i-manager's Journal on Software Engineering, January-March 2007 issue

A PAGING SYSTEM FOR FPGA-BASED CONTROLLERS

Muhammed Abdelati
ECE Department, IUG, Palestine
muhammet@iugaza.edu

Abstract

In this work we design and implement a paging system for controllers which are based on Field Programmable Gate Arrays (FPGAs). A PicoBlaze embedded soft processor within the FPGA is used to monitor the controller and send SMS paging messages when necessary. The messages are transmitted through the GSM network by means of an external modem which is connected serially to the FPGA. The paging system is tested successfully on a speed controller of a servo motor.

1. Introduction

Unlike traditional logic gates which have a fixed function, Field Programmable Gate Arrays (FPGAs) have an undefined function at the time of manufacturing. However, they are configured later using computer aided design (CAD) tools to build digital circuits. The circuit is sketched using schematic capture or it is described textually using a high-level hardware description language such as VHDL. The CAD tool uses the circuit schematic or description code to create and download the FPGA's configuration bit stream [1].

Xilinx is one of the leading companies in manufacturing FPGAs and developing their necessary CAD tools. Due to their efficiency and cost-effectiveness, Spartan-3 FPGA family from Xilinx has attracted the attention of many researchers and manufacturers [2-4]. These FPGAs are not only capable of implementing hardware designs, they are also capable of holding one or more embedded microprocessors giving the researcher the ability to partition his design across hardware and software. This hybrid approach allows enhancing designs by utilizing the computational strengths of the software approach and the speed of hardware approach.

There are literally dozens of microprocessor architectures and instruction sets. Modern FPGAs can efficiently implement any 8-bit or 16-bit microprocessor, and available FPGA soft cores support popular instruction sets such as the PIC, 8051, AVR, 6502, 8080, and Z80 processors. The Xilinx PicoBlaze microcontroller, which is chosen for our work, is specifically designed and optimized for the Virtex and Spartan series of FPGAs and CoolRunner-II CPLDs [5]. The PicoBlaze solution consumes considerably less resources than comparable 8-bit microcontroller architectures. It is provided as a free, source-level VHDL file. Because it is delivered as VHDL source, the PicoBlaze microcontroller is immune to product obsolescence as the microcontroller can be retargeted to future generations of Xilinx FPGAs, exploiting future cost reductions and feature enhancements.

It is remarkable that designing and implementing a *System on Chip* (SoC) remained monopolized for long time by industrial countries that have long successful history of VLSI manufacturing. However, the advent of those high density programmable chips provided a chance for researchers in the third world countries to enter this era with affordable resources.

In a previous work we have utilized Spartan-3 FPGAs to implement PID controllers [6]. All PID modules are totally implemented using hardware approach yielding a fast, compact, and novel design. In this research we enhance our design so that it encompasses a paging system. A PicoBlaze-based embedded device is added in order to monitor the PID controller. In emergency situations this embedded device sends alerting messages to an external GSM modem which is interfaced serially to the FPGA. This paging system may be utilized in many applications such as electrical generators, elevators, and rural treatment plants.

The rest of this paper is organized as fallows: Next section describes interfacing a GSM modem to a PicoBlaze microcontroller. It aims to introduce the block diagram of a typical PicoBlaze-based system as well as the GSM modem used in this work. Section 3 describes the remaining components of the realized paging system, and finally in Section 4, comments and suggestions for future work are given.

2. Building a PicoBlaze-based system with a GSM modem

Recently, there has been a rapid development of wireless data applications, such as short message, vehicle navigation, remote monitoring, wireless Internet access, etc. Therefore, more and more devices need to be able to do wireless communication. With this background, many manufacturers introduced the GSM/GPRS modem which allows users to add wireless communication capability easily to their own products, and then, develop many applications based on the GSM network. For example, Sky microwave Corp. developed its *MOD 9001 BENQ GSM/GPRS Modem* which mostly fits the need of SMS data communication [7]. Having a small size makes it suitable for both embedded application and external peripheral equipment. The AT command set and RS232 interface offers easy data connection without any extra circuit control. In our work we interfaced this modem to a PicoBlaze microprocessor as illustrated in Figure 1.

