DOI: 10.35598/mcfpga.2019.011

Application of Xilinx Series 7 on FPGA (XADC)

Valentyna Moroz
Department of Computer Radio Engineering and Technical Information Security Systems
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
valentyna.moroz@nure.ua

***Abstract*—FPGA systems development has long ceased to be limited to simply writing code in hardware description languages (HDL); and as the number of logical resources and the complexity of projects increase, approaches to designing systems on FPGAs have been repeatedly revised. One of the turns of development was the introduction of soft processors into projects — essentially ordinary microprocessors but assembled on FPGA resources. Unfortunately, despite the relative difficulty of developing software-processor systems, many trying to "raise" this topic face difficulties in mastering, because they do not know where to find the necessary information.**

Keywords—programming environment, VIVADO, FPGA, VHDL, XADC, boards, Basys 3

I. Introduction

The purpose of the article is to give a general idea of the stages of assembly of the processor system based on the Microblaze soft processor, using the Xilinx Vivado environment. Unfortunately, within the framework of a single article, it is difficult to describe all the diversity of the process of building soft-processor systems on an FPGA and to describe in detail all the "subtle" moments accompanying it, but it is necessary to start with something. In the article, we will look at connecting XADC, various memory controllers, external interfaces, analyze the operation of components connecting MicroBlaze with peripherals, and much more. But it will be a little later, but for starters.

Today, there is an acute problem of the development of technologies in different spheres of human life. And one of the most developed is the service sector of people. But the goal of the project is to develop a method for implementing the analog-to-digital signal conversion. This topic is relevant because a person needs to convert signals into one form or another. And to realize a very easy way was our goal.

Therefore, we propose using Basys 3 with the use of MicroBlaze. A tutorial for university students with a detailed explanation of each step in the work of VIVADO (programming environment) or in the creation of the simplest schemes is attached to this project.

Using this programming environment, it is possible to implement various schemes for measuring temperature, voltage, alarms (if you connect a speaker to the board) or a normal flashing LED.

II. Goals And Results

The development of any processor system built on FPGA resources consists of two fundamental parts: the assembly of the hardware platform HW - hardware, and the development of the SW - software executable program.

The HW part is developed in the Xilinx Vivado environment in the IP Integrator module (Vivado IPI) and is the creation of the actual instance (or several for a multiprocessor system) of the MicroBlaze core, connecting it with the necessary peripherals and address space allocation. Code development for MicroBlaze is performed in Xilinx SDK in assembly language or C / C ++.

The assembly process of the HW part in Vivado IPI is in many ways similar to that in the previous Xilinx environments — ISE and PlanAhead (where the XPS utility, Xilinx Platform Studio, was used for this), but has a number of differences from it. General principles have been preserved, the differences are found, for the most part, in the representation of the system. There are no differences from the software (SW-part) side: the Xilinx SDK is also used to work with the software part.

The development of a hardware platform begins with the launch of Vivado and the creation of a project.

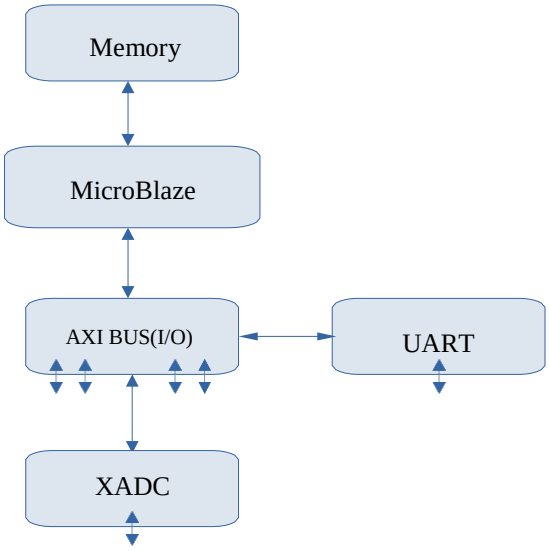

Fig. 1. Block Diagram of Final Scheme of The Project.

Full components in the scheme:

- *clk_wiz_1* – (Clocking wizard) - helps us to manage the clocker frequency and connect it to the other elements which requires;
- *xlconstant_0* – (Constant) applied a constant value (1) to make a trick with the block "rst_clk-wiz_100m";

- *rst_clk-wiz_10Om* – (Processing System Reset) using to reset the entire Microblaze system. This block is required to use Microblaze, but we are not using any "Reset" function;
- *MicroBlaze* – functional block of Microblaze processor;
- *Microblaze_0_axi_periph* – (AXI Interconnect) helps the system to configure the AXI hub, to process the input values;
- *microblaze_0_local memory* – (Local Memory) provides processor to save the data;
- *xadc_wiz_0* – (XADC Wizard) special functional block which provides us the ADC data;
- *axi-gpio_0* – (AXI GPIO) AXI bus module, which getting data from digital inputs (16 switches);
- *axi_uarllite_0* – (AXI uarllite) AXI bus module, provides a connection to PC using UART interface;
- *axi_gpio_1* – (AXI GPIO) AXI bus module, which putting data to digital outputs (16 LEDs);
- *reverse_v1_0* – (Reverse_0) an example of VHDL module to invers the output signal.

