STEP 2: Get the item code from the IBM Keyboard

STEP 2.1: if Item Count equals zero

STEP 2.1.1: If menu key is pressed do device menu operations like calibration, diagnostics, set date/time etc.

STEP 2.1.2: if F1 key is pressed show bill summary.

STEP 2.1.3: if F3 key is pressed provide tare weight operation.

STEP 2.1.4: if ENTER key is pressed in the IBM Keyboard show cannot print since no item is entered

STEP 2.2: If item code is invalid do not process and repeat **STEP 2**

STEP 2.3: If item code is valid show the Item description corresponding to that item code.

STEP 2.4: if the input key is ENTER from the IBM keyboard

STEP 2.4.1: Print the Bill, remove entered items from E2 Prom

STEP 2.4.2: Update Today's, Monthly's, Yearly's Summary.

STEP 2.5: if the input is F4 key perform edit/delete operation for corresponding entered items.

STEP 3: If that particular item is measured in terms of weight get the weight directly from weight scale.

STEP 4: If that particular item is measured in terms of pieces get no. of pieces from IBM Keyboard directly.

STEP 5 : if the input key is ESC Key from the IBM Keyboard, repeat from **STEP 2**

STEP 6: After pressing enter key in the IBM keyboard, store the Items to the E2 PROM, Repeat from **STEP 2.**

3.5 FLOW CHART OF THE POS SYSTEM

The below depicted is the actual flow of the POS system in this project implemented in the microcontroller.

