LIPS

electro®

A look at a 1991 mechatronic lock from Lips Sloten, Netherlands

By Walter Belgers, M.Sc

Lips Eloctro
version 1.0, 30 July 2025

Walter Belgers <walter+lips@belgers.com>

Table of Contents

Introduction

Recently, I acquired a Lips Eloctro lock. It came from the Lips company collection that the current owner is selling. I was intrigued by it, as it is one of the first mechatronic locks for home use, which I had never seen or heard about before. *Mechatronics* is the combination of mechanical, electronic and software engineering. I was able to buy a second lock and some more keys, both old stock, and another Eloctro lock from the Lips collection.

In this paper, I will start by briefly looking at the history of Lips and how lock companies ventured into the field of mechatronic locks in the 1980s.

I will then go into the design of the Eloctro, explaining what are its special features, its patented parts and some background on the design decisions that were made, based on public information from around the time of introduction.

After that, I will show the working of the lock and its components, with images of the locks I own, followed by some insights on how the systems works and some work I have done trying to reverse engineer the locks.

I have not been able to get into contact with people that worked for Lips Sloten or the Lips Eloctro in particular who could provide additional information.

Thanks go out to the users Hobbs, Evo and mh from LPU for their pointers to early electromechanical systems, GBI Telder Kampen for a good deal on Eloctro surplus parts, Jan-Willem Markus for pointers on locks and 80C52 reverse engineering and KIVI for allowing me to browse their old magazines. Special thanks go to Evert Nieuwkoop, who was very helpful in providing information about the working of the Eloctro as explained in documentation from the time which he still owns.

Thanks also go to Jan-Willem Markus and Evert Nieuwkoop for reviewing a draft version of this paper.

If you have found errors, would like to comment or can help with further research, please contact the author at walter+lips@belgers.com.

History

About Lips

Lips was a Dutch company, founded by blacksmith Jacobus Lips in 1871. One of the products Lips sold was safes. In the early 20th century, Lips also started selling mechanical locks, as one of the first companies in the Netherlands to do so. Lips has always been a prominent brand in the Netherlands, both in locks as well as safes.

Lips was bought by the British company Chubb in 1971. Over the years, it created separate entities for locks (Lips Sloten), safes and security. In 1984, Chubb-Lips was sold to Racal and later Williams. Lips continued making safes and mechanical locks. In 2000, the lock factory was bought by ASSA ABLOY, which still sells locks and cylinders under the Lips brand.

The need for mechatronic locks

When the Eloctro hit the market in June 1991, traditional mechanical (pin tumbler) locks had been around for 150 years. Companies had been experimenting with changing the design, to provide more security or to be able to patent the system and control the sales of keys. (Mechanical) master keying had also been around for a long time. In a master keying system, multiple cylinders in the system each have their own key (called a *user key*), but (subsets of) these cylinders can also be opened using *master keys*. In large systems, there can be multiple levels with submasters and a general master key.

Master keying is used in large installations, such as hotels. It allows for each guest to have their own key for opening just their own room, but also have (sub)master keys opening a number of (or all) cylinders in the facility, to be used for cleaning or by the hotel manager.

It was the hotel business that started looking for alternatives to traditional mechanical locks. Upcoming new systems offered them functionality that mechanical locks could not provide, such as the possibility of easily invalidating a key when lost, the possibility to create multiple keys for the same room and doing all of this without needing to go to a locksmith. *Interchangeable Core*s already existed, which allows the core of the cylinder to be physically replaced with another core using a special *change key*. This was still too cumbersome and expensive for a hotel where ideally, you invalidate the user key after a guest checks out without having to replace locks.

From the 1970s, magnetic stripe cards and plastic cards with holes in them were introduced as keys. The stripe cards stayed around for several decades, before being replaced with more modern contactless NFC cards. Hotel locks are currently using their own physical form factor, which differs from what most consumers and companies use. Cards are fairly cheap and can be easily replaced or reprogrammed.

Electronic systems were (and are) also interesting for customers seeking more security control, requiring high-security locks with monitoring features. An electronic lock can register which key is presented to the lock, providing an audit trail. With electronics, more key combinations can be distinguished than with purely mechanical locks, which is a selling point for (very) large installations.

There are also downsides to electrical systems. In most cases, the locks need to be electrically powered. This requires batteries or power lines to be present in all doors. For remote programming, a data connection to a central controlling computer is needed as well[1]. Another downside is the cost for such a system. The higher purchase cost can make sense however, when the lower operating cost is taken into account.

At this point, it is good to realise that different customers have different requirements, such as:

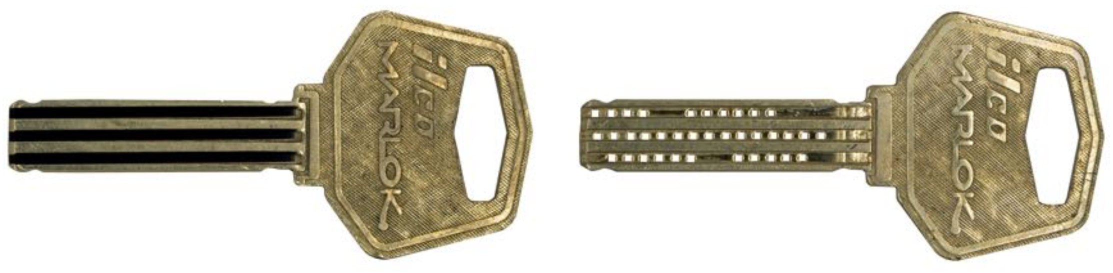
- The price.
- The form factor: can it be fitted in existing doors?
- Functionality: does it support master keying, re-keying, temporary access, etc.?
- How many possible different keys can be distinguished in the system?
- Protection against forced attacks such as breaking, drilling, etc.
- Protection against surreptitious opening such as picking, decoding, etc.
- Key control: who can copy the key, who can get key blanks, can a working key be 3D-printed, etc.?

For a lock manufacturer, all these aspects need to be taken into account, to launch a successful new product onto the market.

Early mechatronic locks

Marlok Company (now Kaba Ilco Corp.) marketed a system called ML-3000 or Solaire. The patent (US 4,432,142) was registered in 1982. This system looks like a traditional mechanical system, with a somewhat normal looking key. The key contains three rows of small holes. The holes are covered so every key looks the same. This has the advantage that a picture or the key will not allow you to duplicate it.

The holes are read by the cylinder using infrared light (which penetrates the covering). The middle row of holes provides timing information, the other rows the actual key code. When the code is correct, the locking mechanism opens.

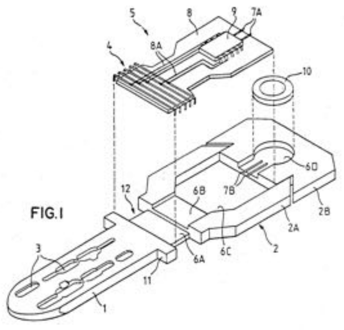


Marlock ML-3000 key (left) and with covering removed (right)

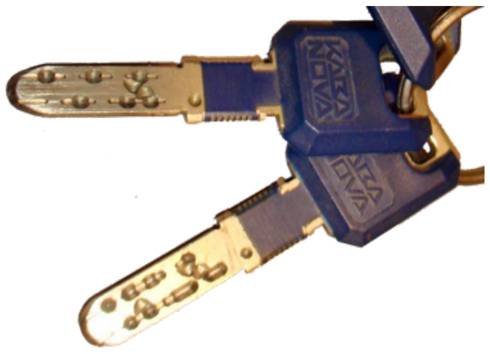
[1] Recent advances have made it possible to have electronic locks that do not need power nor data connections, see for instance the iLOQ system (https://www.iloq.com/)

You could say the holes are the bitting. On traditional keys, there could be 5 positions for pins, where each pin can be of 9 different heights, giving 9^5 (about 59.000) possibilities. The Marlok appears to have 30 positions that can only have two states: a hole or no hole, giving 2^{30} (about 1 billion) possibilities. This system has electronics in the cylinder to detect the holes in the key (and electronics in the door to process that information), but there are no electronics in the key. The key is not used to turn open the lock, that mechanism is separate. The user experience is therefore different from that of a traditional`` key.

In 1985, Bauer Kaba AG (now dormakaba Holding AG), introduced the Kaba Nova system. This implements an electronic extension to an existing Kaba dimple lock. It was patented in 1984 (US patent 4,686,358). It is unknown how the electronics work in the Kaba Nova. Most probably, it just contains a small ROM chip with a static identifier. The contacts are brought out on both sides of the key, making the key reversible. In this system, both the bitting and the identifier need to be correct for the cylinder to open.



Patent US 4,686,358



Kaba Nova keys

A year later, in 1986, ZEISS IKON AG presented the IKOTRON system. Like the Kaba Nova, it was also created by adding an electronic component to an existing mechanical key. The name IKOTRON was a registered trademark from 1991 until 2012. The most important difference with the Kaba Nova is that the IKOTRON key and cylinder connect wirelessly instead of by wire, using a technique described in patent DE 3447 560 C2. It uses a high frequency electromagnetic field, which is always on. A wireless solution is more robust because it eliminates issues with the contacts.

Zeiss-IKON IKOTRON key
Image source: http://ikotron.de/

The Abloy Combisec saw the light in 1988. It is very similar to the IKOTRON: Abloy (now Assa Abloy AB) took an existing mechanical system and added electronics to it. It came with a Psion II organiser with an add-on hardware module, to program keys. This key also transmits data wirelessly.

Abloy Combisec key

There are probably more examples of mechatronic locks in the 1980s, but overall, they were quite a niche product, only used in specific situations such as hospitality, large facilities and for institutions requiring a higher level of security than mechanic locks could provide.

In the 1980s, mechatronic locks were made by traditional lock companies, that had a lot of mechanical know-how. They added electronics to their existing mechanical locks. Nowadays, we see a shift from focussing on the mechanical part to focussing on the electronic part of locks. In the end, it's mechanics that open the lock, but the key is more and more becoming an electronic wireless device, either a key fob, key card or embedded in a mobile phone in the form of an app. New products enter the market that originate from companies or inventors that focus mainly on the electronic part. In those locks, there are often mechanical vulnerabilities.

The Lips Electro was one of the first locks to take the approach of having the code of the key only in the electronics and not in the bitting (as well). With its long tradition in mechanical locks, Lips knew very well how to design a lock to a very high mechanical standard. With the Electro project, Lips sought out to have the same high standard in the electronic part of the lock. Let us now look at how the project came about.

First look at the Lips Eloctro

Market introduction

Lips claims that the Eloctro is the first locking system for which it is impossible to copy keys. The lock needs to be powered, either stand alone, using a box with six 1.5 Volt batteries that is hidden in the door, and/or by using a 9 Volt power supply wired into the door.

The system has keys that work wirelessly and transit a unique code to the lock. Apart from normal (user) keys, there are three special keys. These are a (green) key validation key, a (red) key deletion key and up to four (blue) key suspension keys. The green key is used to program a specific user key into the lock. With the red key, a user key can be deleted. The blue keys are used to program keys for timed access. The system supports four time slots in a day (each one having its own blue key).

The blue keys were not yet available during the market introduction in 1991 but were added in 1995.

The lock comes in three variants:

1. Eloctro Standard

This is the simplest (cheapest) version. The lock has enough memory to store 1000 keys. The Eloctro Standard can be bought from resellers and is immediately ready for use.

2. Eloctro Series

The Eloctro Series adds enhanced key control. The customer is given keys that are for their facility only. Other Eloctro keys (bought at a local shop) cannot be programmed to work with the system. The customer-specific keys can, however, be used in another Eloctro Standard system. The Eloctro Series locks need to be pre-programmed by the factory with a customer identifier.

3. Eloctro System

This adds remote control to the lock. Using an wired interface to each lock, a central computer (PC compatible) can be used to centralise key management. The Eloctro System was introduced in 1995.

Advantages of the Eloctro locks that are mentioned are:
- The flexibility of being able to (de)program keys.
- The fact that keys can easily be bought and used (no delivery times as all keys are identical apart from their electronic id).
- Keys cannot be copied.
- Mechanically secure (anti-saw pins, anti-drill plate, special features to combat manipulation, torque, wrenching, pulling, freezing and glueing).
- The lock can cope with dirt, wear and changes in temperature.

The Eloctro System also comes with a 'programbox', which is a Psion HC 100 handheld organiser, that can be connected to a lock using a special wired key, to program the lock and to read out information from the lock, such as the last 256 key related events the lock has archived. Unfortunately, I do not own a programbox.

Looking at the documentation I have seen, this lock was marketed as Lips Eloctro in the Netherlands, and as Chubb Eloctro in the UK.

Newspaper articles

Looking at newspaper articles from 1991 ([2], [3], [4]), we can learn a bit more about the Lips Eloctro and the development thereof.

We learn that the development of the system has taken years and cost millions of Dutch guilders (millions of euros in today's money).

The key contains a 3x3mm chip that contains an id. The chip is wedged between two ferrite plates, making it durable. There are $7x10^{13}$ possible ids. It is stated that this is the first wireless chip in the world, powered by the lock. The chip is designed by TNO (Netherlands Organisation for Applied Scientific Research) and CME (Centrum voor Micro-Elektronica), both in Delft, the Netherlands. CME was started in the early 1980s by the Dutch government, after a report urged the Netherlands to take a more leading role in chip design and manufacturing.

Keys have a colour code. There are 16 colours available. It is possible to remove individual user keys from a lock with the deletion key, and it is also possible to delete all keys with a specific colour in one go.

The head of Lips product development (mister J.C. Hordijk) mentions the lock is 100% fraud-proof and it can resist electricity, magnetism, radio waves, high temperatures, liquids and chemical substances.

The price is €445 for a lock, €75 for a key validation or key deletion key and €40 for a user key (prices corrected for the 2025 price level).

Patents

As mentioned in the newspaper article, the chip in the key was especially designed and patented. We find patents [5] and [6], which list Evert Nieuwkoop as the inventor and Chubb Lips Nederland bv as the applicant. Evert worked and still works for TNO. Patent [5] is about the lock, patent [6] about the key. Both were filed January 8, 1991, which is just a few weeks before the first mentioning of the Eloctro in the press.

The patents include references to prior work. We find a reference to British patent GB 2 158 870 A, which describes a lock that powers a key wirelessly (using induction) to retrieve a static code. This patent was already filed in 1984 by Finnish company Wärtsilä. Patent GB 2 174 452 A is from 1986 and it describes a similar lock for use in cars.

Patent [5] mentions that the lock will create a alternating magnetic field, powering the key. The key, in turn, changes the modulation of the field. This modulation can be picked up by the lock, decoding a bit stream from it (the key identifier).

This transmission over the magnetic field in itself is already described in the 1984 patent. The *invention* lies in the fact that the inductive coupling is done in such a way that even with a turning key, the inductive elements are placed such that the magnetic fields do not have to be very strong. This in turn lowers the power consumption, which allows for the lock to be made smaller, small enough to be the same size as a traditional mechanical lock.

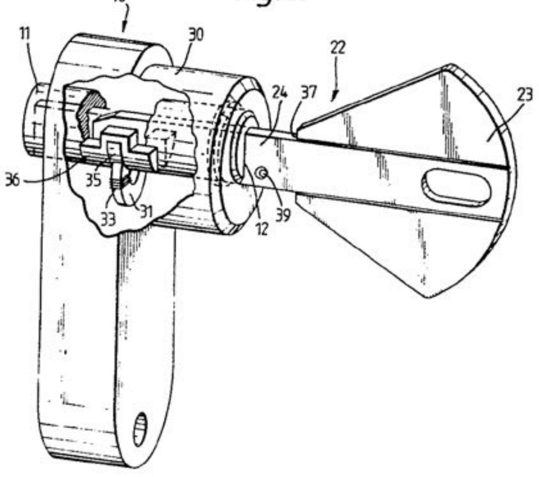


Fig.4.

