STONES UNTURNED: GAPS IN THE INVESTIGATION OF SARASOTA'S DISPUTED CONGRESSIONAL ELECTION

David L. Dill
Dept. of Computer Science
Stanford University
dill@cs.stanford.edu

Dan S. Wallach
Dept. of Computer Science
Rice University
dwallach@cs.rice.edu

April 13, 2007

EXECUTIVE SUMMARY

The November 2006 race for Florida's 13th Congressional District resulted in a 369 vote margin of victory for the winning candidate with more than 18,000 undervotes recorded on the ES&S iVotronic touch-screen voting machines used in Sarasota County. Since then, the losing candidate and a coalition of local voters have filed suit against the state and local election officials (among other defendants), seeking a judicial order to rerun the election. A key question is whether a system malfunction may have induced the undervote rate. We evaluate the two major efforts previously undertaken by the State: a mock election, conducted by the State, and an analysis of the iVotronic source code, conducted by academic computer scientists under contract to the State. Press reports and summaries of the State's findings have created a public perception that the investigation was thorough and that the voting machines have been exonerated of contributing to the undervote. Based on our evaluation of the investigation, this perception is not justified.

There are many significant gaps in the tests conducted by Florida and its experts. The defined scope of the mock election specifically excluded examination of the vote selection process, which, based on voter complaints, should have been a major focus of the investigation. The tests were conducted in an artificial setting with the iVotronics mounted vertically, unlike their horizontal orientation in real elections. Furthermore, the State's report claims that there were no anomalies observed during the vote, yet video recordings of the test show occasional vote selections not registering on the machines.

The State's inspection of the iVotronic's software was also incomplete. The State's academic team read the source code but performed limited hands-on experimentation with demonstration machines. They made no attempt to examine whether the hardware functioned properly, nor did they examine iVotronic machines that were used in the actual election. The team performed no analysis based on compiling and executing the software, either on iVotronic hardware or in a "test harness." Such testing is commonly used to identify bugs that may manifest themselves only under obscure conditions. Likewise, the team did not review internal ES&S documents, such as their bug tracking systems or software repositories, which might contain important clues about the problem. For key issues, including how the iVotronic screen is calibrated and how its smoothing filter operates, the final report contained insufficient detail to determine how these issues may have impacted the undervote.

In total, the State's investigations have provided no persuasive explanation for Sarasota's undervotes. We recommend additional testing and analysis of both the software and hardware used in Sarasota. We also recommend analysis of ES&S's internal documents, including their bug tracking system and other versions (earlier and later) of their software. We estimate that this additional investigation could be conducted by an appropriate team of experts with about a month of work.

1 INTRODUCTION

In the November 2006 general election, in Florida's 13th Congressional District (hereafter, "CD13"), Vern Buchanan was the certified winner, defeating Christine Jennings with a 369-vote margin, but with over 18,000 "undervotes" (i.e., cast ballots with no selection in this particular race) in Sarasota County. The unusually high undervote rate led Jennings, as well as a coalition of non-partisan organizations, to mount legal challenges to the election results.[1]

	Total Votes	%	Election Day	Early Voting	Absentee	Provisional
Vern Buchanan	58,632	47.24	36,619	10,890	11,065	58
Christine Jennings	65,487	52.76	39,930	14,509	10,981	67
Over Votes	1		0	0	1	0
Under Votes	18,412		12,378	5,433	566	35

Table 1: Official election results from Sarasota County.

This congressional district spans five counties; the controversy centers on Sarasota County and its use of the ES&S iVotronic paperless electronic voting system. Table 1 describes the election-day results published by Sarasota County. 12.9% of the votes cast in Sarasota County for the CD13 were undervoted, in contrast with other races that have much lower undervote rates (e.g., 1.14% in the Senate race, 1.28% in the Governor race, 4.36% in the Attorney General race, and 4.43% for the Chief Financial Officer race). Vote tallies from the surrounding counties in CD13 likewise had low undervote rates.

If the iVotronic votes in Sarasota County are considered alone, the CD13 undervote rate was 14.9%. This contrasts with a Sarasota County CD13 undervote rate of 2.5% on absentee ballots (i.e., centrally tabulated optical scan ballots). Without a doubt, the iVotronic votes in the Congressional race exhibit an anomalously high undervote rate. If a result like this had occurred with punch cards or with hand-marked ballots, the inquiry would certainly have focused on a flaw in the tabulating machinery and would have reexamined the original ballots. Unfortunately, the iVotronic does not produce a voter-verified paper record that can be reexamined, so an investigation must follow other avenues.

The subsequent sections of the paper first describe what would be reasonable priorities for investigation of the Sarasota's undervote rate (Section 2), followed by a discussion of the the actual investigation, focusing on the testing performed by the State of Florida (Section 3), and the source code analysis performed by several academic computer scientists under contract to the State (Section 4), with consideration of where both fell short. We finish with conclusions and recommendations for future investigation (Section 5).

2 PRIORITIES FOR INVESTIGATION

We feel strongly that the CD13 undervote merits more extensive investigation, with many obvious questions remaining open. It may not be possible to answer every question, but until relatively simple and straightforward steps are taken to try to answer the open questions, there will continue to be doubt about whether the outcome of the race represents the will of the electorate, and the same problems may arise in the future because we failed to understand and correct the problems that caused the CD13 undercount. The goal of this paper is to discuss where limited resources for investigation could be best employed to maximize the likelihood of gaining more insight into the causes of the CD13 undervote.

For the sake of this discussion, we operate under the hypothesis that the CD13 undervote is most probably the consequence of an accidental problem associated with the iVotronic voting system. We do not

[1] Dill and Wallach are both expert witnesses for plaintiffs in the *Jennings v. Buchanan*-related lawsuits. This document represents our best efforts to present an unbiased consideration of the facts in this case.

propose to investigate whether the CD13 undervote is the result of malicious software or tampering with the votes stored electronically in the iVotronics. While we have no evidence that the latter conjectures are untrue, we also have no evidence in favor of them. Furthermore, if someone had the ability to unduly influence the election outcome, they would be unlikely to choose to create an obviously high undervote rate, rather than making other changes that would be less likely to be noticed. And, most importantly, an effective investigation of malice or tampering would be exceptionally difficult to conduct with limited resources.

Our discussion is divided into two investigative strategies: testing and inspection. The first strategy focuses on the behavior of the systems and their interactions with the voters, while the second focuses on the design and implementation of the iVotronic voting systems. The State of Florida's investigation explored both perspectives, but did so incompletely.

