Emulation of x86 computer on FPGA

Stepan Vyazigin
Department of Artificial Intelligence and Big Data
Al-Farabi Kazakh National University
Almaty, Kazakhstan

Anuar Dyusembaev
Department of Artificial Intelligence and Big Data
Al-Farabi Kazakh National University
Almaty, Kazakhstan

Madina Mansurova
Department of Artificial Intelligence and Big Data
Al-Farabi Kazakh National University
Almaty, Kazakhstan

***Abstract*— It is well known that, emulation in the form of software is the predominant method for engineers to evaluate the capabilities of the studied microprocessors and embedded systems. There are three main criteria for evaluating a model using software tools: modeling speed, model accuracy, and model completeness. The increasing complexity of the processor and the tendency to have an increasing number of processors on the chip put a strain on simulators to achieve all of the above criteria, including accurate fixation of processes in the OS. Thus, the main task in our work is experiments-prototyping using an emulation system and analysis of the results of the described experiments, which satisfies all three criteria. The system is a Board with FPGA, RAM, ROM, real-time clock, DAC chips, and connectors for connecting a monitor, keyboard, and mouse manipulator soldered on it. The system is based on the FPGA Cyclone IV from ALTERA. Which, thanks to a sufficient number of logical cells, allows you to simulate not only a single processor, but also other components of the computer as a whole. Therefore, you can apply architectural changes to the processor and evaluate their impact on the entire system. We use this FPGA-based emulation system to validate the computer's FPGA emulation capabilities. The paper justified the possibility of emulating a computer on an FPGA and its ability to run real operating systems that are not stripped down. The novelty of this project is that unlike other similar projects, the system developed by us allows you to emulate a full-fledged personal computer with an x86 processor architecture, on the basis of which you can emulate more modern computers with processors. For example: Intel Atom or Intel Celeron. However, to achieve these goals, you will need to use a more developed FPGA, based on the methodology proposed in this paper.**

Keywords*— FPGA, emulation, PC, electronics, Pentium

I. INTRODUCTION

Computer architecture research has traditionally used software to emulate a single-core processor such as SimpleScalar [1]. Both previously and today, improving the architecture of processors and memory hierarchies is an urgent task. In addition, there are currently additional optimization requirements across the entire system stack (processor architecture, command set, device drivers, operating system, and applications) with multiple processors. However, the above-mentioned research at the system level is constrained by a certain contradiction between the speed and detail of modeling software and hardware components, and this contradiction is inherent in software simulators, traditionally used for innovations in microprocessor systems. Field Programmable Gate Arrays (FPGA) are considered as a solution to this contradiction and are aimed at developing a new research infrastructure of the system stack that simulates a complete system (processor, video card, sound card, North and South bridges, network modem, etc.) [2]. Flexibility, speed (both development time and simulation time) , and sufficient FPGA capacity allow developers to emulate microprocessor systems and computers in General. It should be noted that with the onset of the 4th industrial revolution [3], in the development and implementation of smart technologies in the urban environment, such as the MQTT Service Broker [4] and With the transition of states to electronic provision of services, many security problems have arisen both for personal data [5] or the Event Handler system as a MQTT [6] and for the protection of systems in general. In connection with the software implementation of various architectures, FPGA-based projects allow not only to parallelize information processing at the hardware level, but also to maximize information protection from hardware backdoors or so-called backdoors on the part of chip manufacturers, for example: Intel Management Engine [7]. However, one of the most difficult issues facing the development of an emulation system on an FPGA is compatibility with existing operating systems (OS). Manufacturers have developed processor cores for FPGA that are very small and simple, but have limited support even for embedded operating systems like Barebone. In addition, in order to run existing OS binaries, including closed source ones such as Windows, it forces developers to consider binary translation of OS files as a solution to the binary translation problem [8]. In this research and convenience in this work, we emulate a version of a commercial desktop computer with an x86 processor on an FPGA to run real operating systems. To be more precise, we replaced the computer with a debugging Board with the necessary components soldered on it. Debugging Board components: FPGA, RAM, ROM, real-time clock, and some other chips are required to connect the FPGA to PC peripherals. The following devices are emulated on an FPGA:

