

Test report 2 - Voting machine ESI2 - Software for elections in Ireland

Testing engineers: H. Schrepf, D. Saborrosch, G. Kilz Date: 2003-09-17
Model: ESI2 Version: ESI2_1 Software version: 2.02 / 1.05 / 2.01

The aim of the evaluation is to prove applicability for elections in Ireland.

Contents

Introduction.....	1
Requirements	3
0. Identification	3
1. General	5
2. General conditions concerning the provision of information by the voting machine	12
3. Installation of a ballot module	16
5. Preparation before the poll	19
6. The poll	21
7. Displaying and printing the total number of votes at close of poll	30
8. Reliability and security of the voting machine.....	33
9. Operability	41
10. Reporting and solution of problems.....	43
13. Conditions for the use of voting machines for two or more polls simultaneously.....	46
14. Documentation	50
Summary	52

Introduction

The evaluation is based on the requirements stated in the document *Requirements for Voting Machines for Use at Elections in Ireland, Appendix, DVREC-2* (5 March 03), and on the document *Functional Specification, Nedap Voting System ESI2, Powervote*, document version 1.9, date 2003-05-28.

First, the requirements for the software of the voting machine will be given. Then the results of the evaluation will be described and summarised.

The voting machine software consists of three source code packages: main board software (MB), connection board software (CB), and display board software (DB). All requirements were tested on the three source code packages. The main board software alone fulfils most of the requirements.

Every voting machine contains the main board program, the connection board program and five instances of the display board program.

The following software versions were tested:

main board software: 2.02
connection board software: 1.05
display board software: 2.01

The checksums for the main board software are: even: 00EA28AA, odd: 010C794E.

The following documents were used:

General documentation:

- Functional Specification, Nedap Voting System ESI2, Powervote, document version 1.7, date 2003-03-05,
- Functional Specification, Nedap Voting System ESI2, Powervote, document version 1.9, date 2003-05-28,
- Project Overview, Nedap Voting System ESI2, revision A, date 2003-05-08,
- Detailed Software Design, Nedap Votingsystem ESI2, ESI2_1 Software, document version 1.2, date 2003-08-05,
- Reliability of the Voting Machine ESI1, document version 1.1, date October 2001, confidential,
- Vote storage, Nedap Voting System ESI1, document version 1.2, date 2001-12-14, confidential,
- Interface Specification Voters Panel & Main Processor Board, revision B, date 2003-05-08,
- Hardware/software specification display board, revision 0.3, date 2003-05-12,
- Hardware and software specifications connection board, Voting machine Ireland ESI1, version 0.0, date 2001-10-05,
- Voting machine Operator's Guide, Powervote, version 0907-2
- List of error/event messages relating to Nedap voting machine type ESI2, Powervote, version 0111.

Test documentation:

- Overview ESI2 software tests, revision A03, date 2003-08-27,
- Integration tests software main board ESI2, revision 1.1, date 2003-08-06,
- State machine tests software main board ESI2, revision 1.0, date 2003-04-01,
- State machine tests software main board ESI2, revision C, date 2003-09-03,
- Service switch, foll.nr. A25, revision A01, date 2003-02-13 (test of service switch),
- Functions tests software main board ESI2, revision 1.1, date 2003-08-14,
- Update report functions tests software main board ESI2, Revision A, date 2003-06-03,
- Communication tests software main board ESI2, revision 1.1, date 2003-08-06,
- Driver tests software main board ESI2, revision 1.1, date 2003-08-14,
- Event and error tests software main board ESI2, revision 1.2, date 2003-09-03,
- Update report Event and error tests software main board ESI2, revision A, date 2003-06-03,
- Connection board function tests, revision A, date 2003-04-29,
- Display board function tests results, revision A, date 2003-03-21,
- Display board function tests results, revision A, date 2003-04-29.
- Testdescription Vote Storage at Power Failure, Nedap Voting System ESI1, Vote Storage, document version 1.1, date 2001-12-17,
- Vote while power down, revision a.00, date 2003-05-09,
- Cast vote beep-interrupt, ESI2, revision A02, date 2003-08-18.

Software update reports:

- Update report software main board ESI2, revision A, 01.04 to 02.00 main board, date 2003-02-18,
- Update report software main board ESI2, revision A, 02.00 to 02.01 main board, date 2003-04-03,

- Update report software main board ESI2, revision A, 02.00 to 02.01 main board, date 2003-04-22,
- Update report software main board ESI2, revision A, 02.01 to 02.02 main board, date 2003-06-03,
- Update report software connection print ESI1, revision A, 01.04 to 01.05 connection print, date 2003-03-19,
- Update report software connection print, revision A, 01.04 to 01.05 connection print, date 2003-04-22,
- Update report software display print ESI2, revision A, 01.06 to 02.00 display print, date 2003-02-18,
- Update report software display print ESI2, revision A, 02.00 to 02.01 display print, date 2003-03-19,
- Update report software display print, revision A, 02.00 to 02.01 display print, date 2003-04-23,
- Voting machines - modifications, received 2003-03-06.

Requirements

0. Identification

The identification of all parts of the system is an essential prerequisite for software tests.

0.0-1 All executable programs shall have a version number, which is shown on request and which cannot be changed during election process.

MB: The main board software contains a macro with the software version number¹. Macros cannot be changed during run time. If they have to be changed, the source code must be compiled and linked anew. The content of the version macro is displayed or printed out without being falsified. It is displayed on request (modus Functions / About voting machine / Versions and checksums / Main board / Hardware and software versions) and printed on request (modus Functions / About voting machine / Print information / Print settings or .. / Print security check). It is also part of the voting result printout and of the open poll statement printout.

CB, DB: The connection board software and the display board software also contain macros for the software version number². The macros cannot be changed during run time. Their contents are sent to the main board software and read out without being falsified. They are displayed on request (modus Functions / About voting machine / Versions and checksums / ...) and printed out on request (modus Functions / About voting machine / Print information / Print settings).

¹ MB/source_def.h::SOFT_VERSIE

² CB/main.h::SWMajor, SWMinor; DB/main.h::SWVersionMajor, SWVersionMinor

Summary: The requirement is fulfilled. All executable programs have a version number which cannot be changed and is shown and printed on request. All software version numbers are part of the "Settings" printout.

0.0-2 All source code modules shall have a version number.

MB, CB, DB: The three source code packages consist of 112 files altogether. All source code files have a version number. The version number management and the revision histories in the files are ok.

Summary: The requirement is fulfilled. All files have a version number.

0.0-3 Each voting machine shall have an identification number, which is shown on request and which cannot be changed during election process.

MB: The main board uses a special memory (Xicor) to save error and information messages and several device characteristics. One of these device characteristics is an identification term (alphanumeric string) called "machine ID." This machine ID can be changed by a function³ which is used in Service mode. This function thus is accessible for service personnel only. Another write access to the memory address of the machine ID does not exist.

The machine ID can be displayed on request (modus Standby) and printed on request (modus Functions / About voting machine / Print information / Print settings or Print security check). It is part of the voting result printout.

Summary: The requirement is fulfilled. Each voting machine has an individual machine ID (alphanumeric string) which can be displayed and printed on request. The machine ID cannot be changed during the voting process.

0.0-4 Each ballot module shall have an identification number, which is shown on request and which cannot be changed during election process.

MB: Each ballot module contains several types of information: module information, election information, display information, layout information, and votes. The module information region contains an identification term (alphanumeric string) called "module ID." A search for write accesses to the module ID shows that it can be changed only if the module is plugged into the programming slot of the PRU and completely erased there before⁴. The PRU has to be in Service mode for that. So the module ID can be changed only by service personnel.

The module ID is typed out without falsifications. It can be displayed on request (Standby mode) and printed on request (modus Functions / Open poll / Print open poll statement). It is also part of the voting result printout.

Summary: The requirement is fulfilled. Each ballot module has a module ID (alphanumeric string) which can be displayed and printed on request. The module ID cannot be changed during the voting process.

³ MB/flash.c::set_machine_ID()

⁴ MB/com_program.c::send_set_module_id_resp_f()

1. General

1.1-1 ***The software shall contain all functions, which are described in the Functional Specification (version 1.7 of 5 March 2003).***

MB: An inspection of the main board software shows that all functions described in the *Functional Specification* are implemented. The main states (Standby, Election, Functions, and Service) and sub states are implemented by special state functions. Combined they form a state machine layer which is supported by layers for man-machine dialog, help functions, hardware drivers, and diagnostics services. The functions are grouped into modules. The structure complies with the description given in the document *Detailed Software Design*.

CB, DB: These software packages contain machine-intimate functions belonging to the hardware driver layer.

Summary: The requirement is fulfilled. All functions described in the *Functional Specification* are implemented. The software structure is clear.

1.2-1 ***The software manufacturer shall have and apply an appropriate development model.***

The document *Project Overview* gives a short description of the Nedap life cycle model for software development⁵. According to this description, each project includes the requirements analysis phase (results: functional requirements, technical requirements), the design and implementation phase (results: implementation plan, source code, software description, test specification), the test phase (results: test plan, test results, changed source code), and the release phase (results: update list, projected changes). According to this life cycle model, the documentation contains the *Functional Specification*, several *Technical Specifications*, the *Detailed Software Design*, the test documentation, and several update lists.

During the software development, a tool for configuration management of source code (QVCS) is used.

The life cycle model used by Nedap is in keeping with the project size and the manpower involved in the development.

Summary: The requirement is fulfilled. The manufacturer has an appropriate development model.

⁵ Document *Project Overview*, p. 4 par. 2.5.3

1.2-2 The source code shall comply with a standardised programming language.

MB, CB, DB: The software is written in the programming language C. The current language standard is ISO/IEC 9899:1999, which basically follows the ANSI C standard commonly used.

A static analyser⁶ was used to prove conformity of the source code with ANSI C. It produced no warnings. The code complies to the standard.

Summary: The requirement is fulfilled. The code complies to the C standard.

1.2-3 The source code shall comply with the current principles of software engineering.

MB, CB, DB: The software must comply with the principles of Structured Programming, Modularisation, and Data Hiding. It also should not use problematic constructs.

Structured Programming:

The software complies with the principle of Structured Programming. The only slight deviation from good programming style is the use of jumps in two functions⁷. These jumps cannot be avoided without the readability of the code being considerably reduced. That's why they are acceptable.

Long jumps are not used.

Modularisation:

The functionality of the software is grouped into several modules. The modularisation is comprehensible and clear. Each function is found in the module supposed to contain it.

The modules communicate mainly by parameters.

Data Hiding:

The principle of Data Hiding is met as far as reasonable. Often variables and constants are grouped together using structures and enumeration types. No variable has a scope greater than necessary. To achieve this, global static variables, local variables, and parameters are used wherever possible. Unions and global variables are avoided as far as reasonable.

Potential Problems:

The software was checked for several constructs which might lead to problems to future software changes. In high-safety software such constructs should be avoided. They are usually forbidden by programming rules issued at the manufacturer's.

⁶ PCLint 7.5 (Gimpel Software); warnings concerning ANSI conformity switched on.

⁷ MB/program.c::erase_eeeprom(), test_module.c::test_module_toestand()

The following problems were found:

Potentially problematic construct	Number of occ.		
	MB	CB	DB
Local automatic variables which are not initialised	*)	*)	*)
Variables which are not used, or written variables which are not subsequently read or read variables which are not written before	6	0	0
Functions which are declared without return type	0	0	0
Function definitions inside header files	0	0	0
Parameterised macros without brackets around parameters or around the macro itself	0	0	0
Side effects in macro "calls"	0	0	0
Multiple selections without default case	1	0	0
Use of variables or constants from other modules in the initialisation of global symbols	0	0	0
Side effects in actual parameter lists	0	1	0
Variables set twice in a statement	0	0	0
Mathematical operations (division, logarithm, radical) which do not test the arguments	0	0	0
Division operator used for arguments with differing sign	0	0	0
Floating numbers compared without tolerance	0	0	0

*) Most of the local automatic variables are not initialised.

The check was carried out using a special tool⁸ and manual inspection.

Compared to the overall code size (about 25000 lines of code), the number of occurrences of problematic constructs is very small. At present, no occurrence leading to a programming error ("bug") was found. As they may all lead to errors, when the software is maintained (changed), it is recommended improving the code in the next update.

Summary: The requirement is fulfilled. The software complies with the current principles of Software Engineering. It is recommended for the next versions avoiding certain problematic constructs which may lead to problems in future.

1.3-1 The diagnostic software detects changes that influence proper functioning of the voting machine.

The correct function of the voting machine depends on the ROM which holds the persistent program copy, on the RAM which holds the temporary (working) program copy, on the rest of the hardware, and on the integrity of the ballot module.

Status changes, failures, and problems are managed with the aid of a status table⁹ contained in the main board software. This status table is permanently updated and checked at regular intervals.

⁸ PCLint 7.5 (Gimpel Software)

⁹ MB/flash.c::config_tbl[]

Persistent program on ROM

MB: The software provides a checksum test to be carried out at the start-up¹⁰. This test checks the ROM area in which the program is burned in and compares the new checksum to a default value burned in at the end of the ROM area. In case of differences, the voting machine is shut down.

Besides that internal checksum, there is an external checksum which contains the internal one. This external checksum is calculated separately for the even and the odd memory¹¹ at the start-up. It is displayed or printed out on request and is also part of the voting result printout.

CB: The connection board software also provides a checksum test at start-up¹². The test checks the ROM area containing the program and compares the result with a default value saved outside the checksum area. In case of differences, an error message is sent to the main board software which then terminates the program of the voting machine.

The checksum itself is sent on request to the main board software.

DB: The display board software works in the same way as the connection board software. A special function¹³ calculates and compares the checksum and initiates an error message which is sent to the main board via the connection board, if necessary. This terminates the program of the voting machine. Besides that, the checksum is sent on request to the main board.

The calculations and the comparisons of the ROM checksums are correct. The output of the checksums is carried out without falsifications. All checksums are displayed on request (Functions / About voting machine / Versions and checksums / <Board> / Checksum(s)) and printed on request (Functions / About voting machine / Print information / Print settings).

Working program on RAM

MB: The start-up procedure begins with a RAM memory test¹⁴. Failures of the RAM lead to an immediate shut down of the voting machine. The normal program flow and the use of variables and stack will start only if the RAM test has been successful.

After start-up, the RAM is tested again while the votes are stored. The RAM will be considered defective if there are inconsistencies between several temporal variables. In such cases, the voting machine will be shut down.

CB, DB: The connection board software and the display board software both contain a start-up RAM test¹⁵. Failures of the RAM lead to an error message being sent to the main board and to a shut down of the voting machine.

¹⁰ MB/romtest.c::eprom_test(), ROM area 0x000000 .. < 0x040000 - 4.

¹¹ MB/romtest.c::calc_program_checksum(), ROM area 0x000000 .. < 0x040000.

¹² CB/init.c::bROMtest(), ROM area 0x0000 .. < 0x4000 - 2.