Knowing where *not* to look can save effort. The cause of the undervote is most likely to be found somewhere between when the voters entered the voting booth and when the vote totals were reported by the voting machines. In particular, we consider it unlikely that any vote corruption was introduced through the county's centralized vote tabulation process, a conclusion we base on the success of Sarasota's recount. For each of the approximately 1500 iVotronics, a pair of election workers inserted a "personal electronic ballot" (PEB), to collect the vote totals from that machine. Once all of the machines in a precinct had their votes stored on a PEB, a hand-held thermal printer was connected to the serial port of the last iVotronic from the precinct and that machine was instructed to print the totals. These totals were brought up to a table where they were announced before various TV cameras and election observers.

The final totals, after the two-day recount exercise, were slightly different from the original totals, but the official results indicate that the changes resulted entirely from reconsidering the absentee votes (using optically scanned ballots) and including provisionally cast ballots. The results of the recount greatly reduce the probability that the high undervote rate resulted from some tampering with data after it left the iVotronics, either in transit or in the Unity election management system.[2] *Therefore, further investigation should focus on the voting machines and their interaction with the voters.*

2.1 Machine behavior and voter interaction

2.1.1 Complaints

An obvious source of clues about the cause of the undervote would be complaints by voters and poll worker incident reports during the election. There were hundreds of complaints, most which point to problems with the interaction between the voter and the voting machines [8]. Many voters complained during the election that their selections registered in the CD13 race screen, but failed to appear on the screen summarizing the voter's selections just before the vote is cast. Others complained that the CD13 race did not appear on the ballot at all. There were also many complaints that the machines were slow to respond, or that the touch-screens had to be pressed for an extended time before they would register a selection.

2.1.2 Smoothing filter

There has been significant press coverage focusing on the software-based "smoothing filter" used by the iVotronic to filter out stray clicks, finger bounces, and other transient effects. ES&S sent a memo in August 2006 to its Florida customers [1] stating:

> It has come to our attention after a number of inquiries from several of our iVotronic 12 inch screen users that some of your screens are exhibiting slow response times. After receiving some

[2] ES&S's Unity election management system runs on a general-purpose PC and performs several functions, including collecting data from individual machines, tabulating votes, and generating reports on the election results.

> of these terminals in our Omaha, NE facility we were able to replicate a slow response during testing.
>
> ...
>
> We have determined that the delayed response time is the result of a smoothing filter that was added to iVotronic firmware versions 8.x and higher. This smoothing filter waits for a series of consistent touchscreen reads before a candidate name is highlighted on the ballot. In some cases, the time lapse on these consistent reads is beyond the normal time a voter would expect to have their selection highlighted. This delayed responses to touch may vary from terminal to terminal and also may not occur every single time a terminal is used.

The memo then goes on to recommend poll workers be trained to help voters with this condition and describes ES&S's efforts to repair the bug. To the best of our knowledge, Sarasota's iVotronic machines were running the software version with the above-described problem and Sarasota's poll workers were not specifically trained to assist voters with this problem.

2.1.3 Touch-screen calibration

Another common worry about the accuracy of touch-screens over the years has been *calibration error*. On any touch-screen display device, the clear, touch-sensitive layer is separate from the part of the screen that displays the buttons. To ensure that every touchscreen press is accurately mapped to screen coordinates, a calibration step is necessary. This process, familiar to anyone who owns a PDA, involves the machine displaying a series of cross-hairs and asking the user to press on the center of each cross-hair. The machine can then compute the necessary correction to apply to screen presses.

Among other procedures, the technicians who prepared the voting machines in Sarasota County were responsible for calibrating their screens. The ES&S iVotronic is unusual in requiring twenty different calibration targets to be touched as part of this process. For comparison, Palm and Windows XP Tablet owners are asked to touch only four targets.

If calibration is inaccurate, voters' touches are less likely to be registered accurately (they may be missed, or even associated with the wrong candidate). There is some evidence to support the theory that machine miscalibration may have been an issue in Sarasota. Stewart, an expert for the Jennings legal team, found a significant correlation between the "clear and test" times for the voting machines and their undervote rate as well as a significant correlation between the number of machines cleared on a given day and their undervote rate [11, 12] (see also the discussion of screen calibration in Section 3.2). Perhaps, in the process of setting up over a thousand machines, technicians grew more careless with calibration as the days progressed.

A test for this, simple to perform, would be to deliberately miscalibrate iVotronic machines and to carefully observe the behavior of a variety of test voters using those machines with the standard Sarasota ballot. Likewise, a number of the iVotronic machines, still sequestered in a warehouse in Sarasota, could be tested to determine how accurately they were calibrated, and this could be compared with the actual undervote rates on those machines. If poorly calibrated machines were observed to correlate with a higher undervote rate, then calibration effects would warrant increased attention. (Jones [9] raises the possibility that calibration can be thrown off when a voter rests one hand on the screen while pressing it with the other hand. This theory could also be tested experimentally.)

2.1.4 Ballot design

Another theory about the CD13 undervote is that it resulted from the particular layout of the ballot, which put the two-candidate CD13 race at the top of the page, above a much longer list of candidates in the Governor's

race. The Governor's race also began with a prominent "State" heading that, it is hypothesized, distracted voters from the CD13 race (see, e.g., Frisina et al.[7]). This hypothesis has been vigorously debated, and we don't want to repeat the arguments on both sides here, except to note that there were many voter complaints of problems that were not consistent with this explanation. However, it would be *extremely difficult* to prove this hypothesis except by a large-scale user study involving hundreds of voters with different backgrounds, controlling for many different factors, including many of those above (e.g., calibration and user interface timing). Such a study would also need to carefully control for prior awareness by the users of the relevance of the experiment to the CD13 contest.

In addition to demonstrating ballot effects, such a study would have to show that they are of the same magnitude as the CD13 undervote. While such a study would yield valuable results for improving the accuracy of elections, we would only propose it as part of the CD13 investigation if ample resources were available. There are many other important questions that can be studied without requiring such a massive effort.

2.1.5 Surprising effects

Computer systems are so complex that the causes of a problem, when discovered, are often surprising. While investigations must be conducted with specific issues in mind, such as those raised in the voter affidavits, they must also be mindful of the possibility that important clues could come from anywhere, such as unexpected behavior that arises in response to a test. Likewise, weaknesses in the vendor's engineering processes may also lead to unexpected problems in practice. Investigators must be alert for such clues and be prepared to pursue them.

It has been recognized that an important quality of so-called real-time embedded systems is that they should behave *deterministically*, meaning that they should behave predictably in the same way in the same circumstances. The same property would be desirable for voting systems for the same reasons. Non-determinism leads to an explosion of different possible system behaviors, which, in turn, often lead to flaws in programs because the programmers don't anticipate all of the possibilities. Non-determinism also makes systems much harder to test and debug; if the system is deterministic, odd or erroneous behavior is easier to reproduce so that it can be investigated and repaired.