The final section of the code involved reading the ADC values and converting to voltages as well as reading the digital input. All of these inputs that were read in were then printed to the serial port in a form such that the raw ADC, voltage value, and digital value could all be read and continuously updated.This was all done in an infinite loop so that measurements were always being made and then reported.

As achievements using GPIO we understood and organized the abilities of AXI hub. How to control and use a digital I/O in C code (via Microblaze). We can read/write the state of digital inputs/outputs like 7-segment dispay, switches, digital I/O, keys, leds, etc. Using special IP-Blocks like UART-LITE helps us to configure the UART port. Also to turn on the XADC, we should use a special block XADC Wizard. It provides the input voltage stage and temperature of microprocessor (FPGA). In our reality, because of the differences between Zynq and Microblaze architectures, Vivado can't provide us same libraries for both of components, and that is the problem what we met. We could get the voltage value of the components, but it should be correctly plugged because we use positive and negative signal in XADC port.

NOTE: The coupled routing and the anti-alias filters might limit the data speeds when used for digital signals. The XADC core within the Artix-7 is a dual channel 12-bit analog-to-digital converter capable of operating at 1 MSPS. Either channel can be driven by any of the auxiliary analog input pairs connected to the JXADC header. The XADC core is controlled and accessed from a user design via the Dynamic Reconfiguration Port (DRP). The DRP also provides access to voltage monitors that are present on each of the FPGA's power rails, and a temperature sensor that is internal to the FPGA.

III. Conclusion

In conclusion, XADC and AXI GPIO modules were implemented on an Arty board along with the MicroBlaze soft processor. The steps for setting both modules in the Vivado software were described. After the modules were set up and the hardware exported for working with the SDK, the code was written to set up channels, initialize the hardware, take measurements and report values from both the XADC and the AXI GPIO modules.

The project examined the work of the microprocessor in the FPGA in the VHDL language in the Vivado programming environment, this expands the field of activity when development and make VHDL programming more flexible, for example, allowing you to work with floating point numbers. With the help of this project, it is possible to carry out various types of work, which in the future will be used for studying by students.

In consequence of this, a tutorial was made, where it is shown how and at what speed an elementary project can be created and then complicate it using different tools of the software environment. The main part of our project is the implementation of the ADC. The paper presents the main scheme of work and its applications in the applications. The results of measurements of the signal at the output of the Basys 3 board were obtained and presented in the SDK (Software development kit).

The signal is the differential voltage on some components of the circuit and the temperature of the processor, as well as the voltage of the analog signal. Results attached to the app. The main objective of the project is to provide students with high-quality material for learning. With the help of FPGA, you can easily integrate different areas of work, both software, and hardware of the project. From this, we can conclude that time is reduced to separate stages of development. Therefore, this project is successful and ready to use.

References

[1] "MicroBlaze Processor Reference Guide. UG984 (v2016.4) November 30, 2016", *Xilinx.com*, 2016. [Online]. Available: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2016_4/ug984-vivado-microblaze-ref.pdf. [Accessed: 20- Jun- 2019].

[2] "Using the MicroBlaze Processor to Accelerate Cost-Sensitive Embedded System Development. WP469 (v1.0.1) June 6, 2016", *Xilinx.com*, 2016. [Online]. Available: https://www.xilinx.com/support/documentation/white_papers/wp469-microblaze-for-cost-sensitive-apps.pdf. [Accessed: 20- Jun- 2019].

[3] "Vivado Design Suite", *Xilinx.com*, 2019. [Online]. Available: https://www.xilinx.com/products/design-tools/vivado.html. [Accessed: 19- Jun- 2019].

[4] "7 Series FPGAs Data Sheet: Overview. Product Specification. DS180 (v2.6) February 27, 2018", *Xilinx.com*, 2018. [Online]. Available: https://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf. [Accessed: 19- Jun- 2019].

[5] "Vivado Design Suite Tutorial. Designing IP Subsystems UsingIP Integrator. UG995 (v2016.4) November 30, 2016", *Xilinx.com*, 2016. [Online]. Available: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2016_4/ug995-vivado-ip-subsystems-tutorial.pdf. [Accessed: 20- Jun- 2019].

[6] "Using Xilinx SDK", *Xilinx.com*, 2016. [Online]. Available: https://www.xilinx.com/html_docs/xilinx2016_4/SDK_Doc/index.html. [Accessed: 20- Jun- 2019].