¹³ DB/init.c::bROMtest(), ROM area 0x0000 .. < 0x4000 - 2.

¹⁴ MB/ramtest.c::ram_test()

¹⁵ CB/init.c::bRAMtest(), DB/init.c::bRAMtest()

Hardware

MB: Hardware failures are managed by means of the status table. The status of each hardware component is evaluated and checked during start-up and run time.

During the start-up a full test of the hardware is carried out¹⁶. This test includes the following hardware components:

- watchdog,
- normal power supply voltage,
- programming voltage for vote storage,
- serial ports COM1 and COM2,
- printer port,
- 4-line-displays on control unit and voting machine,
- keys and buttons of control unit and voting machine, including communication with connection board,
- error and event memory (Xicor).

During run time, the hardware test is permanently repeated. Each time a hardware component is accessed, its status is updated. The evaluation of the hardware status entries is made permanently¹⁷ but at least after each user action. The status check includes the hardware components:

- normal power supply voltage,
- programming voltage,
- number of memory chips in the primary ballot module and the backup module,
- error and event memory (Xicor),
- printer port,
- 4-line-displays on control unit and voting machine,
- keys and buttons of control unit and voting machine, including communication with connection board.

Hardware failures are handled according to their severity. Minor problems lead to an error message only:

- Problems of the printer port only lead to an appropriate error message on the voting machine display. Problems like "Paper empty" may be solved by the polling staff.
- Certain failures of the keys (key pressed too long, more than one key pressed) lead to an appropriate error message and a sustained beep and may be solved by the voter or the polling staff.

In these cases, the problem may be solved and the voting machine continues with normal program execution.

Major hardware failures lead to a shut down of the voting machine. Besides that, the voting machine is blocked for further use ("Machine Block"). Such failures are:

- RAM malfunction,
- loss of a memory chip in the primary ballot module during election process,
- occurrence of new inconsistencies or read/write errors in the primary ballot module.

¹⁶ MB/init_diag.c::init_diag()

¹⁷ MB/run_diag.c::run_diag()

In these cases, the program is terminated and the voting must be continued on another voting machine. Nevertheless, the ballot module is not influenced. It may later be read-out on another voting machine or PRU device.

In all other cases of hardware failure, the voting machine is shut down. After switching-off and -on and repetition of the start-up tests, it will possibly work again.

CB, DB: The connection board software and the display board software both provide a hardware test at the start-up (shorts, breaks)¹⁸. Malfunctions are reported to the main board software and lead to an error entry in the status table. The status check of the display boards is permanently repeated during run time. Minor errors lead to an error message only (key pressed too long, more than one key pressed), all other errors including a lacking communication link lead to a shut down of the voting machine.

Integrity of the ballot module

To ensure the integrity of the ballot module, important data are saved twice or four times in the memory. The most important data¹⁹ are saved four times in combination with a Hamming code. Furthermore, the areas of election information and layout information are protected by independent checksums.

The integrity of the ballot module is checked during the start-up. The check covers:

- equality of all copies,
- validity of Hamming codes,
- validity of checksums,
- plausibility of all data.

The check is carried out for all memory areas including the votes already saved.

Violations of the integrity of the ballot module lead to operational restrictions ("Module Block," no further voting possible, only FUNCTIONS usable).

After the start-up time, the integrity of the ballot module is permanently checked to a reduced extent²⁰. Furthermore, the validity of the data and address lines of the ballot module is checked before and after the votes are stored²¹. Changes in the module status lead to the voting machine being blocked ("Machine Block"), see above, and to a program shut down.

Summary: The requirement is fulfilled. Failures of the ROM containing the persistent program copies are detected at start-up time. Failures of the RAM containing the working copies are detected during start-up and in part during run time. The rest of the hardware is checked during start-up and run time. Violations of the integrity of the ballot module contents are detected at start-up and run time. Failures of hardware and module contents lead to different reactions depending on their severity.

¹⁸ CB/init.c::bSelfTest(), DB/init.c::bSelfTest()

¹⁹ Election numbers, election types, number of preferences per election (poll), votes.

²⁰ MB/program.c::schrijf_stemmen(). The extend is reduced to the most important checks because otherwise the system would take up too much time checking the ballot module.

²¹ MB/keuze.c::data_adres_check()

1.4-1 Any alteration of the installed software by an unauthorised person should be detected.

MB, CB, DB: The persistent program copies are held in ROM memory chips. Changes in these memory chips lead to failures in the ROM test (see requirement 1.3-1) followed by a shut down of the voting machine.

An exchange of the ROM chips including fraudulent presentation of the correct checksums cannot be avoided by software but by means of sealing only.

During run time, the program copies are held in the RAM memory. If the RAM memory is changed (e.g. by electromagnetic fields), the program is no longer normally executed nor capable of producing a regular watchdog signal. In this case, the watchdog circuit leads to a shut down of the program. The program may be reloaded from the ROM and started again.

MB: For the main board software the watchdog circuit itself is tested at the start-up. It consists of the watchdog time-out circuit and a watchdog panic circuit to check the trigger signal produced by the program²².

Summary: The requirement is fulfilled. Alterations of the program in the ROM or RAM memory are detected. Changes in the ROM inhibit the program start, while changes in the RAM lead to a shut down of the program.

1.4-2 Any alteration of the content of the ballot module by an unauthorised person should be detected.

MB: New data may only be written into the ballot module if a certain sequence of write commands is used²³. A wrong sequence of commands or extra bit setting at the port results in the programming sequence being stopped. Furthermore, erasure of data is physically impossible as long as the ballot module is inside its slot and only programming voltage is used²⁴. So it is almost impossible to change the contents of the ballot module in a well-defined manner to add, delete or change particular votes.

To limit the effect of non-specific attacks (e.g. by electromagnetic fields), the contents of the primary ballot module are protected by means of redundancy and checksums (see requirement 1.3-1). Alterations of the contents are found by checking the equality of all copies and the validity of Hamming codes and checksums. A full check is carried out at the start-up and a reduced check during run time. If violations of the integrity are detected, either the machine will be blocked for further use ("Machine Block") or the ballot module will be blocked for further voting ("Module Block").

The only data protected neither by redundancy nor by checksums are:

- The number of programming/erasing cycles for the module. This number serves for informational purposes only. It has no effect on the functionality or other properties of the voting machine.
- The number of released polls for all voters. This number is calculated and written into the ballot module at close of poll. Re-counting the released polls can easily

²² Document *Reliability of the Voting Machine ESI1*, p. 6 par. 3.2 and 3.3

²³ same, p. 6 par. 3.4

²⁴ same, p. 7 par. 3.5

restore this number. Unauthorised changes can be detected by comparing the number with the result of a new count. This comparison is carried out when the votes are transferred to the PC software.

- The checksum for the layout memory and the checksum for the data and address line test. If these two numbers are defective, the checksum tests will fail and the module blocked for further voting.
- The number of deactivations for each poll. These five numbers are written into the ballot module at close of poll. They are for informational purposes only and have no effect on the functionality or other properties of the voting machine.
- The freely programmable texts which are displayed to the voter. If there are falsifications of the contents of the ballot module the displayed text would show slight irregularities (e.g. missing or wrong letters).

Only unauthorised changes of the number of programming/erasing cycles, the number of deactivations for each poll and the freely programmable display texts are not automatically detectable. All these data have no effect on the functionality of the voting machine. Unauthorised changes of the freely programmable display texts may have a slight effect on the usability of the voting machine, but they are easily detectable for the polling staff using the Open poll statement.

Summary: The requirement is fulfilled. Unauthorised alterations of the contents of the ballot module are extremely difficult to achieve. They are either detected or have no effect on the functionality of the voting machine. They may only have a slight effect on the usability of the voting machine. Changes aimed at the addition, deletion or change of particular votes are almost impossible.

2. General conditions concerning the provision of information by the voting machine

2.3-1 *The software shall print or display all candidate information which is contained in the ballot module before and after polling. This information shall not be changeable during election process.*

The layout memory is part of the primary ballot module. It contains the following information:

- row and column of a programmed key (coded as field indices of a table²⁵),
- poll number the key is assigned to (part of the table elements),
- choice number the key is assigned to (part of the table elements),
- name of the candidate the key is assigned to (accessed via an offset in the table elements).

MB: A search for write accesses to the layout memory shows that it can only be changed if its address lies in the range of the programming slot of the PRU. So there are no changes of the layout information during the voting process. Besides the layout memory is secured against unintentional modifications by a separate checksum which is checked regularly during election day.

²⁵ MB/keuze.c::s_kp_toets, p_kp_toets

The layout information can be displayed and printed on request at any time (Functions / Open poll / Show open poll statement on display and ... / Print open poll statement, respectively.). The printout and the display texts list all programmed keys with their poll numbers, choice numbers and candidate names. The software itself cannot verify the correctness of the layout information. It is important to verify the assignment between candidate or party names to the correct keys before voting by a manual test at the polling station or at the polling centre.

Summary: The requirement is fulfilled. The candidate information, which is contained in the ballot module, can be displayed and printed before and after the poll. The information cannot be changed during the voting process.

2.3-2 The software shall print the total number of activations of the voting machine before and after polling.

MB: The total number of activations is computed using the number of voters for each poll (see 2.3-3) and the number of deactivations for each poll (see 2.3-4). The sum is computed correctly. The number of activations is part of the Open poll statement and of the voting result printout (Functions / Open poll / Print open poll statement, Functions / Close poll / Make back up and print statement).

Summary: The software prints the total number of activations before and after polling.

2.3-3 The software shall print and display the number of votes including null votes on the voting machine before and after polling.

The votes are saved in the vote memory of the primary ballot module.

MB: The main board software contains basic functions which search through the vote memory, collect votes and validate them²⁶. The result of the validation depends on the correctness of the Hamming code for each vote part and the equality of the four copies of each vote part. The validation principles are described in the documentation²⁷. These principles are exactly implemented by the collecting and validating functions.

The result of vote validation is used for three counting functions:

- One function²⁸ counts the number of all polls P released during the polling day. This number includes null votes. If five voters are each allowed to select preferences on three ballot papers, P is equal to 15.
- One function²⁹ counts the number of voters per poll $P_1 .. P_5$. These five numbers include null votes. If a voter is allowed to vote for polls 1 + 2, and another voter is allowed to vote for polls 2 + 3, the result is: $P_i = \{ 1, 2, 1, 0, 0 \}$.
- One function³⁰ counts the number of voters P_i for a certain poll i (including null votes), and the separate number of null votes N_i for that poll. If a voter is allowed to vote for polls 1 + 2, and selects preferences only for poll 2, the function gives for poll 1: $P_1 = 1, N_1 = 1$, and for poll 2: $P_2 = 1, N_2 = 0$.

²⁶ MB/stemprog.c::lees_stemmen(), get_next_stemarray(), seek_vote(), validate_vote()

²⁷ Document *Reliability of the Voting Machine ESI1*, p.15 par.5.4, Document *Vote Storage*, p.8 par.1.5

²⁸ MB/geheugenmod.c::get_alle_kiezers()

²⁹ MB/geheugenmod.c::get_kiezers_totaal()

³⁰ MB/geheugenmod.c::get_kiezers_totaal_vk()

The numbers P_i always include null votes.

The sums are evaluated correctly as long as the ballot module is present, defect-free, and contains programmed elections.

The results are used for several output purposes. The number of voters per poll P_i and the number of null votes per poll N_i are part of the Open poll statement and thus can be printed out at any time (Functions / Open poll / Print open poll statement). They are also part of the voting result printout (Functions / Close poll / Make back up and print statement).

The number of voters per poll P_i is displayed on the voting machine when the poll has finished and the election results are evaluated (Functions / Close poll / Make back up - Show data on display). It is also displayed on the voting machine in modus STANDBY which can be switched on at any time between two voters.

The voting machine may be switched on and off at any time. When switching on the ballot module may contain votes or may be empty. The polling staff has to check the display of the number of votes and/or print them out. The display should show 0 votes for each poll at the beginning of voting.

Summary: The requirement is fulfilled. The votes are collected, validated and summed up correctly. The number of voters per poll (including null votes) is displayed on the voting machine on request and printed out on request. It is also part of the voting result printout and can be displayed after polling.

2.3-4 *The software shall print the total number of de-activations of the voting machine before and after polling.*

MB: The software has five counters to count the number of de-activations for each poll. These five counters are summed up correctly. They are held in the error and event memory (Xicor) as long as the polling process lasts. At the end of polling day they are written into the ballot module.

The counting of de-activations starts, when the election has begun. The beginning of the election is detectable for the voting machine only by counting the votes already written into ballot module. For that reason the counting of de-activations starts only when at least one voter has successfully completed his/her voting. If the first voter of the polling day decides not to vote, this de-activation is not counted.

If the polling staff selected a wrong release key on Control Unit and corrects this by turning the functions key into Standby position and back to Voting position again, this is counted as a de-activation, too.

For each voter there is a test of the ballot module being full. This test is done after activation of the voting machine. Thus in the case that the ballot module is nearly full the polling staff may be forced to de-activate the voting machine. This is counted, too.

The five numbers of de-activations are part of the Open poll statement and thus can be printed out at any time. They are also part of the voting result printout.

Summary: The numbers of de-activations are summed up correctly. Each de-activation (after the first voter has voted) is counted independent of reason. The numbers of de-activations can be printed out before and after polling.

2.3-5 *The software shall print or display all information which identifies the voting machine, the software and the ballot module before and after polling.*

The voting machine is identified by its machine ID and the hardware versions. The software is identified by the software versions and checksums. The ballot module is identified by its module ID.

MB: All these identification terms cannot be changed during voting process. They are typed out without modifications.

The machine ID and the module ID can be printed at any time using Functions / Open poll / Open poll statement. They are also part of the voting result printout (Functions / Close poll / Make back up and print statement). They are displayed on the voting machine in Standby mode.

The identification of the main board, display boards, and connection board (hardware versions, software versions, checksums) can be printed at any time using Functions / About voting machine / Print information / Print settings. They can be displayed on request using Functions / About voting machine / Versions and checksums /

Summary: The requirement is fulfilled. The identification of the voting machine, the ballot module, and the software can be displayed and printed before and after polling.

2.3-6 *The software shall print or display all relevant details of the election process (kind of election, date, constituency, poll station number) before and after polling.*

MB: The election is identified by the poll names, the polling date, poll station number and constituency. Polling date and poll station number are overall data valid for all polls held simultaneously. The (short) poll names and the constituency are poll-specific data.

The polling date, poll station number, up to five short poll names and up to five constituencies are held in the ballot module. Furthermore, a so-called "print title" (long poll name) is held for each poll. This string may contain further statements on the poll.

A search for write accesses to these data shows that they can be changed only if its addresses lie in the range of the programming slot of the PRU. So they cannot be changed during the voting process.