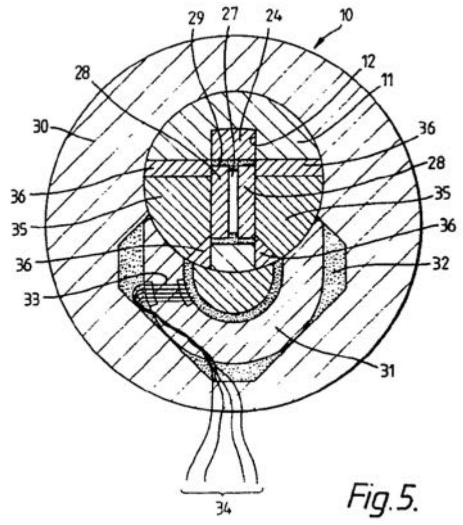


Fig.5.

To make it possible to reduce power, the magnetic field needs to be present as close as possible to the receiving element in the key. Therefore, the magnetic flux created by the coil (which is in the housing) is brought very close to the key by having ferrite (that goes through the coil) extent into the barrel, which rotates when the key is turned. When the key turns, the extension of the ferrite into the barrel is no longer in the right spot, but at that point, the communication has already finished.

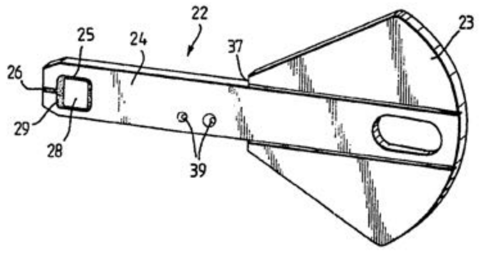
In the pictures above, taken from [5], the coil is 33, and we see the U-shaped ferrite material 31 in the housing extend into the key via 35, creating a toroid to transport the magnetic flux all the way to the chip in the key. The chip is marked as 27, it has two ferrite plates attached to it (28).

The chip in the key is read when the key is inserted and this has to happen quickly for a nice user experience: there must be no noticeable delay between the insertion of the key and the lock allowing the key to turn. When the plug rotates, the ferrite elements are no longer forming a toroid. This is no problem as at that point, communication over the magnetic field is no longer needed. The plug only contains passive elements, so there is no need for (electrical) connections between the (rotating) plug and the housing, which was the case in some other mechatronic locks of the time. Such connections are complicated to fabricate and can lead to reliability problems.

As the system needs little power, the coils (the one in the lock housing but also that in the key) can be kept small. This is what allows the Eloctro key to be like a regular key. The

Nova, IKOTRON and Combisec keys all had electronics into the bow of the key. The fact that the Eloctro uses so little power is the basis for patent [6]. Low power also means a long battery life, and less electromagnetic power leaked from the lock, which would possibly hinder emission standard adherence.

The coil in the key can even be integrated with the chip (integrated circuit) itself. The patent refers to another patent: WO-88/03594 ([7]). The latter was filed by TNO in 1987 (naming Evert Nieuwkoop as inventor).



The key, of which an image from [6] is shown here, has a little slit (26). This prevents the chip from being fully surrounded by metal, which would divert part of the magnetic field from the coil in the chip, which in turn means more power would be needed for the system to work. Another reason for this slit is to prevent a short circuit in the barrel when inserting the key.

The 1987 TNO patent [7] describes the wireless communication of a key with a lock, although in this patent, a more generic token or card, and reader are presented. It mostly deals with the electronics in the token, that need to be able to change the magnetic field such that the reader can extract a code, while still using very little power. For the Eloctro, it also needs to be implemented on an integrated circuit, on which it is nearly impossible to create large capacitors for instance.

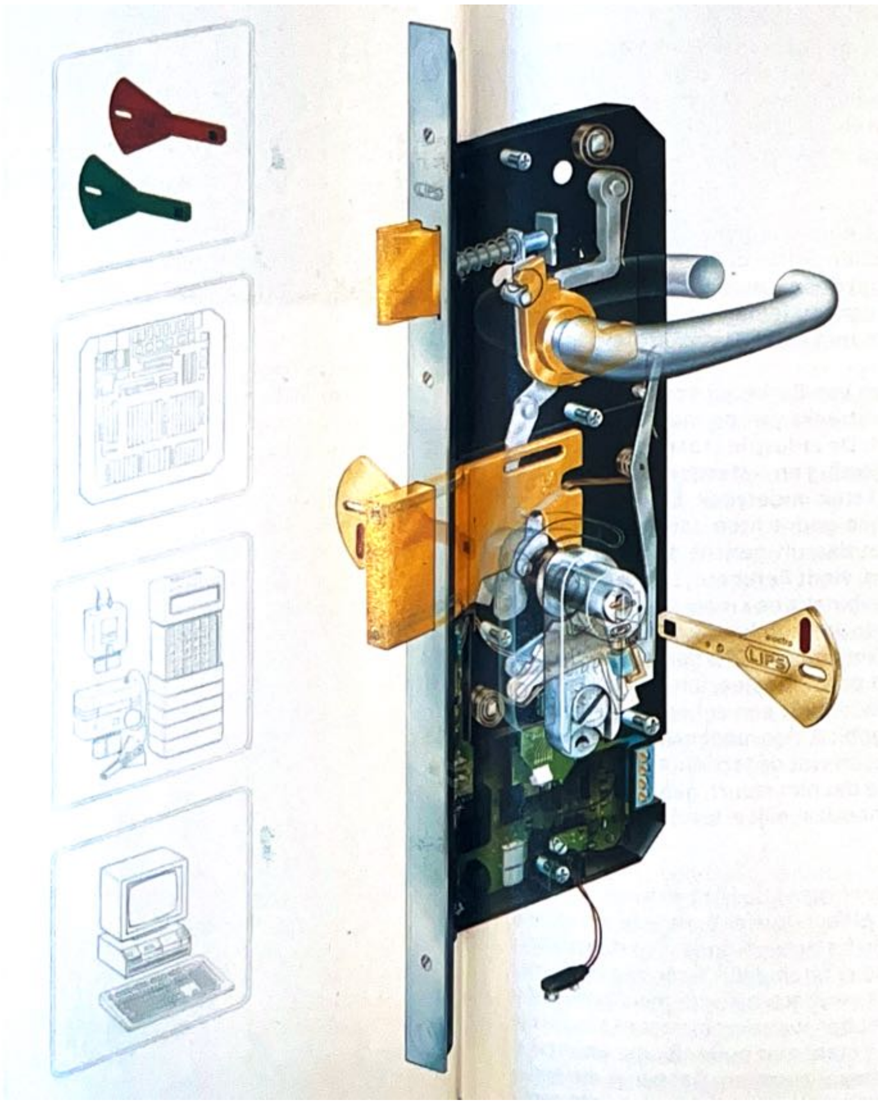
The frequency in which the magnetic field alternates is stated as being "typically 10MHz". To keep the power consumption low, it is important that the oscillator is tuned. Normally, this requires a manual, per-oscillator calibration. The patent describes a change to the electronics that makes the oscillator tune to the right frequency without manual calibration, which simplifies the production process.

Readers with a background in electronics, might want to read [7] for more detailed information.

De ingenieur article

The Dutch magazine "De ingenieur" is a periodical of the Dutch Royal engineering society KIVI. In their first magazine of 1991 [11], an article about TNO-TPD mentions the Eloctro.

It states that TPD (the Technical Physical branch of TNO) was contacted by Lips in 1985 to help create a new mechatronic lock. TPD got CME-Delft (Centrum voor Micro-Elektronica), and Produktcentrum TNO (the branch helping small and medium sized companies with product development) involved in the project. These companies, together with Lips, eventually developed the Eloctro in three parts. Lips was responsible for the locking mechanism, Produktcentrum TNO for the mechanical part, and TPD for the electronics.



The Lips Eloctro system ([11])

After a working demonstration model was built, a small series was made and tested. When that series proved to be reliable, further development was done to prepare the lock for production.

RB Elektronica (Magazine) articles

The Dutch magazine "RB Elektronica Magazine" was a monthly electronics magazine from 1930 until 2003 (under somewhat different names throughout the years).

The May, 1991 edition [9] contains a three-page article about the Lips Eloctro. The article ends with a thank you to E. Nieuwkoop of TNO-TPD, who is the author of patents [5], [6] and [7].

The article gives a very good insight into the requirements and design decisions that were made. It starts by saying that the development took six years of collaboration between mechanic and electronic engineers. (This confirms that the development started around 1985, two years before the IKOTRON key with wireless transponder was introduced. This also means that the 1987 patent [7] that TNO filed, was a result of their involvement in the Eloctro project.)

The original Lips requirements included:
- The look and feel must be like that of a traditional mechanical lock (insert the key and immediately turn the key to open the lock).
- It must be possible to take a random key and program it to be allowed to open a selection of locks.
- Theft or loss of a key should not result in the lock having to be replaced.
- Keys must be very hard to copy.
- The lock must have the same dimensions as existing locks.
- The lock must be able to run standalone on batteries, and have a long battery life.
- The lock must be as safe, or safer, than similar mechanic locks, against outside (mechanical) attacks.
- The lock must resist new types of attacks, such as voltage injections.
- The pricing must be such that, apart from professional institutions (including shops and companies), also consumers should be able to purchase it.

First, the technology to be used for the communication was reviewed. This could be done by using electromagnetic waves (including optical), thermics, mechanics (including acoustic), electronics, magnetics or chemicals. After some research, this list was shortened to acoustic, optic, or electronic inductive. For all three, a study was started to determine the feasibility and possible risks for the project. In this study, the first wireless chips were made in the semiconductor facility of the Technical University of Delft (TUD). When this worked, the technique was patented [7]. A manufacturer was found for mass production of the key chips.

After this phase, Lips decided to stop further investigating acoustic keys. For the other two (optic or electronic inductive), the project was continued by looking at integration in a mechanical lock and coming up with specifications. After this was done, the electronic inductive option was chosen as the one to be implemented.

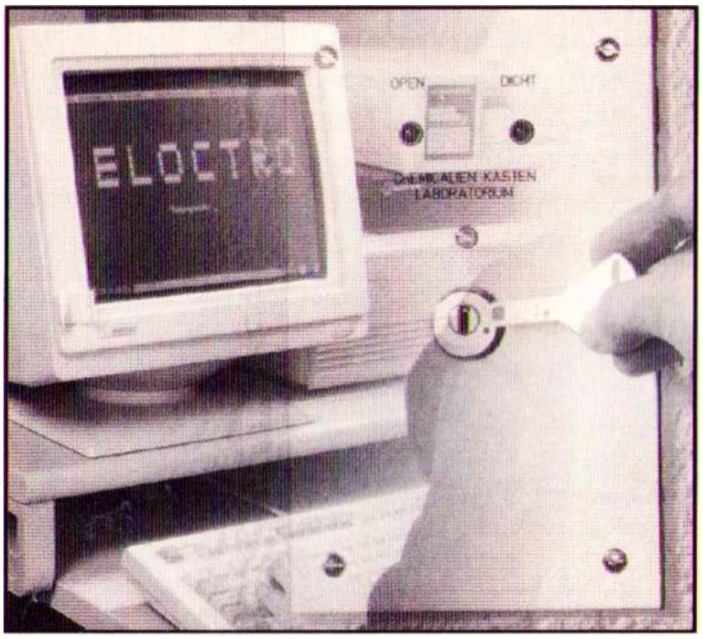
The requirement to have the electronics fit into an existing form factor was not an easy requirement to implement. The article explains this could only be achieved by using multi-layer circuit boards and SMT (Surface Mount Technology) components that are smaller than through-hole components. When the project started, SMT components accounted for less than 10% of the market in which through-hole components were dominant.

To reduce the size even more, an ASIC (application-specific integrated circuit) was developed specifically for the Eloctro. By using an ASIC, multiple general purpose components can be replaced by one custom component, that also uses less power. Alternatives such as using an FPGA (Field-programmable gate array) or PAL (Programmable logic array) would be too big and consume too much power. The ASIC required quite an initial investment, as dedicated chips are produced in small quantities, causing them to be more expensive. Still, the team also worked on keeping the cost down. A microcontroller was added to the design to create more flexibility in the functionality of the end product.

The project was a real collaboration between mechanic and electronic engineers. Some mechanical parts needed to change (like the electronically operated actuator) but the electronic circuit board in turn needed to be shaped in a certain way to leave room in the correct places for the mechanics. The creation of the key was also a challenge, the article states.

Reading the article, I get the impression that a proper analysis of risks (threat analysis) was done. For example, somebody thought about what happens when the battery dies. How do you resolve that situation? This led to the creation of a connection on the outside of the lock to power it. Another example is the implementation of 'fail secure': when the lock is electronically broken, it defaults into the closed (secure) position.

The article, as well as the yearly report of TNO on 1990, explicitly mentions that this is 'as far as we know, the first chip in the world that communicates with its environment wireless in a commercial setting'.



Central computer ([10])

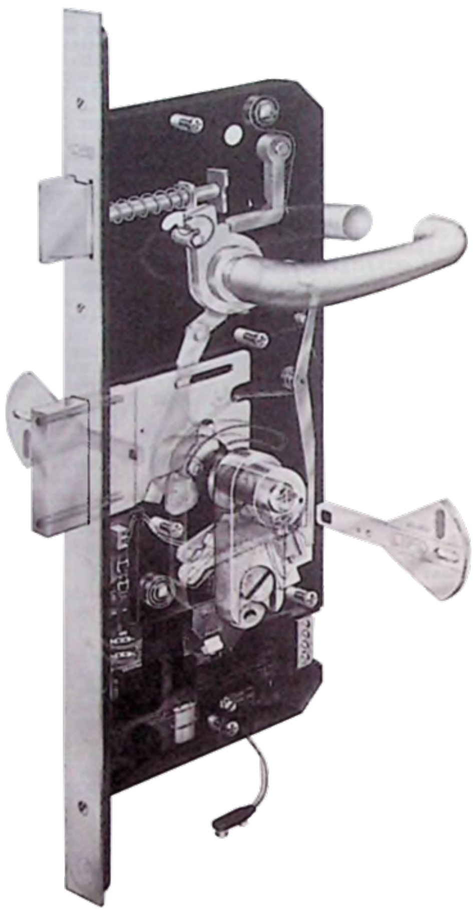
The December, 1995 edition of RB Elektronica ([10]), contains a product news section, that mentions the Eloctro. It states that new functionality has been added to it: the locks can now be networked and managed from a central computer. This is the Eloctro in the *System* variant.

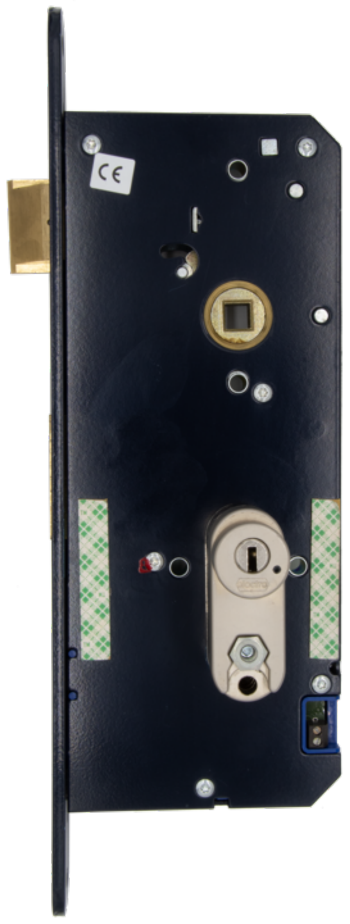
I was unable to find more references to the software used in the Eloctro System nor how it communicates with the locks. We will investigate ourselves in coming chapters.

Disassembly

Lock disassembly

Let us look at a standard Lips Electro. A popular Lips lock format was the 2200 series and during the design phase, that was the size that was aimed for. In the end, the lock became a little bit larger. Below, a press image is shown, next to the actual lock I bought from old stock. I was unable to get this lock to function properly, even though it was new.

