FPGAW: FPGA CONFIGURATION OVER THE INTERNET

Octavian Creţ, Zoltan Baruch and Kalman Pusztai
Department of Computer Science
Technical University of Cluj-Napoca, ROMANIA
{Octavian.Cret, Zoltan.Baruch, Kalman.Pusztai}@cs.utcluj.ro

Abstract

This paper presents the FPGAW project, a hardware/software product that allows the dynamic reconfiguration of Xilinx XC4000 and XC3000 Families FPGA devices over the Internet. FPGAW consists of a hardware part and a software part. The hardware part consists of a special board that allows the dynamic configuration of the two FPGA devices it contains over the ISA bus of a PC host. The software part triggers the configuration of the FPGA devices over the Internet (using the TCP/IP protocol). FPGAW is effective under both Linux and Windows NT operating systems.

Keywords: FPGA Devices, TCP/IP Protocol, XILINX XC4000 Family, Client-Server application, Dynamic reconfiguration.

1. INTRODUCTION

The Internet revolution is the greatest event of the past decade, changing the life of the entire scientific community all over the world. This also had an impact on the hardware design process which has to deal now with complex problems raised by the new remote configuration possibilities. If nowadays most of the hardware designs target the computer peripherals, in the future the hardware design process will be dedicated to control the most important computer peripheral: the Internet.

The FPGAW aims to solve the problems raised by the use of hardware devices in places difficult to access (satellites, meteorological sites etc.) but where it is possible to have a system connected to the Internet through a radio modem or a similar device. Another potential application of FPGAW is to upgrade products such as modems and cellular phones. A new version of the firmware can be downloaded easily, without the need to replace any component of the product.

The application field of an Internet reconfigurable hardware system is considerable, due to its portability and to the intrinsic gain of speed that is specific to hardware systems (compared to software). This is a way – whose importance, as we see it, is continuously growing in the world context – of improving the performance of computing systems, by taking off the task-load of computers and distributing it to reconfigurable devices (like FPGAs) that will perform computations faster and in parallel with the host machines. This way, in the future we will see the development of Hardware Distributed Computing as a very important research direction of the scientific community.

2. THE IMPLEMENTATION

FPGAW is a Client-Server application for LINUX and WINDOWS NT to download a configuration into the FPGA chip through the Internet. It is composed of two main parts: a hardware and a software one.

2.1 The FPGAW board

The hardware part of FPGAW is a special board that is an extension of the Xilinx FPGA Demoboard©. This board was built based on the description given in the **busconf.pdf** file that can be found on the Xilinx website (www.xilinx.com). The implementation of this board allowed us to obtain configuration times of about 5 milliseconds, compared to the 13.5-14 seconds usually needed when using the Xilinx Hardware Debugger program together with the XCHECKER© downloading cable. The FPGAW board must be plugged into an ISA slot of a PC.

The Xilinx XC4000 or XC3000 FPGA device must be set to Slave Serial mode. The chip has 3 inputs used when downloading the configuration, which are: **PROGRAM**, **DIN**, **CCLK**. PROGRAM is used to reset the FPGA chip, DIN is the serial data input and CCLK is the configuration clock, generated by the PC bus. Figure 1 illustrates the connection of the inputs to the data pins of the ISA slot.

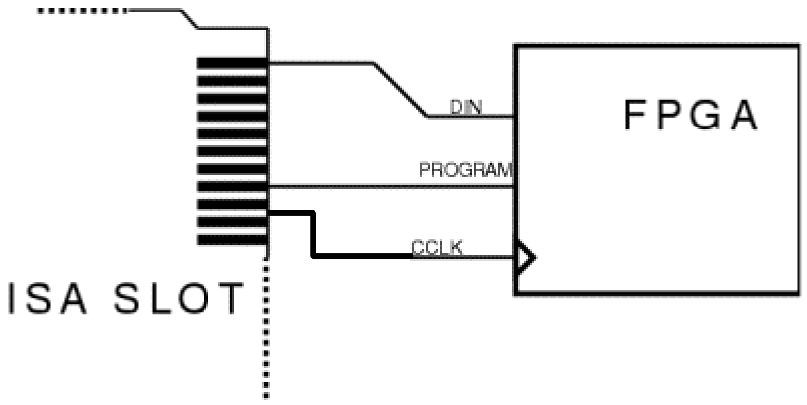


Figure 1. *Connecting the FPGA pins to the ISA slot*

The DIN pin is connected to the D0 line, and the PROGRAM pin is connected to the to D7 line of the bus.

