VOTING MACHINE

Abstract

The project is aimed to make an unsupervised Electronic Voting Machine, with no controls required from the polling officer. It will be used to cast votes for multiple elections taking place simultaneously. Each valid voter is provided with a unique voter ID and a pass code by the election committee to be kept securely, which is used for the voter identification and validation during vote casting. There is no restriction on the number of candidates standing for a particular election. It will tally the votes and also have the facility of recording of actual votes using real-time clock for time stamping of the votes, which will only be read accessible.

1. Introduction

We are fascinated by the fact that the World's Largest Democracy implements elections electronically, while on the contrary it is discouraged by the western countries even now. When we did a comprehensive we came around following facts which are not a part of Indian EVM's driving as motivation for us to implement them in our design.

1. Election Transparency: all the processes of handling and counting ballots to be completely open to public view. Nothing to be hidden or secret – except, of course, each individual's voting choices.
2. Trust of the voter in the EVM registering the vote : We plan to implement voter verifiable paper record (vvpr) to which voter can look up instantly and decide on whether the vote he casted is registered correctly in the system increasing the reliability of the system.

We have made a module which interfaces with the voter and directs to the procedure of voting. The Liquid Crystal Display unit provides the voter friendly interface guiding through the procedure of voting. The keypad is used to enter the details and other actions to be taken by the voter, which are integrated with the display unit. We learnt a lot many aspects of design and especially SD card interfacing is the most interesting part. Also the product can be used for several types of elections not only just the State or Assembly Elections, discussed in later sections.