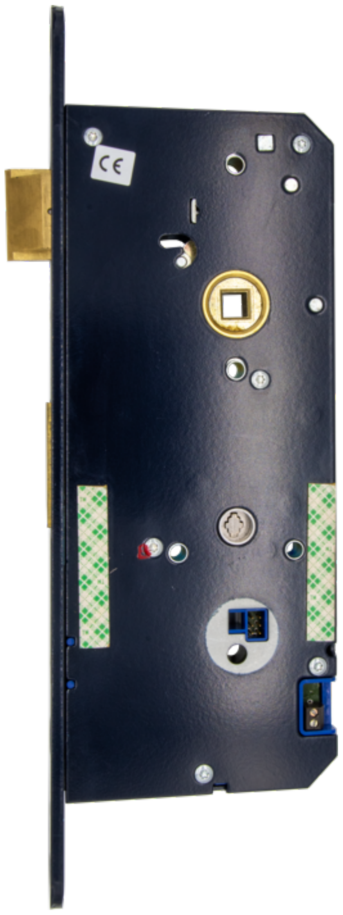

Lips Electro as shown in [9]

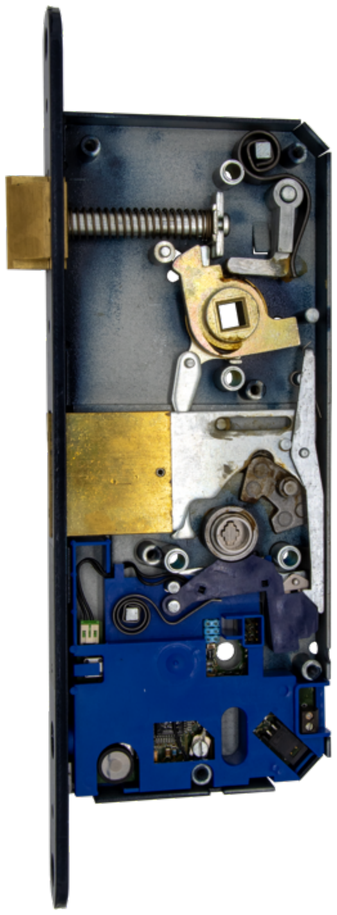

An actual Lips Electro

The lock houses the electronics and the mechanics. The reader (the barrel with the coil, in which the key is inserted), is a separate device that is screwed onto the lock on either one or both sides. Different length readers can be used. On the bottom right is a wire terminal. The press photo has four connections, my lock has two. These two are for external power. The other two will be for locks in the Electro System range (article number E**C**-2*XX*), that can be remotely controlled from a computer. My lock is an Electro Standard (article number E**S**-211). The ES-201 is the normal lock, the ES-211 is better protected against forced attack for use in outside doors. It comes with additional metal enforcements (not shown).

In the bottom right corner, there is a connection for backup power. The press picture shows a connector for a 9 Volt battery. There is also a holder for 6 1.5 Volt batteries, to be put in the door (article number EB10).

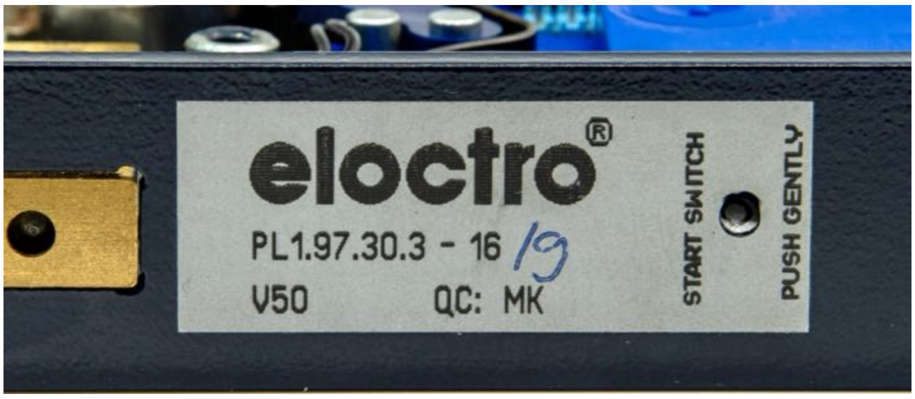
When the reader is removed, we can see a 6-pin connector. Removing the plate reveals the mechanism and also shows the electronics behind a blue enclosure.



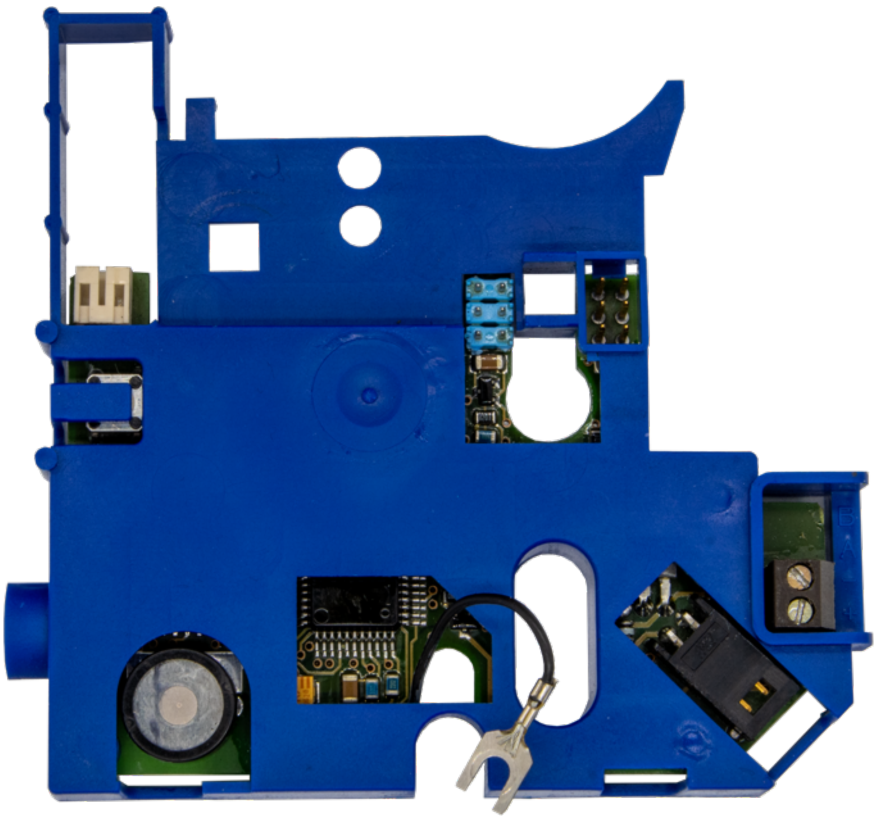


On the right side, we see a vertical piece of metal, that pivots in the middle. This works with an electromagnetic actuator (the purple part is pushed upwards by the spring). This mechanism was designed for the Eloctro and is described in patent [8] which we mentioned earlier.

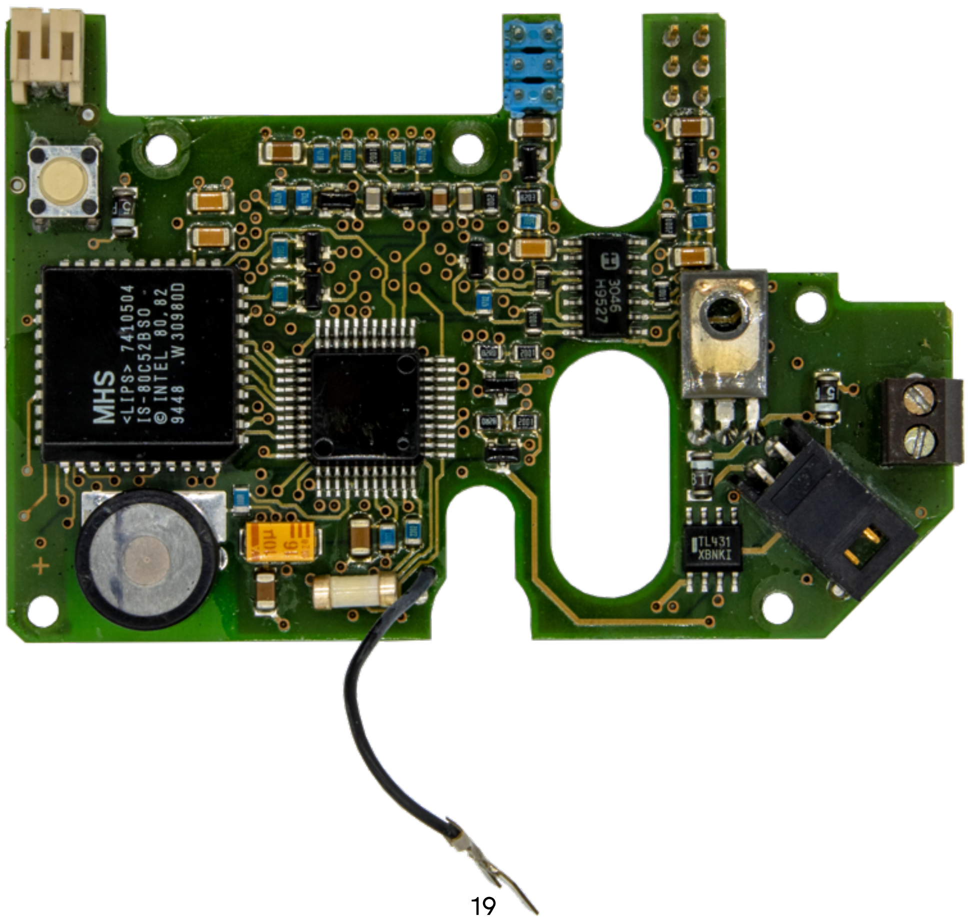
On the side, there is a sticker. It looks like the lock was made in 1997 (possibly week 30, day 3), so the Eloctro has been in production for at least 6 years. "V50" might indicate a release version.



The blue plastic covers the printed circuit board (PCB), which we will focus on. The mechanical part of the lock is rather standard, apart from the patented actuator.



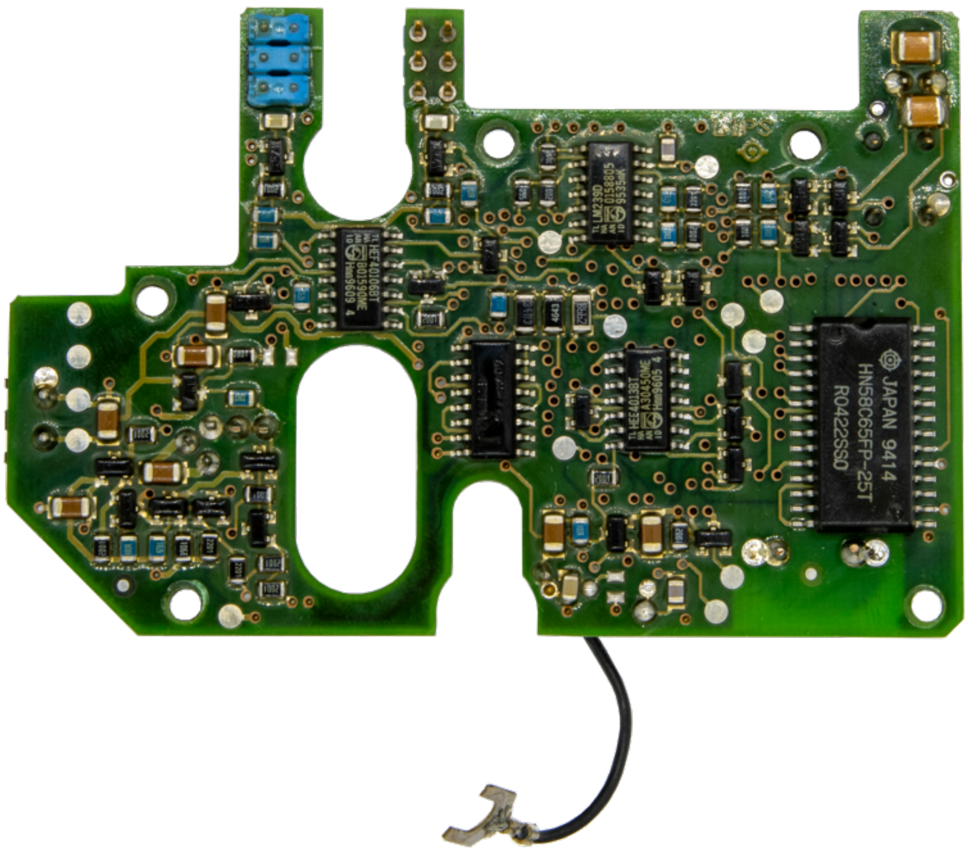
When removing the cover, we see the PCB with all the components.



It is clear that the shape of the PCB was important, to accommodate the mechanics, as we've read in [9]. On the left, we see, from top to bottom, a connector for the actuator, that activates the mechanics to open or close the lock. Under it is a push switch, that can be operated from outside the lock (see the earlier picture of the sticker that labels the switch as "START SWITCH"). Underneath that is a large IC. This is a microcontroller aimed at process control, measurement, and instrumentation. It contains memory that holds its programming. The brand of this IC is Matra MHS SA. This company started in 1979 as a joint venture between French company Matra and Harris Semiconductors, but the latter withdrew in 1989. More interesting is the fact that the IC also has the Lips name on it, which means it was produced specifically for Lips. Once more, this reiterates the large scale of the Eloctro project. At the bottom is a piezo speaker.

Next to the microcontroller, we find another IC. Due to the coating on the PCB, the text on it is barely readable, but this is the ASIC that was designed for the Eloctro.

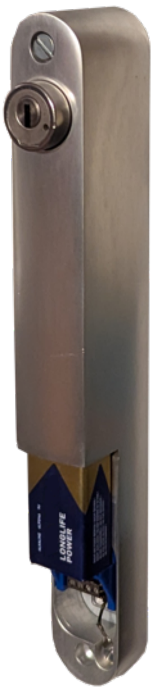
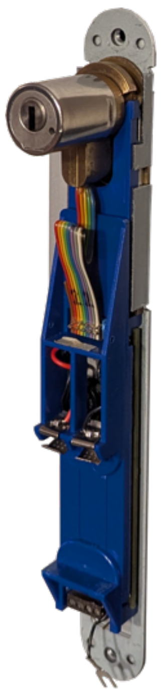
Top right, there are two 6-pin connecters, one pointing up and one pointing down. These are where the readers plug into. On the right are two connectors for supplying the lock with power and backup power. The plastic cover has space for a 2-pin connector next to the 2-pin connector marked + and -. The missing connections are labeled A and B. We will look into those later in this chapter.



Looking at the other side of the PCB, we see some basic components, as well as a memory IC from Hitachi, the HN58C65FP-25. This is a 8192-word x 8-bit EEPROM with an access time of 250ns. This IC can contain 8kB of data and can be (re)programmed. It will retain its contents when power is off. This will store the list of keys and other information.

The IC's are all marked with a production date (year/week). The newest component on the PCB was made in week 9, 1996. The ASIC was made week 6, 1996 and the microcontroller was made week 48, 1994. The lock was probably manufactured in 1997 (according to the sticker). The ASIC was made in larger bulk, or MHS could easily rebrand their stock with another printed label.

