FPGA IMPLEMENTATION OF A CUSTOMIZED PROCESSOR BASED ON RISC-V ARCHITECTURE – CONCEPT AND THEORY OF OPERATION

[1]IONEL ZAGAN, [2]CRISTIAN ANDY TANASE, [3]VASILE GHEORGHITA GAITAN

[1,2,3]Faculty of Electrical Engineering and Computer Science, Stefan cel Mare University, Suceava, Romania;Integrated Center for Research, Development and Innovation in Advanced Materials, Nanotechnologies, and Distributed Systems for Fabrication and Control (MANSiD), Stefan cel Mare University, Suceava, Romania
E-mail: [1]zagan@eed.usv.ro, [2]andy@eed.usv.ro, [3]vgaitan@usm.ro

Abstract - Various real-time time systems, used in fields like automotive, robotics, or industry, are critical in the true sense of the word. For these systems, the time required to switch task contexts and the real-time operating system jitter introduced in treating aperiodic external events are very important parameters. The present paper describes the implementation of the nMPRA concept (Multi Pipeline Register Architecture - n degree of multiplication) based on the RISC-V architecture. The main characteristic of the nMPRA processor is ensuring the deterministic and performance control of a process. The implementation of the nMPRA architecture provides superior performance in terms of response time to external events and the time needed to switch task contexts; the architecture is suitable for real-time small-scale applications due to the resource consumption required to multiply the multiplexed storage elements. The nMPRA processor is a deterministic hardware implementation, due to the integrated preemptive scheduler and ZScale – RISC-V architecture.

Keywords - Pipeline Processor, Real-time Operating System, Real-time System, Scheduler.

I. INTRODUCTION

With the unprecedented development of microcontroller circuits, also called embedded processors, digital computing has penetrated into all fields, and it can be said that there is no field where at least one microcontroller is not present [1]. Under the pressure of costs, times to market, and the competitive environment, most of the embedded applications running on microcontrollers or FPGAs (Field-Programmable Gate Array) tend to be real-time, as a consequence of interacting with the real world.

A digital system is primarily a multitude of combinational and sequential circuits connected together. A FPGA circuit consists of three main elements: Look-Up Tables (LUTs), Flip-Flops (FFs) and routing channels. These constitutive elements, characteristic to the technology of programmable logic, are interconnected in order to form a flexible and state-of-the-art device. All resources used by a FPGA in order for the implementation and development platform to be a flexible, decrease the performances of the entire circuit [2], [3]. Nevertheless, it will never be able to generate a clock signal at speeds equal to those of a dedicated circuit. In a comparison, a ASIC (Application-Specific Integrated Circuit) can reach speeds higher than 4GHz, while a FPGA runs in very good conditions at only 450MHz.

Although FPGA uses more power in comparison to the ASIC circuits, it has a major advantage because it is suitable for small and medium size low-cost implementations with unlimited reconfiguration posibility, at least in theory. Verilog is a hardware description language initially developed by Design Automation in 1984; it is currently known as a industrial standard due to its simplicity and structure similar to the C programming language, and very fast design times. The input/output structures are more or less similar, although it must be taken into consideration the fact that Verilog is a hardware design tool, not a software design tool.

In this project, the RISC-V instruction set and the three-stage pipeline Vscale implementation, with the Verilog Vscale version proposed as ”open source” at Berkeley University, will be used in order to implement the nMPRA and the nHSE scheduler (Hardware Scheduler Engine for n tasks). Moreover, the FPGA Virtex-7 VC707 development kit will be used for synthesis.

The aim of this experimental project is to implement, test and validate the entire nHSE technology as part of the nMPRA microcontroller [4], [5], using the RISC-V instruction set, as well as the implementation of the Z-Scale 3-stage pipeline(Fig. 1) with the verilog VScale variant. Implementation will be done for various configurations of n - the number of tasks, i - the number of interrupts, m - the number of mutexes, s - the number of synchronization and communication events between tasks, and different dynamic schedulers.