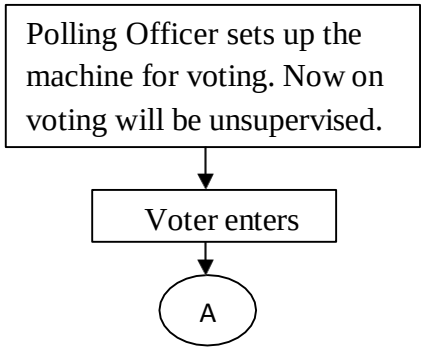
2. Design Approach

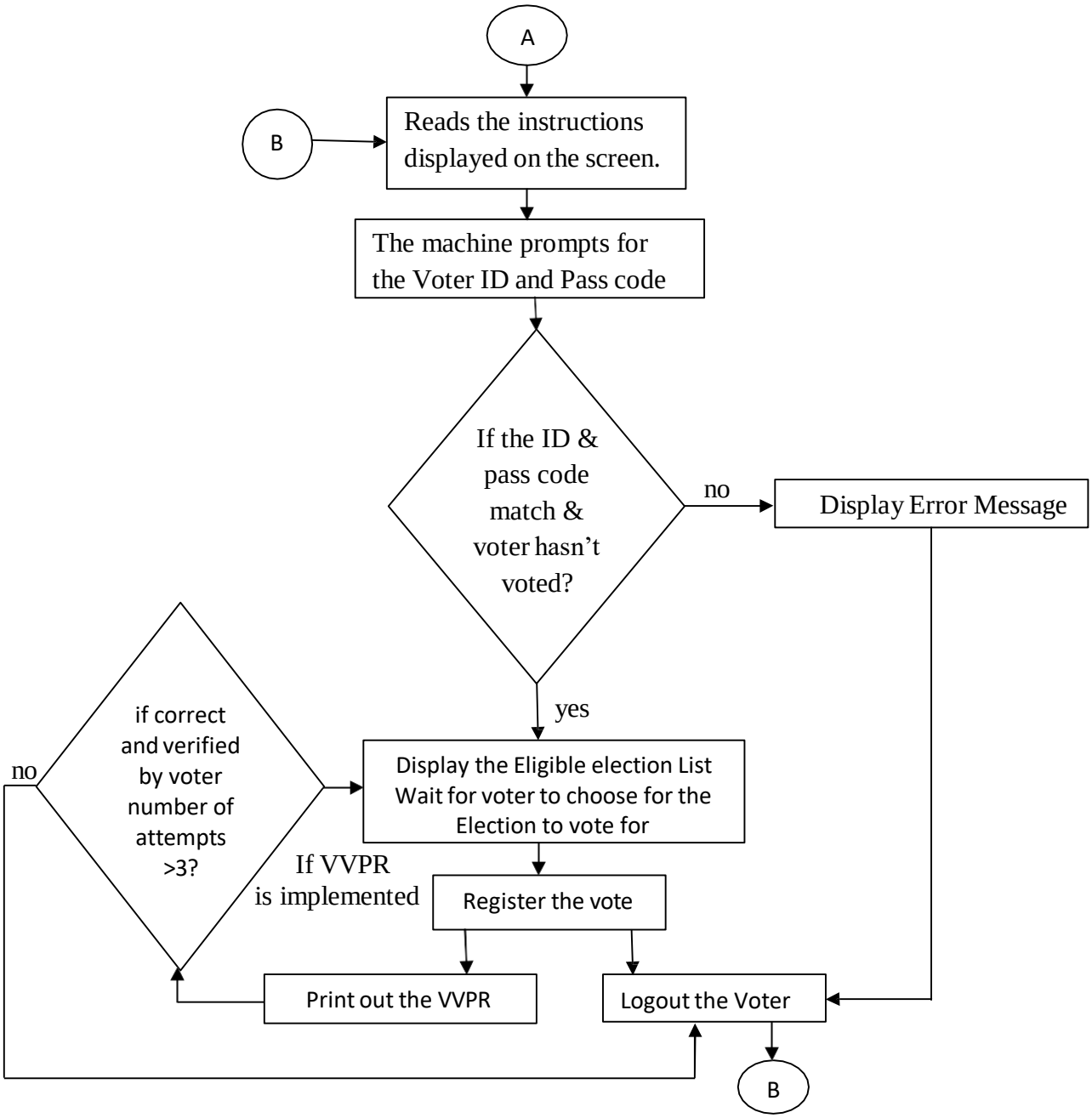
Product Design

FRONT PANEL	
Display Unit	Keypad Unit

BACK PANEL
Sealed Slot for Memory CARD I and CARD II

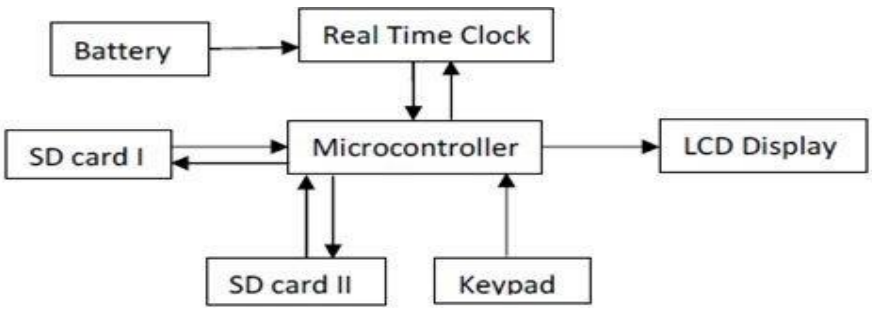
Flow Diagram: How voting will take place:





3. Hardware Design

The hardware required for this design mainly needs to interface various blocks with each other in synchronisation. These are as follows:



3.1 Memory Storage Components

We require a memory which is easy and fast in read write and easily detachable from the system whenever required, without any changes to the internal circuitry. Memory requirement is also high, as it stores a large database of the voters and other details. Therefore, we decided to go for SD Cards. There are two sd cards in the proposed machine:

Card I: It is a Read Write Card that will have data from the Election Commission, serially transferred to the card by the PC:

- Voter Details: Voter ID, Pass code, Election Eligibility.
- Candidate Details: Candidate Name, Party, ID, votes casted in favor,(symbol in case of graphics display).

--Read Only: Voter details section
--R/W: No. of votes casted in the favour of a particular candidate will be updated during voting.

Card II: It will have the stamped vote details, exactly as they are casted by the voters. Data already written will be freeze, to keep details intact and non-modifiable at any point of time. This card can be used in case of any legal concerns regarding the mismatch of the expected and actual results, or if some malpractice is suspected in the tallying process.

3.2 Microcontroller

It will control the interfacing between various blocks as shown above in the machine. It has the voting software programmed in it. We have chosen the Atmega 32 for our design. It has a serial interface, two wire link interface, which are required for SD Card and RTC interfacing respectively. Our system requires a uC with low computational speed, since they happen only at the time votes are casted and that too are not prolonged, making us choose Atmega-32. Initially, we used Atmega16, but due to larger RAM requirements of the software we moved onto Atmega32.We have left many pins of uC unused, this is for further use of these port pins to implement future works like biological unit, printer etc. Also on the same line we shifted from 8bit to 4bit LCD interfacing.

