

Wireless Communication Platform For Controlling Pld's Through Fpga

P. Harish M.Tech, S.Sandeep M.Tech,

Asst.Professor, Dept.of ECE, SVP CET, RVS Nagar, Puttur

ABSTRACT

This paper presents and describes the implementation of a Wireless communication platform for devices synthesized into Xilinx FPGAs. This platform provides a way to control programmable logic devices, as well as data communication using a standard network. The platform modularity allows its use by most devices that can be synthesized into a FPGA, providing they adhere to the required implementation directives (FIFO communication). Presently the proposed platform is being used for the implementation of a remotely accessible, and Controlling of PLD's through FPGA. System provides a bidirectional communication between synthesized devices and client using a wireless network.

Previously PLD's through FPGA is controlled by ETHERNET network, which enables the control of the PLD's with in the small area. To control a device, the Option will be provided in the web page, by opting those options in the web page we can have access to the device in the distant place from the controlling place itself.

The design is based on a traditional embedded system (software and hardware) also synthesized into the FPGA. The system hardware is based on a Spartan-3A and implemented using Xilinx-12.1, Elbert. It runs on the Windows operating system and custom developed applications. Web page was developed by Dot-Net.

1. INTRODUCTION

As the world becomes more dynamic and information is required in real time, computer networks and Internet are increasingly being used in order to meet the demand for fast and remote real time access to information. Consequently, it is important and sometimes fundamental that an electronic device can access computer networks for data communication and also remote control. In a FPGA (Field-Programmable Gate Array) it is possible to synthesize a series of devices, in which the use of computer networks for control and data transfer are often essential, or at least a valuable feature.

Networking adds features that might facilitate enhance and extend the action of digital devices. These functionalities make compatible

equipment very useful for remote controlling of Devices. The implementation of such communication link, between a FPGA synthesized device and a computer requires GPRS modem which is connected to FPGA through UART. This provides bidirectional communication between the synthesized device and an GPRS Modem, Then GPRS modem and a computer communicate with each other using a standard internet network, as presented in Figure1. Considering the desired implementation, this UART and the target device are both synthesized on the same FPGA. In our case we designate this UART as a communication platform

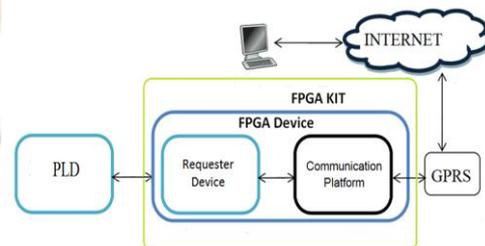


Figure 1. Model of communication between the FPGA and a PC.

As the communication platform must support devices that require networked communication (i.e. requester devices), it is synthesized into the FPGA. The system was developed using the Xilinx ISE-12.1 and includes a Pico Blaze soft core processor. This option has several justifications, namely: (1) the academic focus of its implementation; (2) the user-friendly interface of Xilinx-ISE, which facilitates the inclusion of Pico Blaze and (3) the portability of Windows Operating systems for Xilinx FPGAs.

This solution can be included on most Xilinx FPGAs, and provides a simple interconnection process between the requester device and the communication platform, using the Integrated Software Environment- ISE. Additionally, the robustness and stability of Windows Operating System and the Xilinx-ISE tool, makes the use of Windows Operating System in an system, together with Pico Blaze, a more reliable solution for the communication platform implementation. The communication platform and its operation are presented in section

II and explained in detail in section III. In section IV the results are analyzed, and finally, in section V, we make some comments about the development and use of the proposed solution.

II.COMMUNICATION PLATFORM

The communication platform is based on an embedded system synthesized into the FPGA. The implementation of such communication link, between a FPGA synthesized device and a computer requires GPRS modem which is connected to FPGA through UART. This provides bidirectional communication between the synthesized device and an GPRS Modem, Then GPRS modem and a computer communicate with each other using a standard internet network. Considering the desired implementation, this UART and the target device are both synthesized on the same FPGA. In our case we designate this UART as a communication platform.