This paper is structured as follows: the first section contains a brief introduction, and section II describes similar papers in the field of real-time embedded systems. Section III presents the concept and theory of operation of nMPRA processor based on RISC-V architecture, and section IV describes the practical results obtained during validation of the theoretical elements presented in previous section. Section V concludes the paper presenting final conclusions.

II. RELATED WORK

This section analyzes and compares several processor implementations with functions implemented in hardware. Theoretical and practical aspects related to the schedulers of the analyzed architectures are highlighted, as well as the features of the real time characteristic. The main features of the most representative implementations refer to the implementation of the scheduler, the pipeline, and the type of implementation.

The hthread project, proposed by Andrews et al in [6], is a hardware/software implementation of a multi-thread architecture that is an embedded operating system. The core is preemptive and has 256 software execution threads and 256 active hardware execution threads. To provide the services required for the functioning of the real-time system (RTS), the hthread operation system consists of four different hardware cores.

If talking about processors in the mobile application area, the i.MX 6SoloX processor proposed by Freescale is to be taken into discussion. This implementation is a reliable solution that increases the security of Internet of Things (IoT) applications [7]. Thus, i.MX 6SoloX is the industry's first application processor that integrates a ARM Cortex-A9 core and an ARM Cortex-M4 core into a single chip. This processor has been designed to enable exceptional performance and energy efficiency to real time devices.The processor offers the ability to run an operation system with a user interface on a Cortex-A9 kernel and at the same time benefits from the real-time determinism offered by the Cortex-M4 core. This is fundamental to a wide range of industrial, automotive or medical applications, because they require a modern user interface but, above all, they must be reliable, secure and deterministic, when communicating with other devices in the network.

The SPEAR concept proposed by Delvai and others in [8], based on a single execution path and a three-stages pipeline, simplifies the problems created by WCET (Worst-Case Execution Time), at the same time guaranteeing the predictability of the real-time system.In some cases, when the application tasks have slow execution times and severe time limitations, the overhead introduced by the execution of RTOS core functions can not be neglected, and can generate important interferences in the execution of tasks. In these situations, the predictability of the system can only be guaranteed if the effects of the overhead of the operating system are taken into account in the feasibility analysis for establishing the scheduling scheme.

