Support for Teaching the Subject Digital Computers

Ondrej Karpiš, Tomáš Bača, Ján Šumský, Michal Kubaščík

***Abstract*—**In this paper, a processor emulator is presented which is used in the teaching of the Digital Computers course. The emulator allows you to create a custom circuit that is controlled by an 8-bit processor via the bus. The processor is emulated by software on the computer and is connected to the circuitry through a special converter. Simulation software has also been created to facilitate the students in creating the circuitry, which allows them to create the circuitry and program without the need for any hardware.

***Keywords*—**processor emulator, simulation tools, teaching support

I. INTRODUCTION

Raising a good programmer is not and never has been an easy task. The rapid development of software technologies in recent years has certainly made programmers' jobs easier. On the other hand, it brings increased demands for orientation in the number of different technologies and supporting tools. These advances have largely been made possible by the development of integrated circuit manufacturing technologies. The speed of a computer's computational core and the size of its operating memory represent a constraint for an increasingly smaller set of problems. In most practical applications, programmers are not constrained by hardware at all. Thus, the world of hardware and software is gradually becoming more distant. Nevertheless, knowledge of the basic principles of digital computer operation belongs to the general education of programmers [1]. That is why at the Faculty of Management science and Informatics of the University of Žilina the subject of Digital Computers has been taught for more than 20 years. Of course, the teaching of this subject is also evolving and adapting to the requirements of the time. The main innovation in the teaching of this subject was the creation of a processor emulator, which is presented in this article.

II. PROCESSOR EMULATOR

A digital computer consists of several main parts: processor, memory, input and output devices [3]. One of the most important tasks in a computer is to provide communication between its parts. It is to deepen the understanding of how the processor communicates with the other parts of the computer using the bus that a simple 8-bit processor emulator was developed. This is not a real processor, it has been designed solely for teaching purposes.

The processor emulator is software-hardware based. The operation of the processor is emulated by the software on the PC. The connection to the hardware bus is made using a special converter which is connected to the PC via USB. A development board is connected to the converter, which contains four 7-segment displays, 16 buttons, one LED and a contact field on which the students make their own circuits. Multiple development boards can be interconnected to implement more complex circuits. The program for the processor must be written in a dedicated assembler. The created program is executed (interpreted) sequentially by a software emulator. The speed of program execution depends on the speed of the computer. In the case of communication of the emulated processor with a converter, the speed is limited by the speed

Ondrej Karpiš, University of Žilina, Slovak Republic (e-mail: ondrej.karpis@fri.uniza.sk)
Tomáš Bača, University of Žilina, Slovak Republic (e-mail: tomas.baca@fri.uniza.sk)
Ján Šumský, University of Žilina, Slovak Republic (e-mail: jan.sumsky@fri.uniza.sk)
Michal Kubaščík, University of Žilina, Slovak Republic (e-mail: michal.kubascik@fri.uniza.sk)

of the USB interface.

A. Processor structure

The proposed processor is of the RISC (Reduced Instruction Set Computers) type and uses the Harvard architecture (i.e. it has separate memory for data and program)[4]. Figure 1 shows its block structure. The processor contains four 8-bit general-purpose registers A, B, C, D. According to the result of mathematical operations, the flag bits of the status register F are set: the Z (Zero) flag is set when the result of an operation is zero and the CY (Carry) flag is set when a register overflow occurs. In addition, the F register also contains the IE flag, which can be used to enable or disable interrupts.

The processor also contains three 16-bit registers:

- PC (Program Counter) - pointer to the following instruction,
- SP (Stack Pointer) - pointer to the top of the stack,
- MP (Memory Pointer) - pointer for indirect addressing of external memory.

Either a constant or a pair of A-B registers can be written to the 16-bit SP and MP registers. The C-D register pair must be used to read from these registers.

The stack size is 64 kB (65536 bytes). The program size is limited to a maximum of 65536 instructions. The processor also contains an internal RAM memory with a size of 256 B. This memory is indirectly addressed by general-purpose registers. It is possible to define a maximum of 256 constants within a program that can be read indirectly using the general-purpose registers.

The width of processor data bus is 8 bits, and the width of the address bus is 16 bits. This corresponds to an address memory space and I/O space with a range of 0-65535. The control bus contains 9 signals, including 5 output signals (Memory Write, Memory Read, Input/Output Write, Input/Output Read, Interrupt Acknowledge) and 4 input signals (Interrupt, Ready, Bus Request, Bus Acknowledge).

