IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 4, Issue 4, Ver. II (Jul-Aug. 2014), PP 58-61
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

GSM Based Configuration of FPGA

S. Karthik[1], Prasanna Vishal TR[2], Jayaram SG[3], K. Priyadarsini[4]
[1](Department of Electronics and Communication, SRM University,Vadapalani India)
[2](Department of Electronics and Communication, SRM University,Vadapalani India)
[3](Department of Electronics and Communication, SRM University,Vadapalani India)
[4](Department of Computer Science and Engineering, SRM University,Ramapuram India)

Abstract : *A field-programmable gate array (FPGA) is an integrated circuit designed to be configured by either the customer or a designer after manufacturing – hence "field-programmable". In-flight reconfigurability and dynamic partial reconfiguration enhances space applications with re-programmable hardware and at run-time adaptive functionality attracts the use of FPGAs. FPGAs can provide designers with almost limitless flexibility, but once FPGA is programmed and interfaced with other pheripherals, it's difficult to change the application which is running on FPGA. So, here we use a wireless programming technique to configure the FPGA based on our requirement. A wireless medium is preferred so as to avoid a physical connection with FPGA. In our idea we are going to select the application program in RAM and directly configuring it to FPGA using GSM. Hence we can swap from one application program to another by using a message sent from the user.*

Keywords: *Arduino, AVR, FPGA, GSM*

I. Introduction

In this paper, we focus on selecting different applications in a Data Flash and configuring it to the FPGA. The application is chosen based on the message sent from the user. GSM is preferred as our wireless medium [1] due to its wide range of coverage compared to ZIGBEE or IR. Hence a user can access the FPGA and reconfigure it based on his convenience within minutes from any place. Arduino is interfaced with the FPGA so as to transfer the message.

II. Arduino

Arduino[3] is a single-board Atmel 8-bit AVR micro-controller intended to make the application of interactive objects or environments more accessible. The ATmega328 is a single chip micro-controller of the megaAVR series created by Atmel. The high-performance Atmel 8-bit AVR RISC-based microcontroller combines 32KB ISP flash memory with read-while-write capabilities, 1KB EEPROM, 2KB SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte-oriented 2-wire serial interface, SPI serial port, 6-channel 10-bit A/D converter (8-channels in TQFP and QFN/MLF packages), programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The device operates in the voltage range of 1.8V – 5.5V.

III. Interfacing with Spartan 3e fpga

The Spartan-3E family [14] of Field-Programmable Gate Arrays (FPGAs) is specifically designed to meet the needs of high volume, cost-sensitive consumer electronic applications. Spartan-3E FPGAs are programmed by loading configuration data into robust, reprogrammable, static CMOS configuration latches (CCLs) that collectively control all functional elements and routing resources. The FPGA's configuration data is stored externally in a PROM [2] [4] or some other non-volatile medium, either on or off the board. After applying power, the configured data is written to the FPGA using any of seven different modes.

Now, the Arduino is connected which receives the message from GSM and transmits it to the FPGA.With respect to this application it is not necessary to connect the Arduino directly to the FPGA. For this purpose, we need to note the different memory locations of each program given in Table I and interface it with the Arduino.

There is a data flash memory connected with the FPGA which is used to load the program in the FPGA when booted or on reset. Initially, all the program data which the designer wants to load to the FPGA is placed in this memory. We can access 2^n memory locations using n lines. The application program can be configured in

to the FPGA via data flash or directly using the Xilinx FPGA programmer. The above steps are done by establishing connection to the computer using USB.