The FPGA Demoboard© allocates for itself a range of hardware ports through which a program (like the FPGAW software) can communicate with it. Usually, this is in the range 0x300-0x307. The FPGA Demoboard© contains two FPGA chips. The XC3020 chip can be configured using the port 0x300, and the XC4005 chip can be configured using the port 0x301. Figure 2 shows the configuration process.

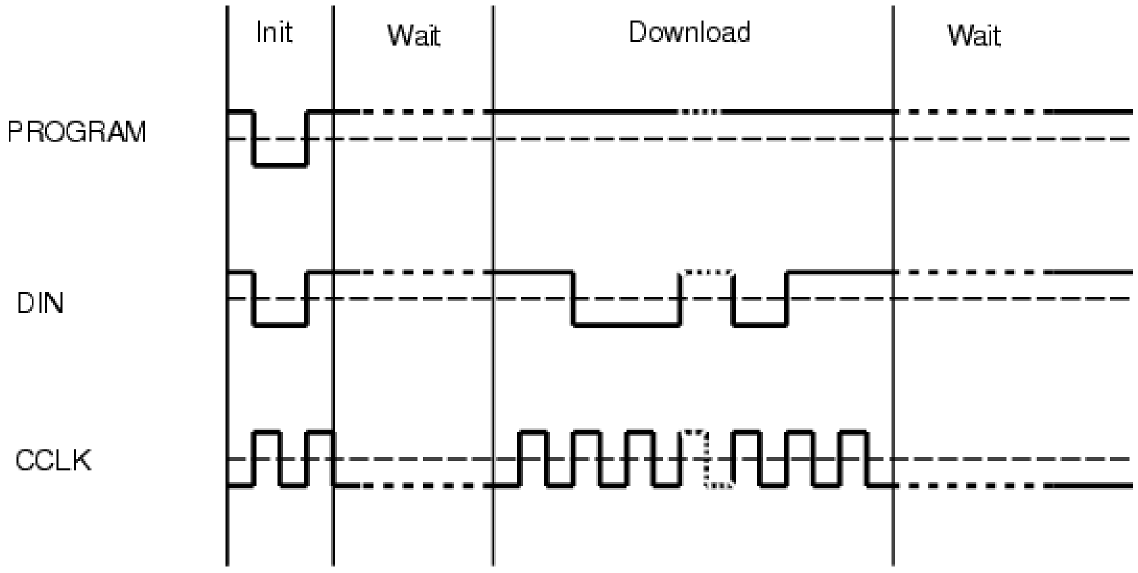


Figure 2. *The waveforms of the configuration process*

2.2 The FPGAW software

The FPGAW software consists of two programs, called Client and Server, that are executed on the Client machine and on the Server machine.

The sources can be compiled both under LINUX and WINDOWS NT operating systems, using the GCC compiler. The graphical version of the Client is written using FLTK, which is a multi-platform GUI library.

To communicate with the 3020 FPGA chip, we write bytes to the port 0x300. The byte corresponds to the D7-D0 lines of the ISA slot. To start the configuration, first the FPGA chip is initialised. This is done by setting the PROGRAM pin to LOW, and then back to HIGH. After that, the chip needs some time to reset itself. The delay in our program is done by reading 9000 times the port 0x300 (because the I/O operations have a constant frequency). Now we can send the data to the chip.

We send the data bit by bit, in serial manner. The data is in bit 0 of the byte we write to the port. As we mentioned before, the PROGRAM pin must be kept HIGH, otherwise the chip will reset. After sending all the data, the chip needs another delay, and than it is ready.

The configuration file (.BIT) of the FPGA device is generated by the Xilinx Foundation Series software. Our program supports *rawbits*, BIT and binary files too.

The *rawbits* files are text files. They start whit a header which contains some specific information, like the name of the file, the creation date, etc. The header is usually a few lines long. After that the configuration bits follow. Each bit is stored in one character, in a human readable manner. The binary files contain only the configuration data. The BIT files are similar to binary files; the only difference is that they have a header too.