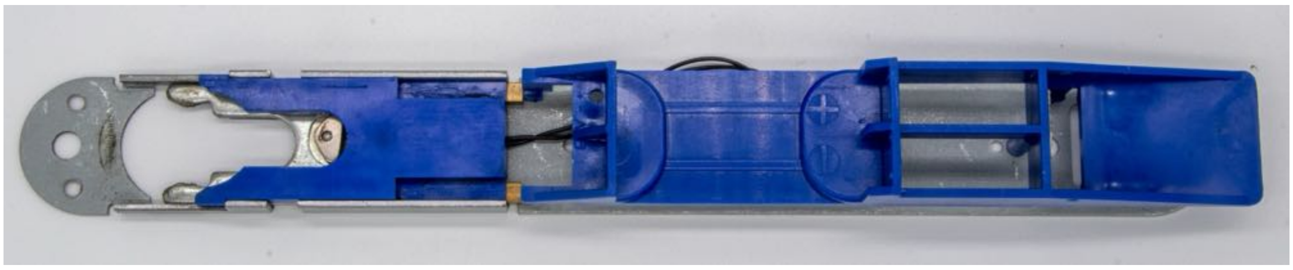
Next, let's have a look at the cylinder in the demo stand, as shown on the title page. This lock uses a different form factor. The cylinder itself is a so-called Swiss cylinder. The mechanical actuator is different from the one we saw earlier, and (backup) power is provided by a 9 Volt battery that sits in the lock (the other lock uses six 1.5 Volt batteries in a separate enclosure). The Swiss format was rather common in large installations in the Netherlands at the time.

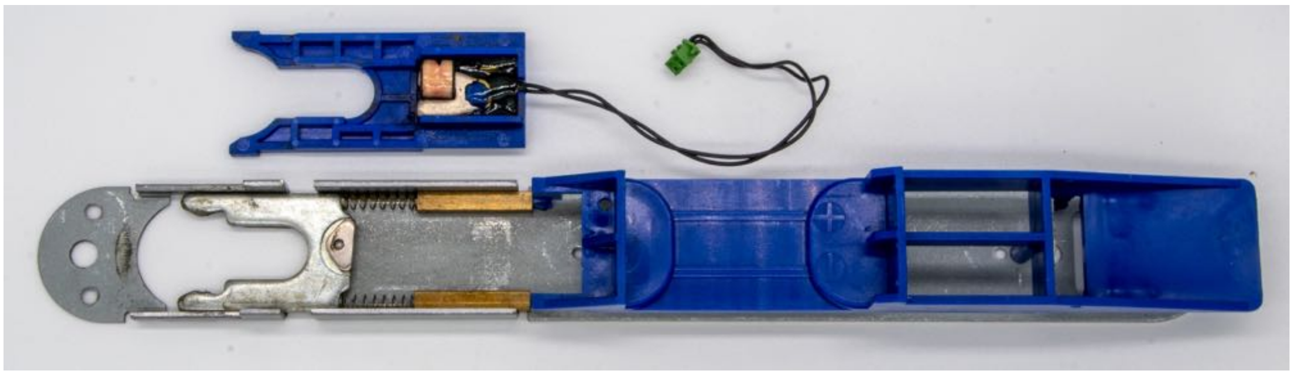
The outside of the door (see cover page) contains just the key reader (cylinder), the inside of the door (see above) contains the cylinder, the locking mechanism, the electronics and the 9 Volt backup battery. The battery is on the bottom behind a metal cover that is tightened with a screw.

With the remainder of the cover removed, we can see that the electronics and mechanics are very compact.

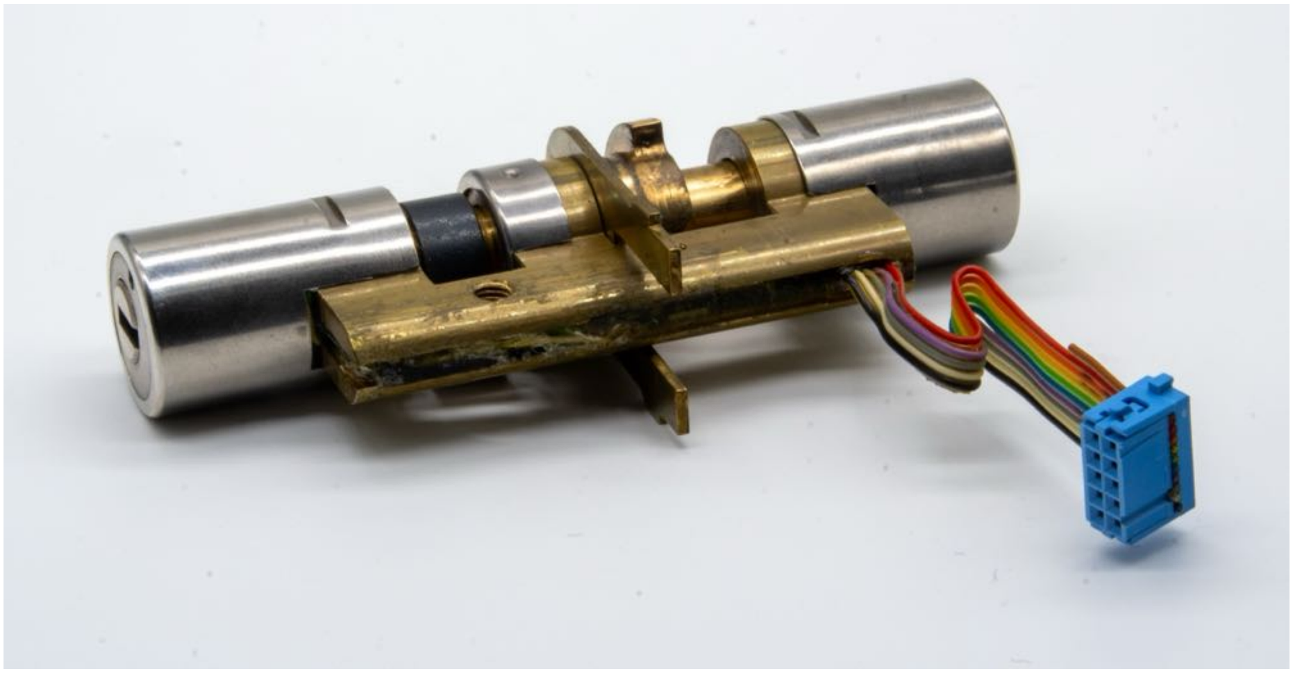
Further disassembly reveals a metal base plate with the mechanical part and the PCB. With the PCB removed, it looks like the image on the next page.



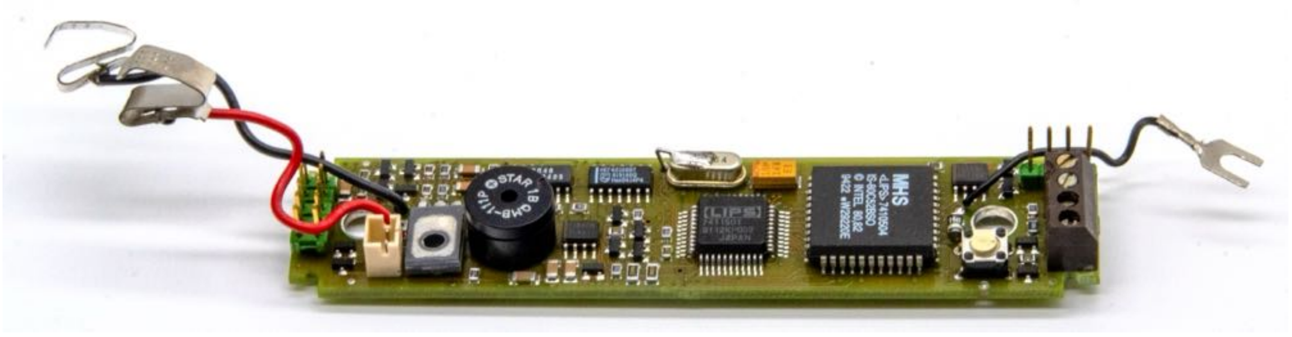
The lock has been rotated 90 degrees counter clockwise in the picture, which means the cylinder is on the left. The blue part on the left of the picture is what keeps the lock from turning.



It slides downwards to release the cylinder. The mechanism is somewhat different than that in the other lock. This lock has the readers integrated into one piece. The wires are led to both sides and they are secured with resin.



The electronics of the lock are similar to that of the previous lock we've looked at.



On the component side, we see the microcontroller, ASIC, piezo speaker, push button and terminals, just as with the other lock. The ASIC is well readable and shows the Lips logo, which makes sense for a component made specifically for Lips. The terminal on this PCB has 4 pins, labeled + and -, as well as A and B. The A and B terminals were missing on the other lock.

When we follow the traces, we see that A and B connect to a Texas Instruments SN65176B IC, which is a device that bridges TTL level logic (binary data encoded by high and low voltages) to RS-422 or RS-485 communication. This implies that A and B (and ground) are used for remote communication with the lock. The A and B connections are not present on the Eloctro Standard lock, which makes sense if this is the remote management feature that is only present in the Eloctro System. RS-422/RS-485 are like a serial connection, but where typical serial communication uses a ground connection and a data connection (that sends data in high/low voltage sequences), RS-422/RS-485 use a differential connection. This means that the data is encoded in a voltage *difference* between the A and B wires. This is more robust against interference, making it possible to use longer wires (over 100 times as long) while still having a working connection.

The SN65176B is not present on the other two locks, but these two do differ. The old stock lock has room on the PCB for the A/B wires, but there is no apparent room on the PCB to place the SN65176B. The other lock from the Lips collection is from 1993 (so before the market introduction of the Eloctro *System*). This lock has a slightly different PCB. On it, there are wires to where the terminal should be, although no terminal is present. Space for a SN65176B is also present, but the actual IC is missing.

As the labelling on the terminal reads "A" and "B", it is to be expected that RS-485 in half-duplex mode is used for locks that support RS-485 communication.

On the back is an 8kB 250ns EEPROM, but for this lock it is an Atmel AT28C64E-25SI instead of the Hitachi used in the other locks. The Atmel is a special industrial/high endurance model that supports 100.000 write cycles.

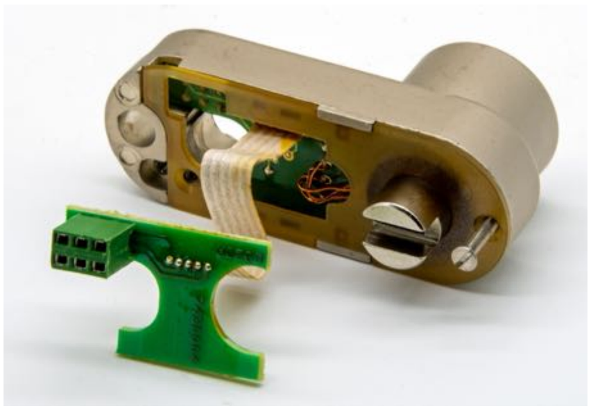
Looking at the date codes, we see the ASIC was produced week 12, 1991, the microcontroller week 22 of 1994. The newest component on this PCB is from week 22, 1994. This matches the sticker on the lock, which mentions PL1.94.37.4 - 100 (possibly day 4 of week 34 of 1994).

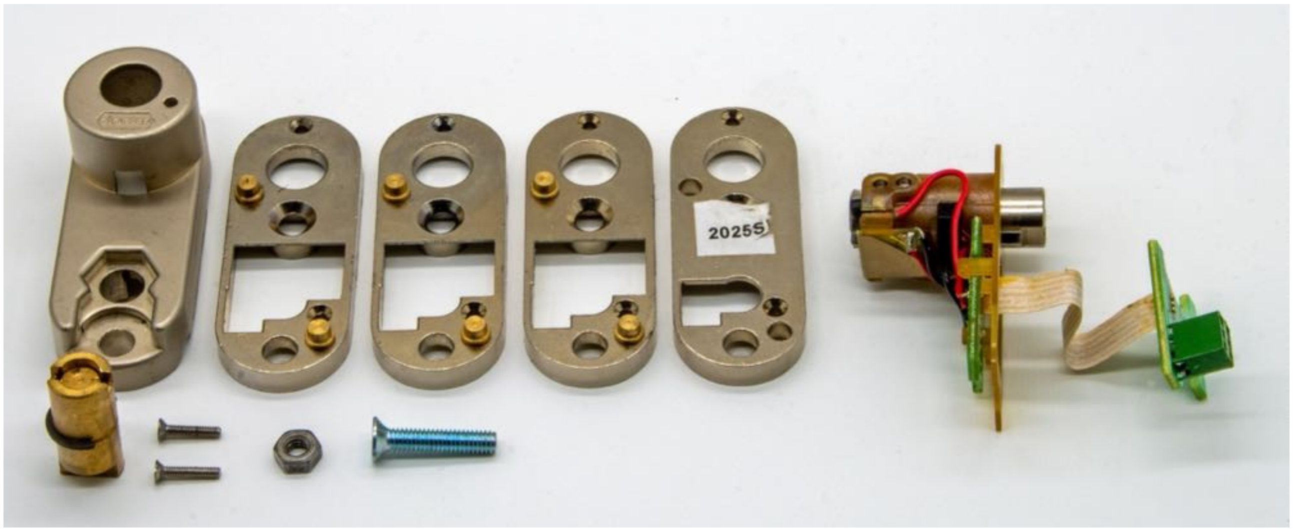
It is most likely that this lock is a preproduction lock of the Eloctro System. The Eloctro System was introduced in 1995. This lock appears to have been made in 1994, and it uses an EPPROM from week 35, 1990, which would probably not have been used if this had been a lock from a (large) production run. The fact that the lock comes from the Lips collection also points into this direction.

It could be the case that this Swiss form factor was a 1995 addition to the "standard" model lock, but this is speculation. I have seen no official mention of the Eloctro in Swiss cylinder format.

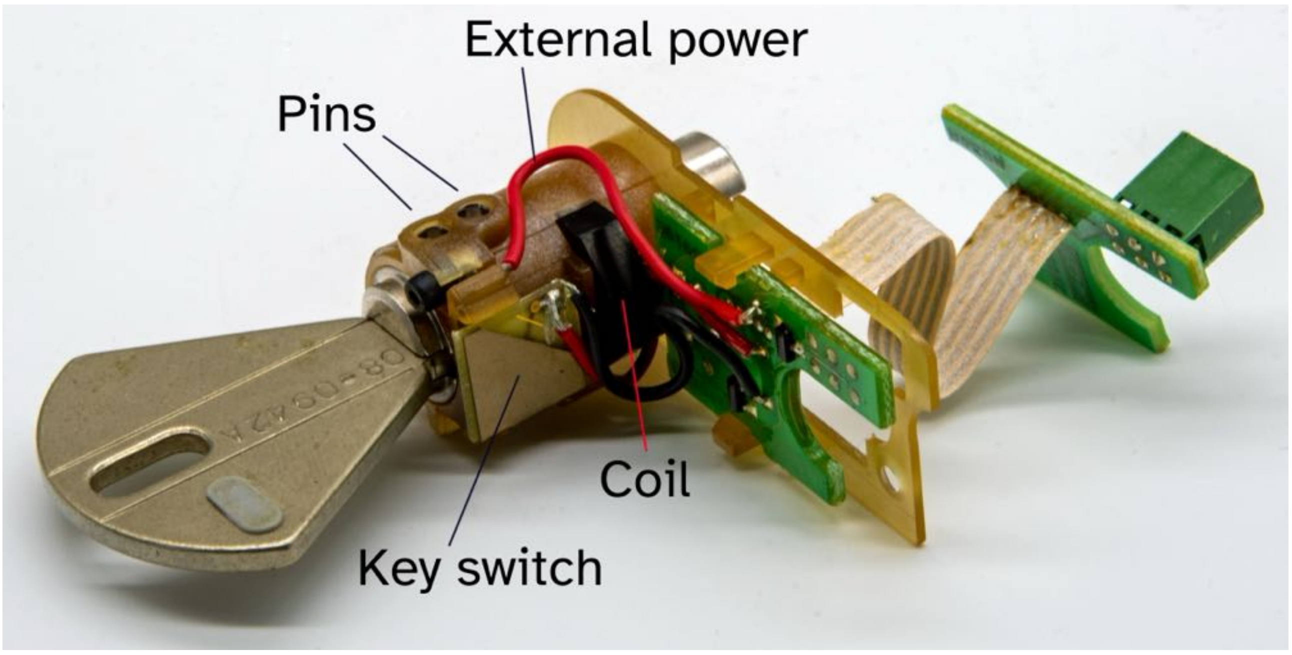
Reader disassembly

The reader has article number EL01. It consists of a holder with a number of metal spacers attached, to vary the length of the reader.





