VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

POINT OF SALES SERVICE SYSTEM USING EMU8086

CSE2006 – Microprocessor and Interfacing

J COMPONENT PROJECT DOCUMENT

FALL 2022 - 2023

Team Members

1. **20BCE0842 ASHWIN SHUKLA**
2. **20BCE2066 KSHITIJ BHOSALE**
3. **20BCE2523 PARTH SETIA**
4. **20BCE2763 BIGGYAT PANDEY**

UNDER THE GUIDENCE OF

PROF. NARESH K

SCOPE

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

DECLARATION

We hereby declare that the J Component project entitled **“ Point Of Sales Service System Using EMU8086** ” submitted by us to the School of Computer Science and Engineering, VIT University, Vellore-14 in partial fulfillment of the requirements for the Microprocessor and Interfacing (CSE2006) course, is a record of bonafide work carried out by us under the supervision of **Prof**. **Naresh K, Associate Professor (Grade 2).** We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other course or degree or diploma of this institute or of any other institute or university

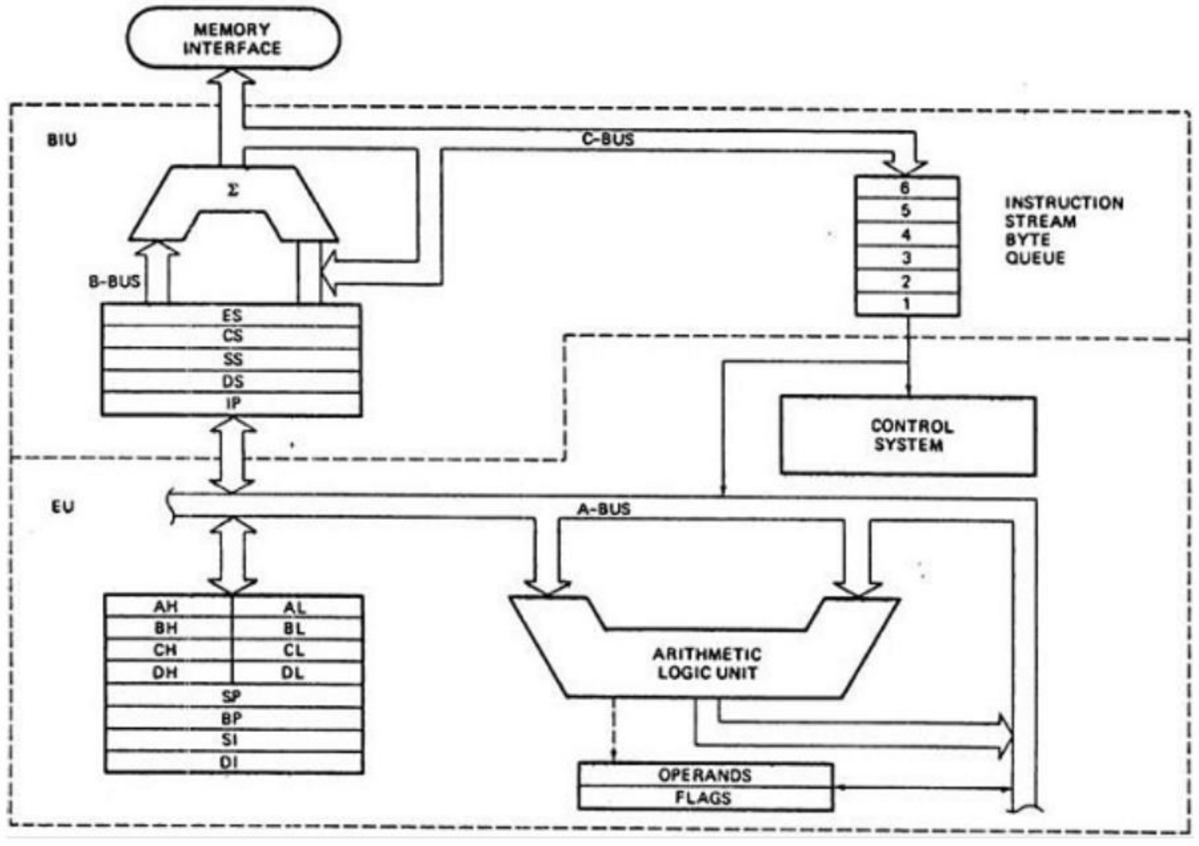
TABLE OF CONTENT

3.1 Arduino UNO 8086 Processor

8086 Microprocessor is an enhanced version of 8085Microprocessor that was designed by Intel in 1976. It is a 16-bit Microprocessor having 20 address lines and16 data lines that provides up to 1MB storage. It consists of powerful instruction set, which provides operations like multiplication and division easily.

It supports two modes of operation, i.e. Maximum mode and Minimum mode. Maximum mode is suitable for system having multiple processors and Minimum mode is suitable for system having a single processor.

