UNITECH 2019 GABROVO

INTERNATIONAL SCIENTIFIC CONFERENCE
15 – 16 November 2019, GABROVO

HACK COMPUTER: FPGA IMPLEMENTATION AND SYSTEM PROGRAMMING SUPPORT

Duško Kovačević, Dejan Vujičić, Slađana Đurašević, Marina Milošević, Uroš Pešović
University of Kragujevac, Faculty of Technical Sciences in Čačak, Serbia

Abstract
The Hack computer is custom-made computer system which is primarily built as a teaching architecture. Its design and functions are relatively simple, but still emphasize all the complexity of modern hardware and software approaches in computer design. In this paper, the FPGA realization of Hack computer is shown. We demonstrated its implementation on Altera DE2 FPGA development system, along with appropriate software support in terms of high-level language compiler, virtual machine translator, and assembler.

Keywords: Hack computer, FPGA, system programming, program translators

INTRODUCTION

Hack computer system is simple enough to be build by students in few hours using basic logic gates. Yet, it is complex enough, to demonstrate to students basic principles of computer hardware and system software [1]. Originally, this computer system is build using Nand2Tetris Software Suite, which enables virtual creation of computer hardware and software components and testing build components on simulator which runs on personal computer [2], [3].

This computer system has been implemented on FPGA platforms by various researchers. In this paper we demonstrated work flow of design and implementation of Hack computer system on Altera DE2 FPGA development board, along with appropriate software support.

The software support of Hack computer is comprised of rudimentary operating system support, high-level language compiler, virtual machine translator, and assembler. Altogether, they ideally represent the architecture of modern software systems.

HACK COMPUTER

Hack computer employs Harvard architecture with independent program and data memory. Program is executed from 32KWord ROM, where instructions are 16 bit long. CPU has 16-bit wide data bus and supports two types of instructions: A-instruction used for loading literals and C-instruction which could perform one of the eighteen basic logical and arithmetical operations. All instructions are one word long and PC is incremented on every new instruction, except instructions with jump conditions which load content of A-register into PC if jump condition is met.

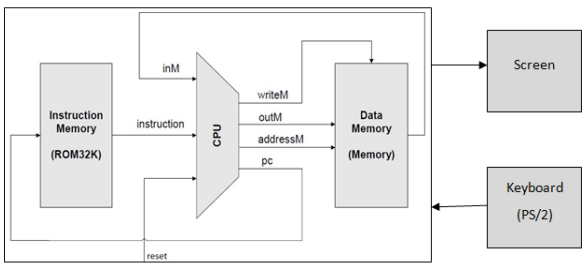

Fig. 1. *Structure of Hack computer*

CPU has two registers, A – register which is loaded by A-instructions, or by ALU output and D register which can be loaded from ALU output, Fig. 2. Content of D-register is used as the first operand and the second operand can originate either from A-register or from Data memory. ALU unit has simple structure and has 6 control inputs which could zero or negate input, select addition or logical AND operation, and negate ALU output. These control inputs have 64 possible combinations which my using De Morgan theorems provide 18 different operations.

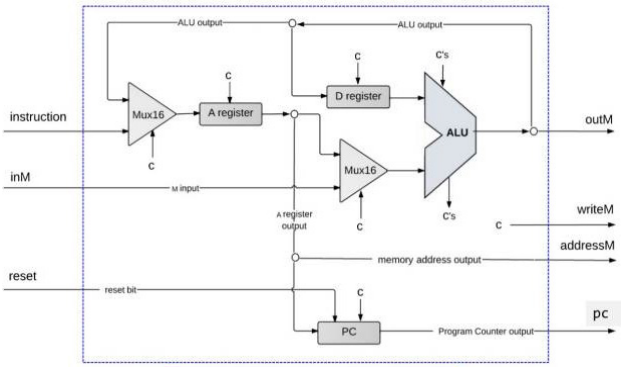

Fig. 2. *Structure of Hack CPU*

Data memory is divided into 16KW RAM memory for program variables, 8KW Screen memory and one memory location for keyboard. Screen memory is used to present monochromatic image on 512x256 screen resolution, Fig. 3 [4].