The connector on the last plate is extended from the actual reader using a flexible wire. On the back of the PCB, we can see the wires going to the coil in the plug.



With he housing completely removed, we can see the elements in the reader.
There are two pins that check the two dimples in the key. In the housing, there are springs that are not shown in the picture. The key switch detects the presence of the key. This allows the electronics to only use power when a key is inserted. The red wire takes external emergency power from the contact in the black tube that can be seen to protrude to the outside of the reader. The housing of the coil that generates the magnetic field is also visible.

Key disassembly

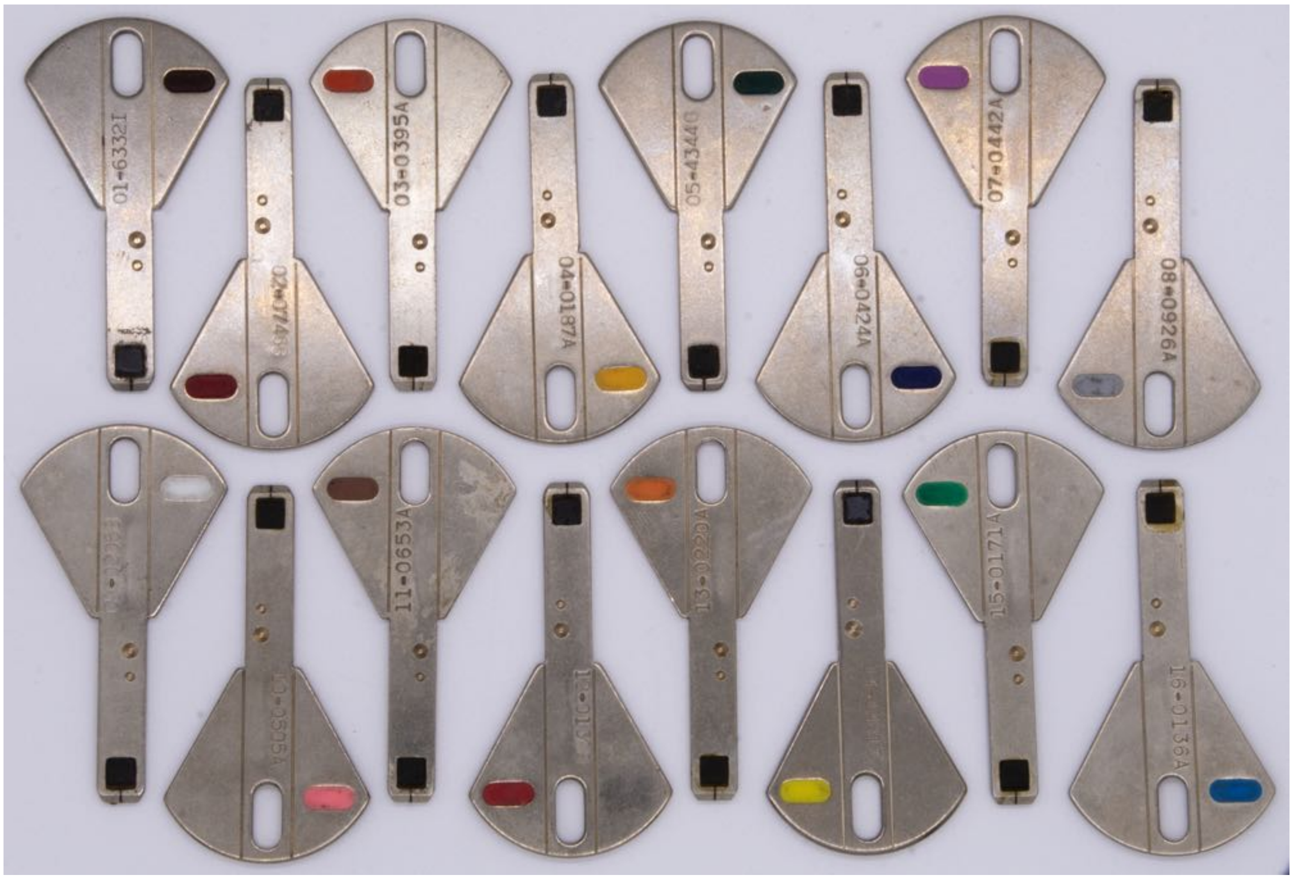
There are user keys, a green key validation key, a red key deletion key, and blue key suspension keys. All my keys are marked 'Lips' except for one blue one marked 'Chubb'. This was probably a key for the UK market.

The user keys have bitting, which only consists of the same two dimples for every key. The green, red and blue keys do not have this bitting, which means they cannot be used to operate the lock by turning the key. At the tip, the ferrite plates in which the chip is embedded can be seen. The chip is glued into the key with epoxy. Each user key has a number stamped in it and a coloured tab.

The key numbers are of the form *XX-YYYYZ*, where *XX* is the group number. With the key deletion key, individual keys can be revoked from a lock, but also groups, for quick revocation of multiple keys from the allow list. There are 16 groups ranging from 01 to 16. Each user key has a coloured plastic tab in the group colour. When I put the keys in order, I was happily surprised to find the electronics colour code, that is also used to encode resistor values in colours for instance, although the place of 'brown' is taken by 'black'. Maybe black was deemed a nicer colour than brown to start the series with.

01 Black	05 Green	09 White	13 Light orange
02 Red	06 Blue	10 Pink	14 Light yellow
03 Orange	07 Violet	11 Brown ("light black")	15 Light green
04 Yellow	08 Gray	12 Light red	16 Light blue

The key number is *YYYY* and *Z* is a letter, I have only seen 'A', 'B', 'C', 'D', 'G' and 'I'. We will look at the meaning of this code later.



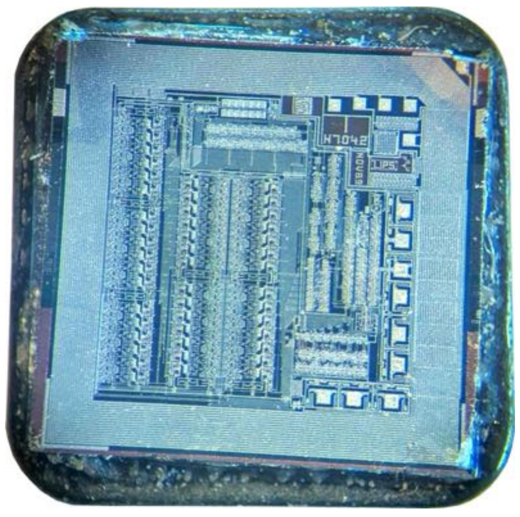
All user key colours (groups)

A user keys weigh about 16.5 grams. The green, red and blue keys are made of a different material, they weigh a little over 5 grams. The coloured keys have no plastic tabs and no dimples. The green and red keys I bought from old stock all have "P A" (green key) or "W A" (red key) stamped into them, instead of a key number. "P" could stand for "programmeren" (Dutch for programming) and "W" for "wissen" (Dutch for deleting). The coloured keys from the Lips collection have different identifiers stamped into them: "LI8" for the green key, "W17" for the red key and nothing for the blue key. It is not known if these refer to a facility code of an Electro Series. None of these keys were programmed to work on the locks I own.

The lock on the title page came with one (working) user key that has no number stamped in it.

Key validation, key deletion and key suspension keys

I proceeded by getting the chip out of a key, using some chemicals to dissolve the epoxy. This leaves two ferrite plates with the chip wedged in between. These plates measure 4x4mm. Using force, I was able to take off one of the plates, revealing the chip die, which I photographed under a microscope after peeling off the adhesive that held it to the ferrite plate. We see it resembles the picture from the patent, clearly showing the coil, the logic centrally on the chip and some test pads on the side. The actual chip has 14 test pads, that can be used for testing and initial programming in the factory.



Chip die on ferrite

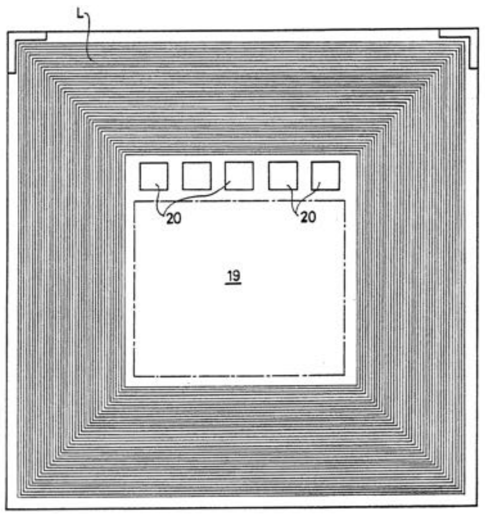
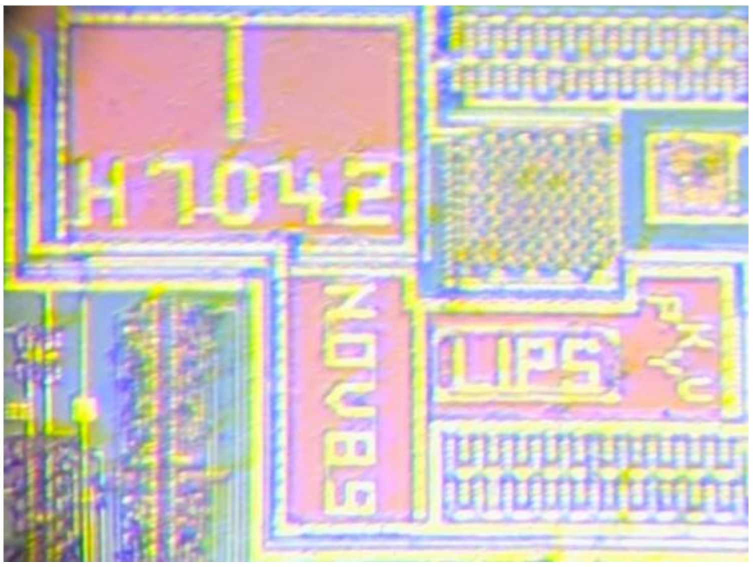


Figure 7 from [7]

Zooming in on the chip, we see markings. "H7042" is probably the internal code of the chip, "NOV89" is probably the date the chip mask was created. This is two years before the market introduction, 4 years into the project. It is unknown how old the key was that I have opened, but it came from an official reseller. It makes sense for the developers to have made a large quantity of such chips after the design was finalised. The cost is in the development of the chip and the creation of the mask. Once that is done, it does not matter much if you make a few or thousands of chips.

Then there is a marking "PKYU", which might have to do with a company responsible for making the chips (or it could be initials of the designers). Somewhat to the left of the "H7042" marking, we see some kind of logo. Below left, the image is shown rotated 90 degrees. It could read "DEM" or "DEMO".



Close-ups of the die

Looking at the die, we can see 46 identical structures making up the left half of the die. These are most probably the bits that hold the code. With 46 bits, there are 2^{46} possible values, which is roughly $7x10^{13}$ or 7.000.000.000.000. This is the number mentioned in the newspaper article [2], so we assume the keys to hold a 46-bit identification number.

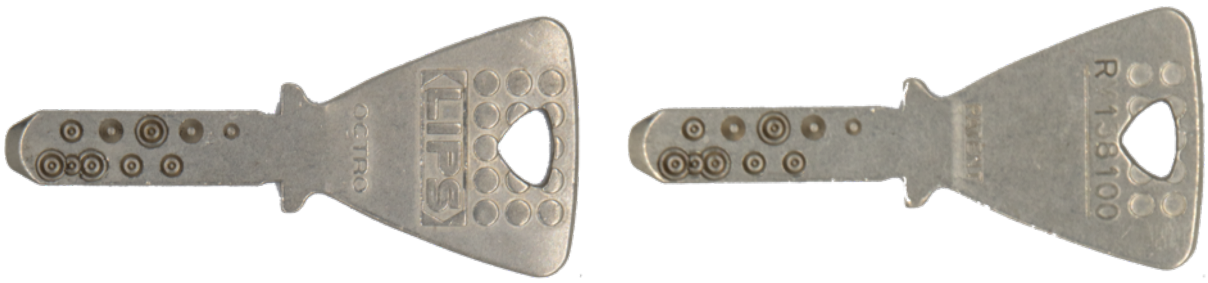
Research

Now that we know what comprises the Lips Eloctro, let's look to see how the system works, what its features are and what attack vectors might be present, in current times and when considering the state of technology when the lock was introduced.

Birth of a new lock

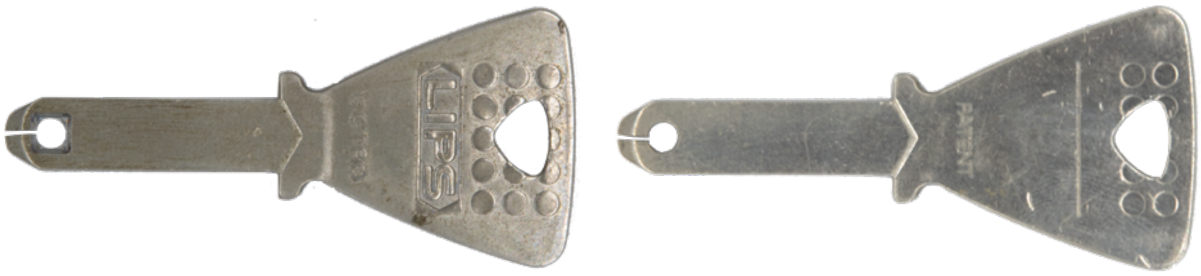
I have in my possession a few prototypes of the Eloctro key, that are from the Lips collection.

But let us first look at a key for the Lips Octro mechanical lock:

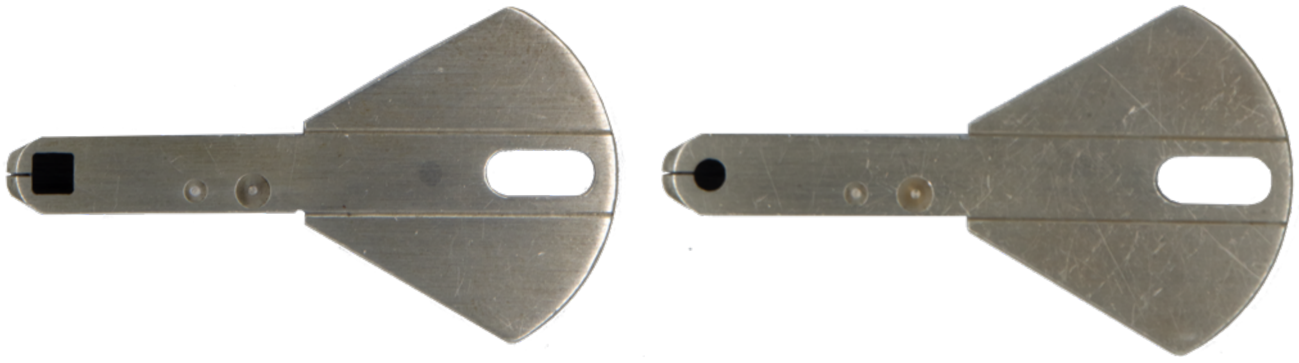


The Lips Octro is a dimple lock with three rows of pins, for a total of 15 pins. Lips introduced it in 1987. You will now understand that *Eloctro* is short for *Electronic Octro*. The Eloctro project started around 1985 but it wasn't until 1991 for the product to be ready. The timeline matches the introduction of the Octro (which was in 1987) with the tests to see if a key using magnetic induction would work.