Architecture:



8086 Microprocessor is divided into two functional units, i.e., EU (Execution Unit) and BIU (Bus Interface Unit).

3.1.1 EU (Execution Unit)

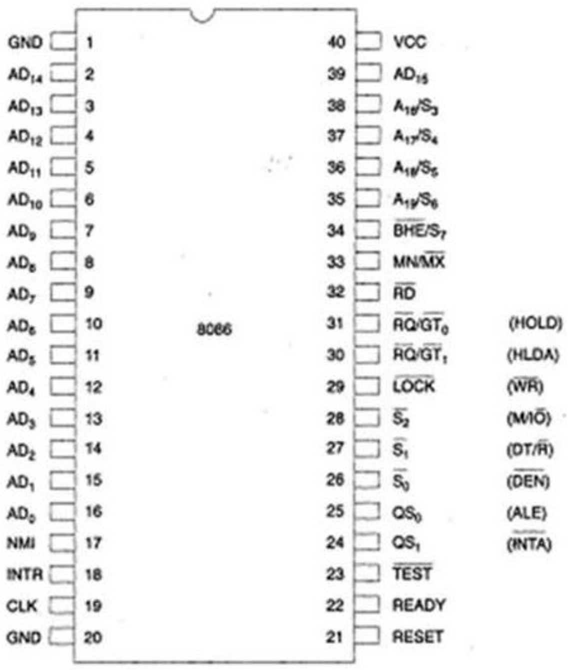
Execution unit gives instructions to BIU stating from where to fetch the data and then decode and execute those instructions. Its function is to control operations on data using the instruction decoder & ALU. EU has no direct connection with system buses as shown in the above figure, it performs operations over data through BIU.

3.1.2. BIU (Bus Interface Unit)

BIU takes care of all data and addresses transfers on the buses for the EU like sending addresses, fetching instructions from the memory, reading data from the ports and the memory as well as writing data to the ports and the memory. EU has no direction connection with System Buses so this is possible with the BIU. EU and BIU are connected with the Internal Bus.

3.2 Pin Diagram:

Here is the pin diagram of 8086 microprocessor –



Let us now discuss the signals in detail −

1. **Power supply and frequency signals:** It uses 5V DC supply at V_{CC} pin 40, and uses ground at V_{SS} pin 1 and 20 for its operation.
2. **Clock signal:** Clock signal is provided through Pin-19. It provides timing to the processor for operations. Its frequency is different for different versions, i.e. 5MHz, 8MHz and 10MHz.
3. **Address/data bus:** AD0-AD15. These are 16 address/data bus. AD0-AD7 carries low order byte data and AD8AD15 carries higher order byte data. During the first clock cycle, it carries 16-bit address and after that it carries 16-bit data.
4. **Address/status bus:** A16-A19/S3-S6. These are the 4 address/status buses. During the first clock cycle, it carries 4-bit address and later it carries status signals.
5. **S7/BHE:** BHE stands for Bus High Enable. It is available at pin 34 and used to indicate the transfer of data using data bus D8-D15. This signal is low during the first clock cycle, thereafter it is active.
6. **READ:** It is available at pin 32 and is used to read signal for Read operation.
7. **READY:** It is available at pin 22. It is an acknowledgement signal from I/O devices that data is transferred. It is an active high signal. When it is high, it indicates that the device is ready to transfer data. When it is low, it indicates wait state.

8. **RESET:** It is available at pin 21 and is used to restart the execution. It causes the processor to immediately terminate its present activity. This signal is active high for the first 4 clock cycles to RESET the microprocessor.
9. **INTR:** It is available at pin 18. It is an interrupt request signal, which is sampled during the last clock cycle of each instruction to determine if the processor considered this as an interrupt or not.
10. **NMI:** It stands for non-maskable interrupt and is available at pin 17. It is an edge triggered input, which causes an interrupt request to the microprocessor.
11. **:** This signal is like wait state and is available at pin 23. When this signal is high, then the processor has to wait for IDLE state, else the execution continues.
12. **MN/MX:** It stands for Minimum/Maximum and is available at pin 33. It indicates what mode the processor is to operate in; when it is high, it works in the minimum mode and vice-aversa.
13. **INTA:** It is an interrupt acknowledgement signal and id available at pin 24. When the microprocessor receives this signal, it acknowledges the interrupt.
14. **ALE:** It stands for address enable latch and is available at pin 25. A positive pulse is generated each time the processor begins any operation. This signal indicates the availability of a valid address on the address/data lines.
15. **DEN:** It stands for Data Enable and is available at pin 26. It is used to enable Transreceiver 8286. The transreceiver is a device used to separate data from the address/data bus.
16. **DT/R:** It stands for Data Transmit/Receive signal and is available at pin 27. It decides the direction of data flow through the transreceiver. When it is high, data is transmitted out and vice-a-versa.
17. **M/IO:** This signal is used to distinguish between memory and I/O operations. When it is high, it indicates I/O operation and when it is low indicates the memory operation. It is available at pin 28.
18. **WR:** It stands for write signal and is available at pin 29. It is used to write the data into the memory or the output device depending on the status of M/IO signal.
19. **HLDA:** It stands for Hold Acknowledgement signal and is available at pin 30. This signal acknowledges the HOLD signal.
20. **HOLD:** This signal indicates to the processor that external devices are requesting to access the address/data buses. It is available at pin 31.

21. **QS_1 and QS_0** : These are queue status signals and are available at pin 24 and 25. These signals provide the status of instruction queue. Their conditions are shown in the following table −

QS_0	QS_1	Status
0	0	No operation
0	1	First byte of opcode from the queue
1	0	Empty the queue
1	1	Subsequent byte from the queue

22. **S_0, S_1, S_2:** These are the status signals that provide the status of operation, which is used by the Bus Controller 8288 to generate memory & I/O control signals. These are available at pin 26, 27, and 28. Following is the table showing their status −

S_2	S_1	S_0	Status
0	0	0	Interrupt acknowledgement
0	0	1	I/O Read
0	1	0	I/O Write
0	1	1	Halt
1	0	0	Opcode fetch
1	0	1	Memory read
1	1	0	Memory write
1	1	1	Passive

23. **LOCK:** When this signal is active, it indicates to the other processors not to ask the CPU to leave the system bus. It is activated using the LOCK prefix on any instruction and is available at pin 29.
24. **RQ/GT$_1$ and RQ/GT$_0$:** These are the Request/Grant signals used by the other processors requesting the CPU to release the system bus. When the signal is received by CPU, then it sends acknowledgment. RQ/GT$_0$ has a higher priority than RQ/GT$_1$.