We don't know to what extent the iVotronic was designed to behave deterministically (although the SAIT report [14] mentions specific programming practices that may lead to non-deterministic behavior), but any observed non-determinism should be treated as an indication of the potential for other hidden problems. The discussion above has already mentioned one such case that is worthy of further investigation: the as-yet unexplained variable behavior of the smoothing filter (see Section 2.1.2).

2.1.6 Combined effects

It is quite possible that there was a system failure which resulted from the *combined effect* of voters and software or hardware that behaved in counter-intuitive ways – or contained outright errors. It's possible that unexpected machine behavior could have caused voters to take actions which caused their votes to be unintentionally lost. Historically, many system failures arise from the combined effects of poorly designed software and its interactions with users. It is not appropriate to dismiss such problems as user errors. The end result is a system failure, in this case, an inaccurate vote, and systems must be designed to minimize such failures. In particular, a machine failure or design flaw – such as a program bug that leads to unpredictable timing in the user interface – could lead to inaccurate voting because of the way that voters react to it.

Leveson [10] summarizes the general concept:

> ...human actions both prevent and contribute to accidents in high-technology systems. Accidents and incidents occur because of human error, machine failure, poor design, inadequate

> procedures, social or managerial inadequacies, and the interaction of all these factors. Assignment of blame to operator error alone ignores the multiple causality of accidents and the relationship between human actions and the context in which they occur. . . . [A]scriptions of blame to humans for accidents in complex systems is often simplistic and may get in the way of reducing risk through improved engineering and management.

An excellent example to illustrate this point comes from a competing voting system from Diebold, which demonstrates how a software bug may manifest only as a result of the occasional idiosyncratic behavior of a few voters. The Diebold AccuVote-TSx voting system would crash on occasion, seemingly at random. The cause was eventually discovered. In rare instances, a voter would touch the "cast vote" button and then drag his or her finger to another part of the screen, outside the boundaries of the button. This action was interpreted as a "drag and drop" operation by the underlying Windows CE system and would cause the voting machine software to receive an unexpected signal, corrupting the machine's internal state and leading the machine to crash. Only some voters will experience such a problem, and the same voter may experience it some times and not others, without knowing how they caused it. Finding this problem required extensive test voting with a variety of test voters [3].

While this example pertains to a system not used in Sarasota, it represents a significant failure of electronic voting systems that is very difficult to discover with the type of carefully scripted testing performed in the State's investigation. To discover this type of problem, it is better to have a variety of people entering votes, who may do a variety of unexpected things, and to have people acting creatively to produce inputs that the machine's programmers may not have anticipated.

2.2 Inspecting the system design and implementation

The State's investigation included inspecting the "source code" for the system [14]. Source code is the human readable representation of a program that the programmers work with. Of course, there is a lot more to a computer voting system than the source code for the voting application.

To begin, the system doesn't execute source code. It executes binary files, which represent machine instructions as a sequence of numbers. Binary files, needless to say, are more difficult for humans to inspect. These binary files are "built" from the sources by running a series of software translation tools, including compilers and linkers. In a sense, investigating the system by reviewing source code is like investigating the collapse of a building by reviewing blueprints. The blueprints have valuable information but the actual building may differ in subtle but significant ways from its blueprints.

Furthermore, the voting application runs in a larger environment of software, firmware, and hardware that supports it. A malfunction in that environment, or a misunderstanding between the application program and the environment, can cause anomalous behavior. According to the SAIT report [14], the ES&S iVotronic contains several "off-the-shelf" items, including a microprocessor, various controller chips, and several software drivers that were not reviewed at all. Any of these components could have contributed to problems, especially since one of them is the software driver for the touch-screen itself.

Finally, the process by which the system was designed, maintained, and enhanced requires careful consideration. In particular, it is standard practice in industrial software development to maintain databases of reported problems with software and hardware, and to carefully track the design changes made to respond to them. Such logs include the date, time, and the names of those making the changes. Parts of the software that are badly designed or just especially hard to get right will have more bug reports and changes than other parts, as the same general types of problems often arise multiple times. Inspection of these reports and changes could yield valuable clues about the causes of problems.

Of particular note, Florida uses an older version of the iVotronic software than that used in many other states. Bugs have certainly been discovered and repaired by ES&S in their subsequent software releases.

Such bugs would clearly have been present in the systems used in the November 2006 Sarasota election. Obviously, these could be relevant to the CD13 undervote and, as such, the ES&S records should be made available for investigation. (See Section 4.6 for more on this topic.)

These two modes of investigation, testing and inspection, are most effective when they are done in tandem. When unexpected behavior of the system is observed in practice, the design and implementation should be inspected more closely to explain it. Similarly, inspection of the design will generate questions about the behavior of the system. These questions can then be answered by testing real machines to see how they behave in practice.

3 "PARALLEL" TESTING

One of the most important parts of the investigation was the so-called "parallel testing" performed by the State of Florida's Division of Elections. In truth, the State's parallel testing is a misnomer, and the testing would more accurately be termed a "mock election." [3]

The State conducted its two tests after the election. The first test was conducted with five spare machines, unused in the previous election, but configured as if they would be used for the general election. The second test was conducted using actual machines used during the election. The Jennings and Buchanan campaigns were allowed to specify two machines, each, based on their serial numbers. Jennings selected two machines with notably high undervote rates. Buchanan allowed the State to select two machines at random. A fifth machine was also selected for so-called *ad hoc* tests.

Test scripts were developed based on votes cast in the general election. In each of the two tests, four of the machines received these scripted test votes while a fifth was reserved for *ad hoc* testing, in which test voters followed no particular scripts. The test voters were all staff members of Florida's Secretary of State's Office. The whole process was recorded on video.

The summary report, produced by the State, claimed that no significant discrepancies were discovered in either test [5]. A few apparent discrepancies after testing were identified as errors by the people entering the test votes, based on reviewing the video recordings.

While the State's testing was time-consuming, it failed to address many of the most important questions about the undervote rate. Despite this, the State auditor's conclusions were quite broad:

> [Based on the parallel tests] the audit team concluded that the iVotronic direct recoding [*sic*] devices correctly captured the voters' selections and accurately recorded the votes cast as displayed to the voters on the review touch screens. ([6], page 2)

This conclusion is not justified, because the testing was not sufficiently thorough, as summarized below.