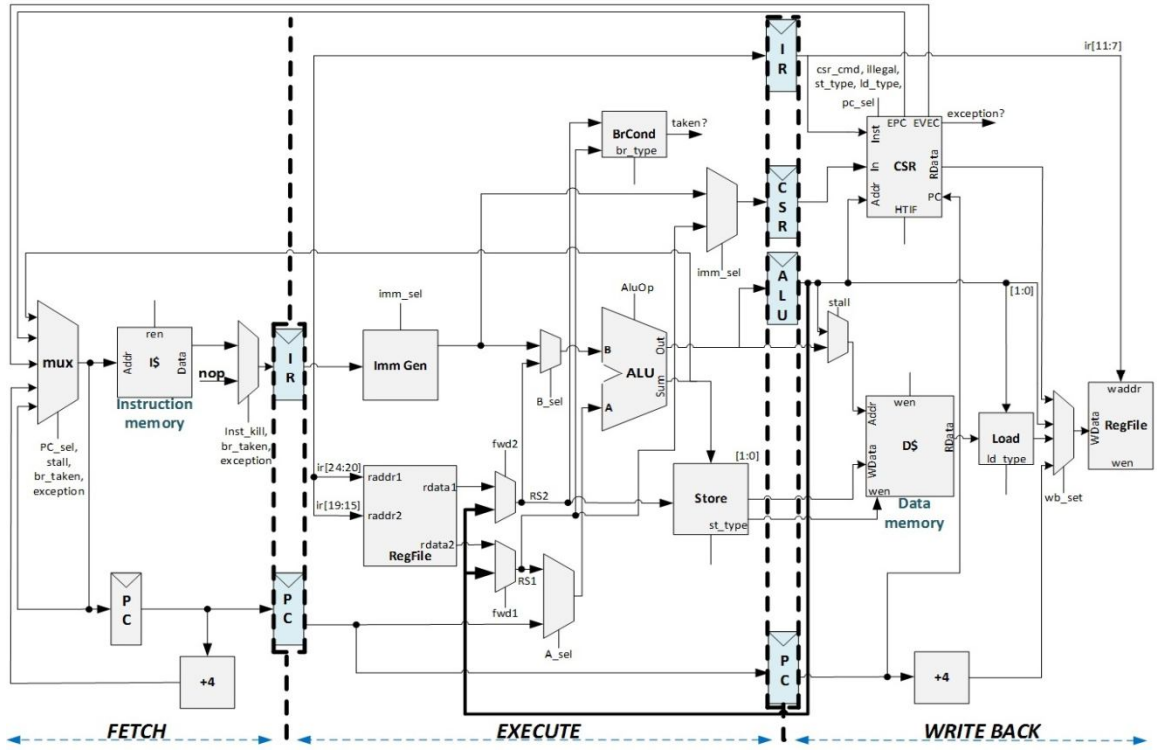


Fig.1. Z-Scale – RISC-V Architecture used to develop the dedicated processor with an integrated hardware scheduler; PC-program counter, IR-Instruction Register, RegFile-Register File, ALU-Arithmetic and Logic Unit, CSR-Control and Status Register.

The time required to switch contexts is the most significant factor in any RTOS [9]. This is an intrinsic boundary of the kernel, which does not depend on the scheduling algorithm or task set structure. For real-time systems, another overhead factor is the processor's time to execute the routine of treating interrupts.

If Q is the tick of the system and σ is the WCET corresponding to the periodic task [10], the

introduced overhead can be calculated as the Ut utilization factor obtained through the relation (1).

$$Ut=\sigma/Q \quad (1)$$

The effects on the scheduling a set of periodic tasks due to interruptions can be taken into consideration by adding the factor Ut to the total usage factor corresponding to the task set.

At the hardware micro-architecture level, there is a continuous research effort on PRET architectures. Firstly, we need to point out that Heckmann and others in [11] and Berg and others in [12] have identified the architectural properties that complicate the WCET analysis and propose certain facilities that make it easier to calculate this coefficient. One solution to guarantee the performance and predictability of the system can be to move the real-time operating system primitives and mechanisms, including the scheduler, into hardware, thus, on the one hand, the over-control of the basic functions implemented by the software is eliminated and, on the other, the jitter effect is significantly reduced.

III. CONCEPT AND THEORY OF OPERATION OF nMPRA PROCESSOR BASED ON RISC-V ARCHITECTURE

The main objective of this project and implicitly of this paper is to describe, implement, test and validate the nHSE technology as part of the nMPRA microcontroller, using the RISC-V instruction set (Fig. 1) in Verilog HDL. In order to accomplish this objective, it was necessary to structure nHSE modules so as to ensure easy adaptation to other implementations (MIPS coprocessor, ARM [13] or non-standard extension of the instruction set to RISC-V, etc.).

The issues covered in this paper relate to: the architecture used for the nMPRA implementation, the integration of nHSE scheduler registers into the nMPRA concept, the resource replication, the assembly line (the number of stages per assembly line), the synchronization and communication mechanisms [14]. Also, the integration into the nHSE of a preemptive scheduling algorithm based on priorities has been taken into consideration.

The project will be completed and synthesized on FPGA, nMPRA + nHSE for n = 2, 4, 8, 16, 32, a variable number of interrupts(i = 2 - 32) and mutexes(m = 2-32), and a variable number of signals(s = 2 - 32). Throughout this paper, the word signal (s) is used to name all inter-task synchronization and communication events [15] validated and treated by nHSE. The nHSE + nMPRA assembly must be implemented to enable the easy synthesis of different configurations for values given to parameters n, i, m and s.

The decision to multiply resources will be taken by analyzing the correctness of the VScale implementation, with three pipeline stages, and the RISC-V instruction set [16], [17]. The basic idea is that any storage element (except data and instruction memories) in the datapath must be multiplied by n and its output signal must be multiplexed n to 1.