3.3. INSTRUCTION SET

The 8086 microprocessor supports 8 types of instructions −

- Data Transfer Instructions
- Arithmetic Instructions
- Bit Manipulation Instructions
- String Instructions
- Program Execution Transfer Instructions (Branch & Loop Instructions)
- Processor Control Instructions
- Iteration Control Instructions
- Interrupt Instructions

Let us now discuss these instruction sets in detail.

3.3.1. Data Transfer Instructions

These instructions are used to transfer the data from the source operand to the destination operand. Following are the list of instructions under this group −

Instruction to transfer a word

- MOV − Used to copy the byte or word from the provided source to the provided destination.
- PPUSH − Used to put a word at the top of the stack.
- POP − Used to get a word from the top of the stack to the provided location.
- PUSHA − Used to put all the registers into the stack.
- POPA − Used to get words from the stack to all registers.
- XCHG − Used to exchange the data from two locations.
- XLAT − Used to translate a byte in AL using a table in the memory.

Instructions for input and output port transfer

- IN − Used to read a byte or word from the provided port to the accumulator.
- OUT − Used to send out a byte or word from the accumulator to the provided port.

3.3.2 Arithmetic Instructions

These instructions are used to perform arithmetic operations like addition, subtraction, multiplication, division, etc.

Following is the list of instructions under this group −

Instructions to perform addition

- ADD − Used to add the provided byte to byte/word to word.
- ADC − Used to add with carry.
- INC − Used to increment the provided byte/word by 1.
- AAA − Used to adjust ASCII after addition.
- DAA − Used to adjust the decimal after the addition/subtraction operation.

Instructions to perform subtraction

- SUB − Used to subtract the byte from byte/word from word.
- SBB − Used to perform subtraction with borrow.
- DEC − Used to decrement the provided byte/word by 1.
- NPG − Used to negate each bit of the provided byte/word and add 1/2’s complement.
- CMP − Used to compare 2 provided byte/word.
- AAS − Used to adjust ASCII codes after subtraction.
- DAS − Used to adjust decimal after subtraction.

Instruction to perform multiplication

- MUL − Used to multiply unsigned byte by byte/word by word.
- IMUL − Used to multiply signed byte by byte/word by word.
- AAM − Used to adjust ASCII codes after multiplication.

Instructions to perform division

- DIV − Used to divide the unsigned word by byte or unsigned double word by word.
- IDIV − Used to divide the signed word by byte or signed double word by word.
- AAD − Used to adjust ASCII codes after division.

- CBW − Used to fill the upper byte of the word with the copies of sign bit of the lower byte.
- CWD − Used to fill the upper word of the double word with the sign bit of the lower word.

3.3.3. Bit Manipulation Instructions

These instructions are used to perform operations where data bits are involved, i.e. operations like logical, shift, etc.

Following is the list of instructions under this group −

Instructions to perform logical operation

- NOT − Used to invert each bit of a byte or word.
- AND − Used for adding each bit in a byte/word with the corresponding bit in another byte/word.
- OR − Used to multiply each bit in a byte/word with the corresponding bit in another byte/word.
- XOR − Used to perform Exclusive-OR operation over each bit in a byte/word with the corresponding bit in another byte/word.
- TEST − Used to add operands to update flags, without affecting operands.

Instructions to perform shift operations

- SHL/SAL − Used to shift bits of a byte/word towards left and put zero(S) in LSBs.
- SHR − Used to shift bits of a byte/word towards the right and put zero(S) in MSBs.
- SAR − Used to shift bits of a byte/word towards the right and copy the old MSB into the new MSB.

Instructions to perform rotate operations

- ROL − Used to rotate bits of byte/word towards the left, i.e. MSB to LSB and to Carry Flag [CF].
- ROR − Used to rotate bits of byte/word towards the right, i.e. LSB to MSB and to Carry Flag [CF].
- RCR − Used to rotate bits of byte/word towards the right, i.e. LSB to CF and CF to MSB.

- RCL − Used to rotate bits of byte/word towards the left, i.e. MSB to CF and CF to LSB.

3.3.4. String Instructions

String is a group of bytes/words and their memory is always allocated in a sequential order.

Following is the list of instructions under this group

- REP − Used to repeat the given instruction till CX ≠ 0.
- REPE/REPZ − Used to repeat the given instruction until CX = 0 or zero flag ZF = 1.
- REPNE/REPNZ − Used to repeat the given instruction until CX = 0 or zero flag ZF = 1.
- MOVS/MOVSB/MOVSW − Used to move the byte/word from one string to another.

3.3.5. Program Execution Transfer Instructions (Branch and Loop Instructions)

These instructions are used to transfer/branch the instructions during an execution. It includes the following instructions −

Instructions to transfer the instruction during an execution without any condition −

- CALL − Used to call a procedure and save their return address to the stack.
- RET − Used to return from the procedure to the main program.
- JMP − Used to jump to the provided address to proceed to the next instruction.