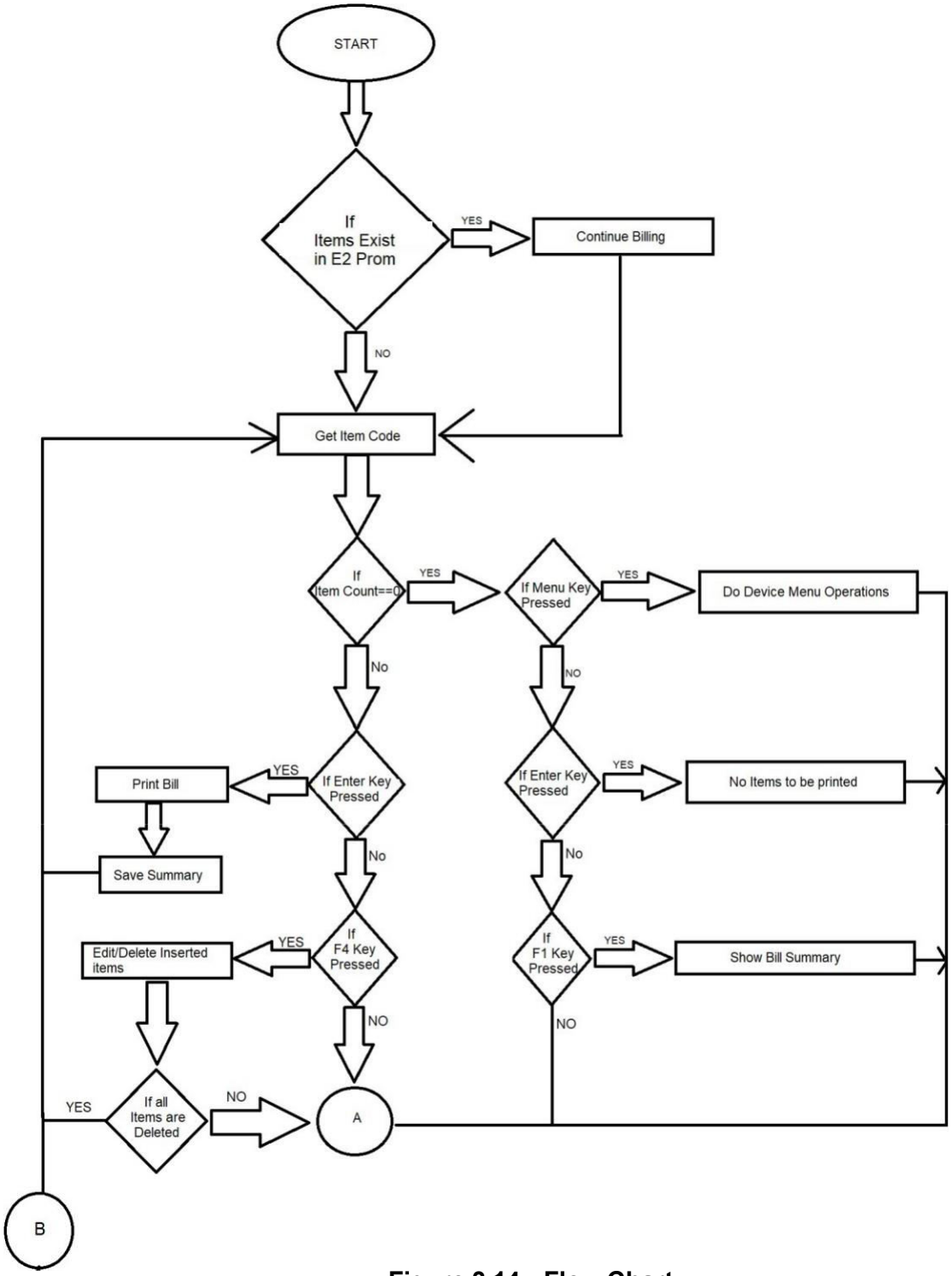


Figure 3.14 : Flow Chart

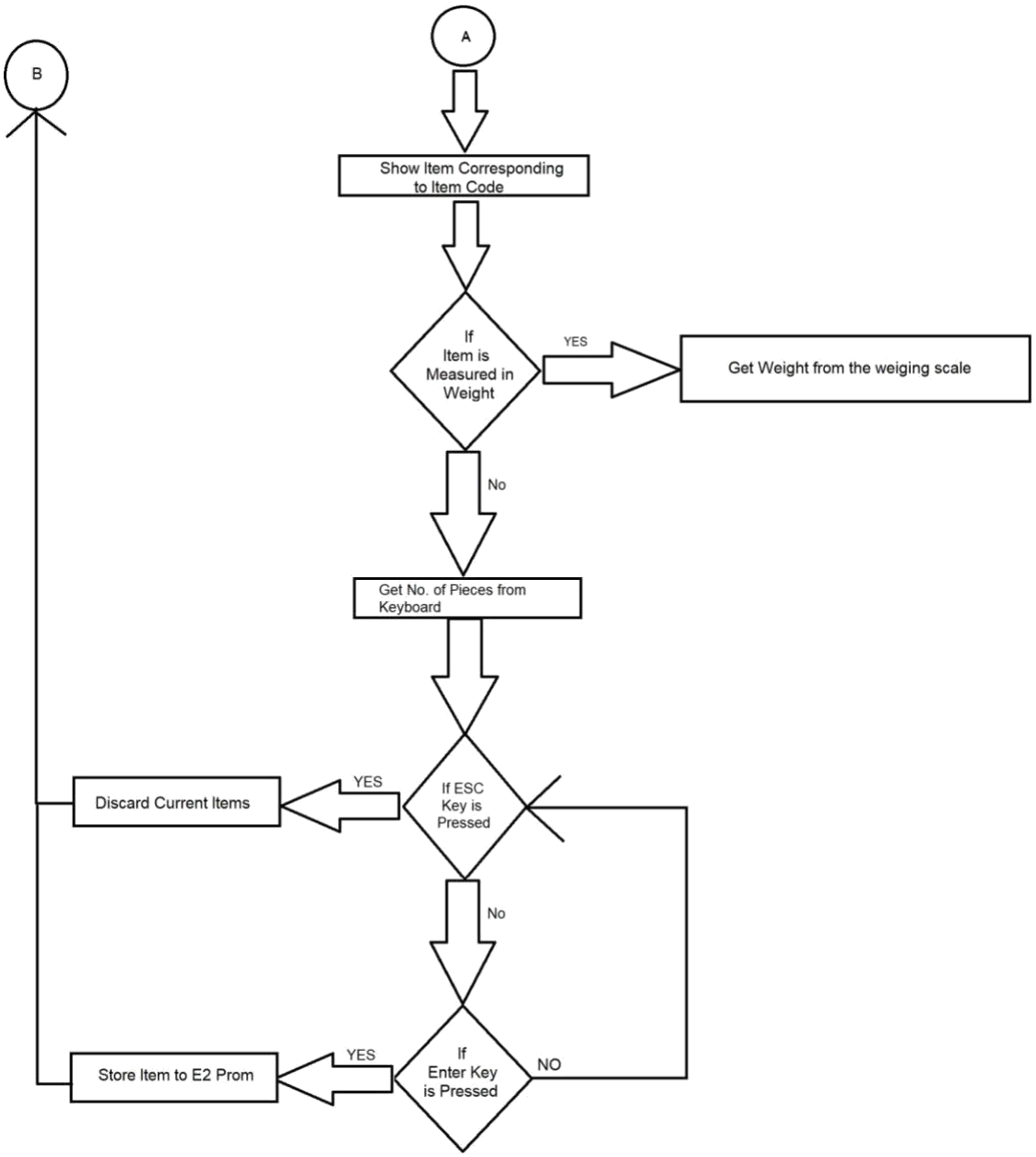


Figure 3.15 : Flow Chart (Cont.)

3.6 E2 PROM

EEPROM (also E^2 PROM) stands for electrically erasable programmable read-only memory and is a type of non-volatile memory used in computers, integrated in microcontrollers for smart cards and remote keyless systems, and other electronic devices to store relatively small amounts of data but allowing individual bytes to be erased and reprogrammed.

EEPROMs are organized as arrays of floating-gate transistors. EEPROMs can be programmed and erased in-circuit, by applying special programming signals. Originally, EEPROMs were limited to single-byte operations, which made them slower, but modern EEPROMs allow multi-byte page operations. An EEPROM has a limited life for erasing and reprogramming, now reaching a million operations in modern EEPROMs. In an EEPROM that is frequently reprogrammed, the life of the EEPROM is an important design consideration.

Flash memory is a type of EEPROM designed for high speed and high density, at the expense of large erase blocks (typically 512 bytes or larger) and limited number of write cycles (often 10,000). There is no clear boundary dividing the two, but the term "EEPROM" is generally used to describe non-volatile memory with small erase blocks (as small as one byte) and a long lifetime (typically 1,000,000 cycles). Many microcontrollers include both: flash memory for the firmware, and a small EEPROM for parameters and history.

3.7 E2 PROM ITEM BACKUP THAT IS ENTERED DURING BILLING

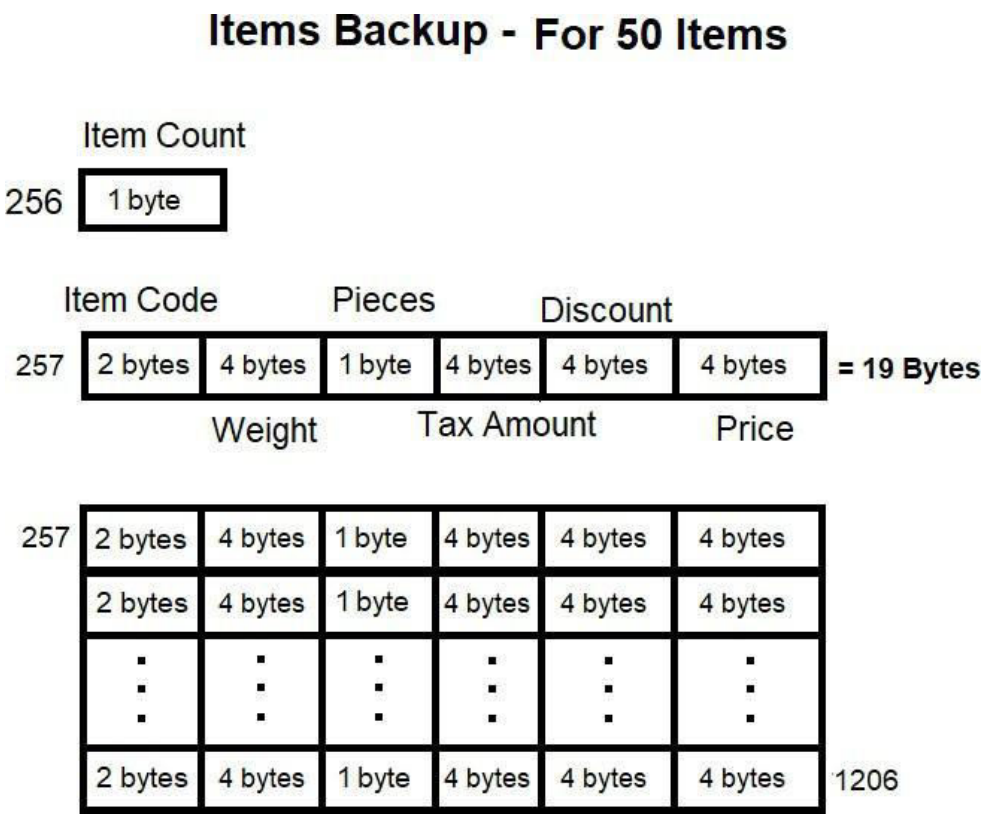


Figure 3.16 : Item Backup Memory Map

The above memory map is for 50 items at maximum.

3.8 E2 PROM TODAY'S BILL SUMMARY

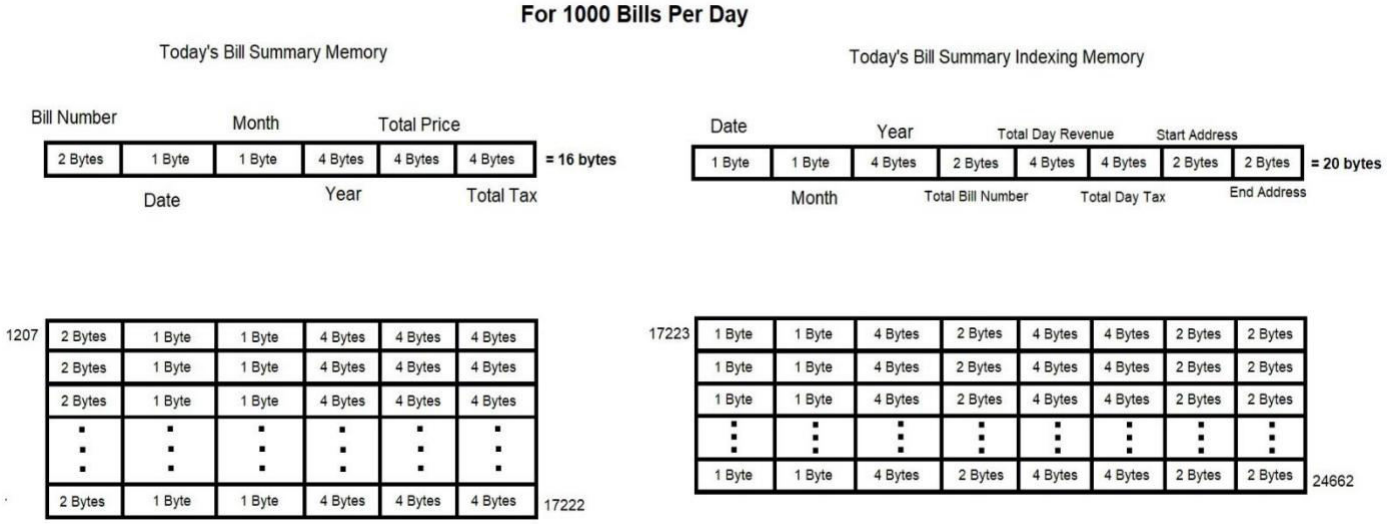


Figure 3.17 : Today's Bill Summary Memory Map

The Indexing memory acts as an index to the Today's bill summary memory which can give a detailed bill report for a particular day. The indexing memory represents the starting and ending memory location of the bill report for that particular date. Hence, today bill summary memory can be used as an circular buffer to store daily bill report, so that memory is utilized efficiently.