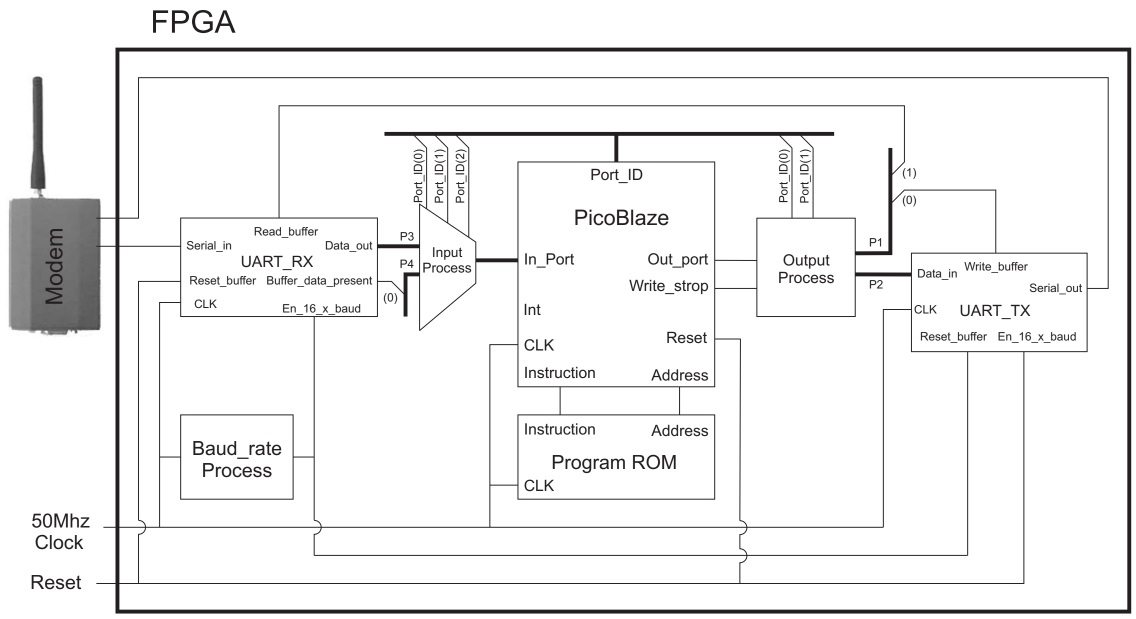

Figure 1. A GSM modem interfaced to a PicoBlaze

At the center of the circuit there is a PicoBlaze microcontroller interfaced to the program ROM. The input process and the output process define the input ports and the output ports respectively. Two input ports and two output ports are necessary to interface the PicoBlaze to the Universal Asynchronous Receiver/Transmitter modules (UART_Tx and UART_Rx). The baud rate process generates the clock necessary for setting the baud rate of the serial communication.

The microcontroller communicates with the modem through the UART modules using a set of AT commands available in modem datasheets [8]. Each AT command line is made up of three elements: the prefix, the body and the termination character. The command line prefix consists of the characters "AT" while the termination character is the carriage return character. This character has an ASCII code of 0Dh and represented below as <CR>. In order to illustrate the basics of handling SMS messages using AT commands, assume that it is required to send a GSM 7-bit alphabet text "Hello world" to a mobile GSM phone whose number is 9363665. This task may be accomplished by implementing the following protocol:

Step 1: The modem should be initialized in the text mode which is appropriate for SMS. This is accomplished when the PicoBlaze sends the ASCII code of the following command:

AT+CMGF=1<CR>

that is equivalent to the hexadecimal data frame

41542B434D47463D310D

On successful reception by the modem, it responds by sending "Ok" back to the PicoBlaze.

Step 2: The destination GSM phone number is specified by using the following command:

AT+CMGS="9363665"<CR>

On successful reception by the modem, it responds by sending ">" back to the PicoBlaze.

Step 3: The text message is delivered to the modem and terminated by Ctrl+z character. This character has an ASCII code of 1Ah and represented below as <CTRL-Z>.

Hello world<CTRL-Z>

On successful reception by the modem it responds by sending "Ok" back to the PicoBlaze.

In each step the PicoBlaze issues a command and waits a reasonable time to read the modem response (about one second). If it indicates a successful reception, the PicoBlaze proceeds to next step. Otherwise, the current step's command is retransmitted. One may program the PicoBlaze to issue an error signal indicating a communication error if the number of trials exceeds a certain threshold (say 5 times). Figure 2 illustrates this protocol using a Sequential Function Chart (SFC).