3.1 Narrow scope

The State's tests were not designed to discover defects in the voting machine that might be triggered by the variation in different voters' interactions with the voting system, even though this would be a seemingly obvious area to study. The mismatch between the reported problems and the investigation is evident in the State's final audit report:

[3] The principle of parallel testing, which has been used in California since 2004, is to simulate an election on machines that is so realistic that a machine cannot determine whether it is being tested, or whether it is being used by real voters in a real election. The defining characteristic of parallel testing is that it occurs on a set of voting machines, chosen at random immediately before the election, that would otherwise have been used for real voting. This aspect of parallel testing was intended to prevent potential malicious software in the machines from using the date and time as a cue to detect that they are being tested.

> Although a number of these voters indicated a problem with their initial and final selection for the 13th Congressional District race, the primary focus of the parallel tests is the review screens. . . . [T]he primary question concerning the accuracy of the iVotronic touchscreen is whether the review screens as presented to the voter and ultimately verified and cast by the voter is in fact what was stored as the ballot image. ([6], Appendix C, page 7 – or page 38 of the PDF)

The State has effectively redefined "accuracy" in a voting system as making a correct electronic copy of a review screen. This is not an appropriate definition, since it deems as accurate machines that display arbitrarily wrong information on the summary screen, so long as the machines faithfully copy the incorrect summary screen to storage. A more common-sense definition of accuracy is included in the Federal Voluntary Voting Systems Guidelines, Volume 1, paragraph 2.1.2.c:

> Record each vote precisely as indicated by the voter and produce an accurate report of all votes cast.

As was pointed out in Section 2.1.6, interactions between users and a system can often lead to inaccurate results, and such interactions may well explain the the CD13 problem. But the State's report clearly indicates their lack of interest in such problems. For example, if the machine changed a vote and the voter failed to notice the error on the review screen, then the State does not it consider it an inaccuracy of the machine, even though the machine (in this hypothetical case) would clearly be at fault. For example, the iVotronic write-in bug mentioned in Section 4.6 does not fall within the scope of problems the State tested for, and would only have been caught during testing by accident, if it were caught at all.

Although the State's tests were not *designed* to find problems in the vote capture process, it is possible that some problems could have been caught serendipitously. We've only been able to identify two sentences in the final audit report directly stating conclusions about vote entry errors. Given this small amount of discussion, it appears that anomalies in behavior did not trigger much curiosity by the State's investigators.

> The parallel tests including a review of the parallel test videos did not reveal or identify any latent issues associated with vote selection or the accuracy of the touch screens' tabulation of the votes cast. ([6], page 5)
>
> . . . In addition, attempts to replicate the published reports concerning voter difficulties in making or changing their vote selections did not materialize during this test. ([5], page 8)

There is no explanation of which "latent issues" were eliminated from consideration, nor even a detailed discussion of what anomalies may have been found and determined to be inconsequential.

We viewed several hours of test voting on videos taken by the State of their testing. During our observation we noted several instances of vote selections not registering the first time the screen was touched. It is difficult to judge from the videos, but it appears that the time required to touch the screen before a vote registers is not consistent, and it is clear that some of the test voters had more trouble getting their votes to register than other voters; this could have been due to differences in how the voters operated the machines, or it could have been due to differences in the machines, themselves. This was not mentioned as a "latent issue" in the State's final audit report.

3.2 Test procedure issues

There were several procedural problems with the State's testing, resulting in a constrained and artificial testing scenario. Insights into the causes of the CD13 undervote may have been missed as a result. Problems could have been missed because the tests failed to reproduce accurately the conditions under which the undervotes could occur.

If the undervote was caused by an interaction between the voters and a machine problem, it may not have been detected during testing because the test voters were casting votes in abnormal ways in an abnormal environment. Likewise, if a system malfunction (such as lost votes in the CD13 race) could only be reproduced under a specific combination of inputs and other conditions, it would be unlikely that the State's tests would have generated that combination of conditions.

The **test scripts** were inappropriate, leading to extremely artificial inputs that bore little resemblance to real voting. The test script for each machine was derived from actual votes cast in the election. However, unnatural "vote patterns" were specified for the CD13 race; the vote patterns were originally specified in the Parallel Test Summary Report [5] and later corrected in the Final Audit Report [6]. There are two patterns for casting Buchanan votes, two patterns for casting undervotes in the CD13 race, and six patterns for casting Jennings votes. In each pattern, an initial selection was made for one of the candidates or no candidate, and then the test voter backed up (in one of two ways) to change the vote to the desired final value.

The actual scripts used to test 8 of the 10 machines are available on the Florida Division of Elections website ("Script and Review Screen Checklists"), but only the first two of the six Jennings patterns were used. In these scripts, *not one test vote started with an initial selection of Buchanan*. All votes, including the Buchanan votes, were cast by first selecting Jennings or by abstaining in the Congressional race. Although Buchanan was selected first in some cases on the *ad hoc* machines, [4] only a small number of tests could have been performed when Buchanan was selected first.

The **screen calibrations** were not examined. Problems stemming from calibration problems would only have been caught if the few machines tested were so grossly out of calibration that blatant errors happened to occur during test voting (see Section 2.1.3). *The State made no attempt to determine which machines were or were not properly calibrated.*

The **screen angle** was incorrect. ES&S iVotronic machines are unusual, relative to other touch-screen machines, in that the screen is typically mounted flat on a table, parallel to the floor, where other voting machines typically elevate the screen such that it is angled to face the voter more directly. During testing, the iVotronic machines were attached to a wall, hanging vertically. As a direct result, *any effects that may have resulted from the screen angle would not be observed.*

The **test volunteers** were a small group drawn from the staff of the Secretary of State's office. The voters were being video recorded and had a second person assisting and checking their work, so it is likely that they were much more careful in everything they did than real voters. In addition, the same people voted on the machines for many hours, and undoubtedly became practiced at tailoring their inputs to avoid any problems with the machines.

Unfortunately, the State's tests would be unlikely to detect a wide variety of problems. If a system bug was triggered only with a certain vote pattern, not included in the test scripts, it might not be detected. Likewise, if poor screen calibration interacts with the way a finger might touch or graze a horizontal screen, this effect would not manifest itself on a vertical screen. Voting tests must mimic the actual voting as closely as possible in order to maximize the chances of discovering problems.

[4] This fact was communicated to us by Dan McCrea of Miami, Florida, who viewed the DVDs for the testing of the *ad hoc* machines.

3.3 Recommendations for additional tests

Additional tests are needed for vote selection and vote capture issues, including the steps of voting that precede reviewing a summary screen. Most of these tests could be conducted with substantially less effort than has already been expended in the State's mock election, simply by directing effort to the most important questions. Since user interaction plays such an important role in many of the voter complaints and hypotheses for what went wrong, those issues should be explored much more thoroughly in the testing. Testing should be directed by specific complaints about the behavior of the voting machines, and observation and analysis of the tests should note any occurrences that might be related to vote selection or capture problems. Factors such as screen angles should duplicate the usage of the machines in the election as closely as possible.

