Journal of Universal Computer Science, vol. 2, no. 6 (1996), 474-483
submitted: 17/4/96, accepted: 17/4/96, appeared: 28/6/96 © Springer Pub. Co.

ASIC Design at Home

Heinrichmeyer Friedrich
(FernUniversität in Hagen, Germany
fritz.heinrichmeyer@fernuni-hagen.de)

Abstract: A possibility to overcome the problem of organizing a diploma–thesis at a distance university with focus on practice in the design of Application Specific Integrated Circuits (ASICs) is presented with the aid of an example. A student of electrical engineering designed a fast arithmetical unit for use in a digital filter with a VHDL software package (ALLIANCE [see laboratory 1994]) freely distributed on the internet. He did his work at home with a standard personal computer running a free Unix clone (Linux [see Johnson 1993]) in summer 1993.

Key Words: ASIC design, integrated circuits, free software, unix, Linux, PC software, diploma thesis

Category: B.7.0, K.3.1

1 Introduction

There are two major drawbacks for distant teaching universities especially concerning electrical engineering:

- missing training of certain computing techniques which students on conventional universities learn by observing tutors in undergraduate courses working on concrete exercises
- limited number of subjects for thesis suitable for dealing with at home (without expensive measurement equipment)

The first point leads to the use of multi media techniques and improved communication channels like ISDN, video conferences and video phones. The second problem could be solved by cooperating with local universities and companies or by more or less software related subjects. This here includes also themes with focus on literature research and programming tasks.

In the field of engineering there is also the possibility and need to work with CAD tools for realizing hardware design (in electrical engineering this means silicon chips or printed circuit boards). This topic will be covered with in the next sections.

2 CAD software for electrical engineering

As today most of distant teaching students are supplied with a personal computer we could imagine that students could use CAD-tools at home to design an electronic circuit or an ASIC chip. Here we have the problem of availability of suitable software. Commercial software for ASIC design could be given away to the student but to communicate with the teacher it is necessary that at least two persons have access to the program. Moreover for solving practical problems over communication lines the software should run on similar machines. Buying the package two or more times is often too expensive.

The demand for similar machines unfortunately sorts out software for workstations, as their prices are too high for students. In Germany the majority of computers are Intel based PCs, running MS-Windows (they where the first ones which allowed us to type in German umlauts). For this software platform only expensive products and limited demo versions are available in the field of electrical engineering. Moreover there are no packages for the full custom design of ASICs at all.

On the other hand many universities all over the world develop CAD-software for silicon chip design dedicated to Unix-Workstations. Since some Years now there are free variants of Unix (Free BSD, Net BSD,...) and Unix clones (with constraints Minix, Coherent^{TM}) available.

The author made especially good experiences with the Unix clone Linux. We can simply recompile graphical editors for silicon chips (e.g. Magic [see Mayo 1994] free source), the electrical circuit simulator Spice (source available from the university of Berkeley under certain constraints [see EECS 1993]) and Alliance [see laboratory 1994].

3 Why Linux?

Linux is a freely distributable Unix^{TM} compatible operating system for Intel microprocessor (386 or higher) based IBM^{TM} PCs and compatibles written by Linus Torvalds from the University of Helsinki, Finland. It was developed by a unique world-wide collaboration of programmers over the internet, and is covered by the GNU ([see Stallman 1993]) General Public License. Linux is a network operating system, much like ones used for years on engineering and professional workstations. The surprisingly high performance of file related operations is caused by extensively buffered read and write (dynamically adapted hard disk cache) and using high performance file system architectures.

It should be mentioned here that Linux is not the only free Unix (clone) available, one could also use FreeBSD or NetBSD and a lot of if not all of the following arguments are also valid for these alternatives.

The following sections mention the points which influenced the decision for Linux sorted by importance in decreasing order.