The polling date and the poll station number are displayed on the voting machine in mode Standby. The short poll names are permanently displayed on the control unit during the voting process and in mode Standby. The constituencies cannot be displayed.

All data (polling date, poll station number, short poll names, print titles, constituencies) are printed on request (Functions / Open poll / Open poll statement) and they are part of the voting result printout.

All data are typed out without significant modifications.

Summary: The requirement is fulfilled. All relevant data of the election process are printed before and after polling.

3. Installation of a ballot module

3.1-1 *Following the insertion of a programmed primary ballot module into the voting machine, the software shall carry out a self check and a check of the voting machine.*

The primary ballot module may only be inserted with the voting machine being switched off. If the power is switched on again after the module has been inserted, the normal start-up test runs. This start-up test includes tests of the software itself (ROM checksums) and a test of the hardware of the voting machine. See requirement 1.3-1 for more details.

Summary: The requirement is fulfilled. After insertion of a ballot module, the voting machine is switched on. Then it executes all start-up tests including a self-check and a hardware test.

3.1-2 *Following the insertion of a primary ballot module and a check of the software and the voting machine, the software shall check the inserted primary ballot module for consistency.*

The primary ballot module may only be inserted with the voting machine being switched off. When the power is switched on again, the normal start-up tests are executed. The start-up tests include a software self-check, a hardware check (see requirement 1.3-1) and a check of the memory contents of the ballot module.

MB: The module check contains the following items:

- existence of both memory chips in the ballot module³¹,
- right type (Ireland) of the ballot module³¹,
- equality and validity (Hamming code) of all votes which have been cast already³²,
- checksum for the layout area³¹,
- checksum for the area containing the election information³¹,
- equality of all data saved twice or four times³³,
- plausibility of all data from the layout area (poll number and choice number of each programmed key)³³,
- plausibility of all data from the module information area (polling date, module ID, major software and hardware versions, poll station number)³⁴,

³¹ MB/test_module.c::test_module_toestand()

³² MB/test_module.c::get_status_stem_geh()

³³ MB/keuze.c::test_programmering()

- plausibility of all data from the election information area (poll numbers, poll types and poll names, maximum number of preferences, constituencies, print titles)³⁴,
- existence of the poll numbers from 1 to maximum³⁴,
- check whether the names of all candidates and all referendum options are printable³⁵.

If the ballot module contains invalid or inconsistent data at start-up or one of the two memory chips inside has been lost, the module itself will be blocked ("Module Block"). The ballot module cannot be used for voting, but it may be used to obtain the voting results or other data. An appropriate error code is shown. After confirming the error display by pressing softkey D, the polling staff may select the modes Standby or Functions, but not mode Election. This is forbidden by a check in the initial routine of the mode Election³⁶.

If both memory chips are lost, the ballot module is considered non-existent. An appropriate error message is shown. The voting machine remains in mode Standby. The mode Functions is selectable, but menu items which use the ballot module will not work. Also voting is not possible since the initial routine of the mode Election³⁶ checks for presence of the ballot module.

In certain severe cases of data inconsistency, the ballot module is considered to be not usable at all. In such cases, an error code is shown and the program is shut down. Such cases are:

- one of the programmed poll types cannot be handled by the software,
- there is no programmed poll,
- one of the programmed polls has an invalid poll number.

In these cases, the voting results might be invalid.

Summary: The requirement is fulfilled. After insertion of a ballot module, the voting machine is switched on. After self-check and check of the hardware, the contents of the ballot module are checked.

3.1-3 *Following the insertion and check of a primary ballot module, the software shall display and/or print the contents of the memory of the ballot module.*

After insertion of a ballot module, the voting machine must be switched on. The start-up tests are executed, including self-check of software, hardware tests and the test of the newly inserted ballot module.

MB: If the start-up tests have been passed successful, the voting machine starts the normal program execution. Now the polling staff may print or display the programming of the ballot module using the menu items of mode Functions (Functions / Open poll / Open poll statement). There is no automatic output of the module contents to printer or display after the start-up tests.

³⁴ MB/test_module.c::test_valid_MI_VI()

³⁵ MB/test_module.c::test_valid_keuze_str()

³⁶ MB/vrijgave.c::init_check_status_vrij()

The printout consists of all relevant data contained in the ballot module:

- ballot module ID,
- polling date and poll station number,
- for each poll: poll number, (short) poll name, constituency, print title (long poll name), number of votes including null votes, null votes, number of activations and de-activations,
- the freely programmable output strings used as messages to the voter or the polling staff,
- the contents of the layout memory with keys / poll numbers / choice numbers and names of candidates or referendum options.

Data for internal purposes (module type, number of programming/erasing cycles, checksums, ...) are not printed.

Summary: The requirement is fulfilled. Following the insertion of a ballot module and the completion of the start-up tests, a printout of the module contents is obtained by simple operations. There is no automatic printout or display of the module contents.

3.2-1 *The software shall only be activated for voting if a module has been installed and the lock on the ballot module slot is in the closed position.*

The voting machine can be switched on only if the lock on the module slot is in the closed position. Otherwise the power supply circuit is not closed.

MB: During start-up³¹ and run-time³⁷ the main board software checks whether both memory chips of the ballot module are available. If no chip is found, the module is considered not available. If one chip is found, the module is considered to be incomplete.

If the module is not available at the start-up, the voting machine will remain in mode Standby and display an appropriate message. The mode Functions is selectable but works only in parts. Those functions which need the ballot module (like Open poll statement) are disabled, other functions (like About voting machine) work. The mode Election is selectable but is left immediately because the initial routine of the mode Election³⁶ checks for availability and status of the ballot module. So the release keys and the keys on the voter panel have no effect.

If the module is incomplete at the start-up, the voting machine will stay in mode Standby and display an appropriate message. The mode Functions is selectable. The mode Election is selectable but is left immediately because the initial routine of the mode Election³⁶ checks for availability and status of the ballot module. So the release keys and the keys on the voter panel have no effect.

If the module is complete at the start-up but one or both chips are lost during run time, the voting machine will shut down³⁷. If this happens during the voting process (mode Election), the voting machine will also be blocked and can no longer be used.

³⁷ MB/run_diag.c::run_diag()

Summary: The requirement is fulfilled. The voting machine may be switched on only if the lock on the memory slot is closed. Voting will be possible only if both memory chips of the ballot module are accessible (readable).

5. Preparation before the poll

5.1-1 The software shall print and display the number of votes including null votes on the voting machine before and after polling.

See requirement 2.3-3.

5.1-2 The software shall only be used for a poll, if it has carried out a self-check and a check of the voting machine.

The voting machine can only be set to the mode Election, if the start-up tests have been successful. These start-up tests include a self-check of the software, a test of the complete hardware and a check of the contents of the ballot module (see requirements 1.3-1 and 3.1-1). During run time, a major part of these checks is permanently repeated.

MB: Depending on the severity of problems found during start-up or run time, there are different reactions of the voting machine:

- If the self-check of the software (ROM checksums, RAM) has not been successful, the program will shut down immediately and operation is no longer possible.
- Many hardware problems also lead to the program being terminated. Operation is no longer possible.
- Simple hardware problems (two keys pressed, key pressed too long, printer not o.k.) lead to an error message. When the problem is solved, softkey C or D are accepted and the program continues with the normal execution including voting. The functionality may be reduced (no printing).
- If the ballot module contains inconsistent or invalid data, it will be blocked. This is marked in the hardware status table⁹ and checked when the polling staff selects the mode Election. If the initial routine of the mode Election³⁶ identifies module blockage in the status table it will display an appropriate error message and inhibit further state changes. The polling staff can only return to the mode Standby or switch off the voting machine. The release keys and the keys on the voter panel stay inactive. Voting is not possible.
- Severe problems with the ballot module lead to blockage of the voting machine. Operation is no longer possible.

Summary: The requirement is fulfilled. The software can be used for a poll only if the start-up and run time tests were successful. These tests include self-checks, tests of the hardware and checks of the ballot module. Any problem occurring during these tests, except simple solvable problems, inhibits the use of the voting machine for polling.

5.2-1 Votes shall only be recorded if the key referred to in 3.2 is inserted in the Control Unit. Withdrawal of the key from the Control Unit will switch the software to standby and prevent votes from being recorded.

MB: The storing of votes is a sub-state of the mode (main state) Election. Votes can be saved only if the mode Election is selected by the polling staff and if any preceding sub-state of Election has been successfully executed.

An analysis of all conditions which must be fulfilled to change to the mode Election yields the following list³⁸:

- the run time tests are successfully executed, and
- the voting machine is not in mode Service (the service switch is not set and the ballot module is not of type "service"), and
- the voting machine is not in mode Standby (the function keyswitch FS is not in key position 1), and
- the voting machine is not in mode Functions (the function keyswitch FS has not been turned while the function button on the control unit has been pressed), and
- several other conditions (availability of the ballot module, ...).

So the mode Election is reached only when the function keyswitch FS is turned around without the function button on the control unit having been pressed.

The main board software basically consists of a main loop³⁹ containing the run time check, a readout of all keys⁴⁰, and a function implementing the changes between the modes⁴¹. So the status of all keys, buttons and switches is permanently checked. A change in the position of the function keyswitch FS results in an immediate change of the mode. As long as FS is in the turned position (without the function button being pressed) the mode Election is preserved. As soon as FS is reset, the voting machine changes to the mode Standby. There is only one exception: during vote storage the keyswitch FS is not checked and thus cannot influence the storing process. Immediately after storage is completed the keyswitch FS is checked again and may cause a change from Election to Standby.

The state transition from Election to Standby takes place without further preconditions having to be met. In particular, the keys pressed by a voter or the preferences already selected by a voter have no influence. All preferences already selected by a voter will be lost if the state changes to Standby before the voter has pressed the CAST VOTE(S) button.

The key can be withdrawn only in its key position 1.

Summary: The requirement is fulfilled. The voting process may start only if the key is inserted in function keyswitch FS and turned without the function button on the Control Unit having been pressed. When the key is reset only or reset and withdrawn, the voting process stops immediately except during vote storage.

³⁸ MB/run_diag.c::run_diag(), functie.c::functie(), functie.c::return_state()

³⁹ MB/stem_comp.c::main()

⁴⁰ MB/admin.c::vul_schakelaar_buffer(), kies_pan.c::haal_VP(), scankeyb.c::haal_KB()

⁴¹ MB/functie.c::functie()

6. The poll

General remarks: The main board software of the voting machine is implemented as a system of main states (modes), sub-states and state transitions, combined with diagnostics routines. Main states (modes) are Standby, Functions, Election, and Service. The main states are divided into several sub-states. The Election mode, for example, is divided into sub-states such as "check status release," "wait for release," "wait for key" etc. Changes between these sub-states are caused by certain conditions, usually by pressing of a certain key. As long as no key is pressed or the condition is not fulfilled, the software remains in its state. In particular, all keys other than the requested one have no effect. The modes/states, sub-states and state transitions (conditions) are listed in the *Detailed Software Design* document⁴². The software complies with these state transition diagrams.

6.1-1 *The software shall not record votes, unless it has been activated.*

MB: The storing of votes is a sub-state of the mode Election. To change to the mode Election the polling staff must turn the function keyswitch FS without pressing the function button on the Control Unit. Besides the voting machine must not be in the mode Service (see requirement 5.2-1).

The mode Election starts with the following checks (sub-state "check status release")³⁶:

- the hardware status table is o.k.,
- the primary ballot module is available and programmed,
- the primary ballot module is not blocked,
- the backup module is available,
- the voting results have not yet been evaluated,
- the primary ballot module is still the same as for the first voter.

If these conditions are fulfilled, the voting machine will change to the sub-state "wait for release."⁴³ If the polling staff then presses one of the release buttons, one or more polls will be activated. The activation is marked by setting the poll status variables⁴⁴ to the value READY_TO_VOTE.

The activation depends on the release key R_x and on the kinds of polls programmed⁴⁵:

Condition	Reaction
R_P (= 'P') pressed	all polls are activated
R_D (= 'D') pressed	polls of type Dail, Europe, and Local are activated
R_E (= 'E') pressed	polls of type Europe and Local are activated
R_L (= 'L') pressed	polls of type Local are activated
All other cases	no poll activated

All other keys have no effect in this sub-state⁴⁶.

⁴² Document *Detailed Software Design*, appendices A1 .. A3

⁴³ MB/vrijgave.c::init_wacht_op_vrijgave(), run_wacht_op_vrijgave()

⁴⁴ MB/functie.ph::verkiezing[].verk_status

⁴⁵ MB/vrijgave.c::vrijgeven_verk_type()

⁴⁶ Except the "Visual Impaired" button

The change from the sub-state "wait for release" to the sub-state "wait for key" (from voter) will be possible only if at least one of the poll status variables is set to READY_TO_VOTE. All subsequent sub-states including "store votes" depend on the successful execution of "wait for release."

If no release key is pressed, the sub-state "wait for release" will not be left.

Summary: The requirement is fulfilled. The sub-state "store votes" will be reached only if the sub-state "wait for release" has been successfully executed. This will be the case only if an appropriate release button on the Control Unit is pressed.

6.2-1 *The software shall not record votes, unless it has been activated.*

See requirement 6.1-1 above.

6.2-2 *The software shall count all activations.*

See requirement 2.3-2.

6.2-3 *After being activated, the software shall switch on LED's at top of those ballot papers on which the voter is entitled to vote.*

MB: When the sub-state "wait for release" has been successfully executed, the software changes to the sub-state "check status election."⁴⁷ This sub-state prepares the voting machine for the next voter. The preparation begins with a check whether the ballot module still has enough memory to save new votes. If there is enough memory, the displays will be prepared. For each poll set to READY_TO_VOTE, the LED at the top of the voter panel is switched on.

Summary: The requirement is fulfilled. After activation the LED's at the top of the activated ballot papers are switched on.

6.2-4 *After being activated, the software shall display the marks "-- --" to the right of those ballot papers on which the voter is entitled to vote.*

MB: After switching on the LED's at the top of each activated ballot paper the sub-state "check status election" also switches on the standby marks (stripes) for the same ballot papers. If the LED's at the top were not switched on successfully the state doesn't try to switch on the stripes.

Summary: The requirement is fulfilled. After switching on the LED's at the top of the activated ballot papers the standby marks (stripes) are also switched on.

6.2-5 *After being activated, the software shall display the preferences chosen by the voter.*

MB: When the sub-state "check status election" has been successfully executed, the software changes to the state "wait for key." This sub-state waits for preferences

⁴⁷ MB/election.c::init_check_status_election(), run_check_status_election()

chosen by the voter on the voter panel. Depending on the preferences already recorded and the pressed key, the software changes to another sub-state:

Non-referendum:

sub-state	voter action	new sub-state
wait for key ⁴⁸	the voter presses a new choice key for an activated poll	show order of preference ⁴⁹
	the voter presses a choice key already chosen for an activated poll	correct order of preference ⁵⁰
	the voter presses the CAST VOTE(S) button and at least one preference is already chosen	test for message ⁵¹

If the voter presses a choice key for an activated, non-referendum poll, the software will change to the sub-state "show order of preference." This sub-state first counts how many choices the voter has already selected for this poll. This number, increased by one, is then written on the LED near the new choice key. Then the sub-state writes key row and column, poll number and choice number into the voter input variable⁵². Each choice number is immediately protected by a Hamming code. In the end, the sub-state changes back to the sub-state "wait for key."