3.9 E2 PROM MONTHLY'S BILL SUMMARY

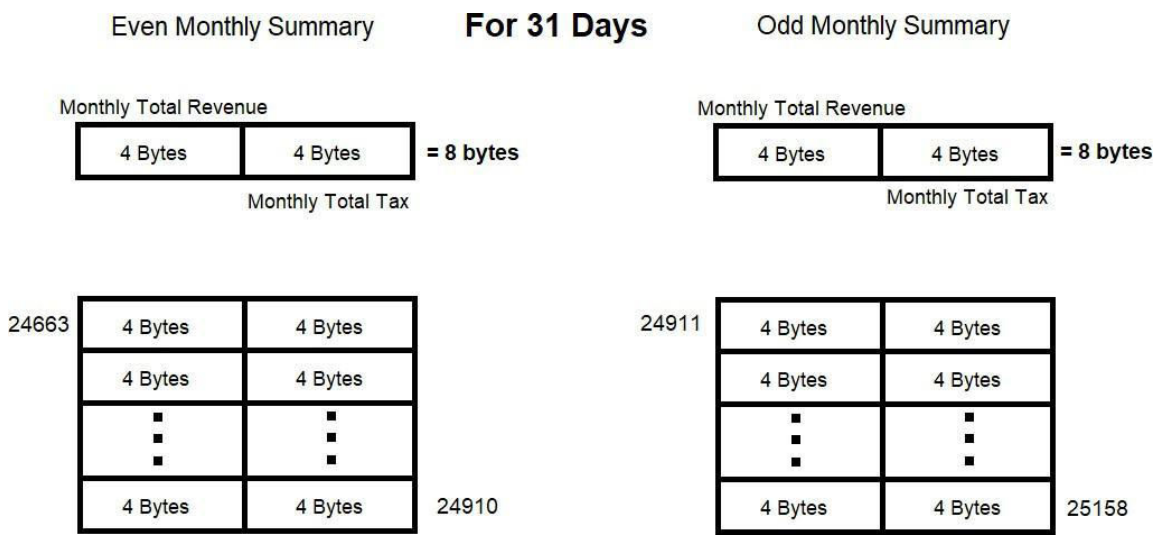


Figure 3.18: Monthly's Bill Summary Memory Map

The above memory map is done for two months of each 31 days assumed.

3.10 E2 PROM YEARLY'S BILL SUMMARY

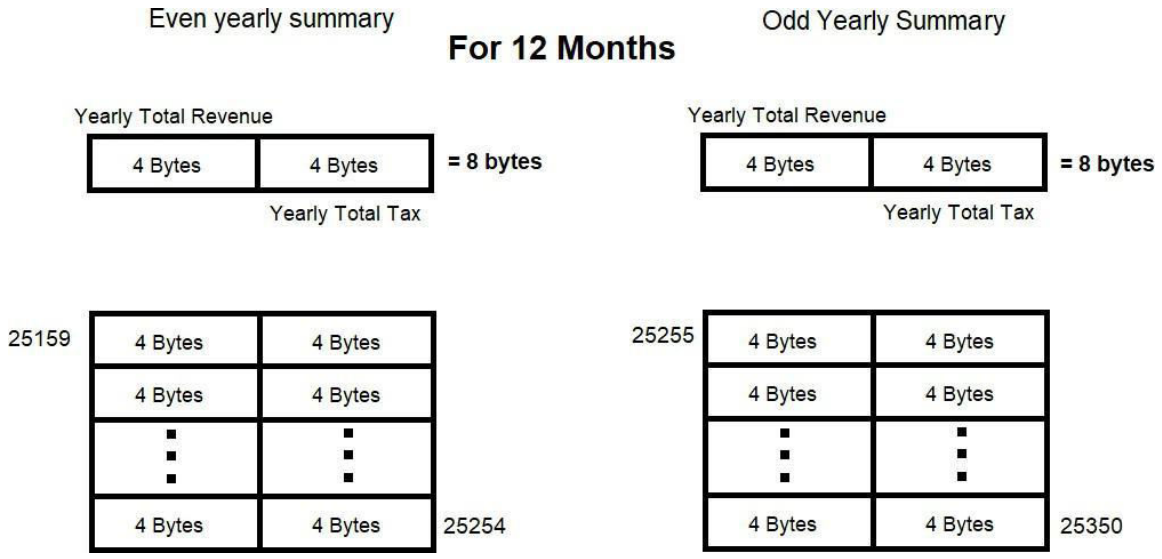


Figure 3.19: Yearly's Bill Summary Memory Map

The above memory map is done for two Years of each 366 days assumed.

3.11 SOCKETS

The android mobiles are used to communicate with the awew 2000 POS indicator wirelessly via socket. The IP address and port number of the indicator is given to the android application.

Sockets allow communication between two different processes on the same or different machines. To be more precise, it's a way to talk to other computers using standard Unix file descriptors. In Unix, every I/O action is done by writing or reading a file descriptor. A file descriptor is just an integer associated with an open file and it can be a network connection, a text file, a terminal, or something else.

To a programmer, a socket looks and behaves much like a low-level file descriptor. This is because commands such as read() and write() work with sockets in the same way they do with files and pipes.

Sockets were first introduced in 2.1BSD and subsequently refined into their current form with 4.2BSD. The sockets feature is now available with most current UNIX system releases.

A Unix Socket is used in a client-server application framework. A server is a process that performs some functions on request from a client. Most of the application-level protocols like FTP, SMTP, and POP3 make use of sockets to establish connection between client and server and then for exchanging data.

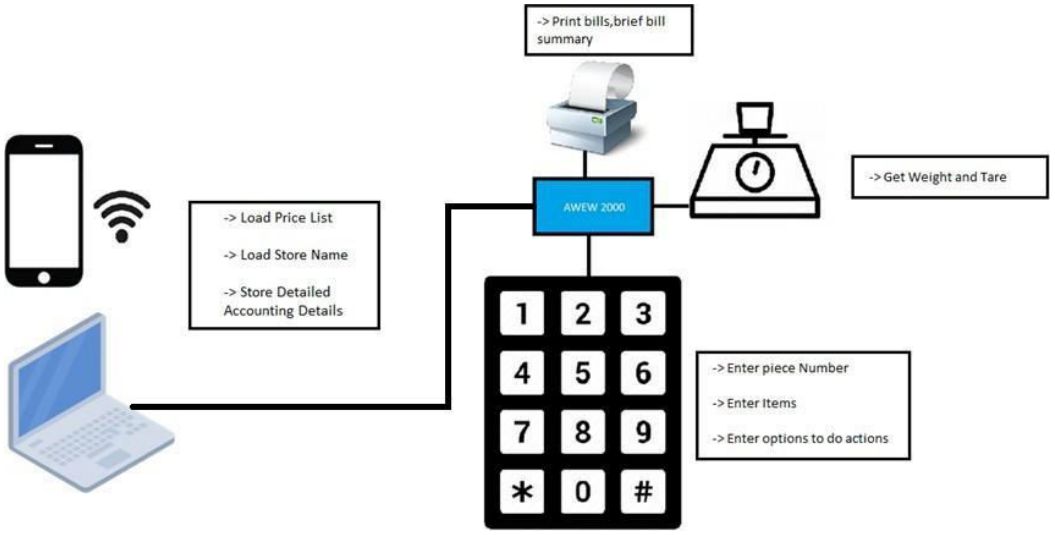


Figure 3.20 : Wireless Mobile Communication

3.12 INFORMATION COLLECTED BY MOBILE

The android mobile can collect,

➔ Today's Summary

➔ Date Wise Summary

➔ Current Monthly's Summary

➔ Previous Monthly's Summary

➔ Current Yearly's Summary

➔ Previous Yearly's Summary

➔ Price List

➔ Stock Report

All the above information's are collected by sending appropriate query.