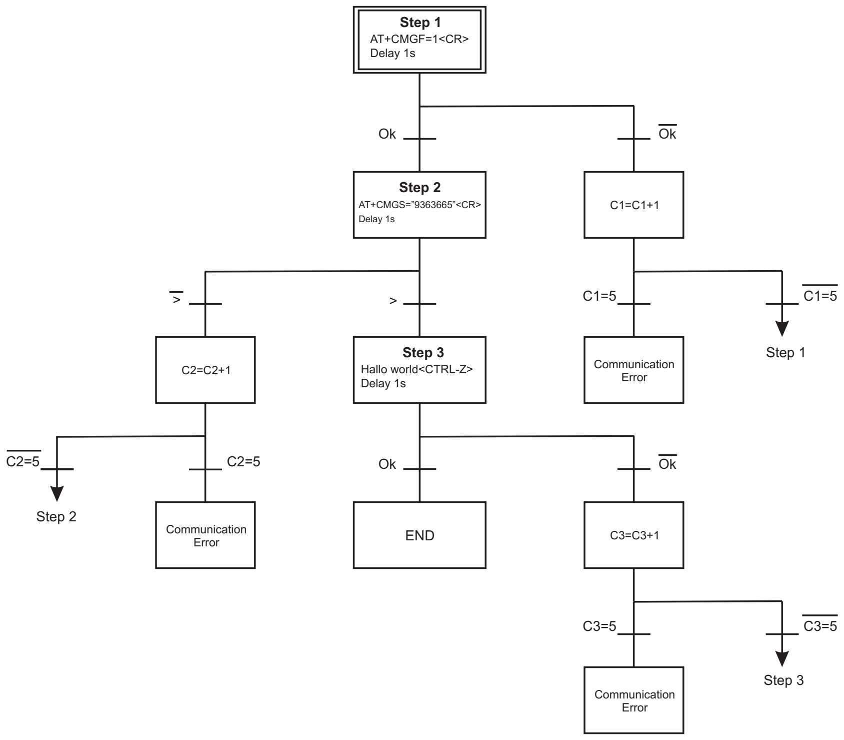

Figure 2. SFC of the implemented message sending protocol

3. System design.

A machine with a paging system is illustrated in Figure 3. The paging system consists of an SMS pager, a GSM network, and a remote monitor. The SMS pager is composed of a message handling unit (MHU), a modem, and an error handling module (EHM). The message handling unit is used to store the text messages of the pager while the modem is used to transmit these messages over the GSM network.

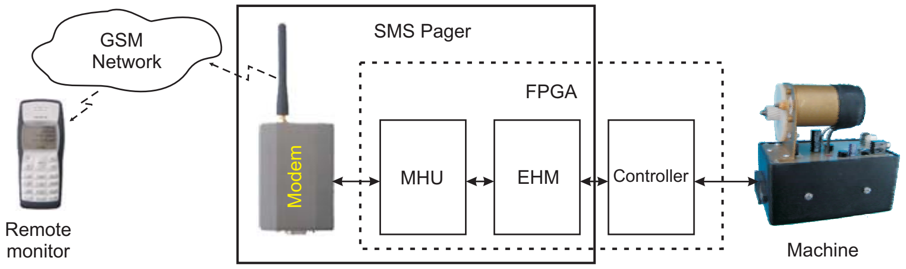

Figure 3. Overall system block diagram

The error handling module is responsible of monitoring and processing alarm signals of the machine. As soon as any of these signals exceeds a predefined threshold (analog signals) or leaves its normal state (digital signals) for a period longer than a preset time, this module sends the identification of this alarm condition to the message handler which in turn sends its corresponding message to the GSM modem using AT commands. Consequently, the modem transmits this message to a remote monitor (a mobile phone) over the GSM network. The EHM may be configured to issue some emergency actions on the machine depending on the alarm priority level.

A microcontroller such as PIC or 8051 with help of some other electrical components may be used to implement both of the MHU and the EHM. However, using FPGAs in system realization makes it feasible to integrate these devices with the machine controller in a single chip reducing the system hardware complexity and cost. Moreover, having sufficient resources, it is also possible to modify a pre-implemented FPGA system to include a paging system with negligible wiring efforts. Thanks to reconfiguration capability of FPGAs.