With modest effort. a broader range of volunteers could be asked to enter votes, not for a full day as in the mock election, but for 30 minutes each, without long waits between the votes. These test voters could be asked to vote however they wish, and, after an initial few votes, given an opportunity to do whatever they can think of to try to "confuse" the machines. A wider range of volunteers would be more likely to provide more varied inputs to the machines, and to react to the machines in more varied ways (including reactions that could explain the undervoting). To find user interaction problems, all unexpected behavior, such as difficulty selecting candidates, selection of incorrect candidates, unexplained timing variation in the user interface, and so on, should be documented and investigated in more depth. Without the constraints of the artificial scripts and long delays in the mock election, much more thorough testing could occur with much less effort.

Experts should also test the systems. For this, all entries on the ballots and many different combinations of votes should be systematically tested. Testers should be able to follow up with new tests immediately if the machine reacts in an unexpected way to a previous test.

Machine calibration issues should be explored by direct inspection of the iVotronic systems. An operator, with a pointed stylus, could press at various points on the screen and photographs or video could be used to determine whether there are calibration errors.

Interface timing complaints should be tested much more systematically. Testers should explore factors affecting vote selection. Tests should include touching the buttons on the screen for various lengths of time to understand the delays imposed by the smoothing filter, noting whether those delays are consistent with any variations in voter demographics, voter behavior, ballot design, voting machine calibration, or other related factors. Special attention should be paid to whether, under any circumstances, delays in updating the displayed selections occur. Any unexpected events should be noted and investigated further, including reinspecting the source code to explain any non-deterministic behavior.

With more resources, an academic-quality user test could be performed as described in Section 2.1.4. Such a test would probably be more costly and time-consuming than everything else we have proposed in this report, but it may be the only route to a definite answer about the effects of ballot design on the CD13 undervote.

4 INSPECTING THE SYSTEM DESIGN AND IMPLEMENTATION

The State commissioned a group of academic computer scientists to perform an analysis of the source code to the ES&S iVotronic machine with the intent of determining whether a software bug may have contributed to Sarasota's high undervote rate [14] (hereafter, the "SAIT report"). The SAIT report found numerous serious problems, including a vulnerability that would allow for the creation of a voting machine virus that might be able to spread from one voting machine to another. Many details, along with many other "unrelated" bugs found, were reserved for appendices that were released neither to the general public nor to the plaintiffs. The SAIT report considered a number of different hypotheses as to how software flaws

may have led to the undervote rate and dismissed them all. The report contained numerous caveats that its analysis could well have overlooked subtle flaws, but the report is being treated by many as conclusively closing the door to further analysis.

In some ways, the analysis in SAIT report is very impressive. The analysis of security issues in the software is especially deep. However, as Section 2 points out, there is more to inspect than just the software's source code.

The primary deficiency in the State's software investigation is its defined scope, as was also the case in the State's parallel testing. There were many aspects of the system design and implementation that were not studied at all, as is clearly explained in the SAIT report. The authors spent very little time doing hands-on experimentation on actual iVotronics. The authors are very careful to be explicit about the scope of their work, and to state their many unverified assumptions. However, these caveats have been lost in the press reports, and, in some cases, even in the latter sections of their own report.

There were also gaps in the software analysis, usually stemming from failure to probe sufficiently deeply into specific areas of interest, or from failure to link the source-code analysis with other aspects of the investigation. Specifics are discussed below.

4.1 Build environment

The SAIT team was given source code for the system, and was able to experiment with iVotronic systems that were purported to be running executable code built from that source code. However, the SAIT team did not build the executables, themselves, from the source code.

> The Firmware Compilation Environment. We assume that the tools used to build the firmware from the source code:
>
> 1. Worked correctly;
> 2. Comply with the ANSI C programming language standard;
> 3. Do not have any bugs or unexpected behavior.
>
> We assume that the firmware image provided to us was compiled correctly from the source code provided to us. We also assume that the firmware image provided to us was the firmware image that was actually executed by the iVotronic machines on Election Day. These assumptions imply that the executable software executed by the iVotronic systems during the election matched the source code we examined. As our study focused *only* on the source code, we did not attempt to reconstruct the executable firmware image. Both ES&S and FLDoS told us that the firmware compilation environment worked correctly. (SAIT report, p. 18)

The lack of a build environment rendered the investigation unable to answer several important questions. First, was the binary executable that ran on the machines consistent with the source code? If not, the explanation for the CD13 undervote may lie in the discrepancy between the executable binary and the source code. Perhaps the source code, as held in escrow by the state, was inconsistent with the compiled binaries. The SAIT team had no way to verify this, further rendering them unable to determine if either malicious behavior or simple bugs might be in the binary executable that are not represented in the source code. (Turing laureate Ken Thompson famously demonstrated the feasibility of malicious attacks where the source code did not correspond to the compiled executable [13].)

More importantly, the inability to build and execute the software limits the ability of the source-code review team to perform a thorough examination. For example, a common analysis technique is to instrument the source program to print or log interesting events, thus improving the examiner's understanding of the

progression of events that occur as the code executes. Likewise, code can be instrumented to carefully study the interaction between the code and the hardware, possibly detecting flaws in the hardware itself. Furthermore, portions of the voting software could be extracted from the main application and executed in a "test harness" where their behavior could be systematically studied. Techniques such as these can identify subtle flaws in the voting system, whether from software or hardware, any of which could have been relevant to the Sarasota undervotes. The SAIT team was unable to utilize these analysis techniques.

4.2 Calibration error

Examination of the source code is important for understanding how the calibration process works internally, how finger-presses are converted to coordinates, and what consequences there might be if an iVotronic is poorly calibrated. Unfortunately, calibration issues were dismissed without a detailed analysis (SAIT report, p. 48). Doing this examination properly would require analysis of the source code as well as instrumentation of actual iVotronic machines to understand the stream of data events that are generated by touches on the screen, particularly with different parts of the finger and pressing at different angles. The SAIT team never attempted to instrument an iVotronic machine in this fashion. Such an effort would require building customized versions of the iVotronic software (see Section 4.1, above).