3.1 GNU software tools

A lot of people have unpleasant memories of Unix due to their experiences with machines controlled through teletype devices. Today the Unix systems are more comfortable first of all on account of the existence of free software. Most important are the GNU (Gnu is Not Unix) packages, e.g. there are shells (command line interpreters) which not only offer a command line history (like “doskey”) but also file name completion (when typing the TAB key the shell completes file names or command names as far as the already typed characters are unique) which together with the ability to use very long filenames (we have to type them actually only at creation time, not when we use them) eases a lot of daily tasks with the computer. Maybe OS/2^{TM} or a coming 32–bit Windows version could also be a platform for GNU software in the future.

It is the experience of the author that students don’t miss a graphical interface like the one of the MS-Windows^{TM} program manager in the first place but a lot of people miss a fully functional shell under MS-Windows. On demand there are program manager clones also installable under Linux. Due to the comfort the GNU shells offer, most users (i.e. all users the author knows) prefer also in the graphical environment X11 a command line interface or join the EMACS community (see section 3.5).

3.2 Net support

Linux has fully functional network support. It can be seamlessly integrated in a workstation network (including Network Information Service NIS). It can also act as server, client or relay for Windows for Work groups, LanManager^{TM} and Novell^{TM} networks and of course gives full access to the "data highway" Internet (if hardware makes it possible).

3.3 Documentation and Support

Software from the internet has as "last resort of documentation" at least the source code in it. Practically there is always some kind of printed or printable documentation. Especially valuable are the advises which gives the internet community to each other by means of net news. As the sources are free, "patches" can be distributed. The user is not forced to wait for new version (and to pay for it, what means aside of the money a large delay as there has to be done a lot of administrative work at the university).

Especially for Linux there are a lot of text books for different degrees of interest available (from a first overview to the "kernel hackers guide", [see Johnson 1993] gives starting points).

3.4 Stability

The system is very stable, when a process ends abnormally the whole system is still stable and usable. Of course there is the advantage that different processes have their own address space. No unwanted writing or reading of data from different processes can occur. Program code segments can not be altered unwanted. Everything works in a "protected mode".

3.5 Programming

For Unix users a very common feature of software in a developing state is that it contains "debugging" info (we have the source code) in a way that a memory dump for a process contains all information necessary to locate bugs in the source code (high level language like C, or PASCAL). Unix systems can be configured in a way that this dump happens whenever a process violates the protection rules and therefore will be aborted. From this information it is possible to "repair" the software and not necessary to wait for future "releases".

As Linux is a 32 bit operating system which provides virtual memory a programmer can concentrate on algorithmic problems and has not to keep in mind the actual size of a peace of memory. If an algorithm works for data < 64 kByte it does also for almost unlimited sized arrays. It is only a matter of time.

Daily problems in managing data, converting to different formats can be solved by short scripts for free available tool languages like "perl". Graphical user interfaces can be relatively easily created with interpreted languages like tcltk, tkperl and more. This interpreted languages can be extended by dynamically loaded libraries (similar to Visual Basic^{TM} under MS Windows^{TM}) to satisfy needs for high speed. The GNU–Emacs Editor [see Stallman 1993] is commonly used as general desktop tool (file manager, calendar etc.) and programming environment (compiling of programs and jumping to error locations in the source files is supported as parallel subprocess!).

Of course all the insight students have gained when working with Linux are applicable to other Unixes and so for dealing with “real” workstations generally. This is valuable for its own sake especially in teaching engineers.

4 What is missing

There is not yet anything which is directly comparable with the standard office packets for MS-Windows (although some people would mention the “Andrew Toolkit” at this point). The user has to manage things a little different if he insists in only using Linux. On the other hand it is possible to run binaries for Intel processor based commercial Unixes and therefore e.g. “Word Perfect^{TM}” or “Lotus^{TM}”.

In 1996 some commercial software in the field of engineering and mathematics is available and also an office software package is at least announced. Apart from this Linux offers a DOS emulator which can run e.g. “Word Perfect” and which we used successfully for a more recent project ([see Baensch 1995]).