TABLE 3.13: QUERY INFORMATION

Query	Description
%LPL$	This query will load the price list csv to the indicator.
%LSN$	This query will load the Store Name csv to the indicator.
%STS$	This query will retrieve the today's summary information from the indicator.
%SDWSDDMMYYYY$	This query will retrieve the date wise summary for the given date (DD), month (MM), year (YYYY) from the indicator.
%SCMS$	This query will retrieve the current month summary information from the indicator.
%SPMS$	This query will retrieve the previous month summary information from the indicator.
%SCYS$	This query will retrieve the current year summary information from the indicator.
%SPYS$	This query will retrieve the previous year summary information from the indicator.
%SIPL$	This query will retrieve the price list csv i.e. Stored in the indicator.
%SSR$	This query will retrieve the stock report of the items from the indicator.
%SISN$	This query will retrieve the store name headers from the indicator.

TABLE 3.14: DATE WISE SUMMARY QUERY EXPANSION

%SDWS	**DD**	**MM**	**YYYY**	**$**
Query to retrieve Date Wise Summary	Date (ex. 09)	Month (ex. 10)	Year (ex. 2020)	Ending Character

CHAPTER 4

RESULTS AND DISCUSSION

4.1 RESULT

The project was tested with many real time inputs example Item code, weight through weight scale, number of pieces and got the expected output as efficient as possible.

4.2 DISCUSSION

This result driven project is considered to be the Version 1.0 of the project. The upcoming versions are developed once the project gets into the market and collects the feedback what users want. The company and R&D team has speculated some of the features that needs to be added in the POS system. The features are voice recognition, multi language facilities to bring nativity into the project, biometrics to access users etc.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

In this paper, the POS system implemented on the microcontroller LPC1548/1549 and making compatible with the awew 2000 device is explained with atmost perfection, as much as possible. This kind of Integrated weight POS system makes the usablity more faster by making an user to stick their eyes directly to the product and the device rather than always seeing the weighing scale each and every time when a product is loaded.

This POS system can print reports and, in many cases, import data directly with suitable PC/Android software which makes the user to well structure their accounting details.

5.2 FUTURE WORK

The above POS System can be made to work with voice recognition which can be compatible with all Indian languages. Multi language bill printer can be implemented such that nativity can be achieved.

REFERENCES

1. Sai, K. (2017). **An Analysis of Point of Sale Systems Physical Configurations and Security Measures in Zimbabwean SMEs**.
2. Yomayra Ramos, Dr. Angel Ojeda Castro (2017) **Point-Of-Sales Systems in Food and Beverage Industry: Efficient Technology and Its UserAcceptance**
3. https://retailexpress.com.au/blog/8-benefits-of-point-of-sale-pos-systems/
4. https://solutiondots.com/blog/point-of-sale-system/
5. https://www.investopedia.com/terms/p/point-of-sale.asp.
6. R&D LCS Controls Pvt. Ltd.

APPENDICES

A) SAMPLE CODE

```
//getting Item Code
short GetCode(ItemMaster g_Items_Available[], //;Bill *Current_Bill, char *Item_Count, short
*Total_Items){
        const unsigned char Digit_Increement_Length =
 2; const unsigned char Code_Total_Length = 4;
        char Inactiveness=0, Inactiveness1 = 0;
        char PieceFlag=0, AnimationFlag = 0;
        char Item_Description[21],Index;
        short Item_Code,Item_Code1=0;
        float Item_Weight;
        unsigned char AcceptBuff[21];
        unsigned char CorrectBuff1[21]={" "};
        unsigned char CodeBuff[21];
        unsigned char digit = 0;
        unsigned char blink = 0;
 memcpy(CodeBuff," ",Total_LCD_Line_Length);
        Keypad_Entry_Flag = 1;
        sprintf(CodeBuff,"Code:"); while(1)

        {
                IBMKeyValue=0;
                memcpy(AcceptBuff,CorrectBuff1,Code_Total_Length);
                if(blink < 10)
                AcceptBuff[digit] = '_';
                if(blink > 25)
                        blink=0;
                blink++;
                memset(LcdBotBuff,' ',16);
                memcpy(CodeBuff+5,AcceptBuff,Code_Total_Length);
```

```
sprintf(CodeBuff+9,"%7.2f",g_Present_Price);
sprintf(LcdBotBuff,"%d",*Item_Count+1);
LcdDisplay(TOP,(char  *)&CodeBuff);
KeySence();
DisplayCurrWt();
Item_Weight=(float)LoadCalibData.CurWeight;
if(digit<=Digit_Increement_Length){
        switch(IBMKeyValue)
        {
                case ATKey_1:
                CorrectBuff1[digit]  = '1';
                digit++;
                AnimationFlag=Item_Code1=Inactiveness1=Inactiveness=0;
                break;

                case ATKey_2:
                CorrectBuff1[digit]  = '2';
                digit++;
                AnimationFlag=Item_Code1=Inactiveness1=Inactiveness=0;
                break;

                case ATKey_3:
                CorrectBuff1[digit]  = '3';
                digit++;
                AnimationFlag=Item_Code1=Inactiveness1=Inactiveness=0;
                break;

                case ATKey_4:
                CorrectBuff1[digit]  = '4';
                digit++;
                AnimationFlag=Item_Code1=Inactiveness1=Inactiveness=0;
                break;

                case ATKey_5:
```

```
CorrectBuff1[digit]  = '5';
digit++;
AnimationFlag=Item_Code1=Inactiveness1=Inactiveness=0;
break;

case ATKey_6:
CorrectBuff1[digit]  = '6';
digit++;
AnimationFlag=Item_Code1=Inactiveness1=Inactiveness=0;
break;

case ATKey_7:
CorrectBuff1[digit]  = '7';
digit++;
AnimationFlag=Item_Code1=Inactiveness1=Inactiveness=0;
break;

case ATKey_8:
CorrectBuff1[digit]  = '8';
digit++;
AnimationFlag=Item_Code1=Inactiveness1=Inactiveness=0;
break;

case ATKey_9:
CorrectBuff1[digit]  = '9';
digit++;
AnimationFlag=Item_Code1=Inactiveness1=Inactiveness=0;
break;

case ATKey_0:
CorrectBuff1[digit]  = '0';
digit++;
AnimationFlag=Item_Code1=Inactiveness1=Inactiveness=0;
break;
```

```
                    blink=10;
                }
        }
            CorrectBuff1[digit] = 0;
            if(strlen(CorrectBuff1) == 3 && ((Inactiveness++)>2 || IBMKeyValue ==
ENTERKEY)){
                Inactiveness=3;
                Item_Code=(short)atoi((const char *)&CorrectBuff1);
                if(Item_Code<=*Total_Items && Item_Code != 0){
                    if(Item_Code1 != Item_Code)
                        { InitializePriceList(g_Items_Available,Item_Code);
Item_Code1= Item_Code;
                    }
                    if(strlen(g_Items_Available[(Item_Code-
1)%5].Item_Description)>7) memcpy(LcdBotBuff+3,g_Items_Available[(Item_Code-
1)%5].Item_Description,7);
      else sprintf(LcdBotBuff+3,g_Items_Available[(Item_Code-1)%5].Item_Description);
                    if(CheckUnit(g_Items_Available, Item_Code)==0)
sprintf(LcdBotBuff+11,"%.2f",Item_Weight/100);
                     else sprintf(LcdBotBuff+11,"Pc");

                    if((*Item_Count) < g_Max_ItemsPerBill
&& CheckUnit(g_Items_Available,Item_Code) == 1){
                        if(PieceFlag == 0) { PieceFlag=1; continue;}
                        CodeBuff[8]=' ';
                        IBMKeyValue = ENTERKEY;
                    }

                     if(strlen(g_Items_Available[(Item_Code-1)%5].Item_Description)>7
&& (Inactiveness1++)>10){
                        Inactiveness1=6;
                        strcpy(Item_Description,g_Items_Available[(Item_Code-
1)%5].Item_Description+1);
                        if(AnimationFlag == 0){
```

```
        sprintf(Item_Description+strlen(Item_Description),"  %c",g_Items_Available[(Item_Code-
1)%5].Item_Description[0]);
                                                AnimationFlag = 1;
                                        }else
sprintf(Item_Description+strlen(Item_Description),"%c",g_Items_Available[(Item_Code-
1)%5].Item_Description[0]);
                                        sprintf(g_Items_Available[(Item_Code-
1)%5].Item_Description,"%s",Item_Description);
                                }
                        }
                }
                LcdDisplay(BOTTOM,LcdBotBuff);
                if((*Item_Count)==0){
                        if(IBMKeyValue  == ENTERKEY)
                        {
                                if(CorrectBuff1[0]==0){
                                        IBMKeyValue = 0;
                                        ClearLcdDisplay();
                                        LcdDisplay(TOP,"Nothing  To Print");
                                        Delay_1sec(3);
                                        ClearLcdDisplay();
                                }
                        }
                        if(IBMKeyValue==F1KEY){
                                IBMKeyValue=0;
                                LcdDisplay(TOP,"* Bill Summary *");
                                LcdDisplay(BOTTOM,"                ");
                                Delay_1sec(5);
                                ShowBillSummary();
                        }
```