Instructions to transfer the instruction during an execution with some conditions −

- JA/JNBE − Used to jump if above/not below/equal instruction satisfies.
- JAE/JNB − Used to jump if above/not below instruction satisfies.
- JC − Used to jump if carry flag CF = 1
- JE/JZ − Used to jump if equal/zero flag ZF = 1
- JNC − Used to jump if no carry flag (CF = 0)
- JNE/JNZ − Used to jump if not equal/zero flag ZF = 0
- JNO − Used to jump if no overflow flag OF = 0

- JNP/JPO − Used to jump if not parity/parity odd PF = 0
- JNS − Used to jump if not sign SF = 0
- JO − Used to jump if overflow flag OF = 1
- JP/JPE − Used to jump if parity/parity even PF = 1
- JS − Used to jump if sign flag SF = 1

3.3.6. Processor Control Instructions

These instructions are used to control the processor action by setting/resetting the flag values.

Following are the instructions under this group −

- STC − Used to set carry flag CF to 1
- CLC − Used to clear/reset carry flag CF to 0
- CMC − Used to put complement at the state of carry flag CF.
- STD − Used to set the direction flag DF to 1
- CLD − Used to clear/reset the direction flag DF to 0
- STI − Used to set the interrupt enable flag to 1, i.e., enable INTR input.
- CLI − Used to clear the interrupt enable flag to 0, i.e., disable INTR input.

3.3.7. Iteration Control Instructions

These instructions are used to execute the given instructions for number of times. Following is the list of instructions under this group −

- LOOP − Used to loop a group of instructions until the condition satisfies, i.e., CX = 0
- LOOPE/LOOPZ − Used to loop a group of instructions till it satisfies ZF = 1 & CX = 0
- LOOPNE/LOOPNZ − Used to loop a group of instructions till it satisfies ZF = 0 & CX = 0

3.3.8. Interrupt Instructions

These instructions are used to call the interrupt during program execution.

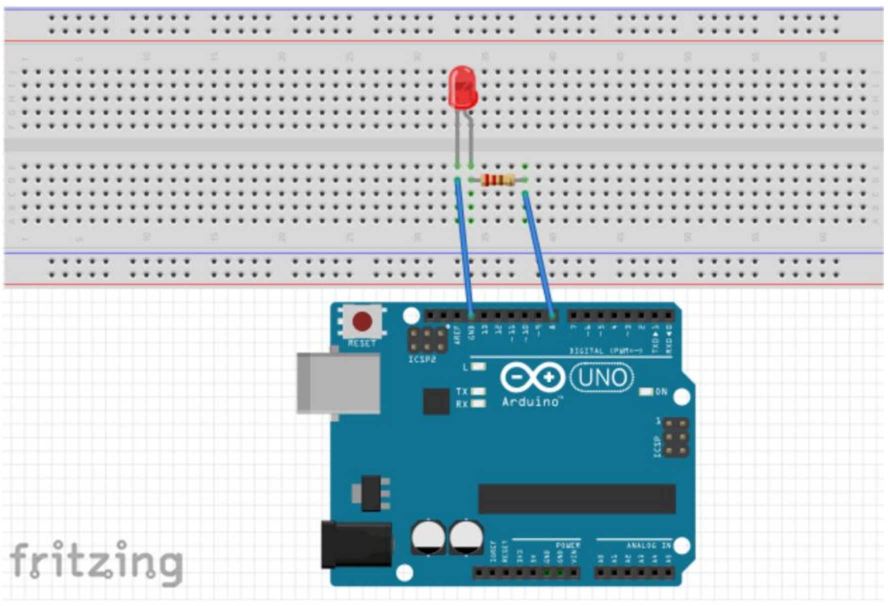
- INT − Used to interrupt the program during execution and calling service specified.
- INTO − Used to interrupt the program during execution if OF = 1