The implementation of a complete embedded system involves the following hardware and software components.

The embedded system hardware

The main hardware component of an embedded system is the processor. There are several types of processors available, that can be used on Xilinx FPGAs, namely the Pico Blaze, Micro, among others . However, an embedded system is not only composed by a processor, but also by buses, RAM and ROM memory and other separate devices such as UART and MAC network controllers. The integration of all these devices within an FPGA and the development of a complete embedded system is normally a complex task.

The Xilinx tool uses intellectual property (IP) cores, to build all the hardware of the embedded system. These include the Pico Blaze soft core microprocessor, the PLB bus, the UART, and all external devices connected to the FPGA, like RAM and FLASH memories. A very critical IP core is the General Purpose Input/Output (GPIO), which has a configurable number of input and output data ports that can be connected to any device synthesized into the FPGA. The GPIO is more general and flexible than a FIFO, and its configuration in the Xilinx12.1 is relatively simple. For this purpose, it is only necessary to configure the ports of the GPIO in a way that meets the connected device port requirements. This process allows the use of the GPIO to interconnect any requester device to the communication platform.

The embedded system software

The embedded system executes its functions according to the software running in the Pico Blaze processor. The first is the execution of the UART and services, for example running a AT

Command Set (on the Pico blaze) that allows a remote computer to connect with it, via the Wireless network, in order to remotely control the requester device. The second function is to manage the Communications with the requester device, namely, managing data transfer between the requester device and the server.

The implementation of the communication platform depends on the requester device, because the GPIO ports must be configured to allow the connection of diverse such devices (and architectures). Given this fact, a generic scenario was defined and implemented, in which a remote computer using AT Commands and FTP sends and receives a text file (with hexadecimal data) from a FPGA synthesized device, via an Wireless network.

III IMPLEMENTATION OF COMMUNICATION PLATFORM

This device, which is identified as the requester device, is composed of a simple forwarder device and two FIFOs, one for data input, called FIFO IN, and other for data output, called FIFO OUT. The device receives the data from FIFOIN and by manipulating it automatically forwards it to FIFO OUT. The communication platform implemented with a GPIO OUT and a GPIO IN, which are connected respectively to the FIFO IN and FIFO OUT of the requester device, sends to FIFO IN the data from the remote computer and receives from FIFO OUT the data from the requester device and sends to remote computer.

Implementation of embedded system Hardware

The system requires a device in the Spartan 3A Starter Kit to implement the complete embedded system. Most of the embedded system devices are synthesized within the FPGA like UART, GPIO, TIMER etc.,then the UART is connected with GPRS module for making Wireless network between the FPGA and Client. All the devices that are synthesized in FPGA are slaves controlled by the Pico Blaze, which is the master and runs at 87 MHz Pico blaze consist of KCPSM3 8-bit controller.

KCPSM3

The Pico Blaze microcontroller is a compact, capable, and cost-effective fully embedded 8-bit RISC microcontroller core optimized for the Spartan-3 family, with support for the Virtex-5, Spartan-6, and Virtex-6 FPGA families. The Pico Blaze microcontroller provides Cost-efficient microcontroller-based control and simple data processing. Its compact yet capable architecture consumes considerably less FPGA resources than comparable 8-bit microcontroller architectures within an FPGA.