5 Resources needed

A relatively cheap personal computer with a price presently around 2000 DM operates as a sufficient hardware base for running programs developed for Unix workstations.

Linux Computers are mainly Intel based, but work is done to include computers with Motorola 68000 and PowerPC based architectures into the Linux community. A distribution for the Atari is already there and some brave enthusiastics use Linux on sunsparc^{TM} workstations.

Today the resources for hard disk space needed for a practically usable Linux system are approximately in the same size or smaller than needed for MS-Windows and an office package for it. More critical are the demands on main memory (RAM). Linux together with the graphical environment X11 works with 4 Megabytes, but the improvement when increasing the amount of RAM to 8 Megabytes when using X11 is considerable. If we want to compare it with OS/2-Warp^{TM} we have to bring Linux to single user mode and then look at the free memory when running X11 (X11 is the hardware independent graphical environment for Unix workstations with integrated network support and freely distributed but copyrighted source code). The free physical memory amount is approximately 700 kByte in the case of 4 Megabytes RAM totally on board. The amount of virtual memory is only limited by the hard disk space and is configurable.

A special feature of Linux compared with other Unix systems is the existence of IBM PC hardware specific graphical software. So there is no need for using X11 in all cases. Without the X11 system 4 Megabytes physical memory is sufficient. This was the case in the project described in the next sections.

6 The project

Within three month a diploma thesis on electrical engineering ([see Rueter 1993]) with the goal of designing an arithmetical unit for digital signal processing from the relatively high level of schematic input down to the physical layout (silicon chip) could be finished successfully.

6.1 Wave digital filters

Algorithms for digital signal processing of image data tend to imply very strong requirements on computing power. Therefore an arithmetical unit for bit parallel computing has been developed (2 port adaptor [see Tomlinson and Mirzai 1989; Lawson and Mirzai 1989]) which is the main building block of "wave digital filters". This filters model digitally electrical circuits by using formulas which are very similar to formulas for wave propagation e.g. in the field of microwave technique. This leads to advantages in computational stability and minimal hardware requirements for certain digital filters ([see Fettweis 1986]).

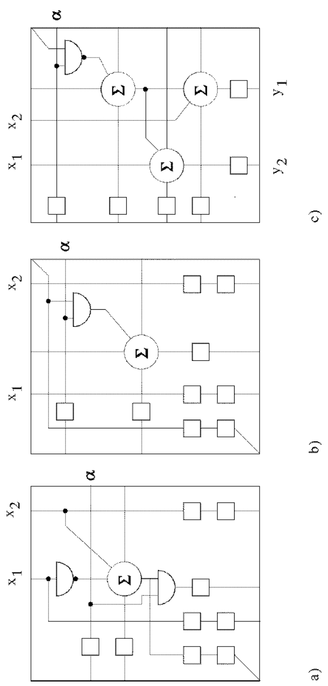


Figure 1: Schematic of systolic cells

6.2 Systolic arrays

The arithmetical unit was realized as "systolic array" [see Kung 1985] on bit level. Systolic array processors organize computations for an algorithm in a way that data is only shifted between neighboring more or less identical processing units at specific clocks ("systolic"). The objective is to maximally exploit parallel and pipelined computing for a given algorithm to achieve maximal speed. Data transport only between neighbors and identical building blocks also brings advantages in implementing special processors on silicon chips. The "systolic" principle on the other side leads to problems as it tends to produce peaks in electrical current consumption.

The logical verification of the design was given by the results of a former thesis (see figure 1 [see Bui 1991]). Within the subsequent work the design of a physical layout should be finished with the aid of the software package "Alliance" available on the internet. So the work had two objectives:

- designing the chip as an exercise down to the silicon level
- gaining experience on the practicability of "workstation software" recompiled under Linux to use for work with students at a distant teaching university.

6.3 VHDL

First of all, the whole building block had to be described in the currently most important hardware description language VHDL (Very High speed integrated circuits Description Language [see Perry 1991; Kunz 1995]) which has some similarities to ADA, PASCAL or MODULA (all these languages are available for developments as GNU software).