3.3 Output Display Unit: LCD

This is the output unit of the machine. Voter Interface will be provided here. It displays all the relevant details about the current election and voter can act accordingly and view the options out here. We have implemented it on the 16X2 display currently since the display items were not complex as we didn't have symbol for a candidate in present machine.

3.4 Input Unit: Keypad

This is the input unit of the machine which takes various instructions from the voters, while voting and also in configuring of the EVM before elections. The voter enters the details asked for identification via voter ID and pass code, and moves further on to the process of voting. Voter ID & Pass code will be in the form of numbers so as to remove any issues related languages, and cultures. Other 4 keys will be as: arrow keys, enter key. This will be specified on the hardware.

3.5 Real-Time Clock (RTC)

It is used for time stamping of votes on the SD Card II. It readily gives current time and date whenever needed. Battery is needed for Real time clock so that in spite of power failure the clock runs. Also there will less burden on the power supply. But we propose to implement it in next stage of our design.

4. Software Design

The software will be programmed in the microcontroller. This will determine how casted votes are handled by the system. The interfacing between various blocks is also integrated with this. Also there will be some encryption for the Voter ID numbers which will be printed on the Paper Record which we have proposed as extension. The software is being implemented in 4 steps (4.1-4.4).

4.1 Validation of Voter

Voter is validated by checking the entered password from the corresponding voterid's passcode entered in voter.txt file.

4.2 Checking the status of vote

Checking the voter status: It first checks if the voter has voted for central elections by searching for the voterID in the file centl.txt.
If no, he is asked to vote. If yes, we check if user is eligible for state election happening.
If the voter is not eligible, he is logged out.
If eligible, similar procedure as of central election is carried out.

4.3 Vote Registration

The list of candidates standing for election under consideration is displayed along with the candidateID and the voter is asked for the candidateID he wish to vote for. The corresponding count in ccand.txt gets incremented by one, thereby giving the count of votes received by a particular candidate. Same procedure is followed for state elections as well.

4.4 Time stamping of vote

Time stamping of vote is done to avoid bogus voting and track them. The vote casted by a voter gets registered at mini SD, in the file centl.txt. Details of the vote casted are given in description of centl.txt file

4.5 File Structure

4.5.1 Card-I (micro SD)

a).Voter.txt – This file has the attributes of the voter namely voterid, voter passcode, voter name and his state of eligibility. All the attributes of one voter are separated by a '|' character and each voter record comes in next line.
The file ends with a ~ character.
For eg. – 1|9|assam|ram means voterid is 1, passcode is 9, voter state is assam and name is ram.

b). Cand.txt – Contains the list of candidates standing for central elections. Each row corresponds to one candidate and has the candidateID, candidate name and number of votes casted for the candidate.

c). Scand.txt – Same as cand.txt but for state elections.

d). Sdet.txt – This stores the name of state in which elections are happening.

4.5.2 Card-II (miniSD)

a). Centl.txt – Stores the vote casted for central elections. Format in which it gets stored is: voterID|candidateID|time at which vote casted|date at which vote casted*
Each vote is separated by a *, and the end of file is marked by ~.

b). State.txt – Same for state elections.

4.6 Assumptions and Constraints of file storage

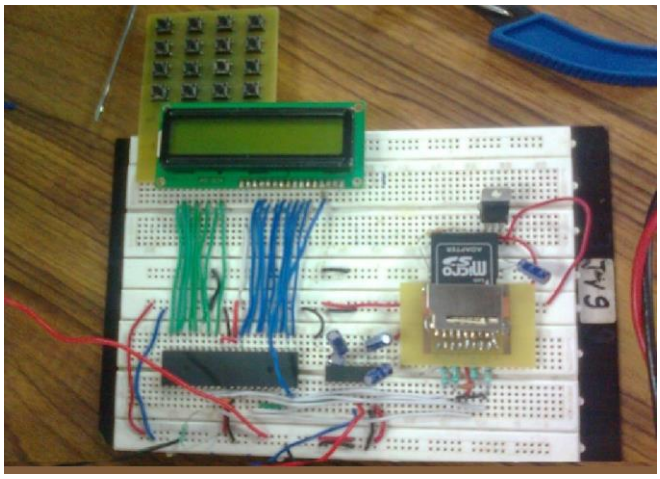
1. VoterID and candidateID should start from one and be consecutive integers.
2. VoterID and passcode can be of maximum 19 characters.
3. CandidateID of 2 characters, candidate name is 30 characters and vote casted is 5 characters.
4. The record of each candidate in cand.txt and scand.txt is occupied fully. That is, if a candidate name is of 10 characters only, still 30 characters should be full. Rest 20 are written by a dummy character, say $. Similar is for all fields in this file. This is done since for updating a file from mid, we need to know the exact character position from where the vote count should be updated.
5. No votes for a candidate has to be represented by a 0.