The Pico Blaze microcontroller is optimized for efficiency and low deployment cost. It occupies just 96 FPGA slices. In typical implementations, a single FPGA block RAM stores up to 1024 program instructions, which are automatically loaded during FPGA configuration. Even with such resource efficiency, the Pico Blaze microcontroller performs a respectable 44 to 100 million instructions per second (MIPS) depending on the target FPGA family and speed grade. The Pico Blaze microcontroller core is totally embedded within the target FPGA and requires no external resources. The Pico Blaze microcontroller is extremely flexible. The basic functionality is easily extended and enhanced by connecting additional FPGA logic to the microcontroller's input and output ports. The Pico Blaze microcontroller provides abundant, flexible I/O at much lower cost than off-the-shelf controllers. Similarly, the Pico Blaze peripheral set can be customized to meet the specific features, function, and cost requirements of the target application. Because the Pico Blaze microcontroller is delivered as synthesizable VHDL source code, the core is future-proof and can be migrated to future FPGA architectures, effectively eliminating product obsolescence fears. Being integrated within the FPGA, the Pico Blaze microcontroller reduces board space, design cost, and inventory.

The Pico Blaze FPC is supported by a suite of development tools including an assembler, a graphical integrated development environment (IDE), a graphical instruction set simulator, and VHDL source code and simulation models. Similarly, the Pico Blaze microcontroller is also supported in the Xilinx System Generator development environment.

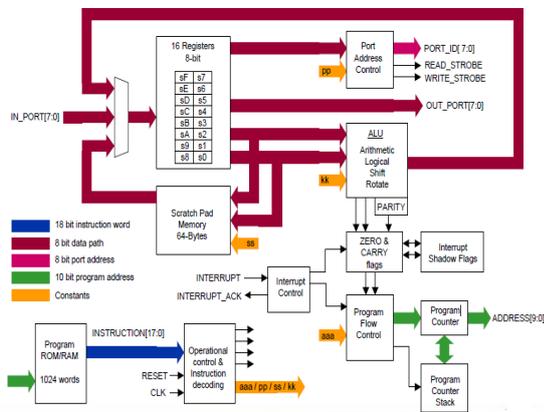


Figure2: the Architecture of KCPSM

PICOBLAZE I/O INTERFACE

To interact with the external environment, a regular microcontroller chip consists of a variety of built-in I/O peripherals, such as a UART, SPI (serial peripheral interface), timer, etc. When starting a new development, we select a microcontroller chip according to the I/O

requirements of the application and may sometimes need to use additional chips to realize less commonly used functions.

Unlike a regular microcontroller, Pico Blaze has no built-in I/O peripherals. It just provides a simple generic input and output structure for an I/O interface. I/O peripherals are Constructed as needed and thus are customized to each application. Pico Blaze uses the **input** and **output** instructions to transfer data between its internal registers and I/O ports, and its interface consist of the following signals:

port-id: an 8-bit signal that specifies the port id (i.e., port address) of an **input** or **output** instruction.

in-port: an 8-bit signal where Pico Blaze obtains input data during operation of an **input** instruction.

out-port: an 8-bit signal where Pico Blaze places output data during operation of an **output** instruction.

Read Strobe: a 1-bit signal that is asserted in the second clock cycle of an **input** instruction.

Write Strobe : a 1-bit signal that is asserted in the second clock cycle of an **output** instruction.

Interrupt: If the INTERRUPTENABLE flag is set by the application code, generate an INTERRUPT Event by asserting this input High for at least two CLK cycles. If the INTERRUPT ENABLE flag is cleared, this input is ignored.

Reset: To reset the Pico Blaze microcontroller and to generate a RESET Event, assert this input High for at least one CLK cycle. A Reset Event is automatically generated immediately following FPGA configuration

Clk: The frequency may range from DC to the maximum operating frequency reported by the Xilinx ISE development software. All Pico Blaze synchronous elements are clocked from the rising clock edge. There are no clock duty-cycle requirements beyond the minimum pulse width requirements of the FPGA

Interrupt_ack: When asserted High, this signal acknowledges that an INTERRUPT Event occurred. This signal is asserted during the second CLK cycle of the two-cycle INTERRUPT Event. This signal is optionally used to clear the source of the INTERRUPT input.