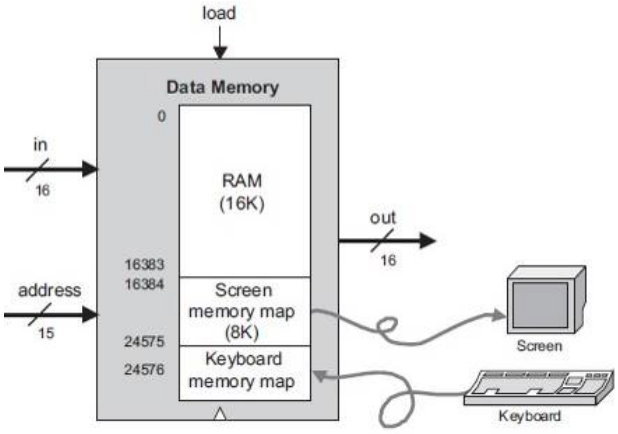

Fig. 3. *Structure of Data memory*

Hack computer was implemented on Altera DE2 development board, equipped with Cyclone II FPGA with 35k logic elements. Among many peripherals, this board also includes 512k SRAM, VGA DAC and PS2 interface. All computer components, except ROM are implemented by FPGA.

Cyclone II has 105 M4K memory blocks which were sufficient to implement Data Memory (Fig. 4), while Program memory was implemented on SRAM chip which was loaded from personal computer using Altera Control Panel application.

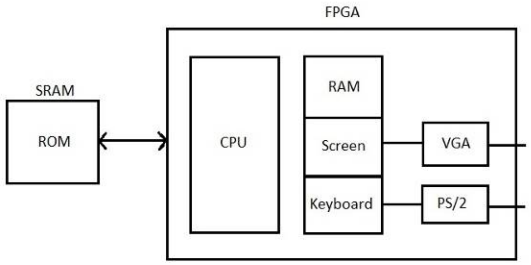

Fig. 4. *FPGA realization block diagram*

CPU is realized in VHDL language, according to the CPU structure shown on Fig2. CPU is composed from two registers, program counter and ALU unit and control unit which decodes instructions and controls all other parts of CPU. Data memory is composed from three different types of memory components which are generated using Memory Compiler. RAM memory which is 16k word is realized as single port memory which is only accessible by CPU on lower addresses. Screen RAM is realized as true dual-port memory which can be read/written from CPU and which is cyclically read by VGA controller and data are send to VGA DAC. Role of the VGA controller is to control position of the beam on screen and to generate strobes for horizontal and vertical synchronization. Based on the current pixel position, appropriate bit is red from particular address in Screen RAM. Based on the bit value, blue color pixel is generated by VGA DAC for value one, otherwise white one is generated.

Since Screen resolution was set to 640x480, pixels outside of 512x256 zone are colored in black. VGA controller cycles trough Screen RAM 60 times per second and it uses 50MHz clock to drive VGA clock to desired frequency using PLL.

Keyboard is implemented as simple dual-port memory, which enables keyboard to write the ASCII code for pressed key in one location, which can be subsequently read by CPU. These three memory components are controlled by address decoder, shown in Fig. 5, in order to activate just one which is accessed according to address ranges presented in Fig. 3.

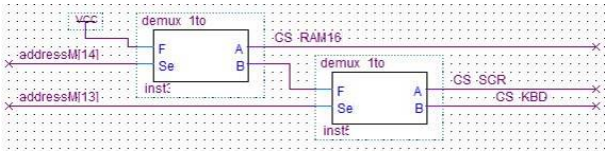

Fig. 5. *Address decoder for Data memory*

PS/2 keyboard is interfaced to FPGA using PS/2 controller which communicates with keyboard using I2C protocol. Since Hack computer uses different key encoding than PS/2 keyboard, appropriate key mapping is added to convert key presses. Structure of complete Hack computer system is presented on Fig. 6.

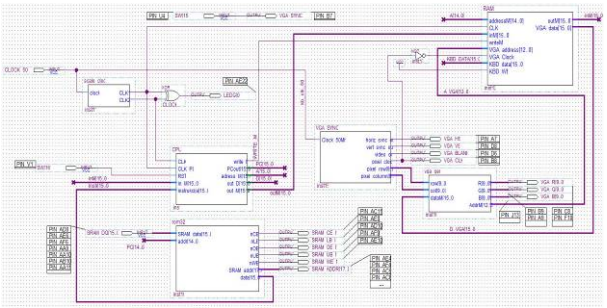

Fig. 6. *Structure of implemented Hack computer*

Hardware components of computer system are tested using simple program written in assembler which fills screen with line segments when any key is pressed (Fig. 6).

```
0   @20480
1   D=A
2   @16
3   M=D
4   @16
5   D=M
6   @24575
7   D=A-D
8   @0
9   D;JLT
10  @24576
11  D=M
12  @20
13  D;JNE
14  @16
15  D=M
16  A=D
17  M=-1
18  @24
19  0;JMP
20  @16
21  D=M
22  A=D
23  M=0
24  @16
25  M=M+1
26  @4
27  0;JMP
```

Fig. 6. *Assembly code for Hack computer testing*

SOFTWARE SUPPORT

Along with the hardware realization on FPGA platform, the Hack computer suite is comprised of assembler, virtual machine interpreter, and Jack compiler. Jack is object-based high-level language with support for classes, basic I/O capabilities, and possibility of library updates [1].