Below is a mock-up Eloctro key. It is just an Octro key, but without bitting and a hole added to hold the chip. We can also see the slit we described before. In this prototype, the hole is square on one side and round on the other side. The initial idea must have been to provide a bit of material for the square chip to rest on. This prototype key has no chip inserted.

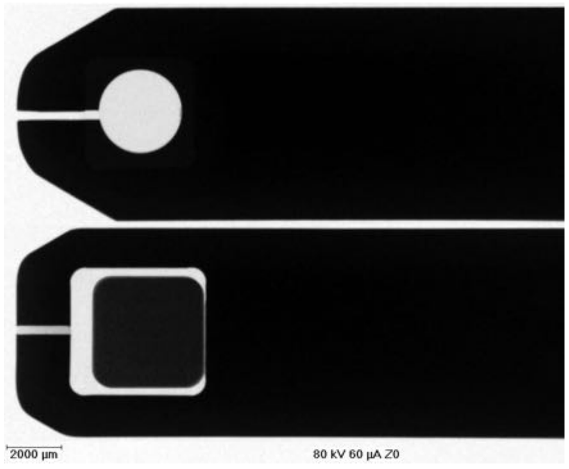


A latter prototype key has the typical bow design of the eventual Eloctro key. The chip is still visible as a square on the front and a circle in the back, which is not the case with the production version. This key has the two dimples that are also in the production version, although they are in a slightly different position, and with a slightly different depths as well.



This prototype key has a thickness of 3mm. The production key is 2.5mm thick, the same thickness as the original Lips Octro key.

To check wether or not an actual chip is embedded in this prototype key, I made an X-ray image of the prototype key and a production key. We see that the production key has a a square piece of ferrite (that will have the chip wedged in between), whereas the prototype does not contain ferrite nor a chip.



Eloctro operation

The Eloctro locks are opened and closed using the metal coloured user keys.

Programming can be done using:
- Three special keys, the green *key validation key*, the red *key deletion key* and one of four blue *key suspension keys*.
- The 'Programbox', of which I have very little information (Eloctro System only).
- Using software on a central computer that is connected to the lock using a serial connection (Eloctro System only).

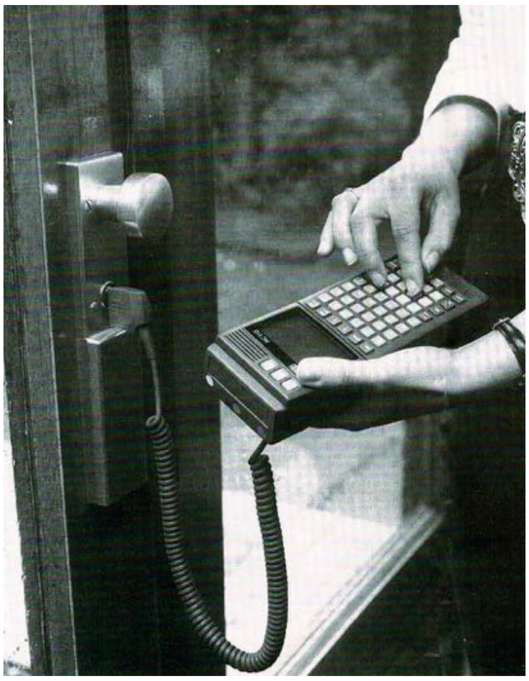
Programming with the coloured keys is done as follows:
- When the lock is new, a first green key needs to be programmed. To do so, insert a key and press the start button after which the lock will give 2 very short and 3 short beeps. Remove the key and insert the green key validation key to be programmed. The lock will give 4 short beeps as confirmation that the green key is now programmed into the lock.

- With this key validation key, other can be authorised. Insert the key validation key (a short beep is emitted). Then, insert the user key, key validation key or key deletion key to be validated. Two short beeps acknowledge the action.
- There is a mode for adding multiple keys, which is to insert the key validation key twice. After the first insertion, a short beep is emitted, after the second insertion two short beeps. Now, one or more user keys can be inserted, with 2 short beeps acknowledging each. After all user keys have been programmed, the key validation key has to be inserted again, yielding 4 short beeps. If a programming action is not completely finished, the lock reverts to the state before the programming.
- With the key deletion key, user keys can be de-authorised. Insert the key deletion key (a short beep is emitted). Then, insert the user key to be invalidated. Two short beeps acknowledge the action.
- There is a mode for removing multiple keys, which is to insert the key deletion key twice. After the first insertion, a short beep is emitted, after the second insertion two short beeps. Now, one user key can be inserted, with 3 short beeps acknowledging it. This is remove all user keys' authorisation that belong to the same group (colour) as the user key used in this process.
- Finally, there is a mode for removing all keys, which is to insert the key deletion key three times, with 1, 2 and 3 short beeps as feedback after each key insertion. To complete the action, a user key needs to be inserted, yielding 4 short beeps.

For the Eloctro Series, the process is the same, but only keys belonging to the facility code (which is hard-coded in the lock) can be used.

Programbox

I have very little information about the programbox. I only found this image from an internal Lips newsletter:



Programbox, picture courtesy of
Lips Nieuws of around March, 1994

The sales documentation states that the programbox can program keys to the lock that you do not need to physically posses. That supports the idea that the programbox emulates any key (or more generally, can send arbitrary bitstreams) to the lock.

The documentation also mentions that the programbox can read logging information from the lock. This means that the lock can alter the magnetic field to send information to the programbox. The communication between the lock and the programbox is thus bidirectional and the programbox can send specific commands (at least one, to read the logging).

I do not have the programbox nor the software, so I cannot reverse engineer it to figure out how the command(s) are sent. It could be that a few 46-bit codes are not used for keys, but to encode commands instead. I would suspect, however, that one of the 46 bits would be reserved to specify wether a code is a key id or a command, to allow for multiple commands and for new commands to be added to later releases.

I would not be surprised if developers to have included undocumented commands that can be used by the factory or service personnel, to perform low-level and/or privileged actions, such as programming the facility code into the lock, or maybe reading/writing the EEPROM. To verify this without programbox, if would require reverse engineering the code in the microcontroller. The PCB of the some of the locks also contain test pads that can be used for programming actions during manufacturing.

The Lips documentation about the programbox says that it is possible to program keys that you do not physically have. That poses the question how these keys can be entered into the programbox for programming. Is the stamped key number enough? After the reverse engineering, which we will discuss next, it seems that that is not the case. A safe assumption is that the programbox works in cooperation with the central computer with Lips software that is used as a central access control system in the Eloctro System.

Wired interface

The Eloctro demo lock has RS485 connections (labeled A and B) but I was unable to get anything out of if. The new lock does not have a terminal for RS485, although the PCB has the TTL-to-RS485 IC. The third lock does not have a terminal and also no TTL-to-RS485 chip.

We will look into the wired interface later, when we start reverse engineering the lock.

Stamped key codes

The user keys have a number stamped on them, in the form *XX-YYYYZ*, where *XX* is the group number. There are 16 groups, so it is plausible that 3 of the 46 bits are reserved for it. The individual code *YYYYZ* (four digits and one letter) has 10.000 x 26 possibilities, which add up to 260.000. My guess would be that 18 of the 46 bits are used for this code, as with 18 bits you can have 262.144 possibilities. That leaves 26 bits.

There could be bits reserved for the type of key. But, since the green, red and blue key cannot be used as a user key (they cannot operate the lock due to lacking dimples), there is no real need to do so. We will later learn that these keys have an identifier unique to them, so a green key cannot be programmed as a user key and vice versa.

The remaining 20 bits will contain bits to be used in the Eloctro Series and Eloctro System, where a facility gets its own facility code. If all 20 bits would be used, that would allow support for over a million facilities. But some bits will need to be used for other things. There might be a bit that indicates the code is not a key code but a programbox command for instance. There will probably also be reserved bits to facilitate the addition of new functionality at a later stage. But even with, say, 15 bits left, that would cater for over 30.000 facility codes.

Memory

The locks have 8kB of EEPROM memory (65.536 bits). It will contain, according to the documentation:
- Known key validation, key deletion and key suspension key codes.
- The list of up to 1.000 allowed user keys.
- Logging of the last 256 key actions.

Storing 1.000 user keys will cost at least 46.000 bits if all bits are used. Storing the logging will cost at least 11.776 bits (one bit to indicate success/failure) if all bits are used. This should fit in the available space and leave room for some configuration information.

In the next chapter, we will see that only 40 key bits are stored.

Reverse engineering

Now, let's disassemble a lock and investigate deeper. I've mainly used the 1997 "New Old Stock" lock for this.

EEPROM

I removed the Hitachi HN58C65FP-25T EEPROM from the lock with a heat gun. This is a 8kB (8192 byte) EEPROM in a SOP28 casing. With an adapter, I converted it to DIP28 and put it in a reader device. At first, the EEPROM seemed empty, but after tweaking the read speed, I was able to dump the memory. I then proceeded to also remove the EEPROM from the lock from the Lips collection that was marked "archief set 4" (archive set 4), to be able to compare a new and a used lock.

The dumps show that the memory is divided into 32 blocks of 256 bytes. Blocks 0, 1, 2, 3, 4 and 5 are empty in the new lock and contain 9 5-byte tuples, where each tuple has it bytes spread over blocks 0, 1, 2, 3 and 4 and an identifier in block 5, in the used lock. This is a list of programmed key ids, but we will look at that more in-depth in the next section.

Blocks 6~23 are empty in both locks. They seem to hold three more lists of at most 256 key ids.

Block 24 seems to contain configuration data. The block begins empty in the new lock and contains 05006FE FF20AFAA 8ABB20AF AA8ABBE8 E8BFCBA9 in the used lock. Further, from 18E4, we see some changes:

Address	New lock	Used lock
18E0	FFFF0000	FFFF0000
18E4	00010000	05010200
18E8	00020804	00020804
18EB	081E0000	081E0000
18F0	00000032	00A50128
18F4	09000000	01000000

Block 25 contains an identification string. The new lock contains ASCII "197303", 6 NULL bytes, ASCII "161EC211-02161971 MK" and 20 ASCII spaces. The text "PL1.97.30.3-16" that is on the physical label can be seen here. The text "EC211" looks like the part number of a Lips Electro Standard. Looking at the packaging this lock came it, we see that it also mentions EC-211. I suspect that each lock has its EEPROM programmed during manufacturing to contain default values and this version string. The "02161971" might be a unique serial number.

The string in the new lock is different. Here, it reads "193153", 6 NULL buses, ASCII "39 1EC201-02095231. HWVOORRAAD", 22 spaces, ASCII "voordeur", 8 NULL bytes and ASCII "19Mar98". The physical label on the look reads "PL1.93.15.3-39", which we see in the text.

The EC201 points to this lock being an Eloctro Standard, which matches we expected, as this lock does not have the RS-485 connection. *HWVOORRAAD* is Dutch for hardware stock and *voordeur* is Dutch for front door. The unique serial number is "02095231".

Block 26 has a difference of 1 byte, the used lock has 5C at 1AF1.

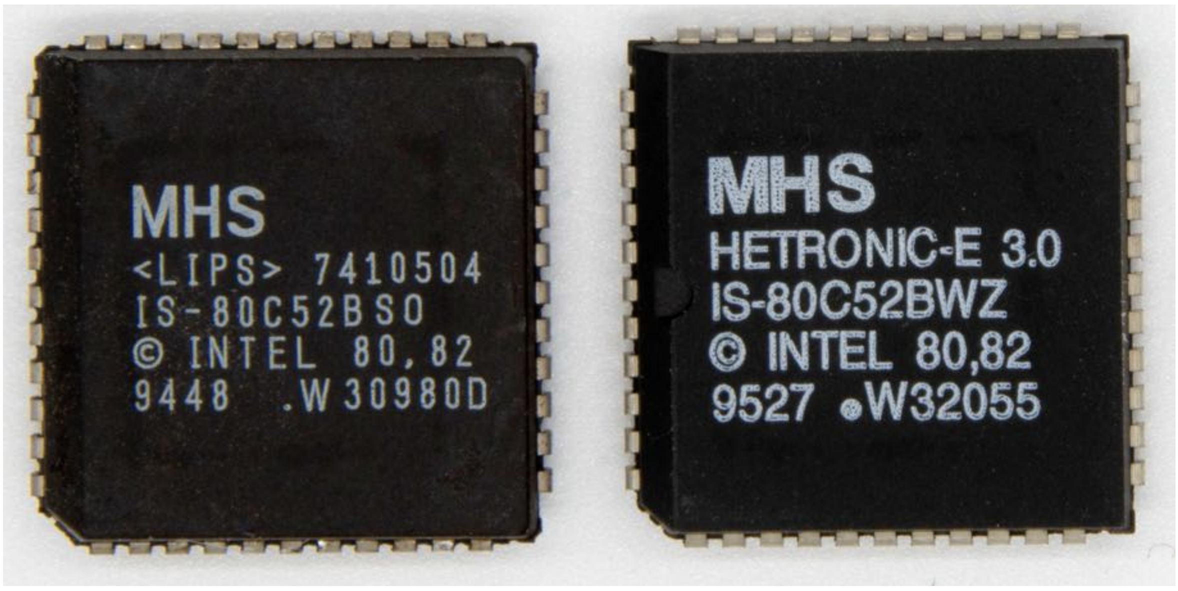
Block 27 is empty in both locks.

Block 28~31 appears to hold logging information, which we will also look into in the next section.

The lists of key id's use 5 bytes per key id it seems. This matches the marketing claim that 1.000 keys can be stored in the lock, as 5kB is reserved for the list (the first 24 blocks). But if that is true, the claim that there are roughly $7x10^{13}$ keys (matching the 46 bits we say in the key chip) is not true. That is to say, $7x10^{13}$ different keys can be made, but the lock will differentiate at most 2^{40} keys (roughly $1x10^{12}$). It seems the marketing information in [2] mentions a theoretical limit, not the technical limit. Still, $1x10^{12}$ is 1.000.000.000.000, which is a huge number.

8052 MPU

The MHS IS-80C52BSO MPU was also desoldered from the 1997 lock using a heat gun. The "IS" stands for the industrial version in PLCC packaging. Again, I used an adapter, to convert from the PLCC44 form factor to DIP40.



Lips version of the 8052 next to an 8052 made for another customer

The 8052 contains code that has been programmed in. This resides in its internal 8kB program memory. The MPU also has limited on-board memory often referred to as IRAM (256 bytes), but it can address an external data memory (XRAM) of up to 64kB. In the Eloctro, the 8kB EEPROM is used to provide external data memory.

An MPU is different from a regular CPU. A CPU runs a program from memory and executes the instructions in order. A CPU can run instructions from RAM. An MPU runs programs based on interrupts. These can be internal, such as timers that trigger at set intervals, or via external triggers via an electrical signal to one of the pins of the MPU. An MPU has a strict separation of the program code and RAM. For more information, see the Wikipedia page on the Intel MCS-51.

There are many vendors that have created (variations of) the original Intel MPU. Some of them have security features that aim to protect the code that has been loaded into the MPU at manufacturing. This MPU was made by Matra Harris Semiconductors SA or MHS for short. MHS has implemented in its version of the 80C52 something called "Secret ROM". When Secret ROM is enabled, an instruction to read internal memory that is coming from external ROM will encrypt the result before giving it out on an external port. This should prevent having the MPU run code from an external memory that reads and exposes the internal memory. MPUs that have this feature have a "C" in their part number. The MPU used here is the IS-80C52**BSO**, where "BSO" is the customer ROM number. There is no "C" prepending it, which means the MPU lacks the Secret ROM feature.