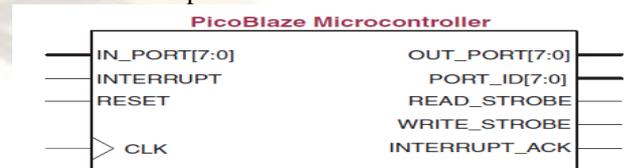


Figure3:PicoBlaze interface Connections

UART

This package contains a pair of macros which have been highly optimised for the Virtex, VirtexE, Virtex-II, Spartan-II, and Spartan-IIe devices from Xilinx. The macros provide the functionality of a simple UART transmitter and

simple UART receiver each with the fixed characteristics of:-1start bit,1stop bit,8 data bits, no parity bit. As well as being able to use these macros as a pair to communicate with each other, they are also fully compatible with standard UART communication protocols such as to a PC (providing level shifting components are employed to generate RS232 signaling).Each macro also contains an embedded 16 byte FIFO buffer, and yet just look at the total size. Their small size makes them an ideal companion to the small KCPSM and KCPSM-II processor macros.

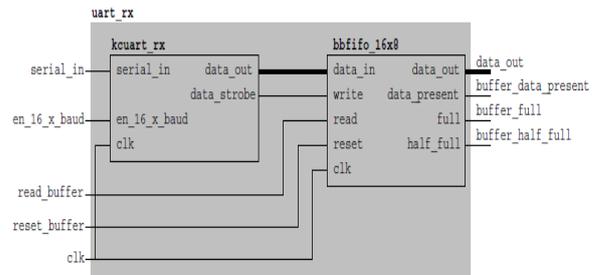


Figure6: Macro internal structural of UART RX

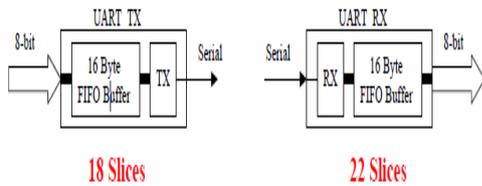


Figure4: Macros of UART TX and RX.

Although standard baud rates from 9600 to 115200 can be supported, these macros are also capable of baud rates exceeding 10 M-bit/second offering an easy way to communicate data between Xilinx devices.

The UART_Tx Macro

The UART transmitter is provided formed by a set of three VHDL files. The top level file 'uart_tx.vhd' is used to combine the FIFO buffer 'bbfifo_16x8.vhd' and the constant(k) compact UART transmitter 'kUART_tx.vhd' modules.

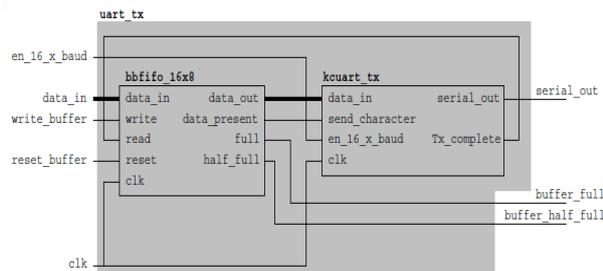


Figure5: Macro internal structural of UART TX

The UART_Rx Macro

The UART receiver is provided formed by a set of three VHDL files. The top level file 'uart_rx.vhd' is used to combine the constant(k) compact UART receiver 'kUART_rx.vhd' and FIFO buffer 'bbfifo_16x8.vhd' modules.

Implementation of Embedded system Software:

In the communication platform, the UART is responsible for managing and implementing the communication with the requester device. The UART is connected to GPRS module, then the communication is established between GPRS and SERVER by using TCP/IP protocol. AT Commands are used for GPRS Module initialization and to establish a connection between server and FPGA.DOT-NET Software is Used to design a Web page, Which is used to control the devices by Providing controlling options in the web-page.

The GPIO speed of operation depends on how long it takes for Pico Blaze to process the instructions of the program. That time is not deterministic, since the microprocessor may have to handle multiple tasks that, in a non real-time operating system, may require a variable number of machine cycles. Therefore, the speed of reading and writing data from/to the GPIO is not precise or even constant.