All headers are automatically stripped down by our program. The program detects the file format after the file extension. If the extension is .RBT or .ASC, the file is in *rawbits* format. If the extension is .BIT, the file is interpreted as a bit file, otherwise the file is interpreted as a binary file.

3. CLIENT-SERVER COMMUNICATION

As we already mentioned, the program is a Client-Server application, which means that it consists of two executables. The Client program has two versions.

The Server program must be run on the machine which has the FPGAW board plugged into an ISA slot. The Server creates a socket, through which it accepts the Clients connections. The socket is handled as a text file. Both the Server and Client can write and read from it. The Server contains the code for downloading the configuration.

The Client program must be run on the station that stores the configuration file. The Client needs two parameters: the name of the file containing the configuration to be downloaded and the IP address of the Server (more information about the program parameters are presented in Section 4). The Client creates a socket too and tries to connect its socket to the Server's. If it succeeds, the Client sends a header, which contains the name of the file containing the configuration, and some commands to the Server. After the header, it sends the configuration data. Because the socket is handled as a text file and the configuration data is stored in the memory in binary format (independently from the file format), the Client must encode the binary data into textual data (like MIME). This is simple: the Client reads three bytes, which is $3 \times 8 = 24$ bits. It divides in four sections ($24 = 4 \times 6$ bits), so now he has four 6-bit values. For each value in the range 0..63 the program assigns a character (a…z, A...Z, 0...9, -, +), and sends this character. On the other side, the Server gets the encoded data, it reads four characters, finds the 6-bit values and creates the three bytes, which is stored in the memory until all the data are read.

After the data are read, the Server write them to the hardware port (as presented in Section 2) or saves them in a file, depending on the commands sent by the Client. After the data are written, the Server sends a response to the Client.

4. PROGRAM PARAMETERS (OPTIONS)

The options of the program are given in the following format: **key=value.**

Table 1. *FPGAW software parameters.*

Option	Description
host = *string*	The host name of the Server (Client only).
ip = *ipnr*	The IP address of the Server (Client only).
port = *nr*	The IP port number through which the Client and Server communicates.
passwd = *string*	Specifies the password.
serial = *file*	Instead of writing to the hardware port, the Server will save the configuration data in the specified file (Server only).
comport = *nr*	Specifies the hardware port (default is 0x300) (Server only).

Option	Description
waitport = *nr*	Specifies what port should be used in the delay procedure (Server only).
send = *file*	This option is mandatory for the Client. It specifies the configuration file (Client only).
filetype = *nr*	Forces the type of the file if the extension doesn't match the file type (0 = binary; 1 = bit; 2 = rawbit).
filemode = *nr*	If this options is given with a nonzero value, the saved file will be an ASCII file instead of binary.
what2do = *string*	Specifies that the Server what should do with the data. The default is to write into the FPGA chip, but if we specify **sock2file**, it will save the data on the disk.
bitheadlen = *nr*	The length of the header in the bit file.

A password can be given to the Server, and then it allows to connect those Clients who know the correct password. If the Server is executed with the **passwd** option, it will save the password. So next time, even if the user doesn’t specify a password, the Server will request the password from the Clients. To disable the password, the user simply has to delete the passwd file from the Server.

The password is encrypted, so if somebody is “sniffing” the network, he won't find the password (unless he knows the encryption algorithm). The stored password on the Server is encrypted too, and the Server compares the encrypted password (which means that if somebody has read access to the Server, he won't find the password).

5. TECHNICAL RESULTS

In the following examples we assume that we have two machines connected to the Internet.. The first machine (*first.utcluj.ro*) is the Server, with the FPGAW board plugged into the ISA slot. The second machine (*second.utcluj.ro*) is the Client. Suppose the configuration file's name is 3020.rbt.