- Pentium compatible processor running at 50 MHz with a 32 KB cache.
- IDE controller.
- SD-IDE interface Converter.
- Intel 8259 compatible programmable interrupt controllers
- Intel 8237 compatible direct memory access controller (DMA)

- Sound Blaster – sound card
- Intel 8254 compatible programmable three-channel timer and counter
- Intel 8042 compatible keyboard and mouse controller
- Standard VGA video card
- 8250 UART-COM port

It is important to emphasize that the FPGA-based computer emulation system allows us to run real operating systems on the FPGA, such as DOS, FreeDOS, Linux, and Windows, and interacts with real peripherals. The ability to emulate a PC based on FPGA provides a powerful tool for research and modification of more advanced microprocessors. Although our proposed emulator system does not contain a modern microprocessor, its applicability to modern architectural research increases due to the advanced modeling capabilities.

II. Features of emulation systems

The concept of using FPGA for faster and more accurate research of the microprocessor development space has recently become widespread, which has led to an increase in the number of publications on this topic [9]. Some of these works focus on speeding up simulation time by offloading highly detailed resource modeling in FPGA, while the software simulator remains the core of the emulation environment [10]. Other studies often focus on a single architectural innovation (for example, transactional parallel systems [11], caching [12], vector-thread processors [13]) and building models of the corresponding hardware based on FPGA. In addition to these approaches, we have implemented a full-fledged microprocessor on FPGA, which allows you to use CPUs with different architectures, for example: x86, x64, ARM, and others. Most of the RTL models of microprocessors have already become available for the SPARC V8 , Niagara, and PowerPC . These cores can be synthesized in FPGAs and are designed to facilitate design, as seen in Jones et al. [14]. Our emulation platform also provides several orders of magnitude faster simulation compared to software emulators such as Bochs and Qemu. Some existing developments in embedded systems that apply add-ons to an FPGA-based core have already been listed above. The utility of the application microarchitecture variation was seen in [17], and its automatic navigation in [18]. In addition, the effect of including user instructions in such kernels has been studied [19]. Unlike Amber (Conor Santifort), Cortex-M1 (ARM), Navre (Sébastien Bourdeauducq), LEON (ESA, Aeroflex Gaisler), OpenSPARCT (Sun) [23], ZPU (Zylin AS) , HIPP [20] and others, we focus on desktop systems, interaction of peripherals and an operating system that supports x86 architecture.

III. FPGA-based emulation system

In our work the emulation environment consists of four main components:

FPGA on which the Pentium processor and PC motherboard are emulated;

- hardware, including the debug Board and peripherals;
- software / operating system;
- necessary software for the implementation of the FPGA project (Quartus II).
- Let's describe each of these four points in more detail.

A. The Emulated hardware

- The processor used in developed emulation system is a recreated copy of the Pentium [25] released after i486 and before Pentium Pro in 1993 using 0.6 micron technology, consisting of 3.2 million transistors and initially operating at a frequency of 75 MHz. It is a 32 bit processor with 5 step pipelining that supports the IA32 instruction set, which includes floating-point instructions using the built-in floating-point module in the pipeline. It is equipped with a level 1 cache of 8 KB for data and instructions.
- IDE hard disk and floppy disk controller, Intel 8259 compatible programmable interrupt controllers, Intel 8237 compatible direct memory access controller (DMA), Sound Blaster, Intel 8254 compatible programmable three-channel timer and counter, Intel 8042 compatible keyboard and mouse controller, standard VGA video card, 8250 UART-COM port were recreated from the technical documentation of the address space distributed by BOCHS[24].