B. Interrupts

The processor allows the use of up to 16 interrupts. When an interrupt is requested via the Interrupt signal, if interrupts are enabled, the processor acknowledges receipt of the request with an Interrupt Acknowledge signal. The processor then reads the interrupt handler number (0-15) from the data bus and calls the appropriate interrupt handler. The names of the handlers are int00 - int15. When an interrupt is invoked, the next interrupt is automatically disabled. If there is an instruction in the handler to enable interrupts (usually this instruction is at the end of the handler, but it can be elsewhere), it is also possible to nest interrupts. The number of nested interrupts is limited only by the size of the stack.

An interrupt can be triggered either by the level of the Interrupt signal (when it is at logic level "1") or when the signal changes from "0" to "1". The specific method of triggering an interrupt can be set in the emulation software.

C. Instruction set

The CPU emulator instruction set contains 69 instructions. These instructions can be divided into five basic groups:

- arithmetic and logical instructions: separate instructions are defined if both arguments are in registers (e.g. ADD, SUB, AND, CMP) and if the arguments are a register and a constant (ADI, SBI, ANI, CMI),
- instructions for shifts and rotations (SHL, SCR, RTR ...),
- data transfer instructions (MOV, INN, OUT, PUS, STR, LDR ...),
- branching instructions: in addition to conditional jumps (JZR, JNC, JE, JL ...), conditional

subroutine calls are also available (CNZ, CCY, CNE, CG ...),

- special instructions: enable/disable interrupts; define constant in the program; communicate with the user via the console - character input and output.

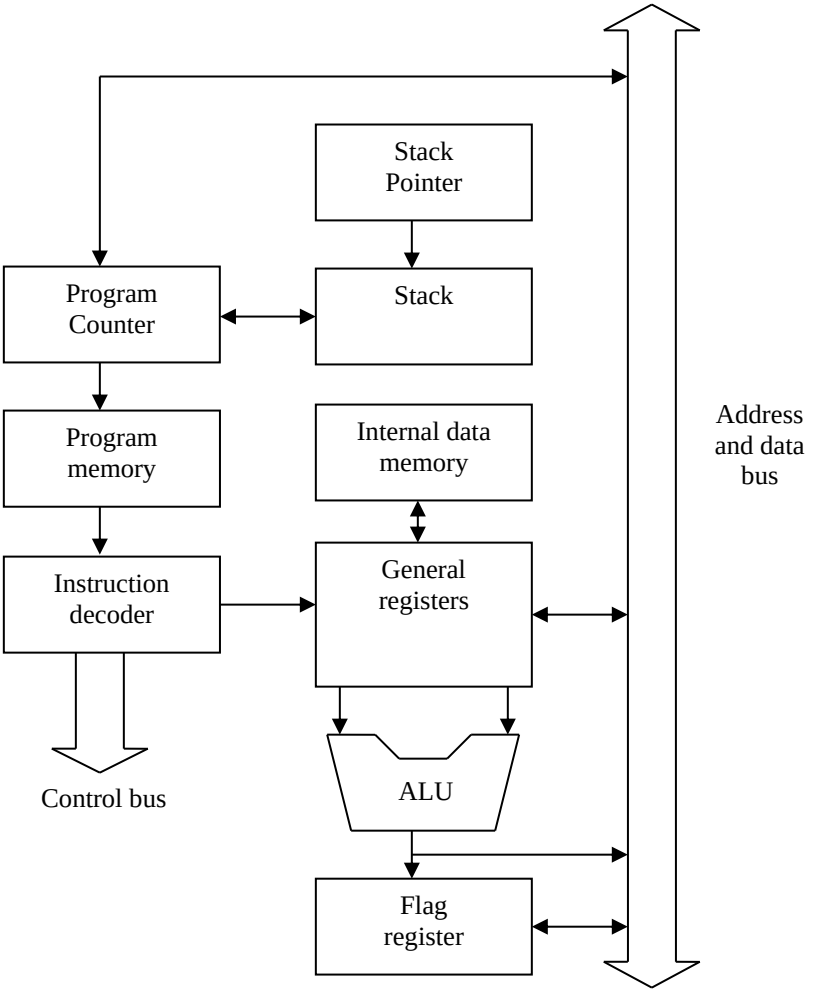

Figure 1 Structure of emulated processor

III. SIMULATION SOFTWARE

In the deployment of the processor emulator in teaching, 20 pieces of converters and 100 pieces of development boards were produced. Due to the high number of students, students had to work in groups. The implementation of the circuits and testing of the developed program was only possible in a dedicated laboratory equipped with converters. The circuits created on the contact field were very sensitive to improper handling. Accidental pulling or breaking of a single wire usually caused a complicated and lengthy search for the fault. This, coupled with limited access to the laboratory, caused problems in completing semester works. For these reasons, a development board simulator was created. Using the simulator, students were able to debug their programs and circuits before implementing them in hardware. They could also work on their semester works in their free time - at home or on campus.