The monitoring and control registers underpinning the nHSE scheduler and the status of tasks are illustrated in Fig. 2 and extensively described in the internal document on which the nHSE scheduler is based. To guarantee the real-time features, the hardware scheduler uses a unified space for tasks and interrupts, and the interrupts inherit the priority of the task to which it is attached.

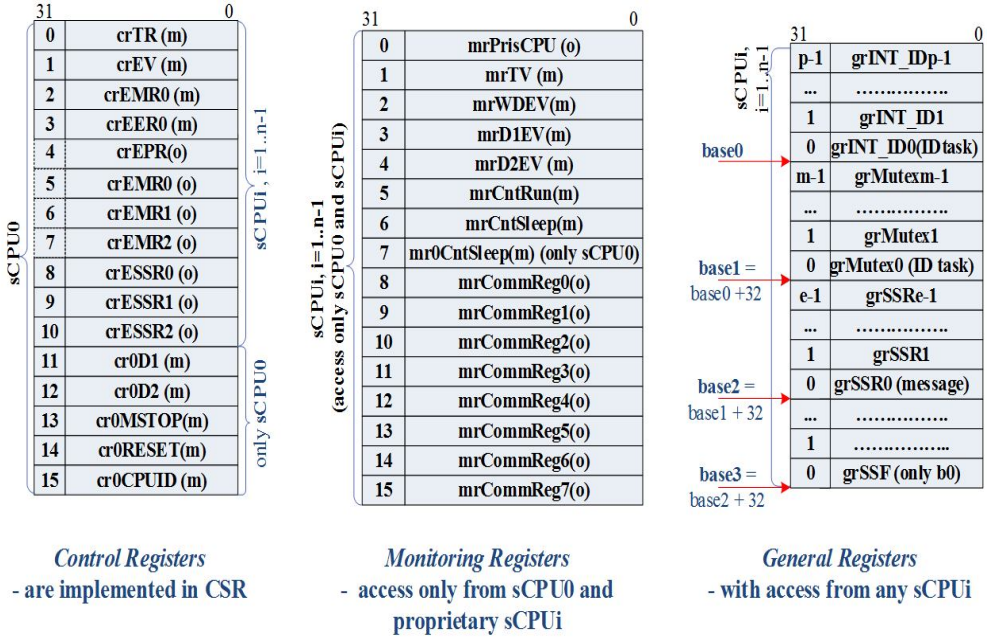


Fig.2. The dedicated nHSE scheduler registers implemented in CSR; cr_ - control register, mr_ - monitoring register; gr_ - global register.

As a result, the structure is composed of multiplex resources multiplied by n, together with the functional blocks between the pipeline registers form a typical RISC-V architecture that will be called semiprocessor (sCPUi).

An important decision for nMPRA is the how to implement timers and counters at the level of each sCPUi (for example, as supervisor registers, etc.), and add a debug port (access to important internal information). Based on these specifications, the following codes were written: the Verilog HDL code for VScale multiplications and extensions, and the Veriolg code for nHSE.

The CSR address convention uses the CSR address bits to encode the default access privileges. This simplifies error checking in hardware and provides a larger CSR space, but constrains the mapping of CSRs in the address space. The implementations could enable a more privileged level to gain access to other access areas of the CSR, at a less privileged level, in order to enable the interception of these accesses. This change should be transparent for the less privileged software.

The attempts to access a non-existing CSR raise an exception to illegal instructions. The attempts to access a CSR without an adequate level of privileges or to write a read-only register also generate exceptions to illegal instructions [18]. A read/write register may also contain some read-only bits, in which case the operation of writing to read-only bits is ignored.

The design is meant to enable the testing of the proposed functionalities. As shown in Fig. 1, Z-Scale is the architecture chosen for nMPRA whereas for the implementation the Vscale architecture has been chosen.

This architecture shown in Fig. 1 is designed with 3 pipeline stages, namely:

1. The stage for extracting instructions (FETCH) and IF/IDEX(Instruction Fetch/Instruction Decode - Execute) pipeline registers;
2. The stage for decoding instructions and executing IDEX/MEMWB (Instruction Decode - Execute/Memory - Write Back)pipeline registers;
3. The stage for memory access, CSR and register file.