The debug Board contains the FPGA (Fig. 1) and the necessary chips and connectors for connecting peripherals. These include SDRAM, flash for bios and vgabios, a TTL logic level Converter for com port, and a DAC for VGA. The FPGA used for emulation is a 90 nanometer Altera Cyclone IV device. A more detailed analysis of the Cyclone IV resources used by the system will be given in section 5.

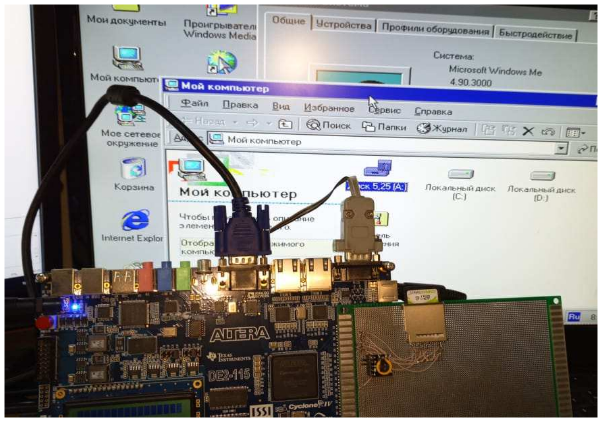

Fig. 1. Image of a PC emulator system based on an FPGA debug Board equipped with different hardware peripherals running Windows ME

B. Emulator debugging Board

The DE2-115 Board from Terasic was chosen as the basis for the main Board on the ALTERA Cyclone IV e ep4ce115f29c7 FPGA. This Board has all the necessary connectors for connecting peripherals. Such as a keyboard, mouse manipulator, VGA display, logical level Converter for COM port, etc. as well as 128 MB SDRAM and 8 MB flash for storing BIOS and VGABIOS, as well as a connector for connecting SD cards. In addition, a second SD card and an external real-time clock are connected to the debug Board. SD cards play the role of IDE hard drives and floppy disks. The other necessary devices are emulated on the FPGA.

1) *Motherboard*

As previously mentioned in the emulation system, the FPGA is taken as a basis on which all the necessary PC components are recreated. The emulator motherboard (Fig. 2) can be divided into 2 main parts: FPGA and peripherals.

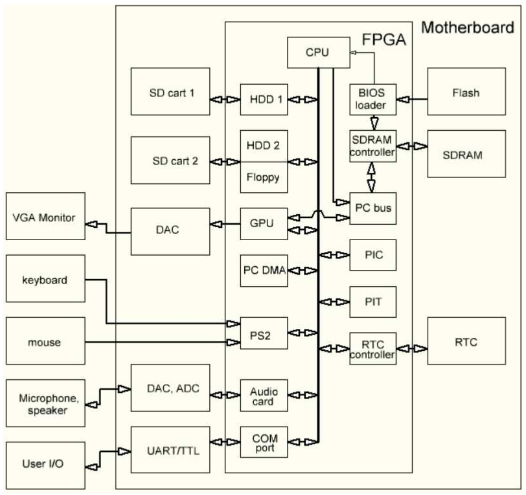

Fig. 2. Block diagram of the emulator

Peripheral devices include:

- Flash memory is non-volatile memory that is used to store BIOS and VGABIOS with factory settings.
- SDRAM is a volatile memory that is used as PC RAM.
- RTC (Real Time Clock) is a CMOS RTC electronic circuit MC146818 designed to take into account chronometric data (current time, date, day of the week, etc.), is a system from an autonomous power source, taking into account devices and a tiny static memory with a very low power consumption in which the basic BIOS settings are stored.
- SD cards - in this case, they were used as hard drives on which the OS was installed. One of the SD cards can serve 2 roles: hard drive and floppy. Switching between roles is carried out depending on the image recorded on the SD card.
- DAC (digital to analog converter) - this device converts a digital signal that outputs the GPU to the FPGA into an analog signal for connecting to a monitor using the VGA video interface standard.
- DAC, ADC (digital to analog converter, analog to digital converter) - this device converts the digital signal that the audio card outputs to the FPGA into analog for connection to speakers or headphones. It also converts an analog signal to digital for connecting a microphone to a sound card.
- UART / TTL is a logic level converter chip from 3.3 V to + -15 V for connecting to the port.