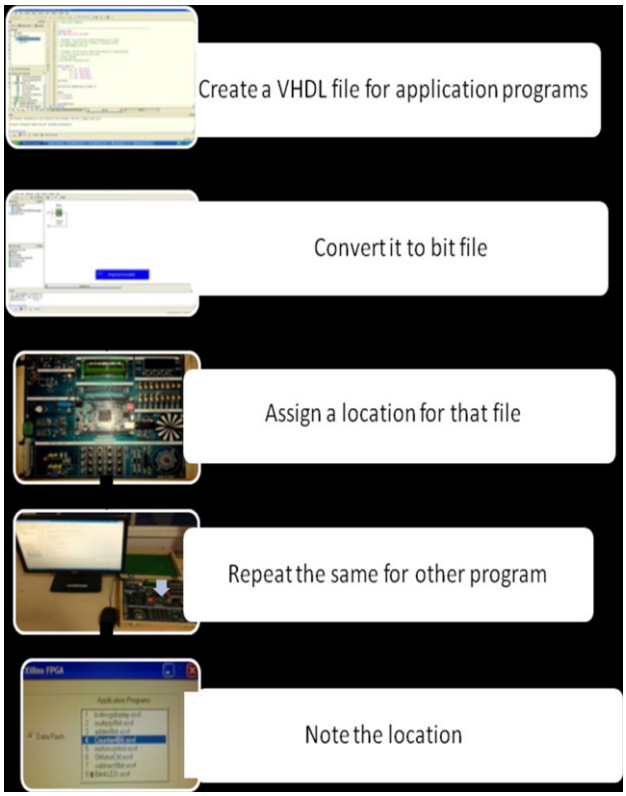

Fig-I: Steps involved in implementation

Table -I: Memory Interface

PIN 1	PIN 2	PIN 3	PIN 4	Mode select
On	On	On	On	Selects Application Program 1
Off	On	On	On	Selects Application Program 2
On	Off	On	On	Selects Application Program 3
Off	Off	On	On	Selects Application Program 4
On	On	Off	On	Selects Application Program 5
Off	On	Off	On	Selects Application Program 6
On	Off	Off	On	Selects Application Program 7
Off	Off	Off	On	Selects Application Program 8

IV. Role Of Gsm

GSM (Global System for Mobile) / GPRS (General Packet Radio Service) TTL –Modem is SIM900 Quad-band GSM / GPRS device [5] which works on frequencies 850 MHZ, 900 MHZ, 1800 MHZ and 1900 MHZ. It is very compact in size and easy to use as plug in GSM Modem.

To interface GSM module with Arduino [6] three connections have to be made between them along with separate power supply.

The three connections are:

1. Connect TX pin of SIM900 GSM/GPRS module to TX pin of Arduino.
2. Connect RX pin of SIM900 GSM/GPRS module to RX pin of Arduino.
3. Connect GND pin of SIM900 GSM/GPRS module to GND pin of Arduino, so that they work in the same logic level.

V. Configuration

Spartan-3E FPGAs are programmed by loading configuration data into robust, reprogrammable, static CMOS configuration latches (CCLs) that collectively control all functional elements and routing

resources. The FPGA's configuration data is stored externally in a PROM or some other non-volatile medium, either on or off the board.

The user types the command needed to select the specific application program and sends it using the mobile. When the command is received, the GSM module sends the command serially to Arduino. Arduino processes the command sent and selects the necessary application program as described in Fig. II and Fig. III.