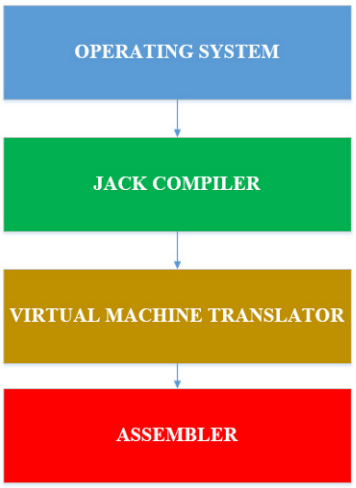

Fig. 7. *The software support of Hack computer*

The Hack software support is represented in Fig. 7. The top level part is operating system, which in fact represent the software libraries to support functioning of Jack programming language and its compiler.

The second layer in software architecture is Jack compiler, which translates code written in Jack programming language into virtual machine code. The main parts of Jack compiler are syntax analyzer and code generator. Within syntax analyzer, there are tokenizer and parser. Tokenizer is responsible for creating elementary tokens, and parser for comprising tokens into statements (Fig. 8).

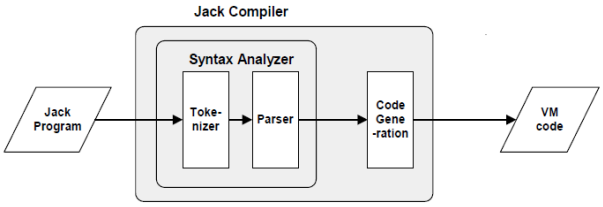

Fig. 8. *Structure of Jack compiler [4]*

The third layer in software architecture is virtual machine translator. Its function is to translate code generated by Jack compiler into assembler code. The virtual machine is based on several memory segments in which the

global variables, local variables, function parameters, etc. are stored. It is somewhat similar in functioning to Java virtual machine and is mainly stack-oriented [5].

The final layer in Hack software support is the assembler. Its function is to translate the assembler code generated by virtual machine interpreter into machine language. The assembler is two-layer, where the first layer is responsible for translating code without populating actual addresses from symbol table, and the second layer populates missing addresses that correspond to ones in the symbol table.

CONCLUSION

The Hack computer is simple, but yet powerful enough to be taught to students as an example of modern hardware and software computer architecture. The FPGA realization of Hack computer is presented in this paper. We successfully demonstrated that although relatively simple, this computer can be implemented in FPGA environment, together with its processor, memory system, and basic I/O support for screen and keyboard.

The software part of the Hack computer is presented as well. It is consisted of several layers, together with high-level language compiler, virtual machine translator, and assembler. In this configuration, it fairly resembles the software architecture of modern object-oriented programming languages and their compilers, primarily based on virtual machine implementation.

The architecture of Hack computer enables improvements in terms of different hardware design and support to various I/O modules. Of course, it would be accompanied by appropriate software support, which can be done as a future work in this direction.

ACKNOWLEDGMENT

The work presented in this paper was funded by grant TR32043 by the Ministry of Education, Science, and Technological Development of the Republic of Serbia.

REFERENCE

[1] Noam Nisan, Shimon Schocken, A Synthesis Course in Hardware Architecture, Compilers, and Software Engineering, SIGCSE 2009, Chattanooga, TN, USA, March 4-7, 2009

[2] Shimon Schocken, Taming Complexity in Large Scale System Projects, SIGCSE'12, February 29–March 3, 2012, Raleigh, North Carolina

[3] Nand2Tetris Official Site, https://www.nand2tetris.org, accessed on: October 2019

[4] Noam Nisan, Shimon Schocken, The Elements of Computing Systems: Building a Modern Computer from First Principles, MIT Press, 2005

[5] Shimon Schocken, Virtual machines: abstraction and implementation, SIGCSE Bull. Vol. 41, No. 3, July 2009, pp. 203-207.