IV. RESULTS

The main result presented on this paper is the communication platform itself, which provides computer network connectivity to synthesized devices, present in the same FPGA. The use of Xilinx tools turns the communication platform implementable in most of Xilinx FPGA models.

At hardware level, the main contribution is the development of a communication platform, based on an embedded system implemented with the Xilinx. The system is based on the Pico blaze soft core microprocessor, configured without MMU, thus reducing the logic occupation of the FPGA by the communication platform, which used 96 slices in proposed scenario. On the software side, the main contribution is the functional operating system port in the embedded system created by Xilinx, with all drivers required to run this project. It is a robust and reliable operating system that provides various computers network services and allows flexibility in creating applications another important software contribution is made by the translators programs, with distinct functions for access and control of the GPIO ports. These programs were developed for

the user space and are invoked as commands. Thus, they may be accessed by the application layer services meaning that, in practice, they can be invoked via the computer network.

The complete communication platform was tested and validated with a generic requester device. The proposed solution is being used for the development of a fully operational of controlling of programmable logic devices coded in VHDL and with Wireless support.

V. CONCLUSION

The objective was the creation of a communication channel with a Spartan3A FPGA Starter Kit, via the UART port. Considering the robustness and flexibility of the platform, it is clear that it comprehensively addresses the communication needs of electronic devices, particularly if synthesized into FPGAs, which may prove to be an important contribution for the remote maintenance of electronic devices. The easy and consistent handling of Xilinx makes most concepts of hardware implementation of an embedded system practically transparent, including the FPGA model.

The communication platform has three main factors directly related to the communication, (1) the configuration of the UART (2) the implementation of the translators and (3) the configuration of the GPIO. The platform development has other requirements, which are related to operation, not functionality. The inclusion of Xilinx tools makes the development of the hardware and software included on the platform simpler and more transparent. As future developments on this area it is planned to further simplify the configuration and implementation of the factors directly related to communication between the requester device and computer networks. This effort will include automating the implementation of the GPRS initialization and establishing a connection between server and FPGA Kit. Although we mostly used existing tools and solutions, there is considerable innovation in the concept of a communication platform in which its implementation is the simplest possible. The idea is the development of an IP core, requiring only basic parameters to automatically implement the whole platform. Thus, it creates a simple environment for implementing the communication with the FPGA, but one that is rich in features and implementation options.

The GPRS Data Rate, is still a problematic factor for the platform, namely when the requester device requires high data rate communications.

As

implementing a multiple requester devices the data communication between devices and server is done serially, which is a delay factor if the data

communication is done parallel the delay gets down.

REFERENCES

- 1) Tanenbaum, A.S., Computer Networks, 4 ed., Prentice Hall PTR, 2003.
- 2) Xilinx, (2010) Pico Blaze Soft Processor. [Online]. Available: <http://www.xilinx.com/tools/microblaze.htm>
- 3) Xilinx,(2003) UART.[Online]. Available: <http://www.xilinx.com/tools/UART.htm>
- 4) Rodrigo Neri de Souza, Plataforma de Comunicac ão Ethernet para dispositivos embarcados em FPGAs da Xilinx, Undergraduate thesis, IFSC, Brazil, 2010.
- 5) Ethernet Communication Platform for synthesized devices in Xilinx FPGA- Daiana Nascimento Muniz,UTFPR, Curitiba, Brazil, Andr e Vaz da Silva Fidalgo ISE Porto, Portugal
- 6) SIM300 AT Commands by simcom
- 7) SIM300 Hardware specifications by simcom
- 8) OpenCores, (2010) Projects :: OpenCores. [Online]. Available: <http://opencores.org/projects>
- 9) Elbert-Spartan 3A FPGA Development board user guide[Online].Available: <http://numato.com>