If the voter presses a choice key which is already contained in the voter input variable, the voter is considered to correct his/her choice, see requirement 6.2-8.

If the voter presses the CAST VOTE(S) button and at least one preference is selected, the software will change to the sub-state "test for message" (see requirements 6.2-9 and 6.2-10).

If the voter has selected all choice keys in non-referendum polls and one option key in each referendum poll, the software will change to "wait for CAST VOTE(S)."⁵³ This sub-state displays an appropriate message to the voter and waits for voter inputs. If the voter presses the CAST VOTE(S) button, the software will change to "store votes," otherwise it will change to one of the sub-states which correct preferences.

Summary: The requirement is fulfilled. If the voter selects a new choice button of a programmed and activated poll, the preference number will be shown on the LED near the selected key.

6.2-6 After being activated, the software shall display "x" in the left hand position of the LED chosen by the voter in the case of a referendum.

MB: When the sub-state "check status election" has been successfully executed, the software changes to the state "wait for key." This sub-state waits for preferences chosen by the voter on the voter panel. Depending on the preferences already recorded and the pressed key, the software changes to another sub-state:

⁴⁸ MB/election.c::init_wacht_op_toets(), run_wacht_op_toets()
⁴⁹ MB/election.c::init_toon_volgorde(), run_toon_volgorde()
⁵⁰ MB/election.c::init_herstel_volgorde(), run_herstel_volgorde()
⁵¹ MB/election.c::init_stem_gedrukt(), run_stem_gedrukt()
⁵² MB/election.c::keuze_array
⁵³ MB/election.c::init_wacht_op_stem(), run_wacht_op_stem()

Referendum:

sub-state	voter action	new sub-state
wait for key ⁴⁸	the voter presses an option key for an activated poll	show or change cross ⁵⁴
	the voter presses the CAST VOTE(S) button and at least one preference is already chosen	test for message ⁵¹
show or change cross	the option key is the first preference	(action:show cross) wait for key

If the voter presses an option key of a referendum poll, the software will change to the sub-state "show or change cross." This sub-state first checks whether or not the voter has already selected an option in this poll. If it is the first selection, the sub-state writes a cross "X" on the LED near the option key. Then the key row, key column, poll number and option number are written into the voter input variable⁵². The option number is immediately protected by a Hamming code. In the end, the sub-state changes back to the sub-state "wait for key."

If the voter had already selected an option key in this poll he/she is considered to correct his/her choice, see requirement 6.2-8.

If the voter presses the CAST VOTE(S) button and at least one preference is selected, the software will change to the sub-state "test for message" (see requirements 6.2-9 and 6.2-10).

If the voter has selected all choice keys in non-referendum polls and one option key in each referendum poll, the software will change to "wait for CAST VOTE(S)."⁵³ This sub-state displays an appropriate message to the voter and waits for voter inputs. If the voter presses the CAST VOTE(S) button, the software will change to "store votes," otherwise it will change to one of the sub-states which correct preferences.

Summary: The requirement is fulfilled. If the voter selects a new option button of a programmed and activated referendum, a cross will be shown on the LED near the selected key.

6.2-7 When a preference is recorded, the software shall display details on the bottom line of the voting machine display screen.

MB: Each time the voter has chosen a choice number for the first time the details programmed for this choice number are read out of the layout memory of the ballot module, put into the fourth line of the display buffer and then shown on the voting machine display. The details will be deleted again if the voter corrects the preference.

In case the "Visual impaired device" is switched on by the polling staff no details are shown on the voting machine. Instead of the candidate details the string "Device for visually impaired active" appears.

⁵⁴ MB/election.c::init_toon_kruis(), run_toon_kruis()

The display of candidate details can be suppressed by programming a special option (KAND_TONEN) when the ballot modules are programmed in the polling centre.

Summary: The requirement is fulfilled. The details of the preference chosen are displayed on the fourth line of the voting machine display except the "Visually impaired device" is activated or a special option is programmed into the ballot module.

6.2-8 The software shall allow to change preferences by pressing one of the selected candidates or referendum choices a second time. The replaced preferences shall not be stored permanently.

MB: If the software is in the sub-state "wait for key" and the voter presses a choice key of a non-referendum which is already contained in the voter input variable, the voter will be considered to correct his/her choice. The same will apply if he/she presses an option key in a referendum which already contains a preference of the voter. The software reacts with actions and state changes as follows:

Non-referendum:

sub-state	voter action	software action	new sub-state
wait for key ⁴⁸	the voter presses a choice key a second time		correct order of preference ⁵⁰
correct order of preference		delete the pref and all subsequent	wait for key

If the voter presses a choice key a second time, this choice key and all keys which were subsequently selected are deleted. Then the software changes again to the sub-state "wait for key" and waits for the next voter input.

Referendum:

sub-state	voter action	software action	new sub-state
wait for key ⁴⁸	the voter presses a second option key		show or change cross ⁵⁴
show or change cross	the first and the second key are the same	delete the cross	wait for key
	the second key is another than the first	delete the first cross, set a new cross	wait for key

If the voter presses an option key a second time, this option is deleted. If he/she presses a second option key the old selection is deleted and the new one is stored into the voter input variable. Then the software changes again to the sub-state "wait for key" and waits for the next voter input.

All variables which are used to store the voters inputs are resetted, when the voter changes his/her preferences.

Summary: The requirement is fulfilled. The software allows preferences to be changed by once again pressing a key. The previous preference numbers are not stored permanently.

6.2-9 *The software shall allow to record the vote, if at least one preference on one ballot paper has been recorded.*

MB: The sub-state "wait for key" may also accept the CAST VOTE(S) button. This button will be accepted only if there is at least one selected preference in one of the activated polls. Otherwise the button will have no effect.

If the CAST VOTE(S) button is pressed and at least one choice has already been selected, the software will change to the sub-states "test for message,"⁵¹ "wait for acknowledge,"⁵⁵ and "store votes."⁵⁶

Summary: The requirement is fulfilled. The software will allow the CAST VOTE(S) button to be pressed if at least one preference in one of the activated polls is chosen.

6.2-10 *If there are ballot papers without recorded preferences, the software shall record the vote after displaying a reminder and receiving a confirmation.*

MB: The sub-state "test for message"⁵¹ first calls a function⁵⁷ which checks all activated polls for the number of selections. If there are polls without any selection, the software will change to the sub-state "wait for acknowledge."⁵⁵ This sub-state displays an appropriate message (which was programmed in the polling centre) and waits for the voter input. If the voter presses once again the CAST VOTE(S) button, the software will change to "store votes."⁵⁶ If the voter presses a choice key for an activated poll, the software will change back to the sub-states "show order of preference," "correct order of preference" or "show or change cross."

If there are no polls without selected preferences, the software will immediately change from the sub-state "test for message" to "store votes."

Summary: The requirement is fulfilled. If there are polls without any selected preference, the software will display an appropriate message and wait for the CAST VOTE(S) button to be pressed again.

6.2-11 *The software shall record the vote, when the voter has pressed the „CAST VOTE(S)“ button or confirmed the button pressing. The process shall be performed without any interruption.*

MB: If the voter has pressed the CAST VOTE(S) button or has confirmed it, the software will change to the sub-state "store votes."⁵⁶ This sub-state first switches off all LED's on the voter panel and displays an appropriate message on the voting machine display. Then the voter input variable is copied to a new vote structure and completed

⁵⁵ MB/election.c::init_wacht_op_bev(), run_wacht_op_bev()

⁵⁶ MB/election.c::init_wegschrijven_stem(), run_wegschrijven_stem()

⁵⁷ MB/election.c::melding()

by poll numbers, null votes, and start/end markers. For all data the Hamming code is kept and added, respectively.

When this is done, the votes are stored⁵⁸ and the software changes to the sub-states "check all stored votes" and "wait for continue." The sub-state "check all stored votes" reads and validates all votes including the new votes. The sub-state "wait for continue" changes the poll status variables from READY_TO_VOTE to VOTED and thus deactivates the voting machine.

The saving process cannot be interrupted by the voter or the polling staff. An analysis of all program steps between pushing the CAST VOTE(S) button and storing the votes shows that the storing process is inhibited or interrupted only in case of major hardware faults. These include:

- the RAM memory which holds the variables is defective,
- failure of read or write accesses to the ballot module,
- the data and address lines between ballot module and voting machine are defective,
- the ballot module contents are inconsistent,
- one or both of the two memory chips of the ballot module has been lost,
- the error and event memory or the displays are defective,
- the CAST VOTE(S) button is not identified.

These hardware components are necessary to store the vote or to maintain the correct operation of the voting machine.

All hardware failures lead to a program shut down as soon as they occur. They will inhibit vote storage after pushing of the CAST VOTE(S) button only in those cases, where the failure event occurs in a very short time-period after the button has been pushed. If they occur earlier, the voting machine will be stopped before the voter pushes the CAST VOTE(S) button.

Summary: The requirement is fulfilled. The software stores the vote when the voter presses the CAST VOTE(S) button or confirms it. The saving process cannot be interrupted by the voter or the polling staff. It can be interrupted only by major hardware faults.

6.2-12 The vote shall be recorded only once.

MB: The variable holding all voter inputs is copied to a new vote structure⁵⁹ and then transferred to the store function⁵⁸. The variable is copied once only, and the store function is also called once only.

The store function saves the vote structure once. Each vote byte of the vote structure is saved four times in four different places of the ballot module. The four copies together form a valid vote byte when the vote is read in again.

Summary: The requirement is fulfilled. Each vote is recorded once only.

⁵⁸ MB/stemprog.c::schrijf_stemmen()

⁵⁹ MB/election.c::run_wegschrijven_stem()::stem_array[]

6.2-13 The software shall count all de-activations.

See requirement 2.3-4.

6.2-14 The software shall record the vote, when the voter has pressed the CAST VOTE(S) button or confirmed the button pressing. The process shall be performed without interruption.

See requirement 6.2-11.

6.3-1 After each vote is stored, the voters panel shall be deactivated automatically. The software will not accept any further votes until a member of the polling staff, operating the Control Unit, reactivates the software. The activation shall not be possible by means of the voting machine.

MB: The sub-state "store votes"⁵⁶ implements vote storage. When the votes are stored, the software changes unconditionally to the sub-states "check all stored votes"⁶⁰ and "wait for continue."⁶¹ The sub-state "wait for continue" switches the poll status variables⁴⁴ for all activated polls from READY_TO_VOTE (activated) to VOTED (inactive) (see requirement 6.1-1). So there are no longer any activated polls. Then the software changes to the sub-states "check status release"³⁶ and "wait for release."⁴³

As long as all poll status variables are set to VOTED, the keys of the voting machine have no effect. The only keys that are active are the release keys R_x and the "Visual Impaired" button. The polling staff may also change the main state (mode) from Election to Standby or Functions by operating the function keyswitch and/or the function button on the control unit.

The release keys on the Control Unit are the only way to release polls for voting.

Summary: The requirement is fulfilled. After having stored the votes, the software automatically deactivates all programmed polls and changes to the state "wait for release." Only the release keys, the "Visual Impaired" button, the function keyswitch and the function button on the Control Unit are active. All keys of the voting machine are inactive.

6.4-1 The software should offer a way to withdraw the activation by means of the Control Unit. Any preferences entered on the voting machine should not be recorded as a vote in this case.

MB: The states "wait for release," "wait for key," "show order of preference" etc. are sub-states of the mode (main state) Election. Changes of the modes have a higher priority than changes between several sub-states of Election. The modes Service, Functions, and Standby have a higher priority than the mode Election³⁸. So the polling staff may switch from Election to Standby at any time by resetting the function keyswitch on the Control Unit. There is only one exception: During vote storage the keyswitch is not checked and so a turn or withdrawal has no effect on the vote storage. If the storage is completed, the keyswitch will be checked again and the mode will be changed to Standby.

⁶⁰ MB/election.c::init_controleer_stemmen(), run_controleer_stemmen()

⁶¹ MB/election.c::init_wacht_op_doorgaan(), run_wacht_op_doorgaan()

If the mode Standby is entered, all poll status variables⁴⁴ will be set to VOTED (deactivated). So the voting process of the current voter is not only interrupted but also aborted. If the polling staff again switches to Election, the voter has to start anew.

The voter input variable⁵² is accessible only for the Election sub-states. Outside Election it is unknown. Inside Election it is saved permanently only in the sub-state "store votes." If this sub-state is not entered, the inputs of the voter will be lost. If the voter starts anew, the input variable will be reset to zero.

Summary: The requirement is fulfilled. The activation of polls can be withdrawn by changing to the mode Standby before the voter pushes the CAST VOTE(S) button. In this case all voter inputs are lost.

6.4-2 The software shall count all de-activations.

See requirement 2.3-4.

6.5-1 The software shall not provide information about any preferences recorded or vote cast by a voter, after the voter has pressed the „CAST VOTE(S)“ button.

MB: Any voter input can be traced on its way through the software. So all possible outputs of vote information can be found. The analysis yields the following trace: If the voter has pressed a key, the key row and column will be recorded⁶². With the aid of the layout memory in the ballot module, the associated poll number, choice number, and candidate details are evaluated⁶³. All these key and choice variables and texts are used only to fill the voter input variable⁵² and the fourth line of the voting machine display.

The voter input variable is used for the following purposes only:

- it collects the voter inputs (row, column, poll number, choice number),
 - it provides the preference number shown on the LED,
 - it provides the information as to whether the choice is selected the first or the second time,
 - it is used to send information about the pressed key to the COM2 port if the "Visual Impaired" function is switched on,
 - it is copied into the new vote structure⁵⁹ which is transferred to the storage function.
- There are no other accesses to the voter input variable or to the new vote structure.

As soon as the voter pushes the CAST VOTE(S) button, the LED's near the choice keys and at the top of the ballot papers are all switched off⁶⁴, the voting machine display is wiped out⁶⁵ and displays a new text, the COM2 port is no longer in use and the vote is stored in the ballot module.

⁶² MB/stem_comp.c::main()::KP_toets, functie.c::functie()::toets_info, vrijgave.c::<>:toets, election.c::<>:toets

⁶³ MB/election.c::<>:keuze, election.c::glob_keuze

⁶⁴ MB/election.c::init_wegschrijven_stem(), kies_pan.c::vp_alles_uit()

⁶⁵ MB/election.c::init_wegschrijven_stem(), screen_func.c::display_screen(), display.c::clear_display()

Summary: The requirement is fulfilled. As soon as the voter pushes the CAST VOTE(S) button, the software provides no longer information about the preferences chosen or the vote stored in the ballot module.