4.3 Smoothing filter

The quotation from the ES&S memo about the smoothing filter in Section 2 raises a number of questions. The memo claims that the effects of the smoothing filter will vary from machine to machine, yet if the smoothing filter is implemented purely in software, identical on every iVotronic, there should be no variation in its behavior from one iVotronic machine to the next, or from time to time. As we discussed in Section 2.1.5, non-determinism is a sign of potential bugs, and, in this case, inconsistent behavior may cause unpredictable problems for the voters. The SAIT report briefly addressed this issue:

> A smoothing filter is a mathematical procedure for damping transient touch screen effects such as the voter altering the position of her finger or changing the pressure exerted by the finger on the screen. The allegation has been floated on Internet newsgroups that the iVotronic touch screen filter could have caused the undervote. No explanation has been offered how the effect would confine itself to a single race on a single screen. The touch screen filter does not act differently on different screens. (SAIT report, p. 48)

The SAIT report does not contain a detailed analysis of the software that performs the smoothing filter, e.g., to see whether it might have bugs or otherwise might interact with other parts of the iVotronic software in an unexpected fashion; there is only the paragraph quoted above. Likewise, it's unclear how the smoothing filter interacted with Sarasota's voters. If the smoothing filter caused voters' genuine presses on the screen to be ignored and the voter went on without verifying their selection, then it is inappropriate to blame the voter when the machine's design is at fault (see Section 2.1.2).

4.4 Non-deterministic behavior

When computer software has latent bugs, these bugs often fail to manifest themselves during normal use and testing. Even when such bugs do manifest, they may not do so in a consistent fashion. Often referred to as "Heisenbugs," a pun on Heisenberg's Uncertainty Principle, it can be quite challenging to locate and repair these bugs. Common causes of Heisenbugs include the use of uninitialized memory, the overflowing of buffers (whether accidental or malicious), or the lack of anticipating some error or exceptional state. The

SAIT report has a redacted appendix that claims to detail a large number of bugs that are "unrelated" to Sarasota's undervotes, but they may in fact be relevant.

Of particular note is the unusual design of the iVotronic's software. In modern computer system design, an *operating system* runs on the hardware, handling the hardware devices' needs, including servicing interrupts and doing input/output operations. Then, *applications* run above the operating system, which provides applications with a simpler, more abstract view of the hardware. An application might then say "give me the next key press" without having to know anything about how that key press is acquired and interpreted. The iVotronic software is constructed in a fashion more consistent with DOS software from the early 1980's (see the SAIT report, Section 6.3). All operating system-like functionality is handled directly within the iVotronic software, which runs directly on the hardware. This means that error-prone and sensitive operations are happening within the main voting application.

Analyzing software built in this fashion for correctness requires significant effort. The SAIT report describes, for example, the way that global state must be correctly handled:

> We then attempted to verify that all such variables were declared as "volatile," so that the compiler would not perform unsafe optimizations (e.g., suppression of apparently-redundant load and store operations) on them. Most of the asynchronously updated global variables were not declared to be volatile, but we do not believe this mattered with the particular compiler used on the iVotronic software. That is, with there being so many cases, if the compiler performed optimizations of the kind that would be unsafe on these variables: (a) the results would probably have been detected in testing; (b) the probability of failure would have been uniform over time, affecting all races with equal probability; (c) the failures would be exhibited in ways other than just undervotes. (SAIT report, page 33)

It's insufficient to state that such problems "would probably have been detected in testing" when so many other problems were clearly not detected in testing. Likewise, there is no reason to assume that such failures would happen uniformly and in ways beyond causing undervotes. Doing this sort of analysis properly would require examination of the actual machine code, generated by the compiler, to see whether it does or does not operate in a safe fashion. Furthermore, the iVotronic software could be executed using debugging or simulation tools that could potentially detect when such problems occur (detailed above in Section 4.1).

The SAIT report then goes on to discuss the various kinds of global state variables in use in the program and the conditions under which they would be safe to use. In addition to examining the source code, it might be appropriate to add new code to the system that deliberately changes the same global variables, or systematically simulates interrupts occurring at various times. If such artificially induced failures lead to undervotes, that would then suggest that genuine failures could also have the same result. This technique is similar in philosophy to a powerful testing methodology called "fuzz testing," which involves feeding random inputs to a program and then running it to see how it fails. *The SAIT report explicitly disclaimed the use of analysis techniques such as fuzz testing and the use of debugging or simulation tools.*

Finally, it might be the case that there are hardware-dependent problems with iVotronic systems, which might well manifest themselves in a non-deterministic fashion. Felten discusses such issues with Diebold's system [4], where a design flaw related to the precise timing of electrical events on the motherboard ultimately led Diebold to replace the motherboards of 4700 Maryland voting machines. None of the analyses yet performed on Sarasota's iVotronic systems have considered the possibility of similar failures. Such problems are difficult to validate, although ES&S may have become aware of such issues in the past and upgraded their hardware designs to address the concern. To the best of our knowledge, the SAIT authors did not have access to internal ES&S documents that might have illuminated this issue, nor did they make any attempt to determine whether the iVotronic hardware, itself, may have suffered from intermittent faults.

4.5 Buffer overflows and viruses

The SAIT report describes how the software engineering practices used to create the iVotronic system leave it vulnerable to a variety of security attacks:

> [T]he iVotronic software copies a variable-length nul-terminated (C-style) string from the ballot definition file into a fixed-size stack-allocated buffer. If the string in the ballot definition is too long, this will overflow the bounds of the fixed-size buffer and overwrite other parts of memory. An attacker could use well-known techniques to exploit this bug, inject malicious code into the address space of the iVotronic machine, and cause the processor to begin executing that malicious code. At this point, the attacker has complete control over the iVotronic: the iVotronic is infected.
>
> We found numerous instances of this type of bug. Misplaced trust in the election definition file can be found throughout the iVotronic software. We found a number of buffer overruns of this type. The software also contains array out-of-bounds errors, integer overflow vulnerabilities, and other security holes. They all arise due to the fundamental architectural flaw of misplaced trust. (SAIT report, p. 57)

The report goes on to detail how a virus could be engineered to spread from one machine to the next via the PEBs (personal electronic ballots), normally used throughout the voting day to activate the machines for each voter. *The SAIT authors made no attempt to examine the actual iVotronic machines, PEBs, CompactFlash cards, or any other materials for evidence of such an attack.*

During the state's parallel tests (see Section 3), the state additionally selected a handful of machines, extracted the EEPROMs containing the iVotronic's software, and used standard commercial tools to compare these binary images to the binary images on file with the State. No discrepancies were found, although it would be reasonably straightforward for viruses to overwrite themselves to remove evidence of their presence. To the best of our knowledge, the state's examination did not look at PEBs or the CompactFlash cards used in the election. *While we have no other evidence to suggest that a virus-based attack may have occurred, neither the state's parallel tests nor the SAIT report made an effort sufficient to rule viruses out of consideration.*

Buffer overflows can occur even without the presence of malicious code. The damage caused by such buffer overflows, if and when they occur, could have a variety of ill effects on the voting system. A number of commercial tools have been developed to identify and repair buffer overflow issues. By using these tools, an examiner could detect if these problems manifested themselves during actual election conditions.