Such a description is named "architecture" description in VHDL. At the time of the project only a subset of the VHDL language was actually implemented in Alliance. No sequential (algorithmic) description of the architecture was available. Unfortunately this kind of description is together with the first mentioned clearly the major advantage of the VHDL language as it enables easy to build verification environments by comparing different descriptions of a given architecture.

The instantiation of certain low level building blocks in a form similar to a net list should behave the same as the sequential description which is very similar to a PASCAL program for the underlying algorithm. For an adaptor network the sequential (or "behavioral") description degenerates to two statements containing simple arithmetic variable assignments.

State of the art now is automatic synthesis of structural or concurrent descriptions out of sequential statements. This means automatically designing circuits to implement algorithms although there are still preliminaries for a successful design of course. "programmers" still have to keep in mind that they design hardware.

6.4 Alliance

As systolic arrays are regular and schematics of the array have very regular structure, the restrictions given by Alliance were not fatal. To overcome the restrictions instances of array cells where generated by a small utility program written by Bernd Rüter.

The main point was that there existed a VHDL based simulator and that there was routing software which actually designed the silicon chip.

6.5 Debugging

Problems arose from the fact that Alliance in the state of version 1.2 in the beginning only worked reliably on the Sun^{TM} workstations of our institute. On "Linux workstations" some utilities produced "segmentation faults". The faulty programs tried to access memory outside of their allowed address space. After the student reported problems we [see Kunz 1993] recompiled the Alliance package with debugging info in it and reconstructed what happened at the home PC of the student. As there was no direct data connection (modem) to the student some hours where spend in transferring typical computer commands and switches (in this case Unix is very similar to MS DOS^{TM}) by spelling them out.