6.5-2 *The software shall not provide information about any preferences recorded or vote cast by a voter to the display of the Control Unit during the election process.*

MB: Tracing of the voter input shows that no information is transferred to the Control Unit (see requirement 6.5-1).

Summary: The requirement is fulfilled. No information about the preferences chosen or the vote cast is transferred to the Control Unit.

6.6-1 *The software shall display the total number of votes including null votes on the Control Unit. The software shall display the number of votes including null votes on the voting machine in stand-by modus.*

See requirement 2.3-3 for the counting of votes and voters.

The number of voters for each poll P_i (or less, if less than five polls are programmed) are permanently displayed on the Control Unit during the voting process. They will also appear on the voting machine display if the status is changed from ELECTION to STANDBY and when the ballot module is ok.

Summary: The requirement is fulfilled. The number of voters per poll (including null votes) is permanently displayed on the Control Unit and is displayed on the voting machine in stand-by modus.

7. Displaying and printing the total number of votes at close of poll

7.1-1 *The software shall display the number of votes and the number of null votes on the voting machine at close of poll.*

See requirement 2.3-3 for the counting of votes and voters.

The number of voters per poll P_i (or less, if less than five polls are programmed) is displayed on the voting machine when the poll is closed and the voting results are evaluated (Functions / Close poll / Make back up - Show data on display). This display does not include the separate number of null votes per poll N_i .

Summary: The requirement is fulfilled. The number of voters per poll (including null votes) is displayed on the voting machine at close of poll.

7.1-2 *The software shall print the number of votes and the number of null votes at close of poll.*

See requirement 2.3-3 for the counting of votes and voters.

The number of votes per poll P_i (or less, if less than five polls are programmed) and the number of null votes per poll N_i are part of the voting result printout (Functions / Close poll / Make back up and print statement).

Summary: The requirement is fulfilled. The number of voters per poll (including null votes) and the number of null votes per poll is printed at close of poll.

7.1-3 The software shall print the total number of activations of the voting machine before and after polling.

See requirement 2.3-2.

7.1-4 The software shall print the total number of de-activations of the voting machine before and after polling.

See requirement 2.3-4.

7.2-1 Activation of the software for the display and printing of the information referred to in paragraph 7.1 shall be accomplished by means of the key referred to in paragraph 5.2. Withdrawal of the key from the Control Unit should automatically stop the printing process.

MB: The close of poll and the output of the voting results (number of voters and number of null votes) are sub-states of the mode (main state) Functions.

An analysis of all conditions which must be fulfilled to change to the mode Functions yields the following list³⁸:

- the run time tests have been executed successfully, and
- the voting machine is not in the mode Service (the service switch is not set and the ballot module is not of type "service"), and
- the voting machine is not in the mode Standby (the function keyswitch FS is not in key position 1), and
- the function keyswitch FS is turned to position 2 with the function button on the Control Unit being pressed.

To reach the sub-state Functions / Close poll, several additional conditions must be met (availability of ballot module and backup module, ...).

So the voting results will be typed out only if the function keyswitch FS is turned to position 2 together with the function button on the Control Unit being pressed. If the function keyswitch is in position 1, the software will immediately return to the mode Standby.

The key itself can be withdrawn only in its position 1 (mode Standby).

The position of the function keyswitch FS is tested in the main loop of the software. When the voting results are displayed, the main loop is permanently cycled. When the voting results are printed, the main loop is cycled at short intervals (after each printed line).

Regardless of the abortion, the voting is considered to be closed. Voting is no longer possible.

Summary: The requirement is fulfilled. The voting result evaluation will be possible only if the function keyswitch is in key position 2 and the function button has been pressed. If the key is reset (and withdrawn), the software will abort the voting result evaluation immediately.

7.2-2 *The back-up module shall be checked to ensure that any existing data in the module are deleted, before the contents of the primary ballot module are copied to the back up module.*

MB: The voting results can be printed (Functions / Close poll / Make back up and print statement) or displayed (Functions / Close poll / Make back up - Show data on display). The sub-states which implement these functions⁶⁶ check whether or not the voting results have already been called. If they have not yet been called, a special flag⁶⁷ indicating the evaluation is set into the ballot module. This prevents further voting. Then the complete backup module is erased⁶⁸ and the primary ballot module is copied to the backup module⁶⁹. The last two steps are unconditionally executed one after the other.

The backup will be made only if the voting results are evaluated at the voting machine (using Functions / Close poll / ...). It won't be made if the (first) evaluation is made via PC.

The backup will be made only if there are votes in the primary ballot module.

Summary: The requirement is fulfilled. The backup module is automatically erased before the backup is made.

7.3-1 *After displaying or printing the total number of votes cast etc, the software shall accept new activation only after the insertion of a new programmed ballot module.*

MB: If the polling staff selects the functions which print or display the voting results (Functions / Close poll / ...) or ask the voting results via PC, a flag will be set into the ballot module⁶⁷. The flag is not erasable without the complete ballot module being erased. The flag is only set if there are votes in the primary ballot module.

The initial sub-state of main state Election, "check status release"³⁶ will read this flag from the ballot module and display an appropriate message if the flag has already been set. In this case, the sub-state is maintained and a change to those sub-states which accept release keys is not possible. So the voting machine may not be used for activation and voting.

The polling staff may only change from Election to Standby or Functions. It also may use a newly programmed ballot module.

⁶⁶ MB/toon_uitslag.c::init_beheer_uit(), init_beheer_display()

⁶⁷ Flag at MOD_UITSLAG_ADR

⁶⁸ MB/program.c::wis_stem_geh()

⁶⁹ MB/program.c::maak_backup()

Summary: The requirement is fulfilled. After displaying or printing of the voting results, voting is no longer possible with the ballot module. The voting machine may be used for voting only, if a new ballot module is inserted.

8. Reliability and security of the voting machine

8.1-1 *The software shall record the vote, when the voter has pressed the „CAST VOTE(S)“ button or confirmed the button pressing. The process shall be performed without any interruption.*

See requirement 6.2-11.

8.1-2 *The vote shall be recorded only once.*

See requirement 6.2-12.

8.1-3 *The software shall not record other information than the vote, when the voter has pressed the „CAST VOTE(S)“ button or confirmed the button pressing.*

MB: The keys pressed by the voter and the associated poll numbers and choice numbers are entered in a voter input variable⁵². This variable is a two-dimensional array containing all selected choices for each activated poll. The variable is updated as soon as the voter selects new preferences or corrects his/her preferences.

If the voter presses or confirms the CAST VOTE(S) button, the voter input variable will be transferred to a new vote structure⁵⁹. During this transfer, the two-dimensional array is changed into a one-dimensional array. The poll number is added to each poll list and then the poll lists are concatenated. In poll lists without any preference a null vote is added. Furthermore, a start and an end marker are added. So the vote consists of a row of data bytes, each with a Hammingcode.

The new vote structure is transferred to the store function⁵⁸. This function duplicates the vote twice and inverts two of the copies. So the vote is saved four times.

Other modifications to the vote are not made.

When the first voter of the day presses or confirms the CAST VOTE(S) button, the identification of the voting machine is written into the module information area of the ballot module. So it is always known on which voting machine the ballot module was used for voting.

Except for the votes of all voters and the voting machine identification, no information is written into the ballot module during the voting process.

Summary: The requirement is fulfilled. The votes of all voters are saved in the ballot module without relevant modifications. In addition to the votes, only the voting machine identification is written into the ballot module during the voting process.

8.2-1 A cast vote must not be lost by a power failure, the failing of one component, through normal use or through failures in the operation of the voting machine.

Failure of a hardware component:

MB: The hardware of the voting machine is permanently checked¹⁷. If hardware components are faulty, the voting process is immediately interrupted or aborted. In case of minor problems (e.g. key pressed too long) the operation is interrupted but the voter can continue as soon as the problem is solved. In all other cases the operation is aborted and all preferences already selected by the voter will be lost. After switching off and on in most cases the voting machine will work again and the voter may repeat his/her voting.

Several hardware failures may even inhibit vote storage if the voter has already pressed the CAST VOTE(S) button. These are failures of hardware components which are necessary to save the vote or to maintain the correct operation of the voting machine, like the ballot module, the RAM memory, etc. (see requirement 6.2-11).

Normal use or failures in operation:

MB: The main board software is implemented as a set of main states (modes), sub-states and state transitions. Each state and sub-state is only capable of reacting to a certain, well-defined set of keys. Keys of this set initiate a state transition, while all other keys have no effect. So the possible ways of faulty operation are reduced to a minimum.

The voter may correct his/her choices as long as he/she has not pressed the CAST VOTE(S) button. He/she is not allowed to select too less or too much preferences. He/she is not allowed to select keys which belong to no poll or keys which belong to a poll which is not activated.

The polling staff may correct a wrong release key by means of the function key FS on the Control Unit. (This is counted as a de-activation of the voting machine.) It may also use the function key to abort the voting process for a voter at any time as long as the voter has not pressed the CAST VOTE(S) button. In this case the preferences already selected by the voter are lost and the de-activation is counted.

In case the voter has pressed the CAST VOTE(S) button, all keys of the voting machine and the Control Unit have no effect.

Power supply failure:

MB: A power supply failure during the voting process aborts normal program execution. All preferences already selected by the voter are lost.

A power supply failure during vote storage triggers a special interrupt routine⁷⁰ which changes the normal vote storage to an auxiliary process which is much faster. The vote is not written into the ballot module but into the error and event memory (Xicor).

⁷⁰ MB/stemprog.c::level3_int_routine()

When the power returns and the voting machine starts again, the vote is transferred from Xicor to the ballot module⁷¹. The data in Xicor are secured against modifications. The restoring function works in the same way as the original store function.

The vote will be transferred only if the module has not been exchanged, the voting results have not been asked, and the data temporarily saved in Xicor have not been modified in between.

Summary: The requirement is fulfilled. A cast vote is always stored except in the following remote and abnormal circumstances: failure of a hardware component necessary to save the vote or to maintain correct operation of the voting machine; power failure combined with a defect of the Xicor.

8.2-2 In case of a failure during election process the software shall provide information, whether the vote of the current voter has been recorded.

MB: During the mode Election, the poll names, the poll statuses and the number of voters per poll are permanently displayed on the Control Unit⁷². The poll statuses and voter numbers are correctly evaluated and promptly updated. The poll statuses change from "Standby" to "Open" (release of a poll), to "Store" (start of vote storage), to "Check" (end of storage, start of memory validation), and back to "Standby" (end of memory validation). The number of voters is increased by one for each released poll as soon as the votes have successfully been stored⁵⁶.

If the voting machine is switched off and on, the status for each poll is VOTED (shown as "Standby") and the number of voters is re-evaluated by counting the votes in the ballot module.

The polling staff may watch the number of voters shown on the Control Unit. If this number increases by one, the vote of the voter is definitely saved.

This is also valid for power failures. If the voter has already pressed the CAST VOTE(S) button and the number of voters increases by one for each released poll, his/her vote has been saved. If the number of voters does not increase, the vote may be saved in the Xicor memory. In this case, the number of voters will increase as soon as the power returns and the vote is transferred from Xicor to the ballot module.

There is no special plain-text message informing about the vote storage success.

Summary: The requirement is fulfilled. The display of poll status and the number of voters for each poll on the Control Unit is correct and up-to-date. As soon as the last vote is stored, the number of voters is increased by one for each released poll. If the number of voters increases after power on, the last vote has been temporarily saved in the error and event memory and has then been transferred to the ballot module.

⁷¹ MB/stemprog.c::check_power_down_stem(), juiste_stem(), herstel_stem()

⁷² MB/vrijgave.c::update_elections(), display_elections(); MB/election.c::update_elections(), display_elections()

8.3-1 The installed ballot module and its contents must be fully maintained in case of a power failure, through normal use or through failures in the operation of the voting machine.

Normal use or failures in operation:

MB: As mentioned in requirement 1.4-2, the programming of new information in the ballot module requires a certain set of well-defined programming statements. Any deviation from the required sequence of statements and any deviation from the required bit sets of the statements lead to the program being terminated. So the writing of new information may take place only by statements programmed for that purpose. An analysis of all accesses to the ballot module shows that only the store function⁵⁸ writes into the ballot module during the voting process. This store function is called after a certain, well-defined set of state transitions at the end of the voting process. In case of faulty operation, these state transitions do not take place.

The store function checks the validity of the vote in RAM and the validity of the destination address in the memory before vote storage is performed. The storage process will be aborted if the assigned storage space is already occupied.

Furthermore, physical measures inhibit deletion or complete overwriting of votes⁷³. Since a certain data line is lacking, the erase statement does not reach the ballot module. Certain bit changes are impossible. So in case a write access would overwrite a previous vote, this process would normally stop during the first few write accesses. In this case, the previous vote would remain countable due to its redundancy.

Power supply failure:

MB: A power supply failure also leads to a loss of programming voltage. Since in the ballot module the votes are stored permanently, power supply or programming voltage failures have no influence on the contents.

Summary: The requirement is fulfilled. The contents of the ballot module are fully maintained in case of power failure or normal or faulty operation of the voting machine. The program is not capable of completely overwriting or erasing votes. In case of normal operation, the program may add new votes.

8.4-1 The functions of the software must be fully maintained in the event of a power failure.

In case of power failure, a 12 V battery may be used to run the voting machine⁷⁴. According to the documentation, all functions of the voting machine are preserved in this case, except the use of the internal printer⁷⁵.

MB: Nearly all data can be both printed and displayed. The only exceptions are:

- The (separate) numbers of null votes for each poll. These numbers are part of the voting result printout and cannot be displayed.
- The constituency. This date is also available without using the voting machine.

⁷³ Document *Reliability of the Voting Machine ESI1*, p. 7 par. 3.5

⁷⁴ Document *Functional Specification* version 1.9, p. 5 par. 1.3

⁷⁵ Document *Reliability of the Voting Machine ESI1*, p. 6 par. 3.1.1

- The list of election types the voting machine is able to handle. This information is used for protocol purposes only.

Summary: The requirement is fulfilled. In case of power failure, a 12 V battery may be used. In this case all functions of the voting machine are preserved, except the use of the internal printer. Hence data which are printed but not displayed are no longer accessible (e.g. null votes per poll).

8.5-1 *The storing of votes in the ballot module must be made in such a way so as to ensure security and continuous self-checking of all data. Each vote should be stored twice in each of 2 independent IC's within the ballot module.*

MB: The store function⁵⁸ gets the new vote structure and duplicates it twice. Two of the four copies are inverted. Each of the independent memory chips of the ballot module gets a normal and an inverted copy of the vote. All four copies are included, when the vote is read out and validated.

Summary: The requirement is fulfilled. Each vote is stored twice in each of the two independent memory chips of the ballot module.