4. DEMO : Blinking LED

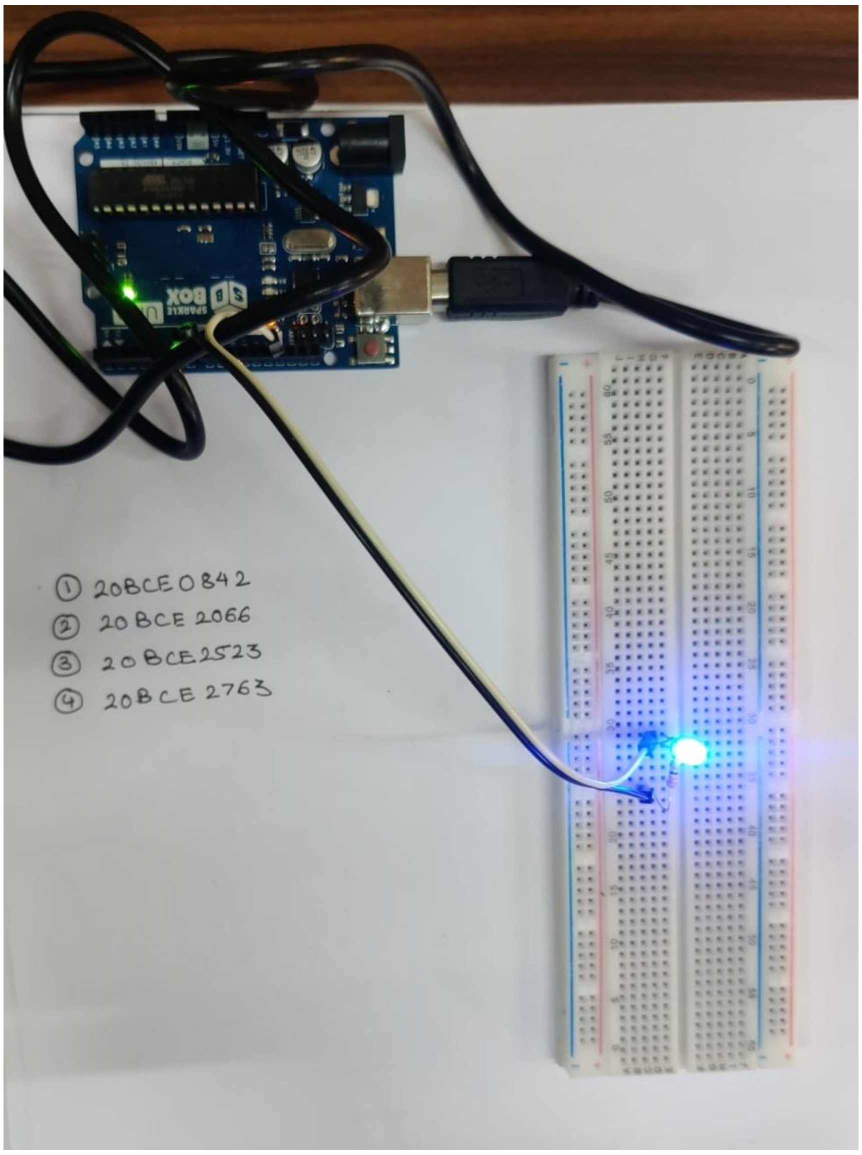
4.1. Sketch:

```
int ledPin=7;
void setup()
{
    pinMode(ledPin,OUTPUT);
}
void loop()
{
    digitalWrite(ledPin,HIGH);
    delay(1000);
    digitalWrite(ledPin,LOW);
    delay(1000);
}
```

4.2. Diagram:



4.3. Output



5. PROJECT PROTOTYPE

5.1. Problem Statement:

Point of Sales System is heavily used in food chain industry and inventory management industry and the cashiers are deeply dependent on it for daily calculations and exchanges with the users or clients .

The existing system do not comply with modern needs of logistics and stock management and thus we indemnified this as a problem and developed our project in that domain . In our project various modern functionalities have been included which makes it more novel and dynamic than the ones existing in the market .

5.2. Objective:

Point Of Sales Service System is made for Inventory Management Industry and is essential for running such business as they help the cashiers to rapidly calculate complex calculations and prevent long queues.

• It also serves to produce statistical reports through the daily sales report , with information such as the total sales if each product in order to aid to accounting.

• Also to make the better decisions in ensuring that the sales grow and products in inventory is consistent .

5.3. Related Work / Literature Review :

1.) Teaching Of 8088/86 Programming With 8086 Assembly Emulator

• April 2020 • Conference: 2nd Joint International Conference on Science, Technology and Innovation, Mandalay (IcSTIM 2019)At: Mandalay Technological University, Mandalay,Myanmar

ABSTRACT : It describes teaching of 8088/86 programming in microprocessor and interfacing course with the aid of 8086 assembly emulator in this paper. Assembly programming of integer instructions, computations, interrupt, control flow and program structures for 8088/86 have been illustrated with typical simple programs and simulation results.

2.)An 8-Bit Scientific Calculator Based Intel 8086 Virtual Machine Emulator

• October 2013 Procedia Computer Science 21 DOI: 10.1016/j.procs.2013.09.070

• Conference: The 4th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN-2013)At: CanadaVolume: 21 • Project: An Efficient Design & Implementation of RSA Cryptoprocessor

ABSTRACT : The calculator were designed over the virtual machine for Intel 8086 microprocessor using EMU8086 emulator software. Several arithmetic and logic operations as well as trigonometric functions were implemented in this paper. Also, a plot function and integration of function tools are to be implemented and added as a separate modules for this design.