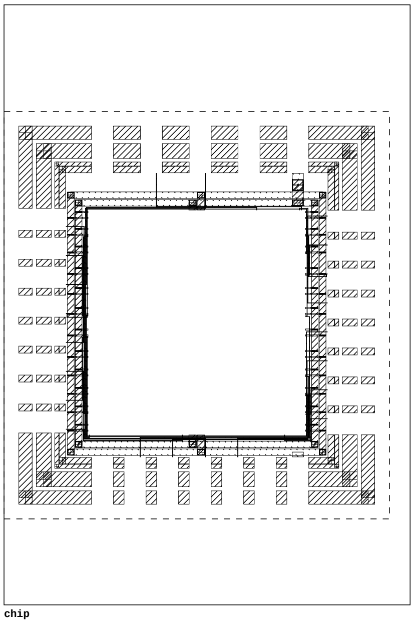


Figure 2: Layout of chip without core

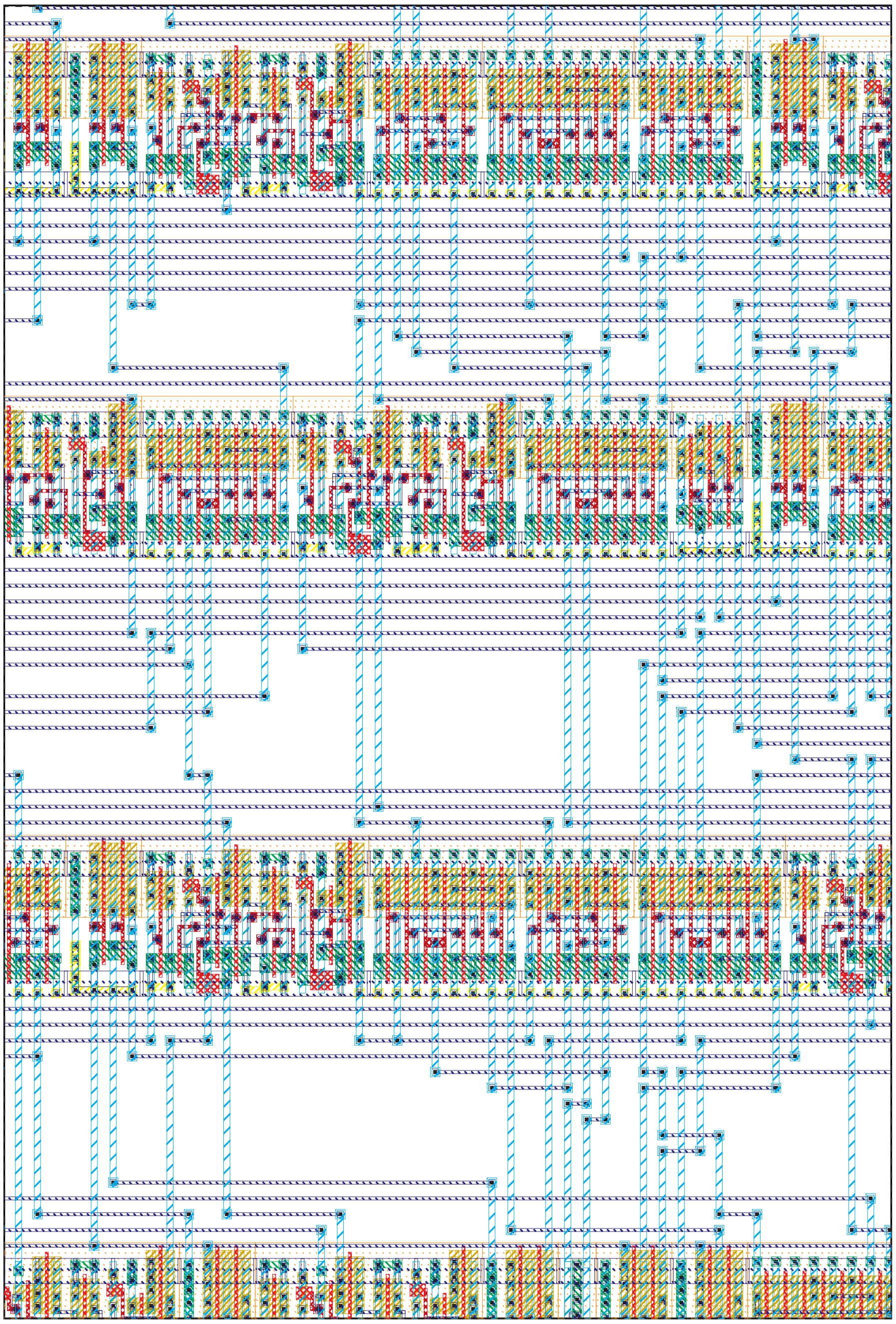
After reproducing the fatal error it was possible to locate the line in the source code (here “C”–programs) which caused the error. We found out that there really where some logical errors in the treatment of data space requested and given back to the operating system which also should have caused the program on the workstation to stop. That means the successful runs at the workstation should have not occured. What ever we regard as right behavior, the error could be isolated and repaired (of course we submitted the patches also to the university of Paris and we where mentioned as contributors in following releases of Alliance).

6.6 Layout result

The selected technology for this experimental project was the “standard cell” approach. Alliance produced a chip layout by arranging lines of well proven geometric structures taken out of a library of standard cells. The layout results (3 and 2) show clearly that the systolic array method can only be exploited with full custom designs (that means the systolic processors themselves have to be designed with e.g. Magic [see Mayo 1994] on a standard cell level). This was not realizable within the constraints of a three month diploma thesis.

6.7 Using Spice

After all problems where solved, the student verified the physical layout with the aid of the circuit simulator Spice [see EECS 1993], developed and distributed by the university of Berkeley. Compiling and installing of Spice was fortunately relatively easy. This software was again available as “C” program source which could be build by heavy



core-2x4 5x5 pages total

Figure 3: Part of inner layout

use of a tool for programmers ("make"). The author applied the free available program "GNU–make" together with the "C" compiler from the GNU software group, he had to struggle with a small bug in the current release of this make program and some small problems with some support scripts. It is worth mentioning that the problems where solved by announcing the problems to the Linux community. Another friendly guy posted a collection of scripts and patches for Linux which enabled us to build Spice successfully.