8.5-2 *As each single preference is recorded, a checksum shall be calculated and stored in RAM.*

MB: The voter may select his/her preferences in several sub-states of the mode Election. Each time a choice or option key is identified, a state transition takes place. The following sub-states^{49,54} transform the key into a choice or option and add a Hamming code. In the end⁵⁶ poll numbers and null votes are added, where appropriate. These numbers are also assigned to a Hamming code.

Summary: The requirement is fulfilled. Each single preference held in RAM is protected by a Hamming code.

8.5-3 *When the voter presses the „CAST VOTE(S)“ button, all checksums of all preferences stored in RAM shall be checked.*

MB: When the voter presses, or confirms operation of, the CAST VOTE(S) button, the preferences, poll numbers, and null votes are converted into a new vote structure which maintains the Hamming codes⁵⁹. The new structure then is transferred to the store function⁵⁸ which checks all Hamming codes for validity before the vote is saved.

Summary: The requirement is fulfilled. When the voter presses or confirms operation of the CAST VOTE(S) button, all Hamming codes in RAM are checked.

8.5-4 *As the voting machine stores preferences, a read back of the last byte of data written should be made.*

MB: The store function⁵⁸ uses a subroutine to save a block of data bytes⁷⁶. This subroutine writes byte after byte. The result is checked after each byte. If the write statement is not successful, it will be repeated up to 24 times.

Summary: The requirement is fulfilled. Each byte of the vote is read back and checked during storage.

8.5-5 *After storing all preferences, the system should check all of these again and their checksums.*

MB: After successful storage of all four copies of the vote, the current memory space is checked to see whether it is empty⁵⁸. If it is not empty the new vote is read and validated using the standard collect-and-validate functions²⁶ of the software. The validation includes a check of all Hamming codes.

Summary: The requirement is fulfilled. After storage of the complete vote, it is read back and validated. The validation includes a Hammingcode check.

8.5-6 *After storing and checking all preferences of the current voter, the software shall check all preferences stored from all voters.*

MB: When the storage process is completed, the program changes to the next sub-state "check all stored votes"⁶⁰. This sub-state evaluates the number of voters by reading, validating and counting all votes in the ballot module^{26,28}. The new sum is compared to the old number of votes (before the current voter has begun).

Summary: The requirement is fulfilled. After storage and checking of the new vote, all votes of all voters are read, validated and counted.

8.5-7 *The software shall display an error message on the voting machine display and on the Control Unit display, if a vote could not be stored.*

MB: Any problem during storage and checking leads to an error code being entered in the hardware status table⁹. The status table is checked by the run time test routine¹⁷ which is executed immediately after the storing process and after the ballot module check. The run time test routine writes the error code into the error and event memory (Xicor), displays the error code on voting machine and Control Unit, and waits for confirmation by the polling staff. After confirmation, the voting machine is shut down. With the exception of power supply problems, all other problems lead to the voting machine being blocked.

The error codes are two- to five-digit numbers. The help desk staff will be able to interpret these numbers with the help of an error codes list delivered by the manufacturer. The manual for the polling staff contains a short list of those error codes which indicate that a vote could not be stored.

The error codes are not only displayed but also permanently saved in Xicor (supplemented by a time stamp). The interpretation of the Xicor memory should be

⁷⁶ MB/stemprog.c::progr_stem()

effected (or at least supported) by the manufacturer, since a service module is necessary.

Summary: The requirement is fulfilled. The software displays an error message (error code) if a vote could not be stored. The manual for the polling staff contains a list of all error codes which indicate that a vote could not be stored. Besides a full list of all error codes will be available for the help desk staff.

8.5-8 *In the event that a discrepancy occurs in the checksums, an error message should be generated. This message should be shown on the voting machine display and on the Control Unit display.*

This is a special case of requirement 8.5-7, see there.

Summary: The requirement is fulfilled. In case of checksum discrepancies an error message (error code) is displayed on the voting machine and on the Control Unit.

8.6-1 *The software should, as far as is reasonably and technically possible, avoid or restrict the possibilities of accidental or intentional incorrect use.*

MB: The software is implemented as a system of modes (main states), sub-states, and state transitions (see introduction to section 6). The narrowed, well-defined effect of keys and buttons widely prevents the voting machine from being incorrectly used.

The following cases of incorrect operation are conceivable:

Incorrect (unintended) operation	Consequences
Voter selects a wrong choice button	No; voter may correct the choice
Voter presses the CAST VOTE(S) button too early; no preferences selected	No; CAST VOTE(S) button has no effect
Voter presses the CAST VOTE(S) button too early; there are still polls without selected preferences	No; voter may continue by not confirming the CAST VOTE(S) button
Voter presses the CAST VOTE(S) button too early; there are still choices unselected	Yes; voter cannot correct this step. He/she casts his/her vote definitely.
Voter presses choice or option buttons for polls which are not released	No; buttons have no effect
Voter presses buttons which do not belong to a poll	No; buttons have no effect
Voter presses buttons of the telephone keypad	No; buttons have no effect
Polling staff selects a wrong release key	No; polling staff may correct the choice by switching the machine to mode Standby and back to Election. This is counted as a de-activation only

Polling staff does not select a release key	No; voting process stops until release key is pressed
Polling staff selects a second release key	No; buttons have no effect when the polls are already released
Polling staff presses function button	No; button has no effect in the mode Election
Polling staff turns function keyswitch FS back to key position 1 before voter is ready	Yes; machine switches to mode Standby; preferences of the current voter are lost
Polling staff turns function keyswitch FS back to key position 1 when voter is ready (pressed CAST VOTE(S))	No; machine switches to mode Standby after storing all preferences of the current voter

When the voter accidentally presses the CAST VOTE(S) button before he/she has selected all preferences he/she intended to select, the vote is saved and the voter is not allowed to continue. The wrong operation is not reversible.

The accidental reset of the function keyswitch FS seems to be the most likely incorrect (unintended) use of the voting machine. If a member of the polling staff resets this keyswitch before the voter pressed the CAST VOTE(S) button, the preferences of the current voter will be lost. He/she has to begin anew.

In both cases, there are no technical means allowing incorrect use to be avoided.

Summary: The requirement is fulfilled. The software restricts the possibilities of accidental or intentional incorrect use.

8.8-1 The votes should be stored randomly in the ballot module.

MB: The votes are stored in the vote memory of the ballot module using a timer-based method. For the first vote a place is selected randomly⁷⁷ somewhere in the middle of the vote memory. The following votes are stored in front of or after the votes stored previously. The decision between beginning and end is also made at random⁷⁸.

The timer used to generate the random place and the random decision is initialised during start-up tests⁷⁹ and then increased every 8 msec. When the first vote is stored, the timer is read out to select the place. The timer can be thought of as a "pointer" pointing to the middle part of the vote memory. It points to the beginning of the middle part when the machine is started and advances to the next memory place every 8 msec. It reaches the end after 16 sec and starts again at the beginning.

To identify the first vote, the time difference between start and readout of the timer must be known with an uncertainty of a few msec. The start-of-timer point is not equal to the switch-on of the voting machine, and the readout point is not equal to the pressing of the CAST VOTE(S) button. Both points are not detectable for an observer.

Knowledge of the storing method is no help in identifying the first vote.

⁷⁷ MB/stemprog.c::genereer_random_start()

⁷⁸ MB/stemprog.c::genereer_offset()

⁷⁹ MB/init_diag.c::init_diag(), timer.c::init_timer()

Since the first vote is unknown, all votes cannot be associated to certain voters.

Summary: The requirement is fulfilled. The votes are randomly stored. They are stored so that nobody will be able to assign certain votes to certain voters. Knowledge of the storing method does not help in identifying certain votes.

8.8-2 *The software shall not provide information about any preferences recorded or vote cast by a voter, after the voter has pressed the „CAST VOTE(S)“ button.*

See requirement 6.5-1.

8.8-3 *The software shall not provide information about any preferences recorded or vote cast by a voter to the display of the Control Unit during the election process.*

See requirement 6.5-2.

9. Operability

9.1-1 *The software shall perform no other actions than described in requirement 6.2, when the voter is acting on the voting machine.*

MB: The Election modules⁸⁰ contain only functions for the Election sub-states, help functions (display functions, test functions, functions to operate the "Visual Impaired" COM port, assignments to change from English to Gaelic and back), and the state tables. The modules comply with the state transition diagrams in the documentation⁸¹ (irrespective of some small documentation problems).

As long as the polling staff does not change from mode Election to another mode, only changes between the Election sub-states are possible.

Summary: The requirement is fulfilled. The software changes between Election sub-states as long as the mode (main state) Election remains unchanged.

9.3-1 *The activation and de-activation for each voter must lead to a visible, audible or tangible feedback signal.*

MB: The status changes of the polls are displayed on the Control Unit. The default status of polls is VOTED, shown as "Standby" on the Control Unit display. When the polling staff pushes one of the release keys, the sub-state "wait for release"⁴³ checks which of the polls should be released and changes their status to READY_TO_VOTE. Then it changes to the sub-state "confirm release"⁸² which updates the display for these polls from "Standby" to "Open." So as soon as the polls are released, their status strings displayed on the Control Unit are changed from "Standby" to "Open."

The sub-state "store votes"⁵⁶ changes these status strings to "Store" for all released polls immediately before the store function is called. The next sub-state "check all

⁸⁰ MB/vrijgave.c, election.c

⁸¹ Document *Detailed Software Design*, appendices A1 .. A3

⁸² MB/vrijgave.c::init_bevestig_vrijgave(), update_elections(), display_elections()

stored votes"⁶⁰ changes these strings to "Check" for all released polls. The last sub-state "wait for continue"⁶¹ changes them back to "Standby" and again sets all poll statuses to VOTED. So as soon as the polls are de-activated, their status strings displayed on the Control Unit are changed back to "Standby."

Summary: The requirement is fulfilled. The status of polls is visible on the Control Unit. A released poll has the "Open," "Store" or "Check" status strings. A de-activated poll has a "Standby" status string. The status strings are changed promptly.

9.3-2 *The selection of preferences by the voter must lead to a visible, audible or tangible feedback signal.*

MB: The software contains a key test function⁸³ which checks a key pressed by the voter. This function produces a standard (short) beep in case of success. So the voter gets an audible feedback when he/she presses a choice or option key belonging to a programmed and released poll. All other keys on the voting machine except the language button English/Gaelic do not produce a beep.

The beep of the voter panel keys may be suppressed when a special option⁸⁴ is programmed into the ballot module at the polling centre.

Furthermore, the sub-states "show order of preferences," "correct order of preferences," and "show or change cross" call a function⁸⁵ which changes the LED's near the selected choice or option keys. They are immediately toggled between stripes and preference number and between stripes and cross, respectively. So they produce an immediate visible feedback. Besides the details of the choice or option key are displayed as long as this display function is not suppressed by a special option in the ballot module.

Summary: The requirement is fulfilled. The voter gets an audible and a visible feedback if he/she selects a choice or option key for a programmed and released poll.

9.3-3 *The pressing of the „CAST VOTE(S)“ button by the voter must lead to a visible, audible or tangible feedback signal.*

MB: The sub-states of Election which accept the CAST VOTE(S) button produce a long beep or a very long beep. The long beep is produced if the CAST VOTE(S) button is pressed and the voter has selected at least one preference for each released poll. The very long beep is produced if the CAST VOTE(S) button is pressed and there are released polls without preferences. In the second case the voter has to confirm the button pressing.

The long beep differs significantly from the normal key beeps (five times as long). So does the very long beep (ten times as long).

⁸³ MB/toetsen.c::is_keuze()

⁸⁴ Flag at MOD_TOETSPIEP_ADR

⁸⁵ MB/kies_pan.c::vp_toon()

When the voter presses the CAST VOTE(S) button (or confirms it), the program changes to the sub-state "store votes."⁵⁶ This sub-state switches off the background light of the CAST VOTE(S) button and displays an appropriate message.

Summary: The requirement is fulfilled. Pressing of the CAST VOTE(S) button produces an audible and visible feedback.

10. Reporting and solution of problems

10.1-1 The software shall only be used for a poll, if it has carried out a self check and a check of the voting machine.

See requirement 5.1-2.

10.1-2 The self check and the check of the voting machine should be repeated after each action.

MB: The run time test¹⁷ is permanently executed. The software consists of modes (main states) and sub-states. Each time a state has been completely executed, the software runs again through the same state (no user action) or changes to a new state (appropriate user action or other conditions). Between two state executions the run time test is performed. It includes a self-check of the software and a check of the hardware (see requirement 1.3-1).

Summary: The requirement is fulfilled. The self-check and the check of the voting machine are permanently repeated but at least after each user action and each time a state has been executed.

10.1-3 Each defect or incorrect working should be reported to the user.

MB: Both the start-up test routine and the run time test routine have their own output functions⁸⁶. If the hardware status table contains an error entry, the output routines will display it on voting machine and Control Unit, write it into the error and event memory (Xicor) and abort or continue the execution of the program.

The problems which may be solved by the voter or the polling staff produce a plain-text message like "Multiple buttons pressed," "Printer without paper or move lever," or "No ballot module present."⁸⁷ All other problems produce a two- to five-digit error code which has to be interpreted by the help desk staff.

At early stages of the start-up test when the RAM cannot yet be used, the success of the ROM and RAM tests⁸⁸ is shown by the display of messages ("4321 * TEST * EPROM *," "RAM *** OK!") letter by letter. If these messages appear only partially, the ROM or RAM tests have failed.

Keys that are pressed a bit longer than is normal (5 sec) produce a beep to get the attention of the voter. When he/she releases the button, the beep stops immediately.

⁸⁶ MB/init_diag.c::handle_status(), run_diag.c::display_diag()

⁸⁷ MB/screen_var.c::kp_e_x, bp_e; str_lang.h

⁸⁸ MB/romtest.c::eprom_test(), ramtest.c::ram_test()

Summary: The requirement is fulfilled. Any defect or incorrect operation is reported to the user. It is reported by plain-text message, error code, or other means.

10.2-1 In the event of a power failure between 500 msec and 2000 msec of pressing the CAST VOTE(S) button, a vote stored in the Eeprom of the machine shall not be lost, providing there is no fatal machine failure on restoration of power.

MB: If the voter presses the CAST VOTE(S) button his/her votes will be given to the vote storage function²⁰. This function performs several checks, enables a certain interrupt routine⁷⁰, and then begins to store the votes into primary ballot module. The interrupt routine will be called if the main power supply is lost during vote storage. It stores the votes into error and event memory (Xicor), which can be done much more faster than storing into the ballot module. If the main power returns, the vote will be read out of the Xicor memory and written into the ballot module.

The interrupt routine starts its guarding function when the CAST VOTE(S) button has been pressed, the votes have been transported to the vote storage function, and the vote storage function has been finished all pre-storing checks. So there is a certain time delay between pressing of the CAST VOTE(S) button and enabling of the interrupt routine. According to measurements at the manufacturer's this time delay is in the range 700 msec.

The interrupt routine is disabled when the votes have been stored successfully into ballot module.