B) SCREENSHOTS

Figure 7.1: Input Data Example In Weight (KG)

Figure 7.2: Input Data Example In Pieces

a. Bill Generated

```
        HEADER1
        HEADER2
        HEADER3
        HEADER4
--------------------------------
08/10/2020 : 12:20:05
Bill No.    1
--------------------------------
Name          Qty  Price   Amount
--------------------------------
Tomato       4.11  38.50   158.24
RedBull         4  54.00   216.00
Semiya(1kg      5  60.00   300.00
--------------------------------
       Gross          :     674.23
       Coinage        :      -0.23
       Net Amount     :     674.00
Count : 3
Tax Amount :    30.39
You Save   :     6.16
--------------------------------
Thank You ! Visit Again !
```

Figure 7.3: Bill Generated

b. Brief Today's Bill Summary

```
        Todays Summary
--------------------------------
Date  :   08-10-2020
Total :      2866.00
Tax   :       105.02
--------------------------------
```

Figure 7.4: Brief Today's Bill Summary

c. Brief Monthly's Bill Summary

```
        Monthly Summary
--------------------------------
Month :      October
Total :      2866.00
Tax   :       105.02
--------------------------------
```

Figure 7.5: Brief Monthly's Bill Summary

d. Brief Yearly's Bill Summary

```
        Yearly Summary
--------------------------------
Year   :          2020
Total  :       2866.00
Tax    :        105.02
--------------------------------
```

Figure 7.6 : Brief Yearly's Bill Summary

e. Detailed Today's Bill Summary

```
DATE : 08-10-2020
|----------------------|
| Total :    2866.00   |
| Tax   :     105.02   |
| Bill Count :     5   |
|----------------------|

|---------------|-----------------------|---------------|
|   BillNumber  |       BillTotal       |       BillTax |
|---------------|-----------------------|---------------|
|             1 |          674.00       |         30.38 |
|---------------|-----------------------|---------------|
|             2 |          216.00       |         16.00 |
|---------------|-----------------------|---------------|
|             3 |           11.00       |          0.00 |
|---------------|-----------------------|---------------|
|             4 |          516.00       |         16.00 |
|---------------|-----------------------|---------------|
|             5 |         1449.00       |         42.64 |
|---------------|-----------------------|---------------|
```

Figure 7.7: Detailed Today's Bill Summary

f. Detailed Monthly's Bill Summary

```
Year : 2020
 |----------------------|
 | Month :     October  |
 | Total :     2866.00  |
 | Tax   :      105.02  |
 |----------------------|

|---------------|-----------------------|---------------|
|         Date  |       BillTotal       |       BillTax |
|---------------|-----------------------|---------------|
|   08-10-2020  |         2866.00       |        105.02 |
|---------------|-----------------------|---------------|
```

Figure 7.8: Detailed Monthly's Bill Summary

g. Detailed Yearly's Bill Summary

Year : 2020

Month	BillTotal	BillTax
October	2866.00	105.02
TOTAL	2866.00	105.02

Figure 7.9: Detailed Yearly's Bill Summary

h. POS_DataTransmitter PC Software

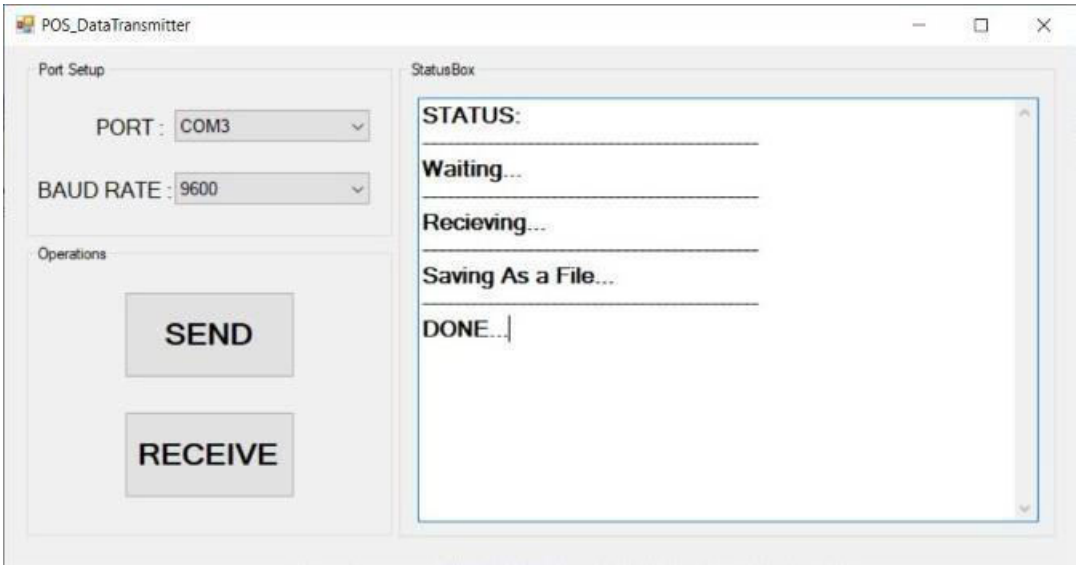


Figure 7.10: POS_DataTransmitter PC Software

i. POS Data Transmitter Android Application

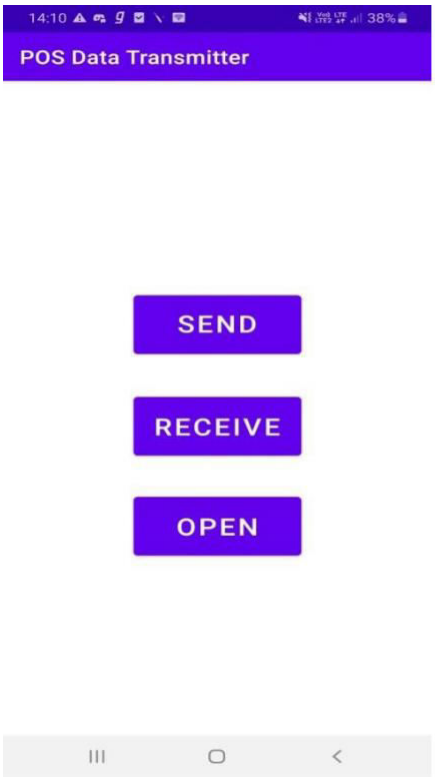


Fig 7.11 Main Page **Fig 7.12 Sending Page**

Fig 7.13 Receiving Page

C) PUBLICATION WITH PLAGIARISM REPORT

Point Of Sale System - A Billing System Infrastructure

Sivakumar G [1], Ravikumar P [2], Vimali J S [3]
[1, 2] U.G Scholar, Department of IT
[3] Assistant Professor, Department of IT
Sathyabama Institute of Science and Technology
kisary1107@gmail.com[1], raviathlet112@gmail.com[2], vimalijsmtech@gmail.com[3]