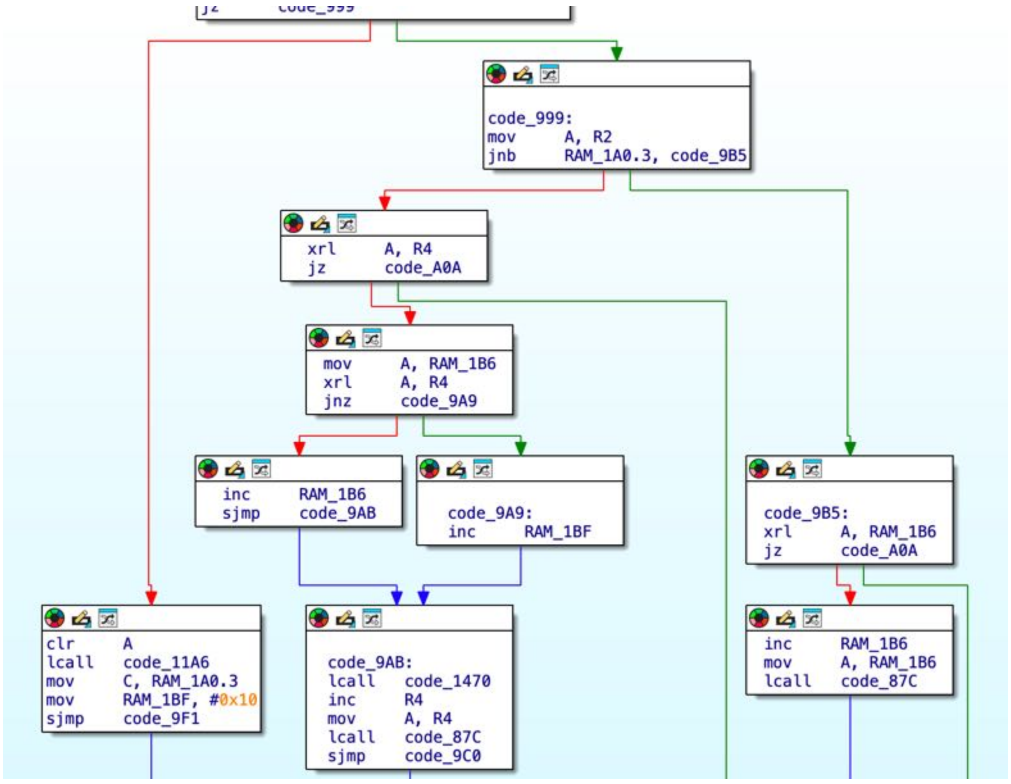
It is not clear why an MPU with the Secret ROM feature was not used. Given the scale of the Eloctro project and the attention to detail, I was surprised to find that code protection was not implemented. It might have been that retrieving and dissembling the code was not seen as a (big enough) threat at the time.

Without the security feature, it becomes possible to read out the contents of the MPU ROM. The technique is to attach an external memory that contains a program to read and expose the internal ROM bytes. When booting the MPU with the EA pin low, it will start running code from external memory. There is a ready-made circuit board that does this automatically (search GitHub for "8051dumper"). Or, you can connect the MPU to a Raspberry Pi to emulate a ROM with a reader program and set the correct values on the pins of the MPU. This is what I did, using a micro-Python program that was kindly shared by Jan-Willem Markus.

I was thus able to dump the internal 8kB ROM. It turns out the program takes up 8093 bytes of the 8192 available bytes. Loading the dump in a disassembler, it looks like valid 8051 code.

If the Secret ROM feature would have been used, reading the ROM this way would not have been possible. In such a case, more elaborate techniques might still have worked. An possible technique would be *voltage fault injection*, but this is a lot harder to do and has the risk (depending on the method used) to wipe the ROM contents. Voltage fault injection, also referred to as voltage glitching, is known as a technique for several decades, but was not really a threat factor in applications such as this when the lock came to market.

It is quite hard to understand the code, even when we know (roughly speaking) what the code should be doing. We also know that there are 4 I/O ports (P0, P1, P2 and P3) on the MPU, where normally P0 and P2 are used to address the external RAM.



```
code_999:
mov     A, R2
jnb     RAM_1A0.3, code_9B5
```

```
xrl     A, R4
jz      code_A0A
```

```
mov     A, RAM_1B6
xrl     A, R4
jnz     code_9A9
```

```
inc     RAM_1B6
sjmp    code_9AB
```

```
code_9A9:
inc     RAM_1BF
```

```
code_9B5:
xrl     A, RAM_1B6
jz      code_A0A
```

```
clr     A
lcall   code_11A6
mov     C, RAM_1A0.3
mov     RAM_1BF, #0x10
sjmp    code_9F1
```

```
code_9AB:
lcall   code_1470
inc     R4
mov     A, R4
lcall   code_87C
sjmp    code_9C0
```

```
inc     RAM_1B6
mov     A, RAM_1B6
lcall   code_87C
```

Part of the program flow

In this case, we can use a measuring device to verify the connection between the EEPROM and MPU. We notice that the 5 high address lines (A8 through A12) are connected as expected, but the lower ones (A0-A7) are not. After some measuring, it turns out that the ASIC sits in between the RAM and MPU for the lower 8 bits of the address (AD0~AD7), AD8~AD12 are connected directly:

MPU pin	ASIC pin	ASIC pin	EEPROM
P0.0/AD0	27	12	A0
P0.1/AD1	26	13	A1
P0.2/AD2	25	14	A2
P0.3/AD3	24	15	A3
P0.4/AD4	23	16	A4
P0.5/AD5	22	17	A5
P0.6/AD6	21	18	A6
P0.7/AD7	20	19	A7
P0.8/AD8			A8
P2.1/A9			A9
P2.2/A10			A10

MPU pin	ASIC pin	ASIC pin	EEPROM
P2.3/A11			A11
P2.4/A12			A12

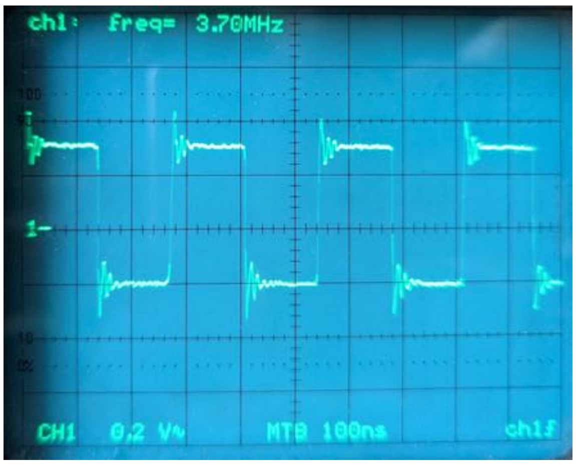
The data lines are connected as we would expect, pins AD0~AD7 from the MPU are connected to I/O0~I/O7 on the EEPROM. Note that this means that both the ASIC and EEPROM share this connection as the data bus is shared with the address bus on the MPU.

Address lines A13~A15 are not needed when no more than 8kB needs to be addressed. We note that A13 and A14 are not connected, but A15 is connected to the ASIC and also the $\overline{CE}$ signal on the EEPROM. In the code we see addresses that have A15 set. Some other notable connections are listed here:

Between	And
MPU P2.7/A15	ASIC pin 33 and EEPROM $\overline{CE}$
MPU RD/P3.7	EEPROM $\overline{OE}$
MPU RST	ASIC pin 3, PCB testpad
MPU not EA	Vcc
MPU ALE	ASIC pin 30
MPU P1.0/T2	GND
MPU P1.1/T2EX	GND
MPU P1.3	PCB testpad
MPU P1.4, P1.5, P1.6	D-type flip flop
MPU P1.7	Reset Button
MPU RxD/P3.0	RS485 RX, PCB testpad
MPU TxD/P3.1	RS485 TX, PCB testpad
MPU P3.2/INT0	Analogue circuitry
MPU P3.3/INT1	Analogue circuitry
MPU T1/P3.5	PCB testpad
MPU not WR/P3.6	ASIC pin 31
MPU not RD/P3.7	ASIC pin 32
MPU XTAL1	ASIC pin 34
ASIC pin 2	EEPROM $\overline{WE}$
ASIC pin 4	D-type flip flop, PCB testpad
ASIC pin 8	Piezo speaker

Note that the other lock has a PCB marked "V2", which has no test pads.

When following the trace on the PCB from the reset button, we see it connects to P1.7 on the MPU (bit 7 of P1). This helps us finding the initialisation code. Similarly, we would want to know what drives the actuator that opens the mechanical part of the lock, but that seems to be triggered from the ASIC (pin 9), so this is masked behind the ASIC "magic". The MPU uses addresses in the 0x8000~0x8007 range to send commands to the ASIC, probably this includes the command to activate the actuator opening the lock.



XTAL1 signal on the MPU

The ASIC is really part of the controlling logic. It also provides the clock. There is a 3.68MHz crystal connected to the ASIC (pins 43 and 44). The ASIC provides the clock signal to the MPU.

There are several ways to get an understanding of how the lock works. The most obvious one is to disassemble the code in the MPU and understand what it does. It helps to know roughly what the functionality of the code is (which we do), but also what I/O takes place. For that, it helps to understand the electronics on the PCB. This is not easy as there is quite a portion of analogue electronics, and the ASIC obscures quite a bit. Reverse engineering the ASIC is too difficult. Let us first start by looking at the expected I/O and then the code.

We expect the MPU to have the following inputs:
- Reading from the EEPROM.
- A signal from the barrel that a key is inserted.
- A key id (possibly as a bitstream, prepared by analogue electronics and possibly the ASIC).
- Serial input for remote logging and programming (not present on this particular lock).
- The reset button.

We expect the MPU to have the following outputs:
- Writing to the EEPROM.
- A signal to the piezo speaker to generate an audible beep, in conjunction with an analogue driver and the ASIC.
- A signal to provide the electromagnetic field to power the key, although this might also be handled by the ASIC on its own in conjunction with the key insertion signal.
- The signal to the actuator the opens the lock, in conjunction with an analogue driver.
- Serial output for remote logging and programming (not present on this particular lock).

For the actual reverse engineering of the code, remember that the MPU is interrupt driven. Interrupts are generated when timers overflow (internal interrupts) or when an external signal is presented (external interrupts). There is one special interrupt, the RESET interrupt but in this case, the RESET function does not really do much.

This calls an initialisation function at 0x1999, we see:

```
mov     TMOD, #0x21 ; '!' ; Timer Mode Register
mov     PCON, #0          ; Power Control Register
mov     TH1, #0xFC        ; Timer 1, High Byte
mov     TCON, #0x50 ; 'P' ; Timer 0/1 Control Register
mov     SCON, #0x50 ; 'P' ; Serial Channel Control Register
```

We notice that timer 0 is set up as a 16-bit timer counting from 0xF447 (the counter counts up and will cause an interrupt when overflowing after 0xFFFF). Timer 1 is set up as a 8-bit timer counting from 0xFC, that interrupts when overflowing after 0xFF at which point it is reloaded to 0xFC again. Given the clock speed, this means that timer 0 causes an interrupt at about 100Hz, timer 1 at around 100kHz.

The 80C52 also has a timer 2 that is used for serial communication. Timer 2 is set to not trigger on external events and to use timer 1 for the baud generation.

Timer 0, timer 2 and serial interrupts are then enabled and timer 1, external interrupts 0 and 1 are disabled. Timer 1 is started.

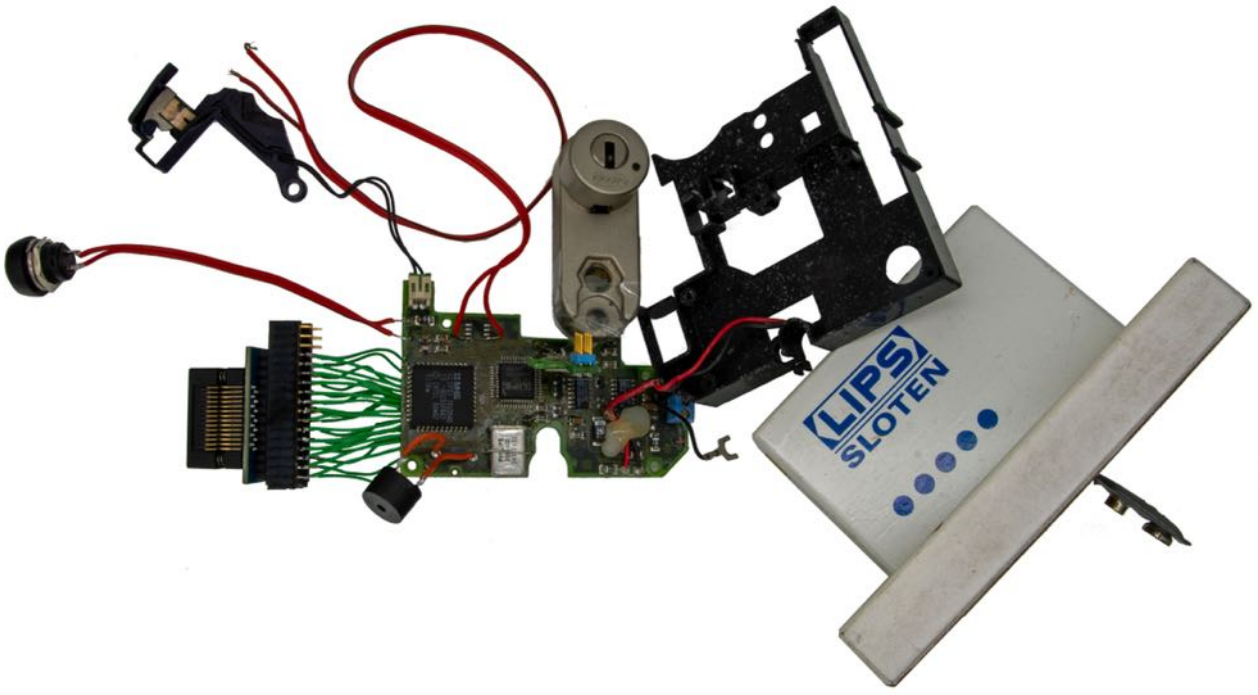
The timer 0 interrupt checks if either of RAM locations 0x1C4 or 0x1C5 are non-zero and if so, decreases memory location 0x45. If memory location 0x44 contains 0xFF, it will also decrease memory location 0x43. This seems to be some kind of delay mechanism. We also notice that the 8052 serial connection is probably used to communicate with the key and/or programbox. After going very deep into the rabbit hole, it still remains unclear to me how the key insertion triggers certain functions.

EEPROM INVESTIGATION

I decided to switch to another approach. The code is hard to read, as it is very compact, interrupt-driven and low-level. What complicates things is that a lot of variables (bits and bytes) are stored in the MPU's memory, where it is not clear what these variables hold.

Having desoldered two EEPROMs from a new and old lock, the new one not working and the old one not accepting any keys in my possession, I changed strategy and started looking at the EEPROM contents. Comparing the EEPROM contents between locks, knowing that the old lock contains programmed key ids and the new one does not, I can deduct where the known key ids are stored, and use that to pinpoint in the code where that data is read or written.

I built this test setup:



The PCB comes from the old archive lock. Since the original buzzer was destroyed during the desoldering of the EEPROM, I put on a new on (with wires). A real reset button is soldered to the PCB (shown left). Finally, and most importantly, an EEPROM socket is connected to the PCB, allowing me to easily take out the EEPROM to read out its contents at will (lower left). One key reader is connected to the PCB to be able to insert keys.

I took the contents of the new locks' EEPROM and used it on the old lock. For the first time, this got a lock to work properly. I was able to reset the lock and program a green programming key into it. After that, I could read out the EEPROM to see the changes. For each key added to the lock, 5 bytes are written in memory locations 0x00XX, 0x01XX, 0x02XX, 0x03XX and 0x04XX, where XX is increased with every next key. The 256 byte space at location 0x05XX seems to hold a status byte for each key, where 02 means the slot is empty and 01 means the slot contains a valid key.

We see three more similar structures of 5 blocks with a 6th block containing only values 02 (starting at 0x0600, 0x0C00 and 0x1200). Since the marketing material mentions the ability to store 1000 keys, I presume this is a concatenated list. Looking in the code, we see a loop at address 0x0A0B, that loops through these lists in concession, confirming this.