The votes can be restored only if certain conditions are met. These are:

- There are no severe software or hardware errors (RAM, ROM).
- The ballot module has not been exchanged.
- The voting results have not been asked.
- The Xicor memory is ok and the vote temporarily saved has not been modified in between.

Summary: The requirement is fulfilled. The vote is not lost in the event of a power failure between 700 msec after pressing the CAST VOTE(S) button and the normal end of the vote storage process. The vote is saved temporarily into error and event memory. It can be restored as soon as the power returns, if the ballot module has not been exchanged or evaluated (poll closed) in between, and there are no severe machine failures (RAM, ROM, Xicor).

10.3-1 In the case requirement 10.2-1 could not be fulfilled the software should display an error message showing that the vote could not be stored.

MB: The interrupt routine⁷⁰ which realises the auxiliary storage of votes into the Xicor memory does not display any error messages. This would not make sense since the voting machine is short before complete power loss.

The function which restores the vote⁸⁹ after the power has returned generates the same error codes as the normal vote storage function⁵⁸. These error codes are linked to certain, special hardware problems like address line problems or checksum inconsistencies. They can be interpreted with the help of the full error codes list available at the help desk or with the help of a reduced list contained in the operator's manual. The reduced list in the operator's manual shows those error codes which indicate that the vote could not be stored.

To detect the auxiliary storage and the restoration of votes the polling staff may watch the number of voters per poll shown on the Control Unit display.

In the case of power loss the following situations may appear:

Observation	Reason
The number of voters increases for each released poll.	The voter had finished (pressed the CAST VOTE(S) button), the votes are stored successfully.
The number does not increase before power loss, but after power return.	The voter had finished, his/her vote was temporarily saved in Xicor and then restored.
The number does not increase before power loss, and does not increase after power return. No error code is displayed.	The voter had not finished yet. OR The voter had finished, but the time was too short to enable the auxiliary storage process.
The number does not increase before power loss, and does not increase after power return. An error code is displayed.	The voter had finished, the vote was temporarily saved and then restored. The restoration was not successful.

Summary: The requirement is fulfilled. If there are problems during restoration of votes error codes are displayed to the polling staff. These error codes are the same as those which may appear in the normal storing process. They can be interpreted with the help of a list available at the help desk or with the help of the reduced list in the operator's manual.

10.4-1 The diagnostic mechanism shall not be accessible to or be capable of being switched off by the user.

MB: An analysis of the main module and the two diagnostics blocks⁹⁰ shows that the start-up test and the run time test cannot be switched off. They are unconditionally called.

Summary: The requirement is fulfilled. The diagnostics mechanism is not accessible to the users. It cannot be switched off.

10.5-1 The diagnostic mechanism must provide the user with messages (text or code) that support and speed up the trouble shooting procedure.

⁸⁹ MB/stemprog.c::herstel_stem()

⁹⁰ MB/stem_comp.c, init_diag.c, run_diag.c

MB: In several situations the start-up and run time tests produce error codes (numbers) or error messages:

Major errors (e.g. inconsistent election information in the ballot module, defective displays)

The error code is displayed. The voting machine is shut down and cannot be used any longer. The error code is written into the error and event memory (Xicor). It must be analysed by the manufacturer's staff or with the help of the full error code list delivered by the manufacturer.

Medium errors (e.g. problems with the "key beep" option in the ballot module, internal printer not found)

The error code is displayed and saved in Xicor. The polling staff may accept the error code by pressing softkey D and then continue. The voting machine may be used to a reduced extent (if ballot module data are defective, the ballot module will be blocked and voting will not be possible; if the printer is not available, printing will not be possible; etc.). Nevertheless, the voting machine may be used to get the voting results.

If the polling staff wants to use the voting machine for further work, it needs the help desk staff with their detailed list of error codes and explanations.

In certain cases, a plain-text message is displayed ("Ballot module full, voting not possible," "Ballot module not programmed," ...). In these cases, the voting machine may also be used to a reduced extent.

Minor problems (e.g. no paper, key pressed too long, more than one key pressed)

An appropriate error message is displayed or a beep is generated. The problem may be solved by the voter or the polling staff. After that, normal program execution continues. The error messages are short and plain ("Printer without paper or move lever," "Printer error," "Multiple buttons pressed," ...). Some of these problems are recorded in Xicor.

Summary: The requirement is fulfilled. The diagnostics mechanism provides error codes and error messages. The error messages are short and plain. The error codes can be interpreted with the help of the full error codes list delivered by the manufacturer.

13. Conditions for the use of voting machines for two or more polls simultaneously

13.1-1 The software shall be able to manage one, two or more (maximum 5) polls that are held simultaneously, including an election and a referendum.

To manage one to five polls simultaneously, the following functions must be implemented:

- maintenance of one to five polls with different attributes,

- separate releases for one to five polls (see requirement 13.1-4),
- free and independent programming of each key for poll 1 to 5,
- free and independent selection of keys belonging to different polls (see requirement 13.1-5),
- secure storage of all preferences including their assignment to different polls.

MB: The software is implemented in such a way, that it can be used for one to five polls. It contains five global poll variables⁹¹ which hold the poll number and the other attributes (short poll name, poll type, print title, number of choices, constituency). The poll number is allowed to lie between 1 and 5, the poll type may be an election (one out of four types) or a referendum. The contents of the poll variables are programmed into the ballot module (election information area). On the polling day the contents are read out and completed by the attribute 'status' and the attribute 'number of voters'.

Each key of the voter panel is programmable for any poll (1..5), and any choice number (1..90) independent of the programming of other keys. The programming is stored in the ballot module (layout memory). Here the programmed keys are stored in a way that connects row, column, poll number, choice number, and candidate name/option text⁹².

The voter inputs are held in a similar way; the selected key (row, column) is linked to a choice number, ordered by appearance, and saved in a list marked by the poll number⁵². When the vote is re-organised in a new vote structure⁵⁹, the connection between choice number, order number (preference), and poll number is maintained. The voter input variable is dimensioned to hold up to 90 preferences for each of up to five polls. The new vote structure is dimensioned to hold up to 90 preferences for each of up to five polls but up to a maximum number of 90 preferences for all polls together.

Summary: The requirement is fulfilled. The software is capable of managing one to five polls. The polls may differ in type and other attributes. Each key of the voter panel is freely programmable for one of the polls. Each voter input is correctly assigned to one of the polls.

13.1-2 The ballot module must be capable of recording votes on 5 ballot papers (maximum 90 preferences) simultaneously.

MB: As mentioned in requirement 13.1-1, each voter input is handled so as to maintain the assignment of choice number, preference, and poll number. Before storing the vote, it is re-organised into a one-dimensional array⁵⁹ of variable length. It may vary in length from 2 byte (one poll released, one or no preference selected) to 95 byte (five polls released, 18 preferences selected for each poll). The array is completed with start and end marker and stored in the ballot module in front of or after the votes previously stored. During the storing process, the number of bytes, their contents, and their order are maintained⁵⁸.

The new vote structure is sufficiently dimensioned to hold the maximum number of polls and preferences (up to five polls, up to 90 preferences per poll, up to 90 preferences for all polls combined).

⁹¹ MB/verkiezing[1] .. verkiezing[5]

⁹² MB/keuze.c::set_programming(), set_keuze_naam()

The store function is capable of storing votes of minimum as well as of maximum length. Neither the storage method nor the structure of the ballot module restricts the number of polls or the number of preferences in any way.

The ballot module is filled voter by voter. Depending on the number of preferences each voter selects the ballot module is filled sooner or later⁹³. The vote storage is sized to save about 29700 vote bytes (about 28000 preferences). If each voter selects the maximum number of preferences (90), the vote storage will allow the votes of about 300 voters or more to be saved.

Summary: The requirement is fulfilled. The ballot module is capable of holding preferences selected from up to five different polls (maximum 90 preferences per voter).

13.1-3 The designation of the selection buttons for each ballot paper shall be stored in tabular form in the ballot module.

MB: The programming of the choice and option keys is stored in the layout memory of the ballot module. It is stored as an array supplemented by a set of strings (candidate names and option texts).

The first part of the layout memory contains an array of all programmed choice and option keys²⁵. The indices of the array are a combination of row and column number of the key. The contents of the array element are the poll number and the choice number the key belongs to. Besides the poll and choice number, an offset is stored in each array element. The offset is used to find the ballot module address at which the candidate name or option text is stored.

The software contains functions⁹⁴ which get a row and column number of a key and furnish the poll and choice number of the key and the candidate name or option text belonging to that key.

The array in the layout memory and the variables used to store the layout information are sized to hold up more than 90 programmed choice and option keys (5 columns and 18 rows). The second part of the layout memory is sized to hold up more than 90 candidate names / option texts with 32 characters each.

Summary: The requirement is fulfilled. The programming of choice and option keys is stored in the ballot module in tabular form (as an array).

13.1-4 The polling staff should be able to activate the voting machine for one to five polls.

MB: The software recognises four different release buttons (R_P, R_D, R_E, R_L). They activate the polls depending on their type (see requirement 6.1-1):

Condition	Reaction
R _P (= 'P') pressed	all polls are activated
R _D (= 'D') pressed	polls of type Dail, Europe, and Local are activated

⁹³ Document *Functional Specification* version 1.9, p. 11 par. 1.8.1

⁹⁴ MB/keuze.c::get_keuze_info(), get_short_keuze_info()

R_E (= 'E') pressed
R_L (= 'L') pressed

polls of type Europe and Local are activated
polls of type Local are activated

So the polling staff is not allowed to release polls completely independent of each other; the polls may be released only depending on their types. Polls of the same type may only be activated together, but not independent of each other. Presidential elections or referendums may only be activated together with polls of all other types. Nevertheless, the polling staff may release the polls according to the description in the *Functional Specification*⁹⁵.

Summary: The requirement is fulfilled. The polling staff may activate one to five polls depending on their type.

13.1-5 *The voter should be able to record preferences starting on any ballot paper. The preferences should start from 1 on each ballot paper (except a referendum ballot paper). The voter may switch between ballot papers until he/she has finished recording his/her preferences.*

MB: The software is implemented in such a way that the voter is allowed to arbitrarily switch between all polls released. The voter input variable⁵² holds all selections already made by the voter. It is a two-dimensional array which contains five lists (for five polls) of entries (selected choices/options).

When the voter presses a certain key, the software looks up the poll number and the choice number of that key and the status of the poll (released/not released). Then the corresponding list in the voter input variable is checked. If the selected choice is already contained, the selection will be corrected. If it is not yet contained, the choice will be written at the end of the list. The number of entries is increased by one and displayed on the LED near the choice key. If the key selected is the first key for that poll, it will automatically get preference number 1.

For referendums the poll list is allowed to contain a single entry only. This may be corrected but not supplemented. The entry has preference number 1 and is displayed as a cross.

Summary: The requirement is fulfilled. The voter may start with any poll and arbitrarily switch between all polls. Inside each poll the preferences start with number "1" and increase by one for each selection (except referendums).

13.1-6 *The votes should be stored randomly in the ballot module.*

See requirement 8.8-1.

13.1-7 *The software shall not provide information about any preferences recorded or vote cast by a voter, after the voter has pressed the „CAST VOTE(S)“ button.*

See requirement 6.5-1.

⁹⁵ Document *Functional Specification* version 1.9, p. 7 par. 1.3.1

13.1-8 The software shall not provide information about any preferences recorded or vote cast by a voter to the display of the Control Unit during the election process.

See requirement 6.5-2.

14. Documentation

Only software documentation is considered here.

14.3-1 The development documentation must include:
(b) the software design of the voting machine and programming unit;
(c) listings of the application source codes;

The documentation includes a description of the software design. The *Detailed Software Design* document describes

- the purpose of the software and all its functions,
- its basic structure as a set of four layers and two diagnostics blocks,
- the error handling principles,
- the purpose and rough contents of all modules including those modules which are used in Programming/Reading Unit only,
- the states and state transitions including the state transition diagrams,
- the ballot module contents including validity conditions for the data,
- information about the compiling and linking process,
- character coding tables,
- data security measures.

The document is supplemented by technical specifications which describe several aspects of the system, like the vote storage process, security measures or the behavior at power failure.

The documents contain all information necessary to understand the internal structure of the software and its operating principles. The documents are readable, understandable, and comply with the source code. They cover the software parts which work inside the voting machine as well as those which work inside the Programming/Reading Unit.

The source code is available in the form of electronic files, which is the most appropriate form for tests and inspections. Listings may be printed, if necessary.

Summary: The requirement is fulfilled. The documentation includes a description of the software design and the source code as electronic files.

- 14.4-1 The test documentation must describe which measurements the manufacturer used at the verification, the validation and the testing of the software of the voting machine and must include at least:**
- a test plan that describes the way in which the functions of the software are tested;**
 - software module test report that describes the results of the module tests;**
 - software integration test report(s) that describe the results of the integration tests;**
 - software system test plan and report(s) containing the system test plan (test specification) and results.**

The documentation includes the description of the software tests. It covers:

- an overview/test plan⁹⁶
- system tests⁹⁷,
- module/integration tests⁹⁸,
- performance tests⁹⁹.

The system tests include the tests of all modes (main states), sub-states, and state transitions. They are complete and correspond to the state transition diagrams described in the *Detailed Software Design* document. The system tests also include the test of the error handling system, i.e. the artificial generation and handling of all possible error codes.

The module/integration tests are focused on the functionality of single functions, groups of functions, modules, and groups of modules. The tests are arranged to find out whether or not certain functionality aspects work correctly. The following aspects are tested in detail:

- check of the ballot module for correct / incorrect data,
- readout of layout information from ballot module,
- readout of votes and validation of votes,
- summing-up of the number of polls, number of voters and number of null votes,
- security measures like checksums and internal tests,
- writing and reading events and errors,
- auxiliary storage process,
- keys of voting machine and Control Unit, LEDs,
- displays, beeper, and background lighting,
- printing and display of all characters,
- connection between main board, connection board, and display boards.

The performance tests include the measurement of the times it takes to program modules, store and check votes, and sum up votes.

⁹⁶ Document *Overview ESI2 software tests*

⁹⁷ Documents *State machine tests software main board ESI2, Event and error tests software main board ESI2, Vote while power down*

⁹⁸ Documents *Integration tests software main board ESI2, Functions tests software main board ESI2, Connection board function tests, Display board function tests results, Test Description Vote Storage at Power Failure, Driver tests software main board ESI2*

⁹⁹ Documents *Integration tests software main board ESI2, Communication tests software main board ESI2*

The tests are performed on the voting machine or on groups of modules (layers) in a special test environment. The documents include the description of test cases and test results and some details about the test execution.

The documents describe extensive tests. Nevertheless several tests could be added to intensify the testing process.

The documents include an introduction describing the aim of the tests and the basic test procedure. Besides a separate software test plan is available.

All documents are identified by title, author, date, and document revision number. All test documents describe the full device under test (hardware versions and software versions of all boards).