The Spice binary built in this way has no limitations compared to popular demo versions of a special PC Version of Spice. The author could for instance simulate a network containing 3 operational amplifiers using a very complicated model for the operational amplifiers which the Spice demo version available for MS DOS rejected.

6.8 Resources used at home

Bernd Rüter was unemployed for the time of the project. He fully concentrated on his thesis so he could stay in the estimated time limit of three month work. His equipment was a 386-DX40 Intel based PC with 4 Megabytes Ram, a 130 Megabytes hard disk, EGA color monitor and a HP Desk-jet 510 printer.

7 Conclusion

A diploma thesis on CAD design of a special arithmetical unit ready for manufacturing a silicon chip could be organized successfully with ASIC design software originally developed for workstation by a student of electrical engineering at the FernUniversität in Hagen at home. Encouraged by this experience another Linux related project followed 1995 [see Baensch 1995]

The free Unix clone Linux and GNU free software showed up to be reliable and suitable for this and similar tasks.

References

[**Baensch 1995**]Baensch A. (1995). Codegenerator für den Signalprozessor TMS 320 C5x. Diplomarbeit am Lehrstuhl für Elektronische Schaltungen der FernUniversität in Hagen.

[**Bui 1991**]Bui X. T. (1991). Integrationsgerechte Umsetzung von Algorithmen zur Signalverarbeitung. Diplomarbeit am Lehrstuhl für Elektronische Schaltungen der FernUniversität in Hagen.

[**EECS 1993**]EECS (1993). Spice Simulation Program. ftp server ilpsoft.EECS.berkeley.edu.

[**Fettweis 1986**]Fettweis A. (1986). Wave digital filters: Theory and practice. *Proceedings of the IEEE* 74(2):270 – 327.

[**Johnson 1993**]Johnson M. K. (1993). Linux Meta FAQ. ftp server tsx-11.mit.edu.

[**Kung 1985**]Kung S. Y. (1985). Vlsi array processors. *IEEE ASSP Magazine* pages 4 – 22.

[**Kunz 1993**]Kunz L. (1993). Private Mitteilung.

[**Kunz 1995]**Kunz L. (to be published 1995). *Entwurf von Wellendigitalfiltern mit VHLD*. PhD thesis FernUniversität in Hagen Fachbereich Elektrotechnik.

[**laboratory 1994]**laboratory M.-V. (1994). ALLIANCE CAD System, Universite Pierre et Marie Curie, Paris France. ftp server cao-vlsi.ibp.fr.

[**Lawson and Mirzai 1989]**Lawson S. S. and Mirzai A. R. (1989). Improved 2-port systolic adaptor for wave digital filters. *IEE Proceedings Part G* 136(3):121 – 125.

[**Mayo 1994]**Mayo R. (1994). MAGIC IC layout tool, University of Berkeley. ftp server gatekeeper.dec.com.

[**Perry 1991]**Perry D. L. (1991). *VHDL*. McGraw Hill.

[**Rueter 1993]**Rueter B. (1993). VHDL–Simulation und Layoutsynthese mit Alliance. Diplomarbeit am Lehrstuhl für Elektronische Schaltungen der FernUniversität in Hagen.

[**Stallman 1993]**Stallman R. (1993). GNU Manifest. ftp server prep.ai.mit.edu.

[**Tomlinson and Mirzai 1989]**Tomlinson A. and Mirzai A. R. (1989). Single-chip 2-port adaptors for wave digital filters. *Electronic letters* 25:1553 – 1555.

Acknowledgements

This work was made possible by the engagement and support of Dipl.–Ing. L. Kunz, Dipl.–Ing B. Rüter Dipl.–Ing. Xuan Thao Bui and Prof. Dr.–Ing H. Wupper.