C. *Testing operating systems*

The challenge of our FPGA-based system is the ability to load real operating systems. We successfully installed unmodified versions of FreeDOS, DOS 6.2, Windows 96, Windows 98, Windows ME, Windows 2000, Windows XP, Tiny core linux, Fedora Core 4, Red Hat 9; the installation procedure did not differ from the usual desktop system except that instead of the installation disk, we used its image downloaded to the SD card.

To measure the OS boot time for each of them, a program was written and added to autoload with the sole purpose of displaying a message via the COM port. The measurements were carried out using the standard utility SignalTap II Logic Analyzer built into Quartus II, the results of which can be seen in Figure 3, the input parameters of which are:

Clock frequency: internal generator at 10 Hz.

Triggers: Start FPGA and start data transfer via COM port.

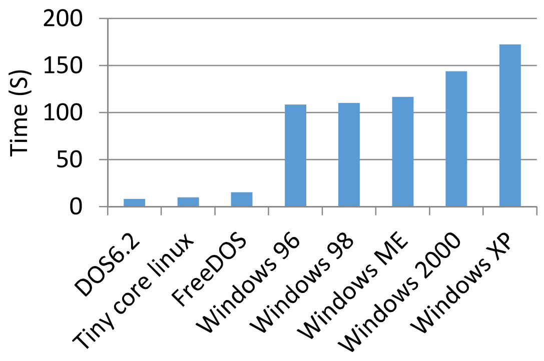

Fig. 3. PC boot schedule with OS

As you can see from the graph (Fig. 3), loading an emulated PC with an OS without a graphical interface takes from 8 to 16 seconds, and with a graphical interface from 100 to 120 seconds. The average startup time of some standard applications can be seen in Figure 4.

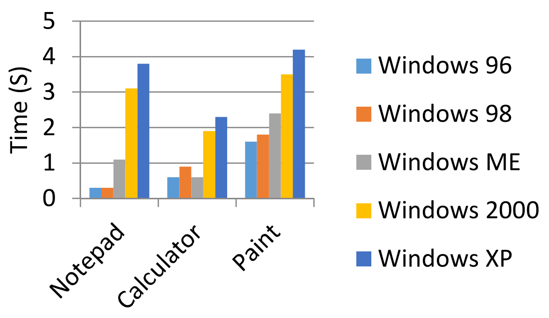

Fig. 4. Application launch schedule

Typing is definitely done at full speed. To measure the maximum search time for text files, the following experiment was carried out:

Files of different sizes with an arbitrary set of characters and the search word at the end are given. The standard application “notepad” was chosen as the search application. The search time was measured by filming the search process with a video camera from the moment the “Search” button was pressed and until the end of the search, the results of which are shown in Figure 5.

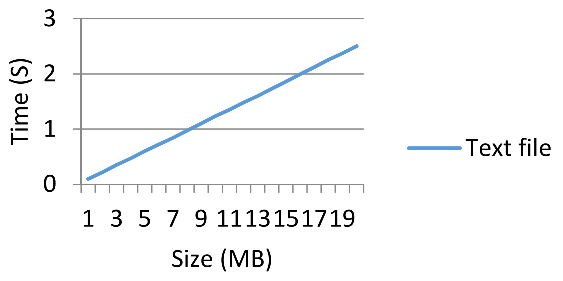

Fig. 5. Graph of delay during search in a text file

Thus, the system is ideally suited as a desktop computer for very simple non-graphical applications. And for more complex ones, you need to replace the video card, the requirements for which depend on the application.