Abstract— **The paper is about the point of sale (POS) system Implemented in Embedded Systems. Point of sale is the payment scenario, when a customer performs payment to the merchant whenever a product is bought. After the payment, the merchant will issue a payment receipt or an invoice regarding the payment, these payment receipts are printed or sent electronically. To calculate the amount to be paid by the customer for what he/she purchased, the merchant uses various devices such as cash registers, weighing scales and to control all these devices a microcontroller is used in hardware. The microcontroller processes all the input data given to it and output's the processed data to the required output console. To process those data, a programmer should give a proper algorithm to the microcontroller through any programming language which is accepted by the microcontroller. Ex. C, Python etc. In this paper a proper algorithm is given to the microcontroller through C programming language, in order to solve the point of sale problem. The problem can be solved by getting data input to the microcontroller through various input devices such as weighing scales, Keyboards and output as a bill to the billing machine. This system is an integrated weighing point of sale (POS) system, such that the weighing scale measured weight for each item is directly displayed in the output screen. POS system enables quick product return processes. When a customer cancels a sale, the merchant can process it through POS system with just a couple of clicks. The stock number of an item that was reduced automatically increases again once a purchase has been cancelled. The Accounting details of a grocery store can be stored in well organized manner either it may be the tax amount, total revenue collected by the store or the detailed item wise stock report using a PC/mobile application.**

Keywords — Point of Sale, Integrated Weighing, Microcontroller, Mobile Application, Weighing Scales

I. INTRODUCTION

The point of sale (POS) is the place or time a retail transaction is performed or completed. During the point of sale, the merchant summarizes the amount owed by his/her customer, indicating that amount either through invoice or through cash register printouts and can show the options for the customers to pay those bills. It can be also called as a point where customer does payment to the merchant for any service done, any exchange of goods or things. The next step to the payment, the customer will be given with the receipt in acknowledgment to the payment customer had done. In this paper a suitable algorithm is given to the LPC1548/1549, an ARM Cortex-M3 processer to perform the task and to overcome the objective of Point of sale activity.

II. PAPER INFORMATION

A. Background of the paper

Each item has its own unique item code number and its quantity is measured in two types, one is weight and the other is pieces. Since, the memory size is limited POS System implemented in the LPC1548/1549 microcontroller can store data of 500 Items, 1000 bills per day and it's summary (max), most recent 2 months bill summary, most recent 2 years bill summary. Each Item entered for billing are stored as a backup at the spot which can handle power cut situation so that it can retrieve all previously entered items when power is back. The non-volatile memory used in this paper is the E2 Prom [9] with a proper data structure. The user is allowed to edit/delete the items that he/she entered. The weight is measured through the high-speed load cell [8] and it sends the analog data to the microcontroller, the microcontroller performs ADC operations and can determine the actual weight through calibrated values. The bill can be printed through a bill printer (i.e.) connected using UART protocol [10] with the hardware. The UART protocol of the microcontroller follows the RS232 [12] serial protocol. A provision has been given to print the brief summary of today's bill, Monthly's bill, yearly's bill. The list of items and shop name are loaded into the hardware through a PC based software (Windows) and an android application, detailed bill summary is retrieved from the hardware through the same software by storing it as a text file in the PC/ so that it can be viewed at any time. All the communications between the hardware and the pc will be serially at present through UART protocol at a baud rate [11] of 9600 at default.

B. Aim of the paper

The main aim of the paper is to describe an integrated weighing point of sale (POS) system, such that the weighing scale measured weight for each item is directly processed by the hardware to get the desired output and to make the billing process more faster [2]. The Accounting details for any store can be stored in a well organized manner, either it may be the tax amount, total revenue collected by the store or the detailed stock report.

Fig 8.1 Published paper page 1

III. Methodology

A. E2 Prom Memory Map

Fig. 1. Item Backup Memory Map

The Fig. 1 represents memory map for 50 items at maximum during billing.

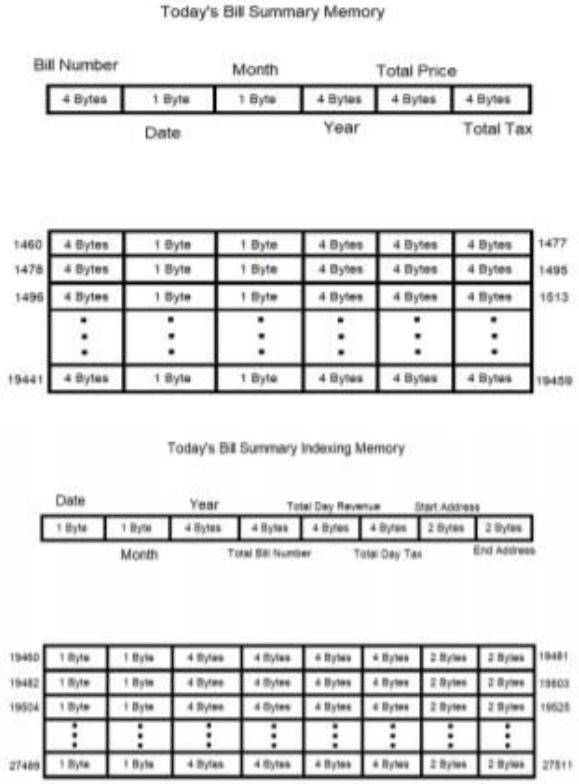


Fig. 2. Today's Bill Summary Memory Map and Index Memory Map

From Fig. 2 Indexing memory acts as an index to the Today's bill summary memory which can give a detailed bill report for a particular day. The indexing memory represents the starting and ending memory location of the bill report for that particular date. Hence, today bill summary memory can be used as a circular buffer to store daily bill report, so that memory is utilized efficiently.

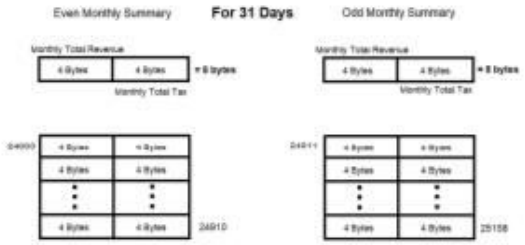


Fig. 3. Monthly's Bill Summary Memory Map

The Fig. 3 represents memory map is done for two months of each 31 days assumed.

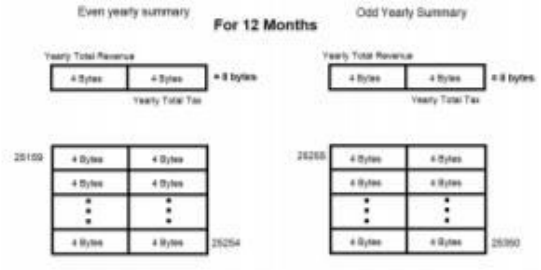


Fig. 4. Yearly's Bill Summary Memory Map

The Fig. 4 represents memory map is done for two years of each 366 days assumed.

B. Sockets

The android mobile is used to communicate with the AWEW 2000 POS indicator wirelessly via socket. The IP address and port number of the indicator is given to the android application [1][6].

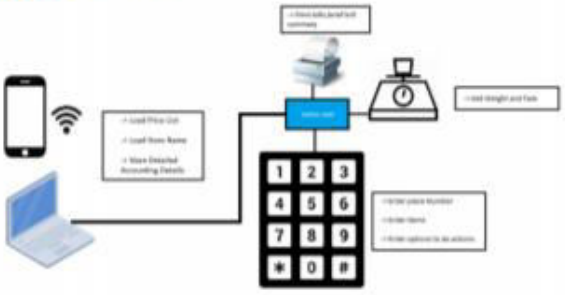


Fig. 5. Wireless Mobile Communication

Fig 8.2 Published paper page 2

The android mobile can collect,

- Today's Summary
- Date Wise Summary
- Current Monthly's Summary
- Previous Monthly's Summary
- Current Yearly's Summary
- Previous Yearly's Summary
- Price List
- Stock Report

All the above information's are collected by sending appropriate query. The information is received via socket by following UDP protocol [5].

IV. Results And Discussions

The project was tested with many real time inputs example Item code, weight through weight scale, number of pieces and got the expected output as efficient as possible.