4.6 Bug tracking and version control

ES&S, like any modern software firm, can be presumed to use modern software development and management tools. Most notably, they must certainly use a *version control system* and a *bug tracking system*. Version control systems allow for all changes to the software to be tracked. If a developer introduces a change that caused problems, the change could be easily identified, undone, and repaired. In a sense, a version control system would allow an examiner to turn the clock forward and backward on ES&S's development efforts, observing changes made beforehand and afterward to the source code used in Florida. When such changes are made, it is also standard practice to annotate those changes to explain what happened (e.g., "fixed bug with calibration"). The code changes and the annotations would provide valuable insight into the processes used to develop the iVotronic system.

In a similar fashion, any modern software development process will include a system to track issues. Consider a hypothetical bug discovered by an ES&S customer and reported to the vendor. The report will

be stored in a database and assigned a tracking number. Subsequent reports may be assigned their own tracking number or may be merged with the original. Bugs are then typically assigned to developers who repair the software. Bug tracking systems often allow developers to discuss possible approaches toward repairing these bugs, retaining these discussions as a record of the developers' thought processes. Also, bug tracking numbers can then be referenced in the source code, inside comments that are read by developers but ignored by the compiler (e.g., "we're doing extra work here on the calibration step to address bug #345"). Bug tracking numbers may also be referenced in the version control system's annotations. All of this data would provide valuable insight into the development process. *The SAIT authors had no access to ES&S's bug tracking system or version control system.*

Given that Florida is running a relatively old version of the iVotronic software (version 8.0.1.2 versus North Carolina's version 9.1.2.0), it's entirely possible that bugs germane to the undervote rate in Florida may have been repaired in the newer version 9 variants of the iVotronic software. Appropriate access to ES&S's internal development processes would greatly assist an examiner in understanding whether such relevant bugs has already been discovered and repaired.

Consider, for example, a recently disclosed bug in the newer version of the iVotronic's software that is used in North Carolina and several other states. The problem occurs under unknown conditions, a small percentage of the time, and could fail to capture the intent of the voter by denying him or her the chance to cast a write-in vote. ES&S notified North Carolina of this serious software flaw in its version 9.1.2.0 iVotronic firmware (reproduced in full in Appendix A), stating:

> The item affecting the iVotronic voting system is a firmware issue which affects the way the iVotronic displays a write-in candidate. The firmware issue is limited to iVotronic version 9.1.2.0 and occurs two to three percent of the time when the iVotronic is being used. When the error is present, the iVotronic does not display a choice for a voter to write-in a candidate's name for a particular office. The display allows a voter to select from the list of predetermined candidates but occasionally may not include a line for a voter to write-in a candidate.

The same issue was discussed in Pennsylvania's amended certification for the iVotronic system [2].[5] This write-in bug, as stated, may not have caused the undervote rate observed in Sarasota. However, this bug may well be the tip of a larger iceberg. If this bug existed in the newer version of the iVotronic software, perhaps the same bug, or some variant on it, existed in the older iVotronic software used in Sarasota. Since ES&S was aware of this bug in their newer software, perhaps they were also aware of related bugs in the software used in Sarasota. An examination of ES&S's internal records would greatly aid the process of uncovering such problems. A thorough examination would look at the steps taken by ES&S to address the write-in bug, among other bugs, and would then examine whether these bugs had an effect in Sarasota.

5 CONCLUSIONS AND RECOMMENDATIONS

Without doubt, the undervote rate in Sarasota County's general election in November 2006 reflected a failure of the ES&S iVotronic systems to accurately capture the intent of many Sarasota voters. While Sarasota County, the State of Florida, and its academic computer science experts have performed certain analyses, we still have no conclusive evidence demonstrating the cause or causes of the unusual undervote rate.

We recommend additional expert analysis of the source code for these voting systems, including debugging and simulation tests which may be more likely to trigger latent flaws if they are present. We likewise

[5] Michael Shamos, one of the co-authors of the SAIT report, also does voting system analysis for the Commonwealth of Pennsylvania, so the SAIT authors were certainly aware of this issue, despite not raising it in their report.

recommend that experts be given unrestricted access to ES&S's internal bugs databases and software repository, where they may find additional evidence that could lead to the discovery of what software bugs, if any, contributed to Sarasota's undervote rate.

We also recommend further analysis of the iVotronic systems used in the election and still sequestered in a Sarasota warehouse. We recommend that a sampling of these machines be carefully examined for evidence of screen miscalibration and touch sensitivity.

These analyses could be performed by a relatively small number of experts in about a month's time. Should such analyses be able to conclusively determine a reproducible explanation for the undervotes, this would have significant ramifications, both for the ongoing legal battle between the two parties for control over Florida's 13th Congressional District seat, as well as for the broader discussion of how voting machines should be designed, tested, certified, and analyzed.

ACKNOWLEDGMENTS

The authors gratefully acknowledge feedback received from our colleagues who reviewed drafts of this report, including Peter Neumann and Dan McRae. We also thank Joyce McCloy of the North Carolina Coalition for Verified Voting for the documents she located.

REFERENCES

[1] L. Bennett. *ES&S letter to Florida users*, Aug. 2006. http://www.ncvoter.net/downloads/ESS_Aug_2006_iVotronic_FL_memo.pdf.

[2] P. A. Cortés. *Ammended Report of the Examination Results of the Election Systems & Software, Inc. iVotronic Touchscreen Voting System with Unity Software*. Commonwealth of Pennsylvania, Department of State, Apr. 2006. http://www.hava.state.pa.us/hava/lib/hava/votingsystemexamination/sec_final_rpt_-_es&sivotronicamended.pdf.

[3] E. Felten. *Diebold's Motherboard Flaw: Implications*. Princeton University, Feb. 2007. http://www.freedom-to-tinker.com/?p=1081#comment-179915.

[4] E. Felten. *Sarasota: Could a Bug Have Lost Votes?* Princeton University, Feb. 2007. http://www.freedom-to-tinker.com/?p=1126.

[5] Florida Department of State, Division of Elections, Tallahassee, Florida. *Parallel Test Summary Report*, Dec. 2006. http://election.dos.state.fl.us/pdf/parallelTestSumReprt12-18-06.pdf.

[6] Florida Department of State, Division of Elections, Tallahassee, Florida. *Audit Report of the Election Systems and Software, Inc's, iVotronic Voting System in the 2006 General Election for Sarasota County, Florida*, Feb. 2007. http://election.dos.state.fl.us/pdf/auditReportSarasota.pdf.