For example, in a previous work we have implemented a PID speed controller of a servomotor using a 200K gate Spartan-3 FPGA chip. Only 9% of the chip's slices was necessary to realize the controller leaving 91% for future upgrade. It is remarkable that this chip is the smallest member in the Spartan-3 family and costs few dollars only.

It is required to modify the controller so that if the motor speed fails to track the reference speed within a preset time window, then the motor is switched off and an alarm paging message is issued. This will protect the motor in case of overload or sensor failure situations.

The modified controller system is illustrated in Figure 4. There is a speed controller module (Speed_Controller) similar to the one presented in [6]. It works to minimize the error signal which is the difference between a reference speed (specified by the operator) and the actual speed (measured process value).

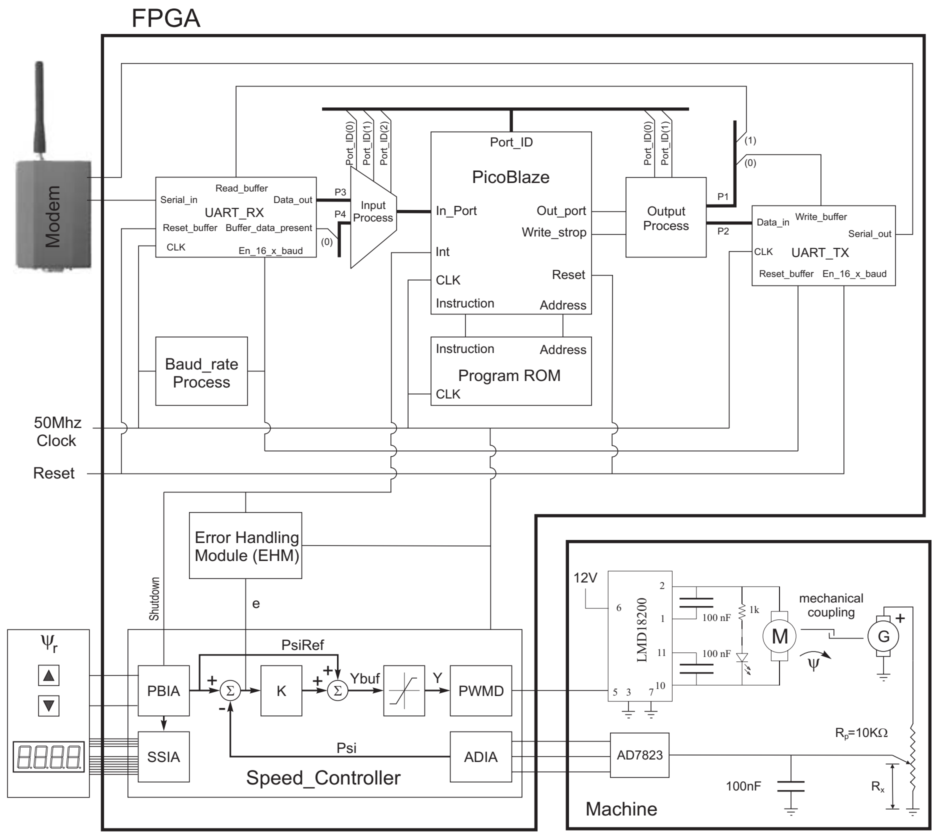

Figure 4. A speed controller with a paging system.

This error signal (e) is applied to an error handling module (EHM) which calculates the average of the error signal (e_{av}) during a moving time window (W) and compares it to a threshold (e_{th}). This threshold equals to the maximum tolerable value of the error signal which certainly depends on the application. Once the averaged error value exceeds that threshold, the EHM triggers the shutdown input of the speed controller as well as the interrupt input of the PicoBlaze microcontroller.

Averaging the error signal is essential for rejecting surge disturbances and improving noise immunity of the system. The EHM is implemented by using an N-Tap finite impulse response filter (FIR) followed by a comparator as illustrated in Figure 5.

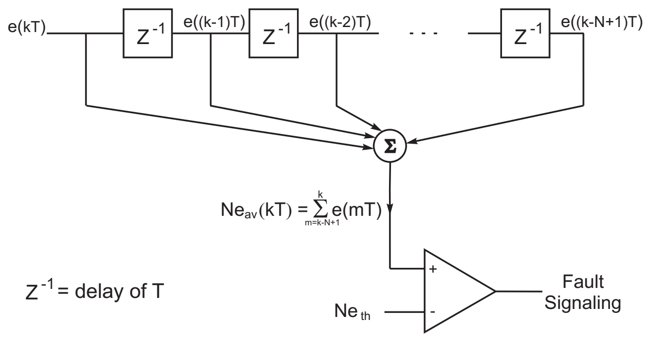

Figure 5. Implementation of the Error Handling Module.