First start the server (note that the $ means the prompt):

```
$ fpgaws
```

and then send the file with the Client:

```
$ fpgawc send=3020.rbt
```

But now suppose we have a small problem: the second machine is at some location where the users don't have access to any IP ports (only 20, 21 and 80). Suppose that on the first machine the port 80 is free (i.e. no web server is installed). Now we can start the Server with the following options:

```
$ fpgaws port=80
```

and then the Client:

```
$ fpgawc port=80 send=3020.rbt
```

If we want to just save the configuration file on the Server, we simply have to add the **what2do** option to the Client's parameters:

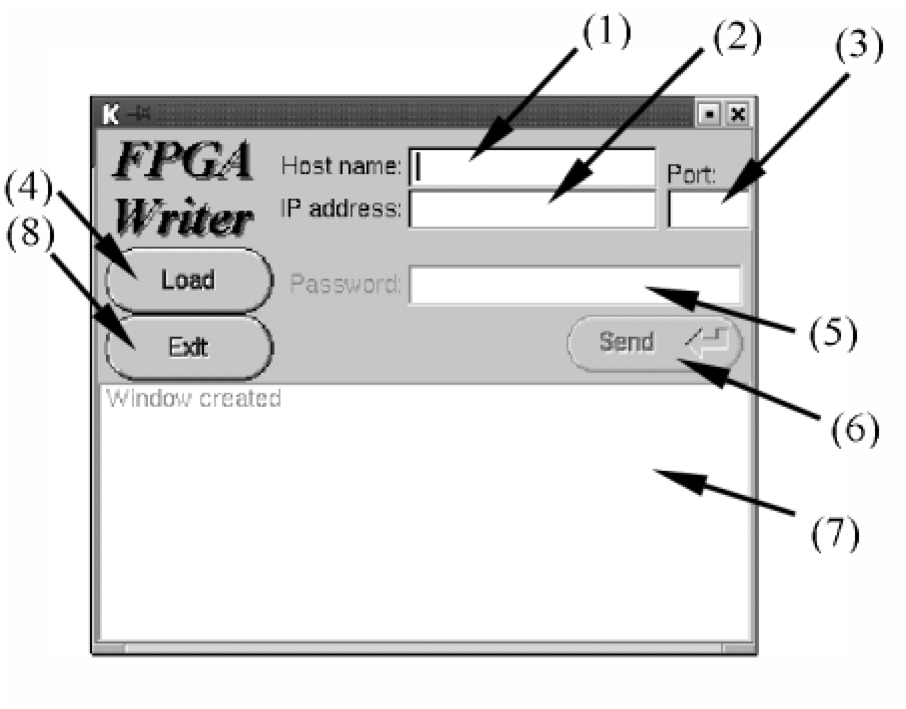
```
$ fpgawc port=80 send=3020.rbt waht2do=sock2file
```

We also wrote a driver for the FPGA chip. It can be used by starting the Server as follows:

```
$ fpgaws serial=/dev/fpgaint filemode=1
```

6. FPGAWCW

This is a more user friendly version of the Client program. After the program is started, a small window appears (Figure 4).



Window created

Figure 4. *The FPGAWCW software*

The signification of the buttons and windows is the following:

1. The host name of the Server.
2. The IP address of the Server. The hostname must be empty to use the IP address.
3. The IP port number, if it is different from the default value.

4. Loads the configuration file.
5. Sets the password.
6. Sends the configuration data.
7. Window for messages from the Server.
8. Quits the program.

The use of the FPGAWCW is very simple:

- Write the host address of the Server;
- Click *Load* and select the configuration file;
- Click *Send*.

7. CONCLUSIONS

In this paper we presented the FPGAW, a hardware / software platform that allows the dynamic configuration of Xilinx FPGA devices on the Internet. The implementation of the FPGAW board, which is an extension of the Xilinx FPGA Demoboard©, allows a significant gain of configuration speed. The FPGAW software is accomplished in two versions: a command-line version and a graphical user interface (GUI) version, and it runs under both LINUX and WINDOWS operating systems. It is composed of two programs: a Client and a Server, thus running as a Client-Server application. The application can be secured using encryption algorithms and passwords, both for the Client-Server communication and for the local storage of the password on the Server machine.

The FPGAW was successfully tested on the Technical University of Cluj-Napoca LAN and the very good results obtained encourage us to continue to develop this application. The functionality of the application will be extended to several types of FPGA devices and it will be used in several larger applications involving dynamic reconfiguration of hardware over the Internet.

8. REFERENCES

[1] Xilinx, Inc. *The Programmable Gate Array Data Book*, San Jose, 2001.
[2] Xilinx, Inc. *Busconf.pdf. Application Note*, www.xilinx.com.
[3] Xilinx, Inc. *Hardware and Peripherals User Guide*, San Jose, 1995.
[4] Jenkins, J. Designing with FPGAs and CPLDs. Prentice Hall, 1993.