The first version of the software was written in C. This made the simulation of the circuit quite fast. However, the program had limited possibilities in terms of visualization and ease of use. Therefore, a new version of the software was created, this time in Java. This made it possible to use the program not only in Windows operating system but also in Linux and iOS.

The new software also brought a more graphically appealing interface, more data display formats, graphical differentiation of individual parts of the program (instructions of different types, constants, labels, comments), the possibility of using more parts of the development system (contact fields, displays, keyboards, LEDs), the possibility of visualizing the current state of the circuits (logic levels of inputs and outputs, content of the registers or memory). The simulation software also indicates some errors that can be caused by improper connection (e.g. short circuit).

The simulation software uses two separate windows - one showing the processor parts (registers, stack, internal memory) together with the user program, and the second window showing the circuit simulator. Both windows are shown in Figure 2.

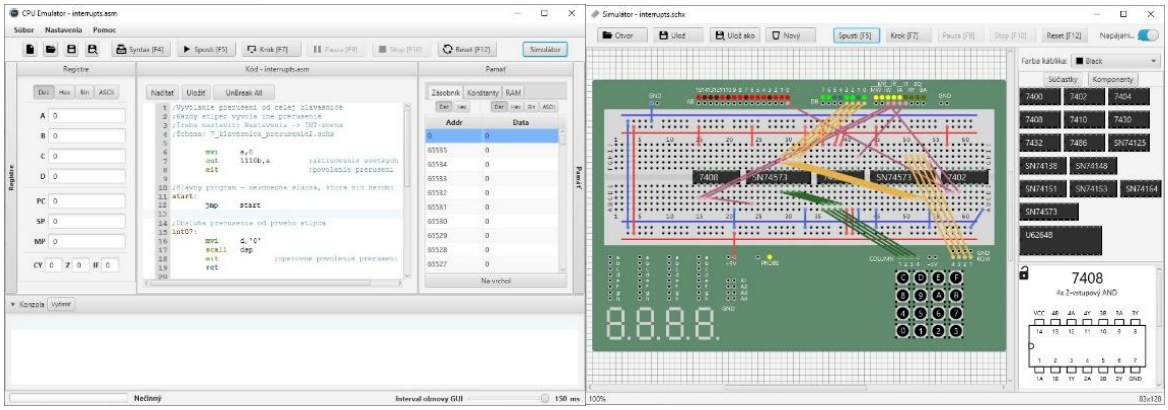

Figure 2 Processor emulator (on the left) and circuit simulator

Of course, the created simulator has some shortcomings. The main drawback is the imperfect simulation of the behavior of the logic circuits involved. In most cases the simulation is sufficiently faithful, but in special cases the simulator behavior is different from the behavior of the real circuitry. The second drawback is the limited number of types of integrated circuits that can be used in the circuitry. Also note that the speed of the simulation is dependent on the speed of the computer. The loss of physical contact with the hardware can also be considered as a disadvantage.

Despite the above disadvantages, the use of a simulator has undeniable positives. Mainly, it simplifies and facilitates the work of students in creating the circuitry and they can spend more time on creating the program or create more complex circuitry. The simulator also eliminates the need for regular repairing of malfunctioning or damaged hardware parts - converters, integrated circuits and development boards. For these reasons, the hardware processor emulator has gradually been replaced by the simulator.

IV. Conclusion

Based on our experience, we can conclude that the use of a processor emulator in the teaching of the Digital Computers course helps students to understand the principles of operation of digital computers. The acquired knowledge is much more durable due to the practical experience. Although most of the students who take this course are software developers, every year there are a few students who are intrigued by the possibility of creating their own circuitry and they produce very original semester works. The interest of these students is the reason why it makes sense to continue to develop similar learning support tools.

REFERENCES

[1] Hassan, M. K., & Rana, M. M. "An Overview of Embedded Systems and Applications in Education," *IEEE Transactions on Education*, vol. 58, no. 3, pp. 123-134, 2015.

[2] S. S. Anwar, S. A. Usmani, and M. R. K. Raju, "A review on embedded systems: Opportunities and challenges in future," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 3, pp. 1005-1017, 2008.

[3] W. Stallings, *Computer Organization and Architecture: Designing for Performance*, 10th ed., Pearson, 2016.

[4] M. Moradi, "A Survey on Modern Microprocessor Architectures," *IEEE Access*, vol. 9, pp. 53323-53340, 2021.