IV. PRACICAL RESULTS BASED ON nMPRA RISC-V CONCEPT

The nMPRA architecture was implemented by multiplying all existing registers in the RISC-V (Z-scale) architecture shown in Fig. 3. All the memory elements were multiplied 32 times, so n = 32.

Registers in the nHSE block were mapped to the CSR area, between the following addresses: 0x200-0x2BF, 0xA00-0xAFF, 0xE00-0xEFF.The programming of internal nHSE registers is done by using the access instructions of these reserved areas.

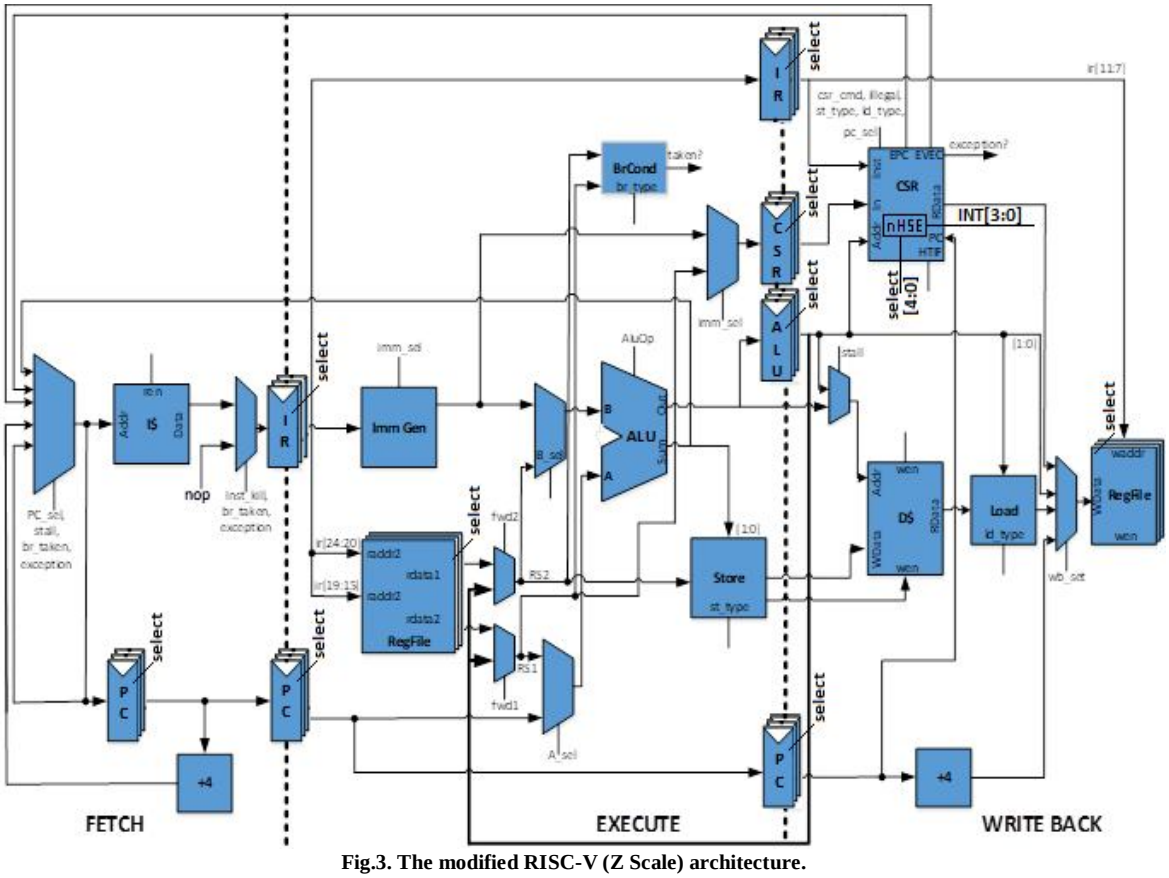


Fig.3. The modified RISC-V (Z Scale) architecture.