3.)An Analysis of Point of Sale Systems

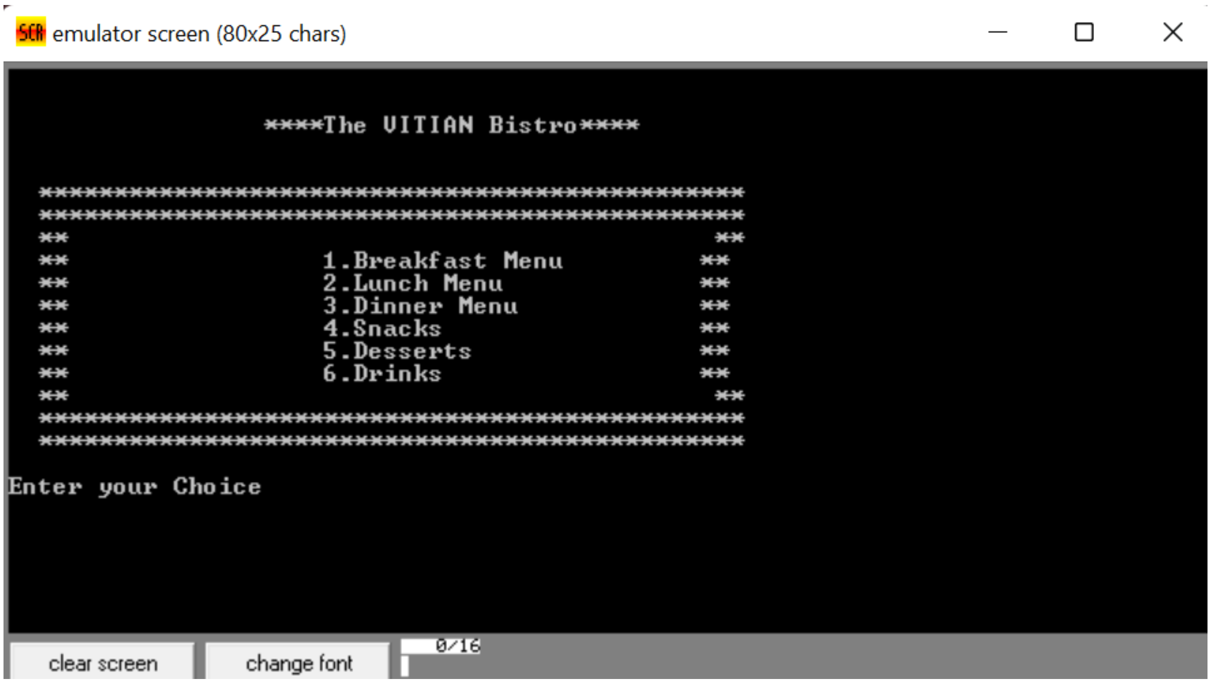
• March 2017IRA International Journal of Education and Multidisciplinary Studies 6(2):181 DOI: 10.21013/jems.v6.n2.p5 **ABSTRACT** : This paper undertook to establish the correlation between the Point of Sale configurations (setup) employed by Small and Medium Enterprises and its existing limitations to be handled if it were to be used for modern requirements.

4.) The Power of Point of Sale: Improving Growth, Profit, and Customer Service in a Retail Business BY Matthew Cote

• October 2018 DOI: 10.26488/IEJ.11.10.1144 • Project: Study of Effectiveness of POS Data in Managing Supply Chain

ABSTRACT - For many small businesses, creating a captivating retail experience is the key to success, and finding the right technologies to enable that experience is crucial for sustaining a competitive advantage. This project is a case study designed to evaluate and select a Point of Sale (POS) system and Inventory Management (IM) system for a small business based upon its specific industry needs

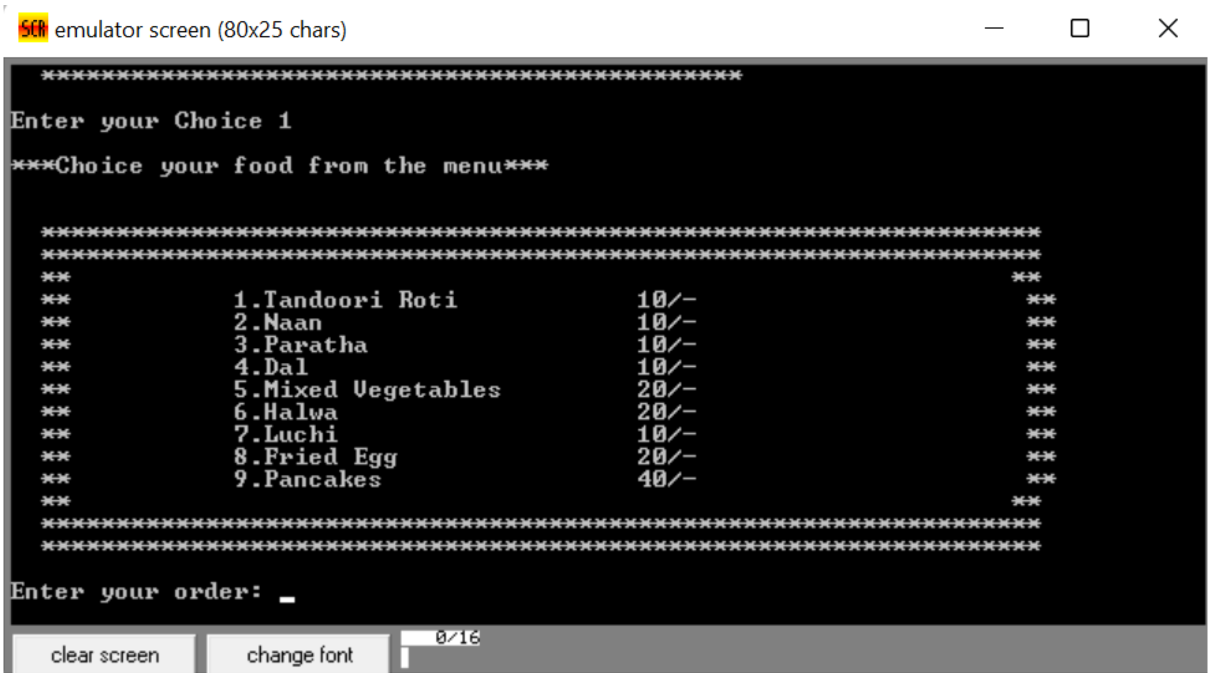
Result/Model Screenshots



```
                 ****The VITIAN Bistro****

  ***********************************************
  ***********************************************
  **                                            **
  **                1.Breakfast Menu           **
  **                2.Lunch Menu               **
  **                3.Dinner Menu              **
  **                4.Snacks                   **
  **                5.Desserts                 **
  **                6.Drinks                   **
  **                                            **
  ***********************************************
  ***********************************************

Enter your Choice
```