The product of the number of tabs (N) and the sampling period (T) equals the duration of averaging window (W). Determination of its value depends on the desired reaction time of fault reporting. In order to minimize false alarm reporting, the value of averaging time window should be set to the maximum tolerable duration of highly classified values of the error signal which exceed the threshold eth. For our servomotor application we set W=2s and eth = 10 divisions which corresponds to 4% of the maximum speed. The value of the sampling period (T) depends on the bandwidth of the error signal. Using unnecessarily small sampling period increases the number of tabs and hence complicating the module implementation. In our servomotor speed control, slow variations in the reference speed are expected and therefore a sampling period of 0.4s is sufficient. This ends up with the following specifications of the EHM:

Threshold (e_{th})	= 10 divisions
Averaging period (W)	= 2 s
Sampling period (T)	= 0.4 s
Number of tabs (N)	= 5

The software program of the PicoBlaze microcontroller is composed of a main code and an interrupt service subroutine. The main code sends the initialization commands of the modem then it enters an infinite loop waiting for interrupt. On the other hand, the interrupt service subroutine carries sending the SMS message characters which is described in the previous section. The KCPSM3 tool is used to convert the software program, which is written using the assembly instruction set of the controller, to the VHDL code of the program ROM. Then, the system VHDL modules are integrated using the ISE tool. This tool synthesizes the design and creates the configuration bit stream of our target FPGA. The created system consumed 15% of that chip's slices.

5. Conclusions and suggestions for future work

Based on utilizing a PicoBlaze soft-processor on FPGAs, a method for implementing a paging system is proposed in this work. This method has great significance for controllers which are already implemented on FPGAs.

The approach used in this work demonstrates nowadays trend in system design which pair hardware with software to take advantage of the strength of each. The speed controller in our example together with the error handling module are totally implemented in hardware while the message handling unit is realized in software all of which are imbedded within the same FPGA chip.

In a future work we plan to improve the system by adding remote administration capabilities to the FPGA-based systems. One possible approach is by allowing the remote operator to send back SMS commands to the controller.

Acknowledgment: The author gratefully acknowledges Dr. I. Abu-Heba and Mr. K. Zaquot for their help in this research.

References

[1] Muhammed Abdelati, “Modern Automation Systems”, IUG, 2006.

[2] Xilinx, “Spartan-3 FPGA Family: Complete Data Sheet” 2004. http://www.xilinx.com/bvdocs/publications/ds099.pdf

[3] Atmark techno, “Atmark Techno Chooses Xilinx Spartan-3 FPGA and MicroBlaze 32-bit Soft Processor”, http://www.atmark-techno.com/en/news/press-releases/040331-spartan-3-microblaze

[4] M. Abdelati and R. Langari, Implementing Sequential Function Charts Using FPGAs, Proceedings of the Seventh IASTED International Conference on Control and Applications, Cancun, Mexico, May 2005.

[5] Xilinx, “PicoBlaze 8-bit Embedded Microcontroller User Guide”, UG129, v1.1, June 2004.

[6] M. Abdelati, “FPGA-Based PID Controller Implementation”, The Islamic University Journal (Series of Natural Studies and Engineering) Vol. 14, No. 1, P. 105-127, 2006, ISSN 1726-6807.

[7] Sky Microwave Co. LTD., “MOD 9001 BENQ GSM/GPRS Modem User Manual”, 2005.

[8] BENQ Corporation, “BenQ M22A GSM/GPRS Wireless Module AT Command List”, 2004.

ABOUT THE AUTHOR

Muhammed ABDELATI received the B.S and the M.S. degrees from Middle East Technical University in 1990 and 1992 respectively, and the Ph.D. degree from Bilkent University in 1997, all in Electrical and Electronics Engineering. He is an Associate Professor at the Electrical and Computer Engineering Department of the Islamic University of Gaza, Palestine. He spent the 2004/05 academic year as a Fulbright visiting Professor at Texas A&M University. His current research interests include industry automation, embedded systems, and Fuzzy logic controllers.