[7] L. Frisina, M. C. Herron, J. Honaker, and J. B. Lewis. *Ballot Formats, Touchscreens, and Undervotes: A Study of the 2006 Midterm Elections in Florida*. Dartmouth College and The University of California at Los Angeles, Feb. 2007. http://www.dartmouth.edu/~herron/cd13.pdf.

[8] F. Gluck, H. Allen, and M. Saewitz. Most callers report voting problems. *Sarasota Herald-Tribune*, Nov. 19 2006.

[9] D. W. Jones. *Observations and Recommendations on Pre-election testing in Miami-Dade County*, Sept. 2004. http://www.cs.uiowa.edu/~jones/voting/miamitest.pdf.

[10] N. G. Leveson. *Safeware: System Safety and Computers*. Addison-Wesley Publishing Company, 1995.

[11] *Plaintiff Jennings Ex. 8.* (prepared by Prof. Charles Stewart III, Political Science Dep't, MIT, for Dec. 19, 2006 evidentiary hr'g), Jennings v. Elections Canvassing Comm'n of the State of Florida et al., No. 2006 CA 2973 (Circuit Ct. of the 2d Judicial Circuit, Leon County, Fla., filed Nov. 20, 2006), reproduced in 2 Appendix to Emergency Petition for a Writ of Certiorari A-579-80, Jennings v. Elections Canvassing Comm'n of the State of Florida et al., No. 1D07-11 (Fla. 1st Dist. Ct. of Appeal, filed Jan. 3, 2007).

[12] C. Stewart. *Declaration of Charles Stewart III on Excess Undervotes Cast in Sarasota County, Florida for the 13th Congressional District Race*, Dec. 2006. http://www.heraldtribune.com/assets/pdf/SH81421120.PDF.

[13] K. Thompson. Reflections on trusting trust. *Communications of the ACM*, 27(8):761–763, Aug. 1984. Also appears in *ACM Turing Award Lectures: The First Twenty Years 1965-1985*, Copyright 1987 by the ACM Press and *Computers Under Attack: Intruders, Worms, and Viruses Copyright*, Copyright 1990 by the ACM Press. http://www.acm.org/classics/sep95/.

[14] A. Yasinsac, D. Wagner, M. Bishop, T. Baker, B. de Medeiros, G. Tyson, M. Shamos, and M. Burmester. *Software Review and Security Analysis of the ES&S iVotronic 8.0.1.2 Voting Machine Firmware*. Security and Assurance in Information Technology Laboratory, Florida State University, Feb. 2007. http://election.dos.state.fl.us/pdf/FinalAudRepSAIT.pdf.

A ES&S LETTER TO NORTH CAROLINA

March 14, 2006

VIA ELECTRONIC MAIL
AND OVERNIGHT DELIVERY

Mr. Keith Long
NC State Voting Systems Project Manager
State Board of Elections
6400 Mail Service Center
Raleigh, NC 27699-6400

Dear Mr. Long:

Pursuant to Section 163-165.9A(4) of the North Carolina General Statutes, Election Systems & Software, Inc. ("ES&S") must provide the North Carolina State Board of Elections (the "Board") with notice of any relevant defect which has occurred in its voting system that will be used in the State of North Carolina (the "State"). In accordance with such notice requirements, ES&S is providing this letter to the State to notify it that ES&S has become aware of two items which could potentially affect the function of the ES&S Model 100 precinct tabulator ("Model 100") and the ES&S iVotronic touchscreen system ("iVotronic"). ES&S has addressed each item and has taken the necessary corrective measures to limit any affect the items may have on the State's ability to conduct its elections. The issues affecting the relevant voting systems and the corrective measures ES&S has taken are outlined below.

The item affecting the ES&S Model 100 is limited in scope to the PCMCIA cards used to load the election definition into the Model 100s. ES&S has identified a batch of PCMCIA cards which were not properly manufactured and as a result may cause the Model 100 battery to "drain" more rapidly than normal. ES&S has identified the customers who may have received the PCMCIA cards from the affected batch and are in the process of working with each customer to verify if the PCMCIA cards are working properly. As the State was one of the customers identified by ES&S as potentially receiving PCMCIA cards from the affected batch, ES&S has notified the State of the issue and has been working with officials from the State to ensure the PCMCIA cards delivered to the State work correctly. To date, ES&S has provided the State with one thousand (1,000) PCMCIA cards which currently reside at the State's warehouse. No PCMCIA cards have been delivered to counties within the State and ES&S has instructed its staff in the State not to send out any PCMCIA cards to any county until the cards have been thoroughly tested to ensure they are operating properly. ES&S will continue to work with the State to ensure that only proper functioning PCMCIA cards are delivered to the counties within the State.

The item affecting the iVotronic voting system is a firmware issue which affects the way the iVotronic displays a write-in candidate. The firmware issue is limited to iVotronic version 9.1.2.0 and occurs two to three percent of the time when the iVotronic is being used. When the error is present, the iVotronic does not display a choice for a voter to write-in a candidate's name for a particular office. The display allows a voter to select from the list of predetermined candidates but occasionally may not include a line for a voter

to write-in a candidate. ES&S has addressed this issue in its latest version of iVotronic firmware, version 9.1.4.0 that is included in Unity Release 3.0.1.0. Unity Release 3.0.1.0 has been successfully tested by an Independent Testing Authority ("ITA") and is awaiting National Association of State Election Directors approval.

It has recently been brought to ES&S' attention that there may be a number of counties in the State conducting local school board elections which will require the use of the write-in option on the iVotronic. Should the State wish to upgrade its Unity software to version 3.0.1.0 which includes the iVotronic write-in firmware enhancement, please advise at your earliest convenience and ES&S will begin the necessary steps to accomplish this upgrade in time for the May primary. ES&S has included a copy of the 3.0.1.0 ITA completion letter and the final full-text test report on CD to be delivered via overnight courier to your attention. Please note that the Model 100 Precinct Scanner and Model 650 Central Count Scanner firmware versions remain unchanged in 3.0.1.0. They are identical to the versions already certified for use in North Carolina.

If you have any questions or need additional information, please contact me directly.

Sincerely,

Timothy J. Hallett, Esq.

cc: Aldo Tesi - President and Chief Executive Officer, ES&S
Eric A. Anderson, Esq. - General Counsel, ES&S
Gary Crump - Chief Operating Officer, ES&S
Ken Carbullido - Senior Vice President, ES&S
Mac Beeson - Account Services Manager, ES&S
Steve Pearson - Vice President Certification, ES&S