The documents are readable and consistent. They comply with other parts of the documentation and with the source code. They are of sufficient detail.

Summary: The requirement is fulfilled. A software test documentation is available. It describes how the manufacturer has tested the modules, the groups of modules and the system as a whole. The tests include module and integration tests, system tests, and performance tests.

Summary

All requirements are fulfilled. A number of issues which do not influence the proper working of the voting machine have been highlighted for the attention of the technical designers.

Requirement	Ful-filled?	Rem.
0. Identification		
0.0-1 All executable programs shall have a version number, which is shown on request and which cannot be changed during election process.	OK	
0.0-2 All source code modules shall have a version number.	OK	
0.0-3 Each voting machine shall have an identification number, which is shown on request and which cannot be changed during election process.	OK	
0.0-4 Each ballot module shall have an identification number, which is shown on request and which cannot be changed during election process.	OK	
1. General		
1.1-1 The software shall contain all functions, which are described in the Functional Specification (version 1.7 of 5 March 2003).	OK	
1.2-1 The software manufacturer shall have and apply an appropriate development model.	OK	
1.2-2 The source code shall comply with a standardised programming language.	OK	

1.2-3	The source code shall comply with the current principles of software engineering.	OK	1
1.3-1	The diagnostic software detects changes that influence proper functioning of the voting machine.	OK	
1.4-1	Any alteration of the installed software by an unauthorised person should be detected.	OK	
1.4-2	Any alteration of the content of the ballot module by an unauthorised person should be detected.	OK	2
2. General conditions concerning the provision of information by the voting machine			
2.3-1	The software shall print or display all candidate information which is contained in the ballot module before and after polling. This information shall not be changeable during election process.	OK	
2.3-2	The software shall print the total number of activations of the voting machine before and after polling.	OK	
2.3-3	The software shall print and display the number of votes including null votes on the voting machine before and after polling.	OK	
2.3-4	The software shall print the total number of de-activations of the voting machine before and after polling.	OK	3
2.3-5	The software shall print or display all information which identifies the voting machine, the software and the ballot module before and after polling.	OK	
2.3-6	The software shall print or display all relevant details of the election process (kind of election, date, constituency, poll station number) before and after polling.	OK	
3. Installation of a ballot module			
3.1-1	Following the insertion of a programmed primary ballot module into the voting machine, the software shall carry out a self check and a check of the voting machine.	OK	
3.1-2	Following the insertion of a primary ballot module and a check of the software and the voting machine, the software shall check the inserted primary ballot module for consistency.	OK	4, 5
3.1-3	Following the insertion and check of a primary ballot module, the software shall display and/or print the contents of the memory of the ballot module.	OK	
3.2-1	The software shall only be activated for voting if a module has been installed and the lock on the ballot module slot is in the closed position.	OK	
5. Preparation before the poll			
5.1-1	The software shall print and display the number of votes including null votes on the voting machine before and after polling. (= 2.3-3).	OK	
5.1-2	The software shall only be used for a poll, if it has carried out a self check and a check of the voting machine.	OK	
5.2-1	Votes shall only be recorded if the key referred to in 3.2 is inserted in the Control Unit. Withdrawal of the key from the Control Unit will switch the software to standby and prevent votes from being recorded.	OK	

6. The poll			
6.1-1	The software shall not record votes, unless it has been activated.	OK	
6.2-1	The software shall not record votes, unless it has been activated. (= 6.1-1).	OK	
6.2-2	The software shall count all activations.	OK	
6.2-3	After being activated, the software shall switch on LED's at top of those ballot papers on which the voter is entitled to vote.	OK	
6.2-4	After being activated, the software shall display the marks "----" to the right of those ballot papers on which the voter is entitled to vote.	OK	
6.2-5	After being activated, the software shall display the preferences chosen by the voter.	OK	
6.2-6	After being activated, the software shall display "x" in the left hand position of the LED chosen by the voter in the case of a referendum.	OK	
6.2-7	When a preference is recorded, the software shall display details on the bottom line of the voting machine display screen.	OK	6, 7
6.2-8	The software shall allow to change preferences by pressing one of the selected candidates or referendum choices a second time. The replaced preferences shall not be stored permanently.	OK	
6.2-9	The software shall allow to record the vote, if at least one preference on one ballot paper has been recorded.	OK	
6.2-10	If there are ballot papers without recorded preferences, the software shall record the vote after displaying a reminder and receiving a confirmation.	OK	
6.2-11	The software shall record the vote, when the voter has pressed the CAST VOTE(S) button or confirmed the button pressing. The process shall be performed without any interruption.	OK	8
6.2-12	The vote shall be recorded only once.	OK	
6.2-13	The software shall count all de-activations.	OK	3
6.2-14	The software shall record the vote, when the voter has pressed the "CAST VOTE(S)" button or confirmed the button pressing. The process shall be performed without any interruption. (= 6.2-11).	OK	8
6.3-1	After each vote is stored, the voters panel shall be deactivated automatically. The software will not accept any further votes until a member of the polling staff, operating the Control Unit, reactivates the software. The activation shall not be possible by means of the voting machine.	OK	
6.4-1	The software should offer a way to withdraw the activation by means of the Control Unit. Any preferences entered on the voting machine should not be recorded as a vote in this case.	OK	
6.4-2	The software shall count all de-activations. (= 6.2-13).	OK	3
6.5-1	The software shall not provide information about any preferences recorded or vote cast by a voter, after the voter has pressed the CAST VOTE(S) button.	OK	

6.5-2	The software shall not provide information about any preferences recorded or vote cast by a voter to the display of the Control Unit during the election process.	OK	
6.6-1	The software shall display the total number of votes including null votes on the Control Unit. The software shall display the number of votes including null votes on the voting machine in stand-by modus.	OK	
7. Displaying and printing the total number of votes at close of poll			
7.1-1	The software shall display the number of votes and the number of null votes on the voting machine at close of poll.	OK	9
7.1-2	The software shall print the number of votes and the number of null votes at close of poll.	OK	
7.1-3	The software shall print the total number of activations of the voting machine before and after polling. (= 2.3-2).	OK	
7.1-4	The software shall print the total number of de-activations of the voting machine before and after polling. (= 2.3-4).	OK	3
7.2-1	Activation of the software for the display and printing of the information referred to in paragraph 7.1 shall be accomplished by means of the key referred to in paragraph 5.2. Withdrawal of the key from the Control Unit should automatically stop the printing process.	OK	
7.2-2	The back-up module shall be checked to ensure that any existing data in the module are deleted, before the contents of the primary ballot module are copied to the back up module.	OK	
7.3-1	After displaying or printing the total number of votes cast etc, the software shall accept new activation only after the insertion of a new programmed ballot module.	OK	
8. Reliability and security of the voting machine			
8.1-1	The software shall record the vote, when the voter has pressed the CAST VOTE(S) button or confirmed the button pressing. The process shall be performed without any interruption. (= 6.2-11).	OK	8
8.1-2	The vote shall be recorded only once. (= 6.2-12).	OK	
8.1-3	The software shall not record other information than the vote, when the voter has pressed the CAST VOTE(S) button or confirmed the button pressing.	OK	
8.2-1	A cast vote must not be lost by a power failure, the failing of one component, through normal use or through failures in the operation of the voting machine.	OK	8
8.2-2	In case of a failure during election process the software shall provide information, whether the vote of the current voter has been recorded.	OK	
8.3-1	The installed ballot module and its contents must be fully maintained in case of a power failure, through normal use or through failures in the operation of the voting machine.	OK	
8.4-1	The functions of the software must be fully maintained in the event of a power failure.	OK	10
8.5-1	The storing of votes in the ballot module must be made in such a way so as to ensure security and continuous self checking of all data. Each vote should be stored twice in each of 2 independent IC's within the ballot module.	OK	

8.5-2	As each single preference is recorded, a checksum shall be calculated and stored in RAM.	OK	
8.5-3	When the voter presses the CAST VOTE(S) button, all checksums of all preferences stored in RAM shall be checked.	OK	
8.5-4	As the voting machine stores preferences, a read back of the last byte of data written should be made.	OK	
8.5-5	After storing all preferences, the system should check all of these again and their checksums.	OK	
8.5-6	After storing and checking all preferences of the current voter, the software shall check all preferences stored from all voters.	OK	
8.5-7	The software shall display an error message on the voting machine display and on the Control Unit display, if a vote could not be stored.	OK	11
8.5-8	In the event that a discrepancy occurs in the checksums, an error message should be generated. This message should be shown on the voting machine display and on the Control Unit display.	OK	11
8.6-1	The software should, as far as is reasonably and technically possible, avoid or restrict the possibilities of accidental or intentional incorrect use.	OK	
8.8-1	The votes should be stored randomly in the ballot module.	OK	
8.8-2	The software shall not provide information about any preferences recorded or vote cast by a voter, after the voter has pressed the CAST VOTE(S) button. (= 6.5-1).	OK	
8.8-3	The software shall not provide information about any preferences recorded or vote cast by a voter to the display of the Control Unit during the election process. (= 6.5-2).	OK	
9. Operability			
9.1-1	The software shall perform no other actions than described in requirement 6.2, when the voter is acting on the voting machine.	OK	
9.3-1	The activation and de-activation for each voter must lead to a visible, audible or tangible feedback signal.	OK	
9.3-2	The selection of preferences by the voter must lead to a visible, audible or tangible feedback signal.	OK	12
9.3-3	The pressing of the CAST VOTE(S) button by the voter must lead to a visible, audible or tangible feedback signal.	OK	
10. Reporting and solution of problems			
10.1-1	The software shall only be used for a poll, if it has carried out a self check and a check of the voting machine. (= 5.1-2).	OK	
10.1-2	The self check and the check of the voting machine should be repeated after each action.	OK	
10.1-3	Each defect or incorrect working should be reported to the user.	OK	11
10.2-1	In the event of a power failure between 500 msec and 2000 msec of pressing the CAST VOTE(S) button, a vote stored in the Eeprom of the machine shall not be lost, providing there is no fatal machine failure on restoration of power.	OK	13
10.3-1	In the case requirement 10.2-1 could not be fulfilled the software should display an error message showing that the	OK	14,11

vote could not be stored.		
10.4-1 The diagnostic mechanism shall not be accessible to or be capable of being switched off by the user.	OK	
10.5-1 The diagnostic mechanism must provide the user with messages (text or code) that support and speed up the trouble shooting procedure.	OK	11
13. Conditions for the use of voting machines for two or more polls simultaneously		
13.1-1 The software shall be able to manage one, two or more (maximum 5) polls that are held simultaneously, including an election and a referendum.	OK	
13.1-2 The ballot module must be capable of recording votes on 5 ballot papers (maximum 90 preferences) simultaneously.	OK	
13.1-3 The designation of the selection buttons for each ballot paper shall be stored in tabular form in the ballot module.	OK	
13.1-4 The polling staff should be able to activate the voting machine for one to five polls.	OK	
13.1-5 The voter should be able to record preferences starting on any ballot paper. The preferences should start from 1 on each ballot paper (except a referendum ballot paper). The voter may switch between ballot papers until he/she has finished recording his/her preferences.	OK	
13.1-6 The votes should be stored randomly in the ballot module. (= 8.8-1).	OK	
13.1-7 The software shall not provide information about any preferences recorded or vote cast by a voter, after the voter has pressed the CAST VOTE(S) button. (= 6.5-1).	OK	
13.1-8 The software shall not provide information about any preferences recorded or vote cast by a voter to the display of the Control Unit during the election process. (= 6.5-2).	OK	
14. Documentation		
14.3-1 The development documentation must include: (b) the software design of the voting machine and programming unit; (c) listings of the application source codes;	OK	
14.4-1 The test documentation must describe which measurements the manufacturer used at the verification, the validation and the testing of the software of the voting machine and must include at least: - a test plan that describes the way in which the functions of the software of voting machine and programming unit are tested; - software module test report that describes the results of the module tests; - software integration test report(s) that describe the results of the integration tests; - software system test plan and report(s) containing the system test plan (test specification) and results.	OK	15

Remarks and hints for use:

1. In the next software version the initialisation of local variables should be improved.
2. Unauthorised changes of the following ballot module data are not automatically detectable: number of programming/erasing cycles, number of deactivations for each poll, freely programmable texts to be displayed to the voter. These data have no effect on the functionality of the voting machine. Falsifications of the freely programmable texts may have a slight influence on the usability of the voting machine, but they are easily detectable for the polling staff using the open poll print-out.
3. The numbers of de-activations per poll are counted correctly. The following should be kept in mind:
 - A de-activation concerning the first voter of the day is not counted. The counting starts when there are votes in the ballot module.
 - A de-activation which is necessary because of the polling staff selected a wrong release key is counted.
 - A de-activation which is necessary because the ballot module is full is counted. The check of the ballot module is done after activation of the voting machine.
4. The software itself is not capable of checking the correctness of candidate names. It is important to verify the correctness before polling by a manual test at the polling station or at the polling centre.
5. There is no automatic check of the ballot module for being empty (0 votes). The polling staff or the staff in polling centre must check the number of votes before poll starts.
6. The voting machine display shows no candidate or option details when the polling staff has activated the "Visually impaired device".
7. The display of candidate or option details is suppressed when a special option is programmed into the ballot module at the polling centre.
8. A cast vote is always stored except in the following remote and abnormal circumstances: failure of a hardware component necessary to store the vote or to maintain the correct operation of the voting machine; power failure combined with a defect of the Xicor memory.
9. The number of voters per poll is displayed on voting machine at close of poll. This number always includes the number of null votes. The number of null votes per poll is not displayed separately.
10. If a battery is used, the internal printer is out of service. Hence data that are printed but not displayed are no longer accessible (e.g. null votes per poll).
11. Slight problems are displayed as a plain-text message or signalled by key beep. Other problems which reduce the functionality of the voting machine are displayed as an error code. This error code can be interpreted with the help of the full list of

error codes available at the help desk. The same list is also necessary, if the error and event memory should be evaluated.

Those error codes which indicate that the vote could not be stored are also listed in the operator's manual available at each polling station.

12. The beep of the programmed voter panel keys is suppressed when a special option is programmed into the ballot module at the polling centre. In this case the voter gets only a visual feedback when he/she presses a choice or option key.
13. The vote which was temporarily saved into Xicor memory is not restored, when the ballot module was exchanged or when the voting results were asked or when the Xicor memory got faulty in between.
14. If there are problems during restoration of votes error codes are displayed to the polling staff. These error codes are the same as those which may appear in case of errors in the normal storing process.
In case of successful restoration there are no messages. The fact that votes were written by an auxiliary storage process and restored after that is only detectable by watching the number of voters on the Control Unit display carefully.
There are no plain text messages like "vote not stored" in the case of errors or "vote restored after power loss" in the case of success.
15. The software module / integration tests should be extended in several cases.

.....
N. Greif
Head of Software Testing Laboratory