It was spring of 2002, and my company was about a year into developing its next version of a fiscal cash register. Fiscal cash registers require certification from a local authority of the country where they operate and must comply with the relevant requirements for its construction and mode of operation. Usually the certifying authority issues a certificate that allows vendors to sell the cash register in the country for a period of three years, during which the producer can’t make any modifications to hardware or firmware. Any change made requires re-certification, which takes time and incurs additional costs, so we wanted to be sure we had no hidden problems.

It was the day before a long weekend and everyone was happily planning their time off. At about noon, however, the production department informed us of a problem that appeared during the tests of the main board in one of the cash register models. More than half of the boards failed at the beginning of a basic test, the first test after assembly.

Such failures in the past had been usually the result of a problem with component mounting, i.e., it
was inappropriately assembled or damaged during the process, but this time everything there seemed OK. One of our colleagues checked the board's components and all appeared well. The next thought was that the tester was damaged; we had noticed that sometimes the same board would pass the test on another try.

After connecting an oscilloscope and making many test attempts, we noticed the source of the problem: a programmable logic chip (GAL16V8) didn’t always propagate a key signal to the other chips.

At first the problem seemed obvious – the GAL hadn’t been programmed before soldering or contained the wrong contents, and so didn’t work correctly. But when we unsoldered some GALs and checked them, they all had appropriate contents and they all passed the test vector procedure. Yet they still weren’t able to work on the board. Component problem? We had used these components in many other boards in different cash register models, so we decided to check those other boards as well. We found only one that behaved the same way.

We prepared a special version of the failing board – with a socket for GAL. We then collected different revisions of GAL chips and started testing. After some minutes we realised that it was only the newest version of the GAL that caused a problem. It seemed impossible – a well-programmed GAL, tested in the programmer using specially prepared test vectors, and half the time it couldn’t work in a circuit?

We analysed the board configurations in which they were working, and we noticed that the GAL had problems only where it was functioning as an address decoder. On boards where its functionality was as a multiplexer and decoder, the problem didn’t exist. This discovery explained why only two kinds of boards had a problem with new GALs, but we wanted to find out why the board sometimes did pass the test.

We connected the board to the tester and observed the GAL’s power supply during repeated tests. It turned out that only when the power supply on the GAL fell below 1V (for older versions of the GAL, 2V) before restarting the test, the board passed. It might seem that cash registers never work in such conditions, but when somebody turns off a cash register and very quickly turns it back on, such might be the case. Under a quick power cycle, the voltage on GAL doesn’t fall below 1V and cash register goes dead.

We prepared a special testing circuit to reliably simulate the problem during production test. We also described our observations of the GAL’s behaviour and sent them to the manufacturer. In the meantime, we had to prepare a remedy that allowed us to “repair the GALs” so they would no longer fail in this way. After many attempts we noticed that it was enough (after power up) to change the state one of GAL’s inputs to restore it to proper operation.

We made a small modification on our boards. The GAL had a free pin, so we connected that to the µC reset signal. The production department could now continue with assembling the devices.

After some weeks we received a possible explanation for this odd behaviour from the GAL’s manufacturer. The latest GAL revision contained a newly optimised internal structure which, among other changes, had altered the reset threshold of its internal circuits. It was probably the source of
our problem with this board.

Lesson learned: An improvement to one component does not necessarily lead to the improvement of the whole design.

Stayed tuned for part 2 of this series, where we solve another problem with a cash register board.

Andrzej Winczura has worked in the R&D department of ELZAB (an electronic cash register manufacturer in Poland) for almost 20 years, starting as a hardware and firmware designer, then developing applications for embedded systems.

Related articles:

- Programmable logic: A practical introduction for beginners
- The new tool: configurable hardware
- Programmable logic devices
- Perform five common debug tasks with an oscilloscope