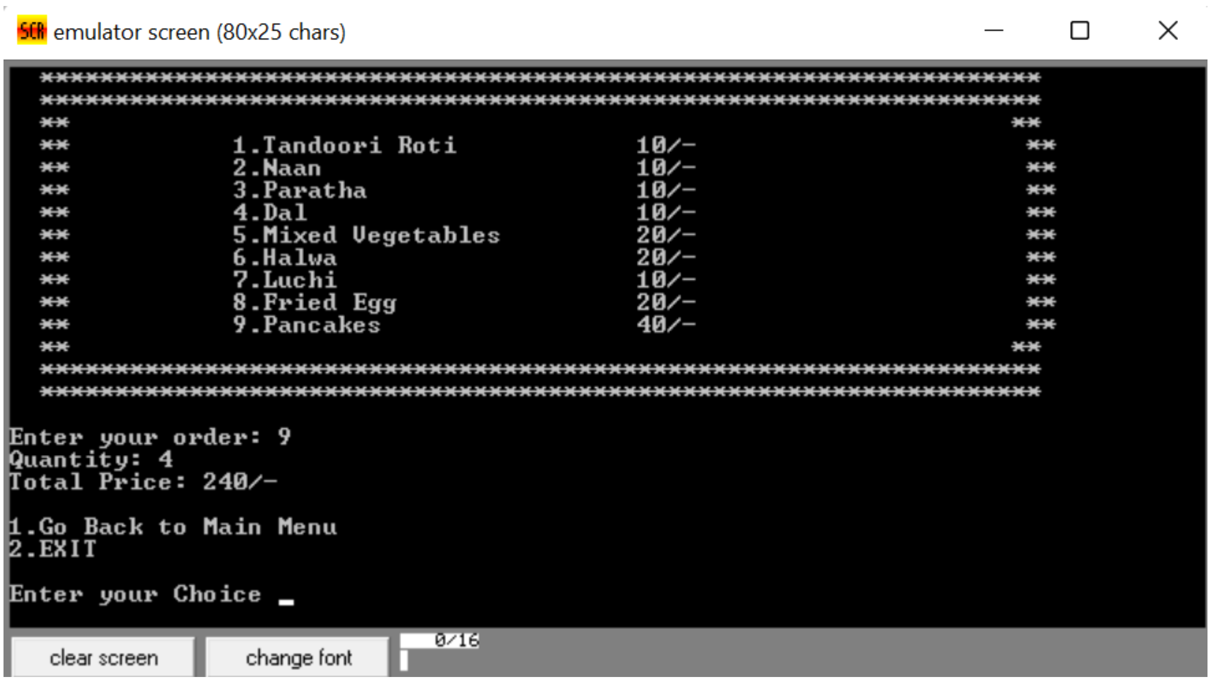
```
  ***********************************************

Enter your Choice 1

***Choice your food from the menu***


  *******************************************************************
  *******************************************************************
  **                                                               **
  **            1.Tandoori Roti              10/-                   **
  **            2.Naan                       10/-                   **
  **            3.Paratha                    10/-                   **
  **            4.Dal                        10/-                   **
  **            5.Mixed Vegetables           20/-                   **
  **            6.Halwa                      20/-                   **
  **            7.Luchi                      10/-                   **
  **            8.Fried Egg                  20/-                   **
  **            9.Pancakes                   40/-                   **
  **                                                               **
  *******************************************************************
  *******************************************************************

Enter your order: _
```

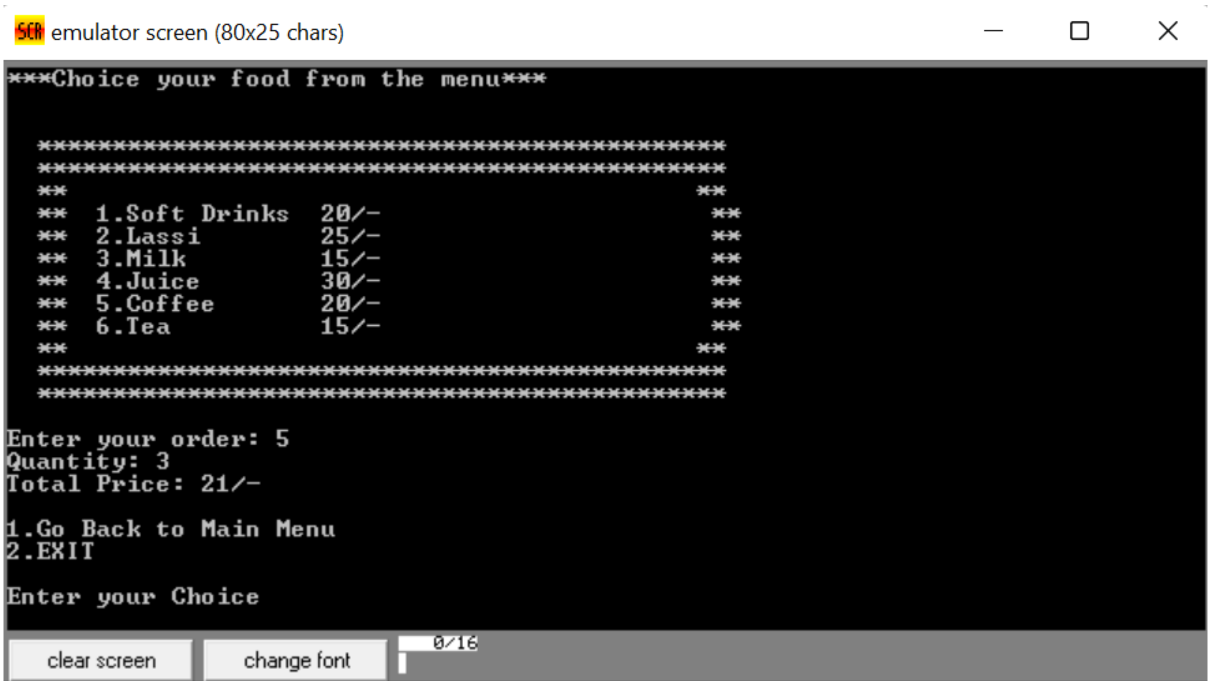


```
  *******************************************************************
  *******************************************************************
  **                                                               **
  **           1.Tandoori Roti            10/-                      **
  **           2.Naan                     10/-                      **
  **           3.Paratha                  10/-                      **
  **           4.Dal                      10/-                      **
  **           5.Mixed Vegetables         20/-                      **
  **           6.Halwa                    20/-                      **
  **           7.Luchi                    10/-                      **
  **           8.Fried Egg                20/-                      **
  **           9.Pancakes                 40/-                      **
  **                                                               **
  *******************************************************************
  *******************************************************************

Enter your order: 9
Quantity: 4
Total Price: 240/-

1.Go Back to Main Menu
2.EXIT

Enter your Choice _
```