A. Outputs And Screenshots

Fig. 6. Input Data Example in Weight (KG)

Fig. 7. Input Data Example in Pieces

Above Fig. 6. and Fig. 7. represents the indicator's output when a material is measured in weight or piece.

```
          HEADER1
          HEADER2
          HEADER3
          HEADER4
--------------------------------
08/10/2020 : 12:20:05
Bill No.    1
--------------------------------
Name          Qty  Price   Amount
--------------------------------
Tomato       4.11  38.50   158.24
RedBull         4  54.00   216.00
Semiya(1kg      5  60.00   300.00
--------------------------------
        Gross         :     674.23
        Coinage       :      -0.23
        Net Amount    :     674.00
Count : 3
Tax Amount :    30.39
You Save   :     6.16
--------------------------------
Thank You ! Visit Again !
```

Fig. 8. Bill Generated

The above Fig. 8. represents the final bill generated by 32 chararacter thermal printer.

```
        Todays Summary
--------------------------------
Date  :   08-10-2020
Total :      2866.00
Tax   :       105.02
--------------------------------
```

Fig. 9. Brief Today's Bill Summary

```
        Monthly Summary
--------------------------------
Month :       October
Total :      2866.00
Tax   :       105.02
--------------------------------
```

Fig. 10. Brief Monthly's Bill Summary

```
        Yearly Summary
--------------------------------
Year  :          2020
Total :      2866.00
Tax   :       105.02
--------------------------------
```

Fig. 11. Brief Yearly's Bill Summary

Above three figures Fig. 9, Fig. 10 and Fig. 11 is the brief summary printed by the printer.

Fig 8.3 Published paper page 3

DATE : 08-10-2020

Total :	2866.00
Tax :	105.02
Bill Count :	5

BillNumber	BillTotal	BillTax
1	674.00	30.38
2	216.00	16.00
3	11.00	0.00
4	516.00	16.00
5	1449.00	42.64

Fig. 12. Detailed Today's Bill Summary

Year : 2020

Month :	October
Total :	2866.00
Tax :	105.02

Date	BillTotal	BillTax
08-10-2020	2866.00	105.02

Fig. 13. Detailed Monthly's Bill Summary

Year : 2020

Month	BillTotal	BillTax
October	2866.00	105.02
TOTAL	2866.00	105.02

Fig. 14. Detailed Yearly's Bill Summary

Above three figures Fig. 12, Fig. 13 and Fig. 14 is the Detailed summary stored in PC/mobile via the native software application.

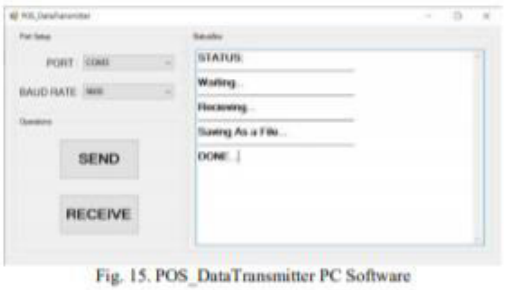


Fig. 15. POS_DataTransmitter PC Software

Fig. 15 represents the screenshot of the PC application that communicates with the indicator via serial port.

Fig. 16. POS Data Transmitter Android Application

Fig. 17. Android App Send Page

Fig 8.4 Published paper page 4

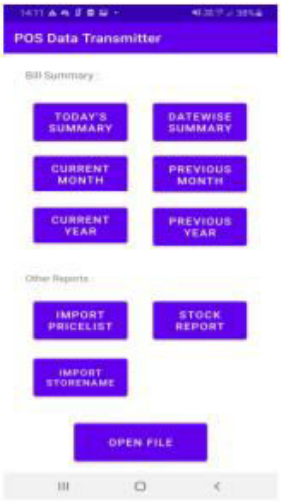


Fig. 18. Android app Receiving Page

The above three figures Fig. 16, Fig. 17, Fig. 18 represents the screenshot of the android application that communicates with the indicator via socket.

V. Conclusion

In this paper, the POS system implemented on the microcontroller LPC1548/1549 and making compatible with the awew 2000 device is explained with atmost perfection, as much as possible. This kind of Integrated weight POS system makes the usablity [3][4] more faster by making an user to stick their eyes directly to the product and the device rather than always seeing the weighing scale each and every time when a product is loaded. This POS system can print reports and, in many cases, import data directly with suitable PC/Android software which makes the user to well structure their accounting details.

Acknowledgement

The paper would have not been completed without the help of the organization LCS Controls PVT. LTD., Kottivakkam, Chennai. Thanks for the organization for providing the indicator, load cell and many other connectors. In regards I would like to give and dedicate this whole project to the organization.

References

[1] Sai, K, An Analysis of Point of Sale Systems Physical Configurations and Security Measures in Zimbabwean SMEs.,International Journal of Education & Multidisciplinary Studies, Vol 6, issue 2, 2017, pp. 181 – 190.

[2] Yomayra Ramos, Dr. Angel Ojeda Castro, Point-Of-Sales Systems in Food and Beverage Industry: Efficient Technology and Its User Acceptance, Journal of Information Sciences and Computing Technologies, Vol 6, issue 1, 2017, pp. 582 – 591.

[3] Ali Sajedi, Mehregan Mahdavi, Amir pour shir mohammadi, Minoo Monajjemi nejad, Fundamental Usability Guidelines for User Interface Design, IEEE, 2008, pp. 106 – 113.

[4] Ugochi Oluwatosin Nwokedi, Beverly Amunga Onyimbo and Babak Bashari Rad, Usability and Security in User Interface Design, I.J. Information Technology and Computer Science, 2016, pp. 72-80.

[5] Ming Xue, Changjun Zhu, The Socket Programming and Software Design for Communication Based on Client/Server, IEEE, 2009, Pacific-Asia Conference on Circuits,Communications and System, pp. 775-777.

[6] Rolou Lyn Rodriguez Maata, Ronald Soriano Cordova, Balaji Sudramurthy, Alrence Halibas, Design and Implementation of Client-Server Based Application Using Socket Programming in a Distributed Computing Environment, IEEE International Conference on Computational Intelligence and Computing Research, 2017, pp. 1- 6

[7] Gowri, S. and Divva, G., Februarv. Automation of garden tools monitored using mobile application. IEEE International Confernce on Innovation Information in Computing Technologies, 2015, pp. 1-6.

[8] Andrew Gilman, Donald G. Bailey, High-speed Weighing Using Impact on Load Cells, IEEE International Confernce on Innovation Information in Computing Technologies, 2005, pp. 1 – 6.

[9] Bruce Euzent, Nick Boruta, Jimmy Lee, Ching JenqS, Reliability Aspects of a Floating gate E2 Prom, IEEE International Electron Devices Meeting, 1984, pp. 11 – 16.

[10] Umakanta Nanda, Sushant Kumar Pattnaik, Universal Asynchronous Receiver and Transmitter (UART), IEEE 3rd International Conference on Advanced Computing and Communication Systems, 2016, pp. 1 – 5.

[11] Xingchun Liu, ,Yandan Liu, Multi-functional Serial Communication Interface Design Based on FPGA, IEEE 3rd International Conference on Computer and Communications, 2017, pp. 758 – 761.

[12] Xinghai Han, Xiangxin Kong, The Designing of Serial Communication Based on RS232, IEEE First ACIS International Symposium on Cryptography, and Network Security, Data Mining and Knowledge Discovery, E-Commerce and Its Applications, and Embedded Systems, 2010, pp. 382 – 384.