The documentation of Lips states: "*There are four BLUE 'time-zone' Control keys available for each Chubb ELOCTRO installation. They are coded T1 to T4 and are programmed into each lock at the factory for complete security.*" The time zone a specific time zone key can be used for, is probably kept in the status byte. Concluding, the EEPROM starts with 4 lists of at most 256 key ids each, providing a list of a maximum of 1024 authorised keys.

Since I have no working time zone keys, I cannot verify what status byte would be associated with time zoned keys.

KEY ID STORAGE AND LOGGING

We add some keys and check the EEPROM content. The keys added are an initial green programming key (using the reset button), two more green keys, three red keys, and user keys 08-0942A, 08-0926A and 08-175A. Each learned key adds 5 bytes, one in each of the first 5 blocks of the EEPROM. The bytes added are:

key	Bytes
Programming key 1 (green)	*60* 1B C2 *FC FF*
Programming key 2 (green)	*70* 3F 93 *FC FF*
Programming key 3 (green)	*70* 3B D6 *FC FF*
Deletion key 1 (red)	*70* 2C A7 *FD FF*
Deletion key 2 (red)	*38* 3E 8B *FD FF*
Deletion key 3 (red)	*78* DC D7 *FD FF*
08-0942A	*78* A8 93 *AE BF*
08-0926A	*B8* 9D DB *9E BF*
08-0175A	*E8* ED F2 *AF BC*

Interestingly, the logging section starting at 0x1C00 only contains the first, fourth and fifth byte, shown in italics in the table above. It is impossible to add the blue key. The blue keys need to be programmed by the factory. When inserting the blue key, we see it being logged as 00 DD FF, which means the key id is 00 *xx xx* DD FF, but we don't learn the full key id. There are only 256 slots for the logging, matching the marketing claim.

When deleting a key, we see that the key data is still present in the key list at 0x0000, but the indicator byte starting at 0x0500 is changed back from 0x01 to 0x02. Also, the second key id byte (starting at 0x0100) is changed into 0x00. When the same key is then reprogrammed into the lock, the second key id byte is restored (and the indicator byte is set to 0x01 to mark it a valid key).

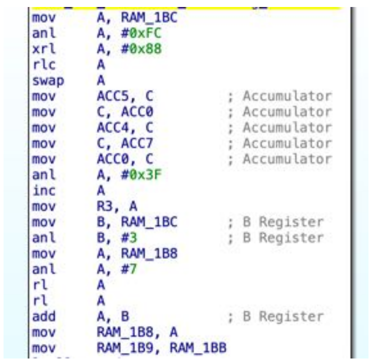
It is interesting to know if the id that is stamped on the key is the actual number in the key chip. The stamped id cannot be immediately found in the key bytes, but the number might be obfuscated. In current mechanical locking systems, the bitting code is also stamped on the key, where some kind of (secret) translation algorithm is used. Only those with access to this secret algorithm can translate the number stamped on the key into an actual bitting code. This algorithm can be very simple, such as with the Winkhaus X-pert, where random nines are added to the code to obfuscate it. When leaving these out, the remaining number is the bitting code.

Winkhaus X-pert
Image source: Winkhaus

In high security systems, the code may be completely random, with the translation being a large table of values, only known to the manufacturer.

For the Eloctro, we suspect the translation being done in the code, as the logging can be printed out via the serial interface and thus these numbers need to be translated back.

After looking at the code for a while, it did not become clear to me how the translation works. At address 0x0775 in the code, there is some key bit processing done but it is hard to make out what actually happens. What does not help is that the code is optimised such that sometimes a call is made halfway a multi-byte instruction, which is hard to decompile.



```
mov     A, RAM_1BC
anl     A, #0xFC
xrl     A, #0x88
rlc     A
swap    A
mov     ACC5, C          ; Accumulator
mov     C, ACC0          ; Accumulator
mov     ACC4, C          ; Accumulator
mov     C, ACC7          ; Accumulator
mov     ACC0, C          ; Accumulator
anl     A, #0x3F
inc     A
mov     R3, A
mov     B, RAM_1BC       ; B Register
anl     B, #3            ; B Register
mov     A, RAM_1B8
anl     A, #7
rl      A
rl      A
add     A, B             ; B Register
mov     RAM_1B8, A
mov     RAM_1B9, RAM_1BB
```

Bit juggling at 0x0775

I opted for another approach: I authorised all 67 keys I own into the lock and read their ids from the EEPROM. Then, I tried to reverse engineer how the stamped code relates to the bits that were recorded. I have in my collection keys 01-0175A and 08-0175A, which differ just in the group code, and 16-0714A and 16-0715A, which differ in the key number by just one. Using that as a starting point and with some scripting, I was able to (almost) determine the algorithm.

Let us say that key bytes 1, 4 and 5, written in binary notation, have the following bits: *abcdefgh*, *ijklmnop* and *qrstuvwx*. To go from stamped key code to these bytes, perform these steps:

- Subtract 1 from the group number and make it a 4-bit binary. This become bits *rstv*.
- Make the key number a 10-bit binary. This becomes bits wx*ijklmnop*.
- The letter determines bits *ugh*: for A or B take 000, for I take 001, for C take 010, for D take 011, for G take 100. Note that I have not got any keys that have bits *ugh* be 111.

The algorithm converts a stamped key id unambiguously to a real key id, but for the reverse, we cannot differentiate the letters A and B. There are more issues, though. I have in my collection the key 01-6332I, that have bits *ugh* 101. That would mean that an *I* could translate in both 001 and 101, which more make the algorithm ambiguous in both directions. Another issue is with key 01-0958D. According to the algorithm described above, if should have an A or B instead of a D.

The bits *q* and *u* are always 1 for all the keys I tested. Looking at the code, we see that these bits are inverted and prepended to the group id. It might be that these are used for the Eloctro System with enhanced key control.

The bits *abde* differ between keys, but I did not find a relation between these bits and the stamped key number. They might be bits coming from the key but not be part of the stamped key number.

Bit *c* is only 1 for my blue time zone key, all the others have *c* set to 0. At first I thought this bit indicates a time zone key. But, eventually I also read out the EEPROM of the lock on the front page, on which I cannot program keys. All the keys in the allowed key list and in the logging, have bit *c* set. Possibly, this bit indicates the key is a restricted key and the lock is an Eloctro Series.

We now know that the stamped key number is mapped to 17 bits. As we have seen, 5 bytes or 40 bits are stored per key. This means that key bytes 2 and 3, that are stored in the key id storage, but not the logging, must be part of the key id, but these bytes cannot be determined by using the stamped key number only. To test wether or not these bytes actually matter, I changed byte 2 of an authorised key. The key was then no longer accepted, so it appears all 5 bytes of the stored key id are checked. That, in turn, means that multiple keys may exist with the exact same code stamped on them, that are not identical. We will discuss key numbers in a later chapter when we look at possible risks.

Note that of the 46 bits the key can store, at most 40 (5 bytes) are used to identify the key. If bits *q* and *u* are always 1, only 38 bits remain.

Using the algorithm, we can calculate what key number should be stamped on the green and red keys. The green keys are all 16-1020A (or 16-1020B). The red keys are all 16-1021A (or 16-1021B), except for the red key labeled "W17", which is 16-3069C. The blue key is 16-0989A (or 16-0989B). The unstamped key that I cannot program in the test setup, which is probably a special Eloctro Series key, is 02-1529I. This is weird, because it has a light green tab, and group 02 is the brown group. Maybe this key was made outside regular production. We have already established that the corresponding lock might be a pre-production lock.

The blue key might have belonged to the Lips archive lock. In its original memory contents, we see that only a key deletion key and one user key (07-0906A/B) were authorised. One key validation key and 6 user keys have been added in the past but are now marked invalid.

When going through the log file (containing 256 entries), we see entries for blue time zone keys. One is 16-0989A/B (which might very well be the one I have in my possession), another is 16-0990A/B. No keys have ever been programmed for specific time zones though. It is conceivable that all time zone keys have a specific key number.

For the Eloctro Series key (that can only be used on locks programmed to accept a specific series of keys), nothing in the key bytes 1, 4 and 5 is different from the normal keys. The conclusion is that the series code is hidden in bytes 2 and/or 3. Looking at bytes 2 and 3, we see that in byte 2, bit 3 is always 1, and in byte 3, bit 7 is always 1. The other bits appear not to be related to the stamped key number. If the mentioned bits are indeed always 1, only 38 bits remain.

So far, we've seen the main key authorisation list starting at 0x0000 and the logging starting at 0x1C00. There is one more list, starting at 0x1800, that contains key ids (5 per key) in sequential order, one after another. Only the old lock has something here, the new lock and the lock on the cover page do not contain any ids. The key numbers stored are timezone key 16-6142T, 07-0906A (or B) and 05-0459A (or B). These keys did not come with the lock and it is unclear what is kept in this list, the only reference I found in the code is where its contents is sent out over the serial link.

RS485/422

The new lock that I examined, has no RS485 connector. Although there is physical space for a connector, there are no traces on the PCB. Apparently, different PCBs were made for different models of the Eloctro.

Taking apart the second Eloctro (the archive set), we see a different PCB. On this PCB, there is also no terminal to connect RS485. But, there are traces on the PCB. They lead to an empty spot on the PCB for the RS485 converter chip. In the test setup, I soldered wires to the PCB but there was no usable signal on them.

The third lock (depicted on the front page) has an RS485 terminal and the converter chip. However, there seems to be no data coming over the connection.

In the 8052 code, we see a timer value of 0xFC and with a clock frequency of 3.68MHz. We can calculate that the serial port runs at around 38400 Baud. The internal clock of the 80C52 runs at 1/12th of the external clock, which is around 307kHz (timer period is 3.26μs).

I tried fuzzing communication to the lock, in case a special wake-up byte needs to be sent, but I did not get any output. It could very well be that none of my locks are set up to support the serial connection. That is to say, none is an "Eloctro System" lock. The lock shown on the front page, has a production date in September, 1994. The Eloctro System came to market in 1995, but, as we've seen before, it is believable that this lock was a preproduction version that might have been an Eloctro System, but it is more likely to be an Eloctro Series.

Wether or not the programming of the MPU is the same on all three locks I have not tested, but the MPU I took the code from is from the 1997 lock. I would imagine that the MPUs all get the same code, with the enabling of functions done via the EEPROM.

So far, I have been unable to find out how to change the settings (probably stored in the EEPROM in the 0x18E0-0x18FF region) such that the lock would turn into an Eloctro System.

Security threats

When the Eloctro came to market, is was said that keys were impossible to copy. For the mechanical Octro, the statement at the time was that it was *almost* impossible to copy a key.

How difficult would it have been to copy a key in 1991 when the system was introduced? It would have been hard indeed, a lot harder than copying a Lips Octro key (which was already hard to do because of the tight tolerances in the lock where a few hundreds of a millimeter make all the difference).

With just a key, it would take a lot of reverse engineering just to figure out how to read the code from the chip, let alone copy it. The Eloctro keys have a number stamped into them but as we have seen, this does not provide enough information to copy the whole key id. An attack could have been to insert a key to be copied into a standard Eloctro lock and then reading the key code from memory. But, if a key is rejected, only 3 of the 5 bytes are stored in the logging. A possible attack would be to quickly program a key into a lock you carry around. Then, at your leisure, you can retrieve the full 5 byte key id from the EEPROM. You would then also need a programbox, of which you would need to change the programming, to emulate this captured key. This attack would also work on the restricted Eloctro Series keys, as they can be programmed into non-Series locks.

The marketing material states that the programbox can be used to program keys into the lock, without having the physical key. So, there must be a way to transfer the key code to the programbox. The programbox will probably not be able to read keys the electronic way, as that would make the programbox much more complicated. The images of the programbox also do not support this idea.

At first I thought the programbox can be given the key number stamped on the key as input to then emulate it. The programbox would then need to be able to translate the stamped code into the real code. We have seen that the stamped number alone is not enough to identify a key. One possibility would be that only 3 key bytes are stored by the proqgrambox, with a special override bit set to have it be accepted, but this is unlikely. More likely is that the programbox works in conjunction with the central software, that connects to the lock via RS485 and can thus retrieve the full id. The statement of Lips that it can program keys you do not physically have is true, but the key must then have been programmed into any lock or the central system at one point. Further research into the programbox is needed to draw conclusions.

The attack described above, where an adversary programs a key into their own lock, reads out the full id from the EEPROM and using a special reprogrammed programbox to emulate that key, would still be conspicuous, as it is clear to see that a non-standard key is used.

Copying the key into the same form factor would have been only possible with a very large budget. This would –and still is– a theoretical attack, even for adversaries with huge budgets.

Another possible attack is to reset the lock. For this, the adversary would need to be able to open the lock in the first place, or find it in an open state. Then, the steel plate needs to be removed with a screwdriver to reveal the reset button. After the reset, a new key can be programmed into the lock. This attack has limited use, as the original keys are no longer programmed to operate the lock. Also, this would not work on the Eloctro Series with its enhanced key control. Also, although a lock can be reset this way according to the manual, I have found that none of the locks in my possession could be reset. I had to resort to manipulating the EEPROM instead.

It is interesting to note that the attacks on the Lips Eloctro have not become substantially easier over the past decades. It is still very expensive to create a duplicate key that looks like the original key. Reverse engineering the protocols and communications has gotten easier, thanks to better and cheaper tools. The desoldering and decompiling I have done would have only been feasible for people working in the electronics industry at the time.

I would say that in today's world, the Lips Eloctro can still be considered to be a high security lock.

Conclusion

The Lips Eloctro has been an interesting lock to look at. In the end, the Eloctro key is still almost impossible to copy or emulate, even many years after the lock was designed. Nowadays, hobbyists do have the tools to desolder, decompile and reverse engineer the software in the lock, although that is also still a specialists job.

I was able, in the end, to get two out of the three non-working locks operational again.

There still are questions though, specifically around the RS485 interface, the time zones, how the configuration settings are stored and how the programbox is used.

If readers of this article want to help to further unravel the software, or have more information about the programbox or even have a programbox, please contact the author via the e-mail address walter+lips@belgers.com.

Bibliography

[1] Pulford, Graham W. *High-Security Mechanical Locks*, Elsevier Buttersworth-Heinemann, 2007

[2] Het vrije volk: democratisch-socialistisch dagblad, "Superveilig en flexibel slot moet de wereld gaan veroveren", Rotterdam, 19910125, page 2, retrieved from Delpher on 20250301 via https://resolver.kb.nl/resolve?urn=ddd:010963668:mpeg21:p002

[3] NRC Handelsblad, "Het einde van de sleutelbos", Rotterdam, 19910131, page 35, retrieved from Delpher on 20250301 via https://resolver.kb.nl/resolve?urn=KBNRC01:000030307:mpeg21:p035

[4] Algemeen Dagblad, "Chips in slot en sleutels", Rotterdam, 19910202, page 73, retrieved from Delpher on 20250301 via https://resolver.kb.nl/resolve?urn=KBPERS01:003102002:mpeg21:p00073

[5] UK Patent Application GB 2 252 356 A, Chubb Lips Nederland bv, category "Locks with inductive code transmission", 19911218

[6] UK Patent Application GB 227 3128 A, Chubb Lips Nederland bv, category "Keys", 19911218

[7] Patent Cooperation Treaty Patent Application WO-88/03594, TNO, category "Identification System", 19871112

[8] European Patent 0 392 596 A3, Chubb Lips Nederland BV, category "Lock with an electromechanical release mechanism", 19900405

[9] RB Elektronica Magazine, "Elektronisch sluitsysteem", May 1991, edition 5, pages 40-42

[10] RB Elektronica, "Elektronisch slot", December 1995, edition 12, pages 38-39

[11] De ingenieur, "Research in Nederland", January 1991, edition 1, pages 48-53