An example of a program code that program the nHSE internal registers is listed in the listing below:

```
lui x16, F0000     //x16 = F0000
ori x16, x16, FF  //x16 = F00000FF
csrrw 202, x16    //cr0MSTOP = F00000FF
//is activated sCPU 0, 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31
. . . . .
lui x16, 0//x16 = 0
ori x16, x16, 7     //x16 = 7
csrrw AB0, x16   //grINT_ID0 = 00000007
//INT0 is assigned to sCPU7
. . . . . .
lui x16, 400          //x16 = 0
ori x16, x16, F     //x16 = 0000000F
csrrw 281, x16     //mrTEV2 = 0000000F
//the timer recharge value for sCPU2
. . . . . .
lui x16, 0  //x16 = 0
ori x16, x16, 1      //x16 = 1
csrrw 240, x16     //crTR0 = 00000001
csrrw 209, x16      //cr0PageReg = 1 - change to page 1
//enable timer interrupt for sCPU0 and enable registers page for sCPU1 (the register dedicated only sCPU1 is accesible)
. . . . . .
```

Next we will show the operation of the RISC-V - nMPRA - nHSE processor when running a program that activate the timer events for sCPU0, sCPU1 and sCPU2 at different time intervals. The INT0, INT1, INT2, INT3 interrupts are assigned to sCPU7, sCPU5, sCPU7, sCPU4 in this order. Prioritization of interrupts is given by the sCPU number to which it is assigned.

Interrupts are disabled by the software written in the sCPU interrupt routine (sCPU0 = maximum priority, sCPU31 = minimum priority).Fig. 4 shows that at the occurrence of two simultaneously interrupts, the one with the highest priority (sCPU5) is processed, uiSelectCPU[4:0] = 5.The delay of interrupt processing is maximum two clock cycles.

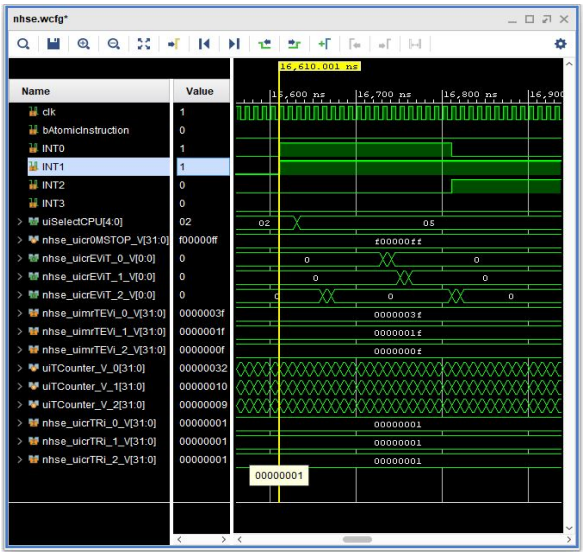


Fig.4. Simultaneous interrupts activations.

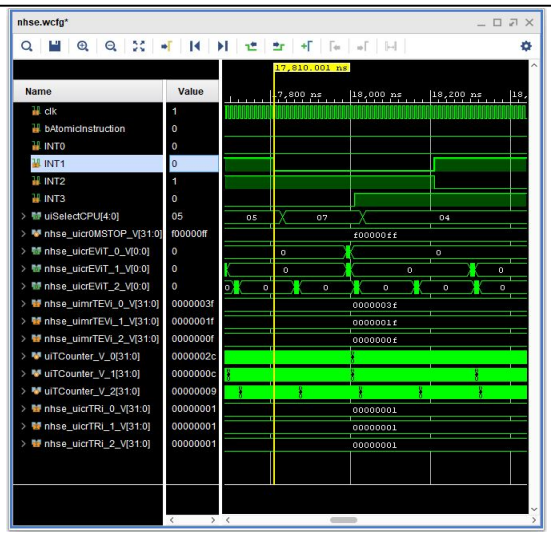


Fig.5. Interrupts nesting.

Fig. 5 shows that after the higher priority interrupt is deactivated, the less priority interrupt is processed (uiSelectCPU[4:0] = 7). When INT3 (most priority) is activated, sCPU4 (uiSelectCPU[4:0] = 4) is selected. It is also observed that as long as the interrupts are active, the events from the timers 0, 1 and 2 are not processed, the events of the interrupts are the most priority.