Fig 8.5 Published paper page 5

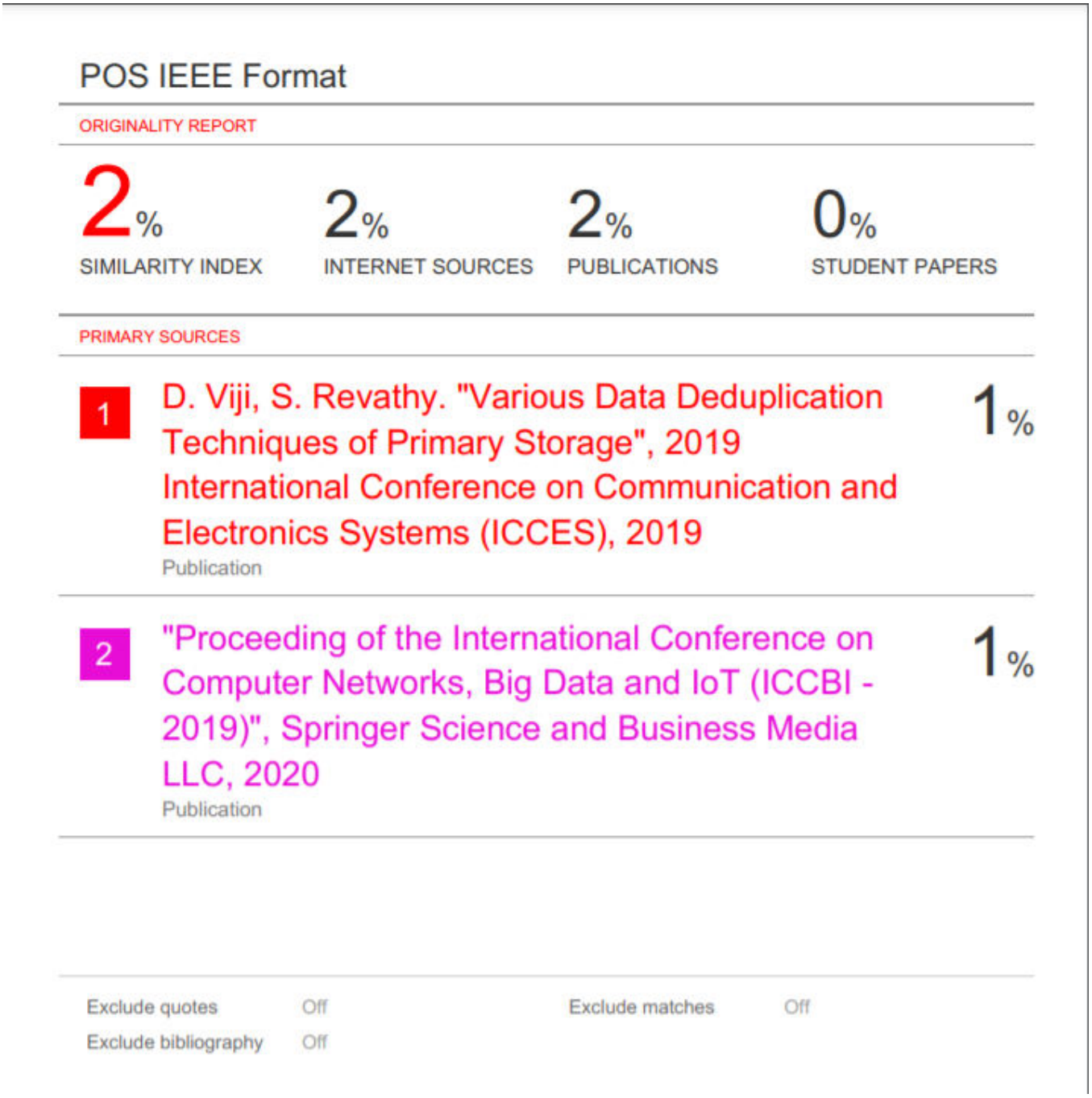
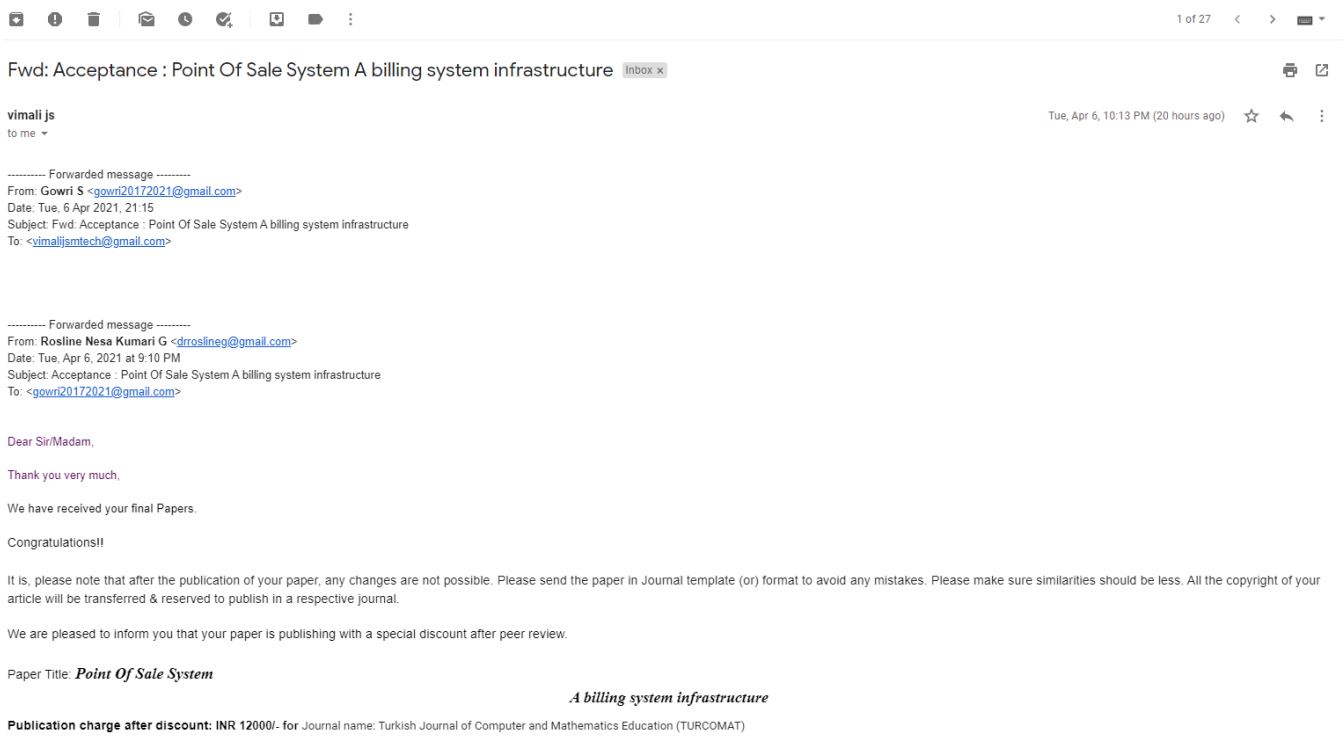


Fig 8.6 Plagiarism paper



1 of 27

Fwd: Acceptance : Point Of Sale System A billing system infrastructure Inbox ×

vimali js
to me

Tue, Apr 6, 10:13 PM (20 hours ago)

---------- Forwarded message ---------
From: **Gowri S** <gowri20172021@gmail.com>
Date: Tue, 6 Apr 2021, 21:15
Subject: Fwd: Acceptance : Point Of Sale System A billing system infrastructure
To: <vimalijsmtech@gmail.com>

---------- Forwarded message ---------
From: **Rosline Nesa Kumari G** <drroslineg@gmail.com>
Date: Tue, Apr 6, 2021 at 9:10 PM
Subject: Acceptance : Point Of Sale System A billing system infrastructure
To: <gowri20172021@gmail.com>

Dear Sir/Madam,

Thank you very much,

We have received your final Papers.

Congratulations!!

It is, please note that after the publication of your paper, any changes are not possible. Please send the paper in Journal template (or) format to avoid any mistakes. Please make sure similarities should be less. All the copyright of your article will be transferred & reserved to publish in a respective journal.

We are pleased to inform you that your paper is publishing with a special discount after peer review.

Paper Title: ***Point Of Sale System***

A billing system infrastructure

Publication charge after discount: INR 12000/- for Journal name: Turkish Journal of Computer and Mathematics Education (TURCOMAT)

Fig 8.7 Paper Acceptance