D. *Development of an emulator on FPGA*

For PC synthesis and placement, we use Quartus II 16.0 64 bit, and for routing emulated peripherals, we use The qsys subprogram built into Quartus II. The entire compilation process takes about 1 hour to synthesize, map, place, route, and generate the bit stream, followed by an additional 2 minutes to load the bit stream to the device. This is orders of magnitude faster than the manufacturing time of the silicon implementation of the processor, which can be inserted directly into the motherboard. From a debugging perspective, Modelsim is used to simulate VHDL in a step with a software simulator that simulates the original behavior of the processor and emulated devices. A set of regression tests is used to make sure that the processor is still running on x86. Regression tests are a subset of those used to test the original Pentium.

IV. Description and some features of SOFT processors

This section will describe the improvements of the integrated Pentium processor when increasing the amount of memory caches, as well as the ability to connect already emulated processors to our system.

A. *Third-party SOFT processors*

Today, there is a fairly large selection OF soft processors for FPGA, both old and relatively new, but in most of them the main distinguishing feature is the RISC architecture that allows you to run Linux on them at best.

For this system, the following SOFT processors were tested for the role of the CPU: Amber-a processor compatible with ARM A23; VexRiscv-a processor with the RV32I instruction set; as well as a number of other processors such as: LEON; OpenSPARC; CPU86; ZetCPU (x8086); OpenRISC, NIOSII. The space occupied by these SOFT processors is shown in figure 6.

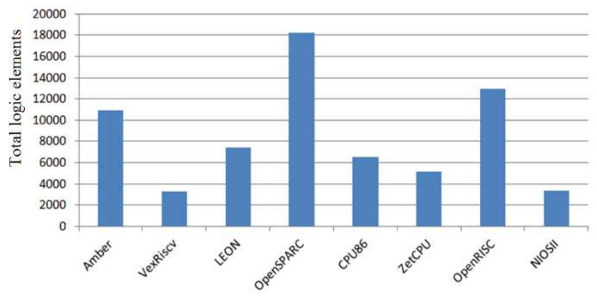

Fig. 6. Space occupied by various SOFT processors

B. *Emulator debugging Board*

The cache level 1 of the selected Pentium processor is 8 KB, which is of course small by today's standards, but enough to demonstrate the possibility of changing the processor configuration. Recall that there are two similar caches, one for data memory, the other for instruction memory, each of which has a size of 8 KB and is a two-way associative set of 32 bytes per cache line [21]. In our experiments, the cache sizes were increased 4 times, and became 32 KB 8 band associative caches. The LRU algorithm, which determines which row is pushed out in the full set, has also been extended to handle sets of 8 cache rows. Instruction and data caches can be individually configured for 8 KB or 32 KB versions. for larger volumes, you need to replace the FPGA, but in this work, we always keep them the same size 32 KB.

V. Experiment with the emulator system

In this section, we analyze and test a PC system with an emulated Pentium processor based on FPGA to obtain the following results: Allocation of system resources of the emulated PC according to the CAD stream; Comparison between the original cache of level 1 8 KB and our extended cache of level 1 32 KB. We will look at each stage in more detail. Note that the number of elements used in this paper is considered in terms of FPGA resources. However, the FPGA resource analysis can be used for preliminary estimation of the number of transistors when implemented on a silicon wafer [22].

A. *Distribution of the emulator system volumes*

In this work, we emulated a computer based on a Pentium compatible processor in VHDL for Cyclone IV E EP4CE115F29C7 and noticed that more than half of the device's resources were used; the corresponding data is shown in table 1, taken after completing high-level synthesis and mapping the model in Quartus.

TABLE I. Use of system resources in Cyclone IV

Resource	Used space	Percentage utilization
Total logic elements	104731	91 %
Total combinational functions	95179	83 %
Dedicated logic registers	71273	62 %

In our experiment, 91 % of the logic elements were used to store all the system logic. In addition, 62 % of the register blocks were used. Although more than half of the FPGA resources were used for emulation, there are still enough unused elements on the FPGA to extend the PC's capabilities. Figure 7 shows a breakdown of each Cyclone IV resource used by different blocks in the system.