Fig. 6 shows how to handle timer events when there are no interrupts. Timer events occur depending on the reload value: mrTEV0 = 0x3f, mrTEV1 = 0x1f, mrTEV2 = 0xf. Timer counters (uiTCounter) decreases until they reach 0 value. Then, they generate a timer event (crEVi[0] = 1), triggering the sCPU for which the event was generated.When the timer event is disabled (crTR1[0] = 0), it is noticeable that the uiSelectCPU[4:0] signal is no longer 1 when the timer event occurs at sCPU1.

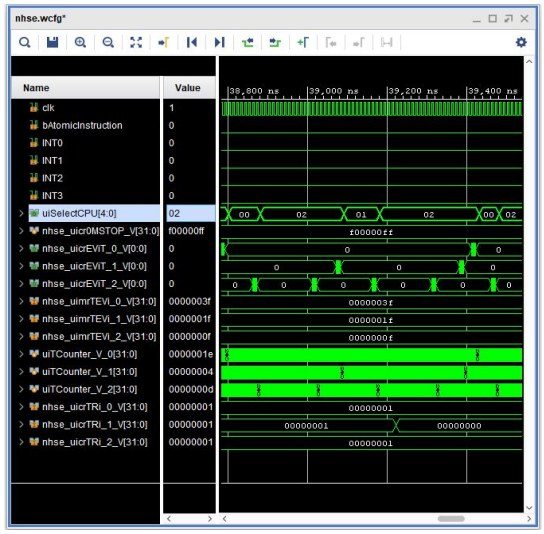


Fig.6. Timer events handling.

Following the implementation in the FPGA, some studies could be performed on the architecture complexity, resource requirements, the impact on the work frequency and power consumption, as well as

the execution of some test programs meant to highlight the performances of this new technology (the degree of technological maturity - TRL4).

CONCLUSIONS

This study argues that the nMPRA architecture is scalable and flexible and that it can be successfully used in small-scale RTS in a 4, 8 or 16 sCPUis implementation.
Future research directions will continue the studies related to the hardware implementation of real-time operating systems, taking into consideration the nMPRA architecture and its nHSE defining component. The performance of the nMPRA architecture can be enhanced by designing a debug and trace module to enable a better control of the system and a high diagnostic capability, providing system designers with flexible responses to the imposed requirements.

ACKNOWLEDGMENTS

This paper was supported by the project "Demonstrator experimental de laborator bazat pe nHSE - sistem de operare de timp real integrat în hardware - implementat pe o arhitectură ZScale - RISC V", acronym: nHSE-RiscV, Contract no. 219PED / 2017, PN-III-P2-2.1-PED-2016-1460, PNCDI III, using the infrastructure from the project "Integrated Center for research, development and innovation in Advanced Materials, Nanotechnologies, and Distributed Systems for fabrication and control", Contract No. 671/09.04.2015, Sectoral Operational Program for Increase of the Economic Competitiveness co-funded from the European Regional Development Fund.

