LAP!1

The unusual story of a 24-years-old microprocessor and a little wooden suitcase

The closed case is not recognizable as a computer, easy to lock and easy to carry.

The idea to build my own computer was born when I – as a little child – read a very impressive description of the Turing Machine by Roger Penrose. I did not understand every detail, but I could imagine some very mechanical and chaotic device in an old-fashioned suitcase, which would work on a kind of paper stripe. Although I did not take it very seriously, this funny idea did not get out of my mind.

Years later I bought a book about microcontrollers and started to learn how to program the modern version of Turing's idea. By that time I had grown a demand for a mobile computer. I dreamed about programming in the forest and word processing on trains. But a laptop was much too expensive for an A-level pupil like me, thus the old idea of building my own computer was more current than ever.

Soon I recognized that those 8 bit controllers I had worked on were not quick enough for a word processing application. So, I went to the local library and looked out for books about "real" processors. In the 1980s the town library of Cologne had a very sophisticated selection of computer science books. I went there in 1999 and they still had all these books which were exactly what I needed, because literature about modern Pentiums or PowerPCs would have been too difficult for me. Several of these books dealt with machine language programming, and most were based on the M68000 instruction set. Thus, the problem which processor to choose was quickly solved.

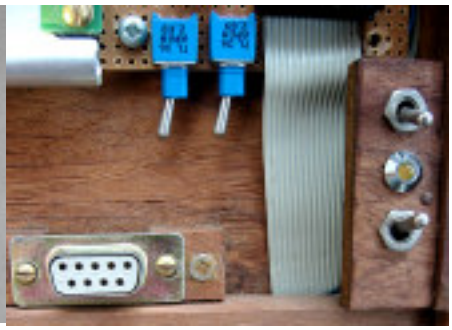
The next step was to compose a basic development environment. I drew a sketch of some glue logic which met the most basic needs of the M68000 and went to my local semiconductor dealer to order two processors. I received only one, because this was the last one to get from this source.

My first 68k program was hand-assembled and consisted of a simple NOP loop. I made several assembling mistakes and it took me some time until the first regular strobe signals could be seen on the oscilloscope screen. But from that moment on I knew I would not give up!

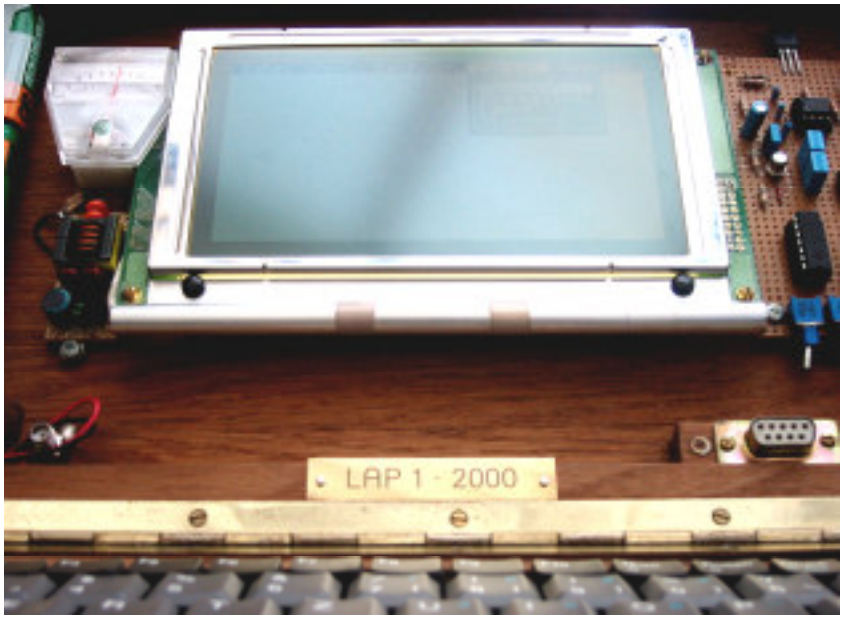
A friend of mine allowed me to use his internet access to search for datasheets, an assembler and a compiler. The assembler was easy to find, but the only 68k freeware compiler I could get was a funny undocumented tool which was written in times before ANSI C was invented and the compiler and me spent a lot of time to find out which syntax he preferred. By this we got best friends!

Keyboard and OS allow to enter all Greek letters. For this reason, LAP!1 defines its own ASCII subset.

A fly examining the memory connections.



Clockwise: display power switch, CFL backlight switch, main power switch, FLASH access indicator LED, power source switch and serial connection.



On the left: battery, voltage indicator and voltage converter for CFL backlight. Below: connector for AC adapter. The passive LCD display shows 16 lines by 40 columns and is readable even in direct sunlight. In this case CFL backlight is not needed and power consumption is reduced by 60%.

There was a lot of programming to do. As I designed my own mainboard architecture, there was not a byte of foreign code that would work on my system, and as the compiler was not very tolerant concerning standard C programs, adapting existing C programs was as difficult as writing them completely new. But this was more a consequent design principle than a problem to me. Later, I enjoyed the ability during debugging sessions to go down deeply into the self-written OS routines to find out what happens there.

Compared to the software, the hardware was finished relatively early. The last change was increasing the amount of memory from 256KB to 1MB. From the beginning on I feared that the mainboard would get too huge to fit into a case with base measurements just a bit bigger than a DIN-A4 sheet. But although I did not make use of any SMD components, the final mainboard measurements were surprisingly small.
As soon as these measurements were “stable”, I drew plans of the case and went to a cabinetmaker who realised my ideas (I am not a specialist in working with wood…)

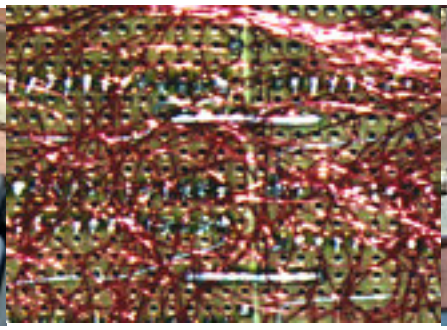
While the casing was worked on, I finished the software. The operating system (LAP!OS) was declared to be ready when it was able to run programs (one program at a time) and implemented a small file system. LAP!OS is stored in ROM and offers a great variety of functions that allow programs to display data on the text display. Menus and dialog boxes are also supported, although the OS itself is a command line OS.
The most powerful application has become LapWrite, my word processor. Although the text display does not show formatted text, LapWrite supports different styles for paragraphs. Using the serial connection, data can be imported directly into Word Documents on a Windows PC and appear already formatted.

Thursday, 8th of June 2000, was the great day when case and electronics came together. After that, only a few software details have been added.

Today, my good old 8 MHz LAP!1 stands next to my new 2.4 GHz Notebook and boots much quicker. Although I often prefer the comfort of a modern computer, LAP!1 is still my first choice for word processing. Its little weight and the 4-6 hours lasting rechargeable battery packs have not been beaten so far.

Notebook, photos and text
by Friederich Kupzog,
case by Sidika Kordes

Mechanical details. The chain keeps the display part in position and allows different angles.



Free wiring reduces costs and capacitive coupling between signal lines.

This cable connects the mainboard with the display controller.