5. Block Wise Testing

5.1 Secure Digital Card

- Micro SD Card: 2GB
- Mini SD Card: 2GBB
- Serial Clock Frequency (SCK): 0.5 MHz
- Protocol – Interfaced with Atmega16 through the SPI peripheral. The card is acting as slave and uC as master.

SD card is interfaced with the atmega16 using FAT32 filesystem. Two pins are used as chip select for selecting between the cards and hence the communication is done independently between them. Since two cards were used, MISO, MOSI and SCK pins of uC are multiplexed to both the cards. Communication is done block wise at a time and each block size is 512 bytes. The resistors at these pins are provided to match the current specifications of SDcard. SDcard runs on a supply voltage of 2.7-3.6V, we have used a fixed voltage regulator of 3.3V(LM1117).



5.2 Keypad

We have used the logic of scanning the rows and columns of the 4X4 matrix keypad by pulling rows low one by one and then checking the corresponding column press for that low row. If a key in that row is presses then a column intersecting with that row will be low, hence key detection was done. Also we introduced a delay of 20us in order to check that

really a key was pressed and hence again took the input for that key. The program flow diagram is as follows:

- Select a 8bit Port for attaching keypad
- Set 4 bits for row connection: output pin configuration
- Set remaining 4 for column connection: input pin configuration
- Initialize output pins as all 1's
- Run a loop in which each row is pulled down at a time
- Check for column and wait for key de-bouncing time and then again read the input pins.
- Wait for input port to settle
- Then according to the read value assign the symbol to that key using a switch case.

5.3 LCD Interfacing

- 16X2 Hitachi HD44780

LCD is interfaced in 4 bit mode, that is we have used only 6 pins of uC:
4: data line (DB7-DB4)
2: control pins→ RS(register select) and E(enable)
We only write data onto LCD. Instead of 8bit mode 4bit was chosen since our application is not high speed and delays due to the introduction of multiplexing data line twice, doesn't affect the system speed pertaining to our requirement. Its important to note down the sequence of instructions to be followed, else the 4bit implementation won't give correct results. The program flow diagram is as follows:

- There's an initialization sequence to be followed mentioned in the datasheet, as a 8 bit interface to set the LCD in 4 bit interfacing, done in init_lcd() function.
- Define 2 functions for writing data and command words by selecting data and instruction register respectively.
- Sequence of instruction to be followed is:

1. Put the word to be written in the respective register on data lines.
2. Select the data/instruction register to be written in.
3. Strobe the enable line.

Repeat this for lower nibble of the 8bit word.

- Also define a function for selecting the position on the LCD to be written, move_to(x,y).

This function writes the DDRAM address to position the cursor. As mentioned in datasheet for 16X2, 2 line display address are as follows.

Display position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DDRAM address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F

5.4 Real Time Clock

Chip used: Maxim DS 1307
Protocol: Serial interface using Two-Wire interface using Inter-Integrated Circuit(I^2C) multi-master serial bus.
RTC interfaced with LCD.
Specifications of communication:

RTC

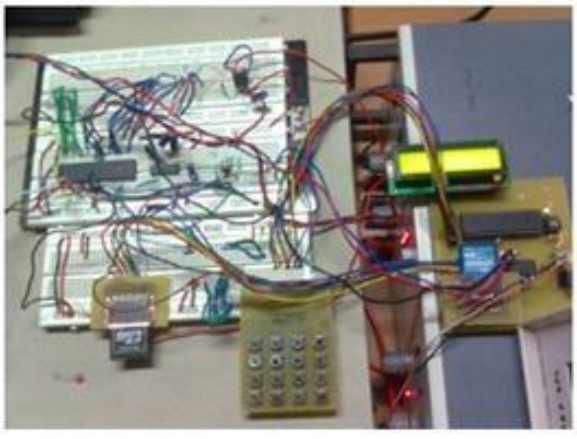
- uC Frequency : 1MHz
- Serial Clock Frequency (SCL): 28KHz (calculated from data sheet).
- Serial Data Line: SDA

Currently RTC resets on each power shut down since there is no back battery put into the system.