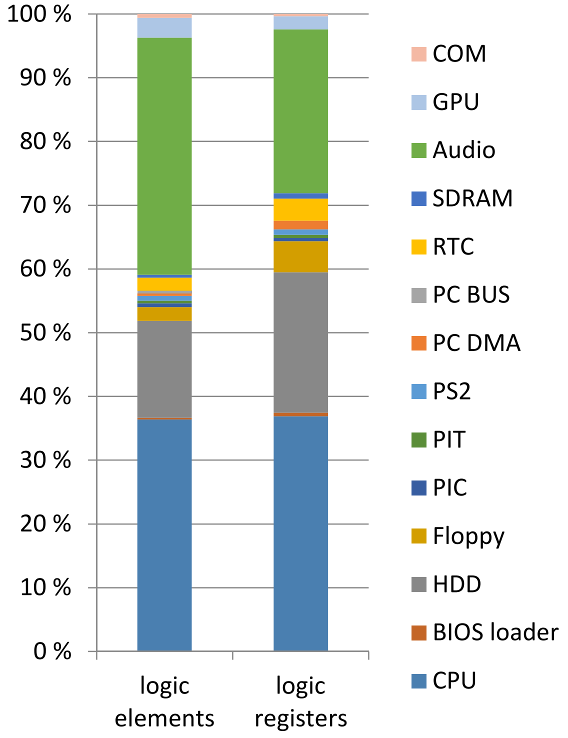

Fig. 7. Space occupied by PC components

As you can see from the graph (Fig. 7), Most of the logic elements and registers were used by the processor, sound card, and hard disk controller. Cyclone IV logic elements were used mainly by FPU, ALU, address generation and cache, conversion of SD-IDE interfaces and audio codecs. The entire memory hierarchy in the experiment (including caches and the bus interface) required approximately 30 % of the logic elements, assuming that even when considering logic alone, almost half of the chip is allocated for communication, leaving the other half for management and actual calculations. Special attention should be paid to the organization of access to RAM and the relationship between a large number of modules via DMA, which is the main factor affecting both the speed and the amount of space occupied.

B. Changing the size of the level 1 cache for the emulated processor

Figure 8 shows the additional FPGA resources consumed when increasing the L1 cache from 8 KB (in 2 directions) to 32 KB (in 8 directions). The expansion required about 24 % more logic, as well as more than 50 % more registers, which made this growth in the L1 cache very expensive in terms of volume. However, the performance gain is quite significant. On average, the performance improvement reaches 16 %, sometimes reaching 40 %. Despite the presence of a significant number of publications devoted to the study of Cache functioning and its interaction with the CPU, OP, OS, and other computer components, this work is of interest from the point of view of controlling the actions of the operating system, such as clearing the cache and replacing it, while maintaining high simulation speeds. However, such studies have not received sufficient coverage in the literature.

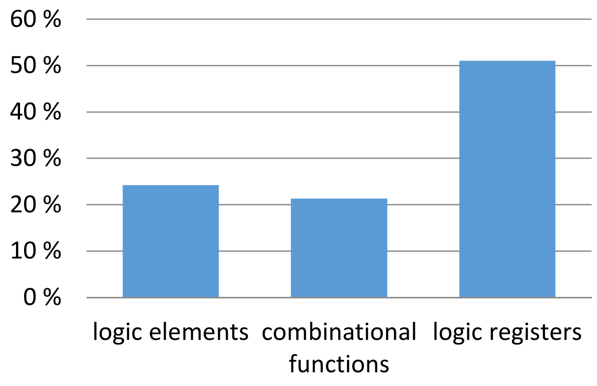

Fig. 8. Increasing the space occupied by the level 1 cache

C. A comparison of the performance of emulators

As you know, today there are several PC emulators such as: Limbo, Qemu, Bochs, DOSBox, VirtualBox, VMware Workstation, Wine, Simics [15], SimOS [16], etc. each of these emulators has its advantages and disadvantages. Some emulators were not affected in this work. For example: VirtualBox and VMware Workstation-do not emulate the processor but use the host processor; Wine and Limbo-are focused on Android devices. To confirm the effectiveness of our emulator, the following experiment was performed:

The conditions of the experiment:

1) 3 PC emulators: Qemu; Bochs; FPGA emulator developed by US

2) processor Frequency 50 MHz

3) 128 MB RAM

4) Same Windows XP image

To conduct experiments, we developed a specialized program that emits the activity of an office computer. Namely:

1) waiting for work to start;

2) open the directory and select documents;

3) typing;

4) move the text editor window.

During the experiments, the load on the emulated processor was recorded. The results of which can be seen in figure 9. Where from 0 to 1 second-waiting for work to start; from 1 to 4 seconds - opening directories and selecting documents; from 4 to 6 seconds-typing; from 6 to 8 seconds-moving the text editor window.

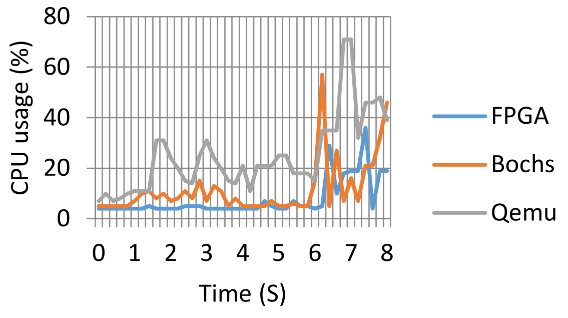

Fig. 9. CPU usage

As you can see from the graph, the FPGA emulator we developed shows good results compared to other emulators in our experiment. Especially good results are seen when working with files and text, while the CPU load does not exceed 5 %, which is 3 times less compared to Bochs and 6 times less compared to Qemu. In this case, the text set that is fed to the emulation system is comparable to the text set in Bochs and in our experiment loads the processor by 4-7 %, in contrast to Qemu which loads the processor by 11-25 % (Fig. 9). Which allows you to make the following conclusion. The FPGA emulation we created is as good as commercial PC emulators, and in some cases even better. It performed best when interacting with external devices such as the mouse and keyboard manipulator (Fig. 9). It should also be noted that during the experiments it turned out that the Bochs emulator does not work well with the mouse manipulator.

VI. Conclusion

The FPGA-based PC emulator is a powerful tool for researching architectural improvements to processors and other desktop components. Its ability to quickly prototype architectural changes and measure their impact at the application level in the presence of a real operating system provides a more realistic research tool without the expensive costs and long design times associated with silicon-based creation.

The system we emulated showed that it can be used to develop and achieve greater efficiency of the source computer by optimizing the entire system stack: architecture, device drivers with installed instructions, operating systems and applications without limiting the time of simulation of the software simulator. For illustration, we used FPGAs with fairly limited functions, but using more advanced FPGAs will allow us to emulate modern multiprocessor and multicomputer systems based on our methodology.

References

[1] T. Austin and D. Burger. The SimpleScalar Tool Set Version 3.0, 1998.

[2] G. Gibeling, A. Schultz, and K. Asanovic. RAMP: The RAMP Architecture and Description Language. Technical Report, 2006.

[3] A. Romanovs, I. Pichkalov, E. Sabanovic, J. Skirelis. Industry 4.0: Methodologies, Tools and Applications. 2019 Open Conference of Electrical, Electronic and Information Sciences (eStream). Vilnius, Lithuania, April 2019.

[4] A. Zabasta, N. Kunicina, K. Kondratjevs, A. Patlins, L. Ribickis, J. Delsing. MQTT Service Broker for Enabling the Interoperability of Smart City Systems. 2018 Energy and Sustainability for Small Developing Economies (ES2DE). Funchal, Portugal, July 2018.

[5] P. Dorogovs, A. Romanovs. Overview of government e-service security challenges. 2015 IEEE 3rd Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE). Riga, Latvia, Nov. 2015.