REFERENCES

[1] W. Stallings, "Computer Organization and Architecture," 10th Edition, 2015, ISBN:978-0134101613.
[2] M. Shahbazi, P. Poure, S. Saadate, and M. R. Zolghadri, "FPGA-Based Reconfigurable Control for Fault-Tolerant Back-to-Back Converter Without Redundancy," IEEE Trans. Ind. Electron., vol. 60, no. 8, pp. 3360–3371, Aug. 2013. doi:10.1109/TIE.2012.2200214.
[3] M. Shahbazi, P. Poure, S. Saadate, and M. R. Zolghadri, "Fault-Tolerant Five-Leg Converter Topology With FPGA-Based Reconfigurable Control," IEEE Trans. Ind. Electron., vol. 60, no. 6, pp. 2284–2294, Jun. 2013. doi:10.1109/TIE.2012.2191754.
[4] V. G. Gaitan, N. C. Gaitan, and I. Ungurean, "CPU Architecture Based on a Hardware Scheduler and Independent Pipeline Registers," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 23, no. 9, pp. 1661-1674, Sept. 2015.
[5] E. Dodiu and V. G. Gaitan, "Custom designed CPU architecture based on a hardware scheduler and independent pipeline registers – concept and theory of operation", 2012 IEEE EIT International Conference on Electro-Information Technology, Indianapolis, IN, USA, 6-8 May 2012, ISBN: 978-1-4673-0818-2, ISSN: 2154-0373.
[6] D. Andrews et al., "hthreads: A hardware/software co-designed multithreaded RTOS kernel", in Proc. 10th IEEE Conf. Emerg. Technol. Factory Autom., pp. 331-338, Catania, Italy, Sep. 2005.
[7] Electronica-Azi, nr. 210, Decembrie 2016, Anul XV, http://electronica-azi.ro.
[8] M. Delvai, W. Huber, B. Rahbaran, and A. Steininger, "SPEAR – Design-Entscheidungen für den Scalable Processor for Embedded Applications in Real-time Environments", Tagungsband of Austrochip 2001, pages 25–32, Vienna, Austria, Oct. 2001.
[9] J. Echague, I. Ripoll, and A. Crespo, "Hard real-time preemptively scheduling with high context switch cost", In Proc. of the 7th Euromicro Workshop on Real-Time Systems, Odense, Denmark, 14-16 June, 1995, doi: 10.1109/EMWRTS.1995.514310.
[10] S. Altmeyer and G. Gebhard, "WCET analysis for preemptive scheduling", 8th International Workshop on Worst-Case Execution Time WCET Analysis (WCET'08), pp. 105-112, Prague, Czech Republic, July 2008, doi: 10.4230/OASIcs.WCET.2008.1664.
[11] R. Heckmann, M. Langenbach, S. Thesing, and R. Wilhelm, "The influence of processor architecture on the design and the results of WCET tools", Proceedings of the IEEE, vol. 91, no. 7, pp. 1038-1054, July 2003, doi: 10.1109/JPROC.2003.814618.
[12] C. Berg, J. Engblom, and R. Wilhelm, "Requirements for and design of a processor with predictable timing", seminar 03471 - Design of Systems with Predictable Behaviour, 2004, ISSN: 1862-4405.
[13] C. A. Tănase, "An approach of MPRA technique over ARM cache architecture," in 2016 International Conference on Development and Application Systems (DAS), Suceava, Romania, 2016, pp. 86-90. doi: 10.1109/DAAS.2016.7492553.
[14] I. Zagan, V. G. Gaitan, "Improving the Performances of the nMPRA Processor using a Custom Interrupt Management Scheduling Policy," Advances in Electrical and Computer Engineering, vol.16, no.4, pp.45-50, 2016, doi:10.4316/AECE.2016.04007.
[15] I. Zagan, V. G. Gaitan, "Implementation of nMPRA CPU Architecture based on Preemptive Hardware Scheduler Engine and Different Scheduling Algorithms," IET Computers & Digital Techniques, accepted manuscript, September 2017, doi:10.1049/iet-cdt.2017.0163
[16] A. Waterman, Y. Lee, R. Avizienis, H. Cook, D. Patterson and K. Asanovic, "The RISC-V instruction set," 2013 IEEE Hot Chips 25 Symposium (HCS), Stanford University, CA, USA, 2013, pp. 1-1. doi: 10.1109/HOTCHIPS.2013.7478332.
[17] Andrew Waterman, Yunsup Lee, David Patterson, Krste Asanovic, "The RISC-V Instruction Set Manual Volume I: User-Level ISA, Version 2.1,"EECS Department, University of California, Berkeley,May 2016. This document is also available as Technical Report UCB/EECS-2016-118.
[18] Andrew Waterman, Yunsup Lee, David Patterson, Krste Asanovic, "The RISC-V Instruction Set Manual, Volume II: Privileged Architecture Version 1.7,"EECS Department, University of California, Berkeley,This document is also available as Technical Report UCB/EECS-2015-49.

★★★