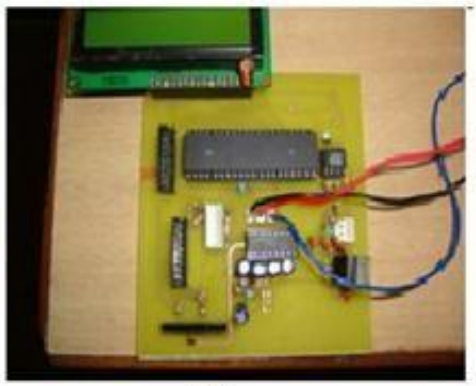
6. Test Results and Discussions

The blocks tested are running successfully:

- Serial communication with the PC from the uC is implemented successfully (serial port used). This was performed in 3 steps- character by character mirroring, buffer transmission, modify the received buffer and then transmit. We observed this transmission on the DSO as well to verify whether the serial communication is working perfectly or not.
- MicroSD card interfaced with uC and following steps were executed.
  1) A single block was read/write/deleted from the card.
  2) Card was formatted in FAT32 and then the files were read/write/deleted from the uC.
- LM317 was used first which is a variable regulator due to ease of access. It did work out for while we used only one card in the circuit. But on introduction of second card in the system, loading occurred and the supply voltage of regulator was not within the range 2.7-3.6, though it was supposed to be at 3.3V(set voltage).
  Due to this, one microSD card was blown. Hence, to account for the huge sensitivity of microSD**,** we moved on to LM1117, for using this. These variations were removed by the fixed voltage regulator LM1117.
- Before writing the EVM software, card to card communication was tested, as a prerequisite step.
- Software issues :- "voter.txt" is not taken by the avr compiler and hence this was passed character by character as 'v','o','t','e','r' .
- When we implemented the writing of data from one card to another, via the uC, we encountered the problem of RAM flow. Each block is of 512 bytes and we required two thereby leading to >1K of RAM. This caused RAM overflow of Atmega16 and problems like main reset were encountered. Hence we switched to Atmega32, our final processor of the EVM.



Bread Board Implementation of EVM



PCB

Testing on PCB

7. Future Plans

1. Facility to provide printer extension in the machine, so that voter can be sure of his vote casted after seeing it on a hard copy. This will help to track the malfunctioning of EVM and have a backup of votes in that case.
   Voter Verifiable Paper Record (VVPR): (if implemented)
   It will have following details on it:
   1. Candidate name for which voter voted
   2. Encrypted Voter ID

   We propose to have this paper record behind a glass shield so that voter doesn't tamper with it.
2. Graphics/touch screen implementation for display unit.
3. Facility of increased font size for poor eyesight voters.
4. Biometric Voter Identification.
5. LAN connectivity of the machine.

8. References

1. Datasheets: Atmega16,32, DS1307, LCD: Hitachi HD44780, max232, voltage regulator 7805, Low drop out linear regulator 1117
2. Web sources :
   http://www.dharmanitech.com/,
   http://www.schneier.com/blog/archives/2004/11/the_problem_wit.html
   http://www.tiresias.org/research/guidelines/evoting_projects.htm#australia
3. Comprehensive Study: Documents in pdf format as follows

a) FAQ's: Indian EVM
   Published by: Election Commission of India
b) Electronic Voting Machines from Bharat Electronics Limited, India
c) Facts About Electronic Elections: www.VotersUnite.Org
d) Paper: On Voting Machine Design for Verification and Testability
   Authors: Cynthia Sturton, SusmitJha, SanjitA.Seshia, DavidWagner
e) Report: Electronic voting – challenges and opportunities
   Published by: Ministry of Local Government and Regional Development, Norway

9. Appendix

9A Schematic Developed