[6] A. Zabasta, K. Kondratjevs, J. Peksa, N. Kunicina. MQTT enabled service broker for implementation arrowhead core systems for automation of control of utility' systems. 2017 5th IEEE Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE). Riga, Latvia, Nov. 2017.

[7] Security researchers lift lid on snafu at Black Hat Europe. Intel Management Engine pwned by buffer overflow. URL:https://www.theregister.com/2017/12/06/intel_management_engine_pwned_by_buffer_overflow/

[8] G. Gibeling and J. Wawrzynek. A Universal Processor for RAMP. Technical Report, 2006.

[9] International Symposium on High-Performance Computer Architecture. Workshop on Architecture Research using FPGA Platforms, San Francisco, 2005.

[10] D. Chiou, H. Sunjeliwala, D. Sunwoo, J. Xu, and N. Patil. FPGA-based Fast, Cycle-Accurate, Full-System Simulators. In Workshop on Architecture Research using FPGA Platforms in the 12th International Symposium on High-Performance Computer Architecture, 2006.

[11] C. Kozyrakis and K. Olukotun. ATLAS: A Scalable Emulator for Transactional Parallel Systems. In Workshop on Architecture Research using FPGA Platforms in the 11th International Symposium on High-Performance Computer Architecture, 2005.

[12] S.-L. Lu, E. Nurvitadhi, J. Hong, and S. Larsen. Memory Subsystem Performance Evaluation with FPGA based Emulators. In Workshop on Architecture Research using FPGA Platforms in the 11th International Symposium on High-Performance Computer Architecture, 2005.

[13] J. Kasper, R. Krashinksy, C. Batten, and K. Asanovic. A Parameterizable FPGA Prototype of a Vector-Thread Processor. In Workshop on Architecture Research using FPGA Platforms in the 11th International Symposium on High-Performance Computer Architecture, 2005.

[14] P. Jones, S. Padmanabhan, D. Rymarz, J. Maschmeyer, D. V. Schuehler, J. W. Lockwood, and R. K. Cytron. Liquid Architecture. In International Parallel and Distributed Processing Symposium: Workshop on Next Generation Software, 2004.

[15] P. S. M. et al. Simics: A Full System Simulation Platform. IEEE Computer, 35(2):50–58, 2002.

[16] M. Rosenblum, S. A. Herrod, E. Witchel, and A. Gupta. Complete Computer System Simulation: The SimOS Approach. IEEE parallel and distributed technology: systems and applications, 3(4):34–43, Winter 1995.

[17] P. Yiannacouras, J. G. Steffan, and J. Rose. Application-Specific Customization of Soft Processor Microarchitecture. In FPGA '06: Proceedings of the 2006 international symposium on Field-programmable gate arrays. ACM Press, 2006.

[18] D. Sheldon, R. Kumar, R. Lysecky, F. Vahid, and D. Tullsen. Application-Specific Customization of Parameterized FPGA Soft-Core Processors. In IEEE/ACM International Conference on Computer-Aided Design (ICCAD). ACM Press, 2006.

[19] P. Biswas, S. Banerjee, N. Dutt, P. Ienne, and L. Pozzi. Performance and Energy Benefits of Instruction Set Extensions in an FPGA Soft Core. In IEEE International Conference on VLSI Design (VLSID). IEEE, 2006.

[20] Hifn. 4450 HIPP III Storage Security Processor, 2006.

[21] Intel. The Pentium Datasheet, 1997.

[22] I. Kuon and J. Rose. Measuring the Gap BetweenFPGAs and ASICs. In FPGA'06: Proceedings of the 2006 international symposium on Field-programmablegate arrays. ACM Press, 2006.

[23] Sun Microsystems 2006. OpenSPARC. Sun Microsystems

[24] Bochs 2019. XT, AT, and PS/2 I/O port addresses.

[25] Intel 1997. MultiProcessor Specification.