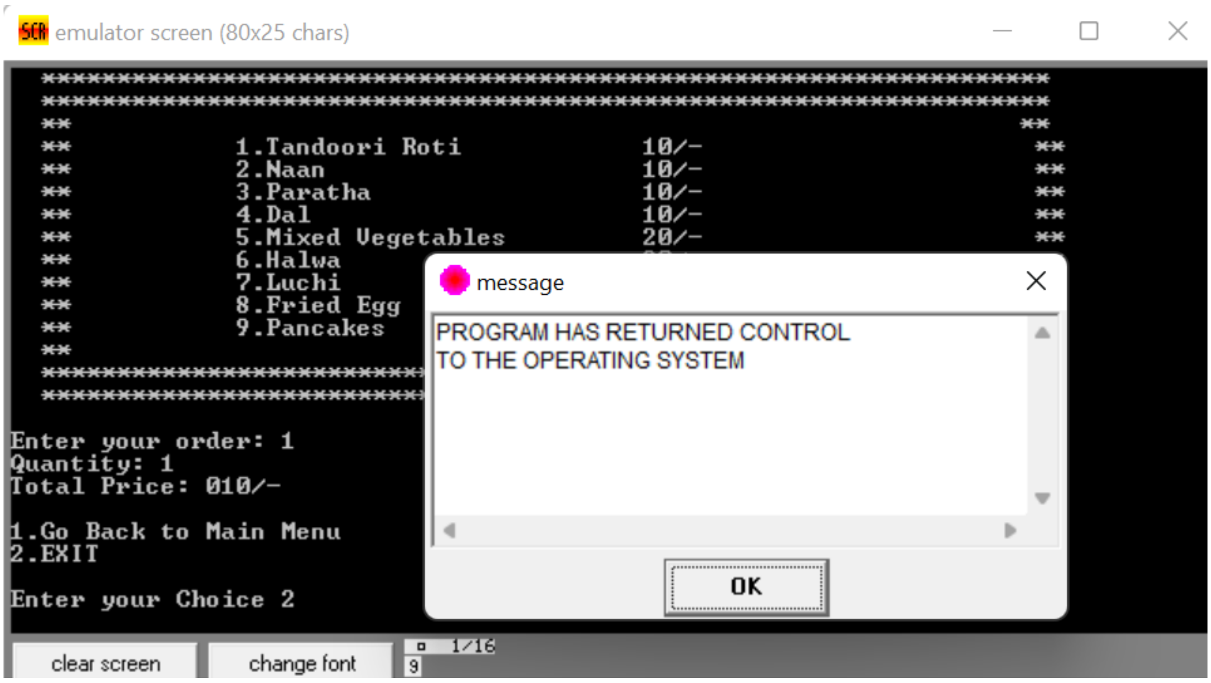
```
***Choice your food from the menu***


  **********************************************
  **********************************************
  **                                          **
  **  1.Soft Drinks  20/-                      **
  **  2.Lassi        25/-                      **
  **  3.Milk         15/-                      **
  **  4.Juice        30/-                      **
  **  5.Coffee       20/-                      **
  **  6.Tea          15/-                      **
  **                                          **
  **********************************************
  **********************************************

Enter your order: 5
Quantity: 3
Total Price: 21/-

1.Go Back to Main Menu
2.EXIT

Enter your Choice
```



```
emulator screen (80x25 chars)
    ******************************************************************
    ******************************************************************
    **                                                              **
    **          1.Tandoori Roti            10/-                     **
    **          2.Naan                     10/-                     **
    **          3.Paratha                  10/-                     **
    **          4.Dal                      10/-                     **
    **          5.Mixed Vegetables         20/-                     **
    **          6.Halwa
    **          7.Luchi
    **          8.Fried Egg
    **          9.Pancakes
    **
    *************************
    *************************

Enter your order: 1
Quantity: 1
Total Price: 010/-

1.Go Back to Main Menu
2.EXIT

Enter your Choice 2
```

```
message
PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM
OK
```

Conclusion:

Development of successful software as Point of Sales System to be used in business to increase productivity and decrease long queues , further to increase dependency on machine rather than humans.