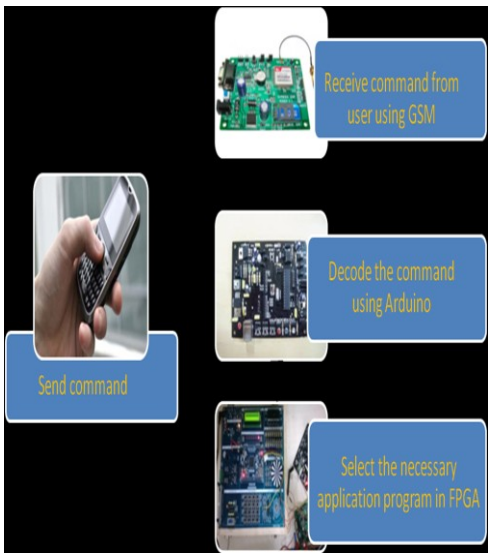

Fig-II: Overall process

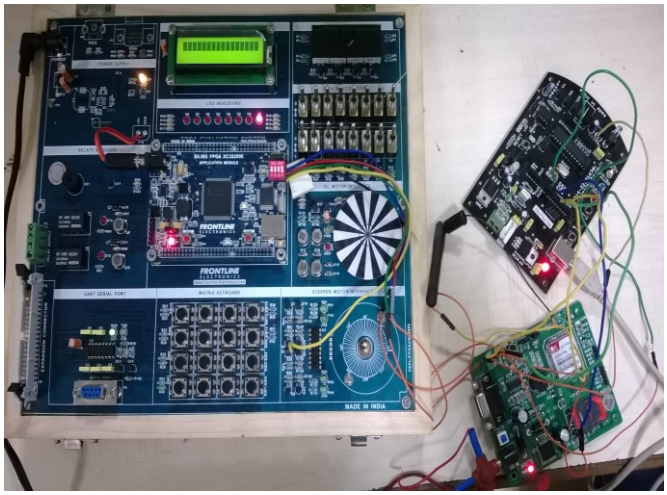

Fig-III: Working model

VI. Applications

Applications of FPGAs include digital signal processing, ASIC prototyping, medical imaging, computer vision, speech recognition, cryptography, bioinformatics, computer hardware emulation, radio astronomy, metal detection and a growing range of other areas. Few of those are as follows:

Software-Defined Radio (SDR): The ability to adapt to varying waveforms is a paramount goal in military SDR.

Field testing: By designing with reconfigurable components, applications can be reconfigured in the field without needing to go back to the lab for re-engineering.

*Airborne applications***:** Susceptibility to radiation-induced single event upsets (SEUs) makes it important to monitor and reconfigure devices with bit failures.

*Remote sensors***:** The ability to reconfigure devices that are difficult to reach physically makes it important to be able to update "over the air."

VII. Conclusion

Current FPGA capabilities in the area of run-time reconfiguration are maturing to meet the needs of military and wireless communications customers. As the capabilities of programmable logic devices grow, there will be an increased demand for flexible reuse of FPGA resources. Advances in this area will continue to be made in device configuration and reconfiguration speed, built-in error detection and recovery and ease of design of the reconfiguration modes. Designers are more likely to continue developing sophisticated designs that require robust implementations of partial reconfiguration. For this reason, work will continue both in developing the capabilities and ease-of-use of partial reconfiguration, as well as software partial reconfiguration .As is the case with many FPGA capabilities, having the technology available today is not sufficient to compel.

References

[1]. Guifen Gu and Guili Peng, The survey of GSM wireless communication system 2010 International conference on computer and information application (ICCIA)

[2]. Malik U and Diessel O, On the placement and granularity of FPGA configurations 2004 IEEE International conference on Field-Programmable technology

[3]. Steven F.Barrett, Arduino Microcontroller Processing for Everyone (Morgan and Claypool publications 2010)

[4]. Lawal N, Thomberg B and O'Nils M, Architecture driven memory allocation for FPGA based real-time video processing systems

[5]. 2011 VII Southern Conference on Programmable logic (SPL)

[6]. Dehng G.K, Kuo C.F and Wang S.T, A single-chip RF transceiver for quad-band GSM/GPRS applications 2004 IEEE Radio frequency integrated circuits (RFIC) Symposium

[7]. Vandana Pandya and Deepali Shukla, GSM Modem based data acquisition system International journal of Computational Engineering research, 2(5), 2012, 1662-1667

[8]. B. Fiethe, H. Michalik, C. Dierker, B. Osterloh, G. Zhou, Reconfigurable System-on-Chip Data Processing Units for Miniaturized Space Imaging Instruments, Proceedings of the conference on Design, automation and test in Europe (DATE), pp. 977-982, ACM, 2007, ISBN 978-3-9810801-2-4

[9]. M. Cassel, D. Walter, H. Schmidt, F. Gliem, H. Michalik, M. Stähle, K. Vögele, P. Roos, NAND-Flash Memory Technology in Mass Memory Systems for Space Applications, Proceedings of the conference on Data Systems In Aerospace (DASIA), May 2008

[10]. M. Ullmann, M. Huebner, B. Grimm, J. Becker: "An FPGA Run-Time System for Dynamical On-Demand Reconfiguration", RAW04, Santa Fee.

[11]. M. Hübner, J. Becker: "Exploiting Dynamic and Partial Reconfiguration for FPGAs - Toolflow, Architecture and System Integration"

[12]. Cristiana Bolchini, Davide Quarta, Marco D. Santambrogio: "SEU Mitigation for SRAM-Based FPGAs through Dynamic Partial Reconfiguration"

[13]. Davin Lim and Mike Peattie: Two types for partial reconguration: Module based or small bit manipulations, xapp290 (v1.0). May 2002.

[14]. Davin Lim and Mike Peattie: Difference-based partial reconguration, xapp290 (v2.0). Dec 2007. www.xilinx.com/support/documentation/data_sheets/ds312.pdf