Development Environment Using FPGA for Domotics Applications Based on X10 Technology

Manuel D. Cruz[1], Juan A. Ortega[1], Ángel Barriga[2] and Alejandro Fernández-Montes[1]

[1] Department of Languages and Computer Systems, University of Seville, Spain.
[2] IMSE CNM-CSIC / University of Seville, Spain.

Abstract—This communication proposes a basic software and hardware architecture of a controller for the X10 technology interface CM11A, oriented to the world of home automation. The implementation of the system is based in the use of programmable devices such as FPGA (Field Programmable Gate Array). With this controller an end user will be able to control and to manage a set of devices distributed in a domotics space.

I. Introduction

In order to find home automation low-cost and little-invasive technological solutions, and easy handling for the end user, the use of FPGA embedded systems will be an implementation option to have into account [1]. Home automation or domotics are automation technologies applied to buildings. This automation technologies use control devices, sensors and actuators. These devices communicate between them by different communication schemes such as specific buses, wireless communication, or electrical network. From end of the nineties there appeared different standards of communication like CEBus, LonWorks, KNX [2, 3].

The use of the electrical network facilitates the installation of these technologies in the buildings since it is not needed of any additional communication infrastructure. In this sense, devices and communications protocols adapted to this media have been developed [4, 5]. One of the most widespread is the X10 protocol [5, 6]. This one was developed in 1978 by engineers of the company Pico Electronics Ltd. in Glenrothes, Scotland. Since then the technologies based on the utilization of the electrical network known by Power Line Carrier (PLC) have been spreading, appearing a great variety of commercial products and standards [7].

In the area of home automation architectures focused on the education, the systems based on FPGA open a broad field of possibilities [8]. The present communication focuses the process of implementation of a controller capable of communicating with all the devices connected to an X10 network distributed in a domotics space. In particular the developed system focuses in an application to the illumination control. With the control of the illumination integrated in a domotics system it is possible to obtain an important power saving and an increase of the comfort. The next section describes briefly the X10 protocol. Later the FPGA platform is presented. Finally, the last section describes the design of the system.

II. The X10 Protocol

The X10 communication protocol allows controlling remote devices using the electrical network and the specific modules to which they are connected. The control signals are based on the transmission of RF pulses (120 KHz). The transmissions are synchronized with the crossing by zero of the alternating current (AC). The binary one is represented by a pulse of 120 KHz. during 1 millisecond whereas binary zero is represented by the lack of this 120 KHz pulse.

The protocol [9] consists of address bits and commands. In an X10 network can coexist up to 256 different interconnected modules. The way of identifying them is by a code (*housecode*) composed by a letter (from A to P) and by a number (from 1 to 16, the device code). These modules can recognize (depending on its characteristics.) up to 16 different operation codes.

An X10 transmission from the controller, implemented in a development board (DB), to the X10 interface consists of the communication of a *housecode* and device code combination followed by the command code. The sequence of the transmission can be seen in the figure 1. We can observe what each one of the frames of bytes means. This format of transmission is always the same regardless of command or address.

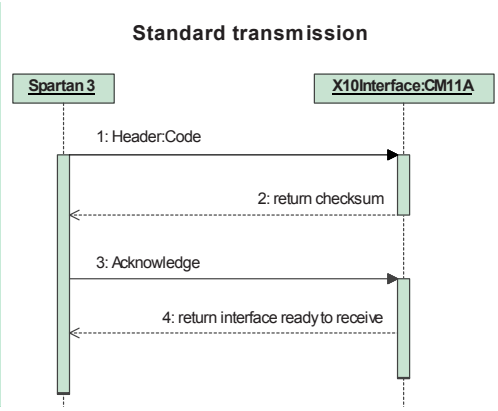

Fig. 1. X10 standard transmission

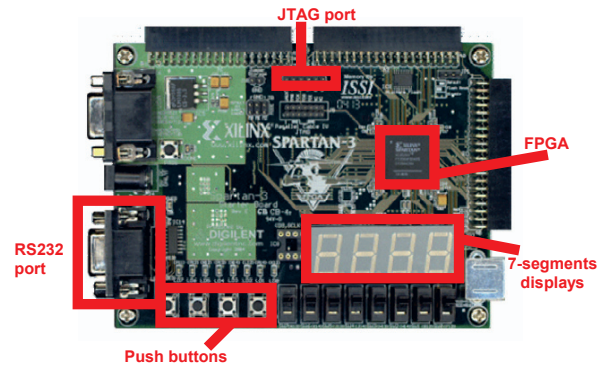

Fig. 2. Xilinx's Spartan-3 development board

1. The DB transmit the *header:code/command* to the X10 interface.
2. The X10 interface receives the frame and makes a checksum. It transmits it back, to the DB.
3. The DB checks the checksum and sends an acknowledgement (ACK) frame.
4. The interface receives the ACK and informs that it is ready for more transmissions.

We can observe a code example in the System Design section, subsection B.

There are an extensive range of products based on X10 protocol which can be classified in the following categories: controllers, modules and complements. The system describe in this paper is composed by a CM11A model PC programmer (belonging to the category of the controllers.) and a lamp module (that receives commands from the electrical network and acts on a lamp.).

III. Hardware Development Platform

The control system has been implemented on an FPGA development board containing a Xilinx Spartan-3 [10]. The system operates with a clock frequency of 50 MHz determined by an oscillator included on the board. The board incorporates a set of elements that facilitate the system development and allows diverse applications. It has two SRAM memory modules up to 256 KB that store the application. The RS232 port allows the communication with the X10 interface.

The address (*housecode*) and the command (*operationcode*) are formed activating four push-buttons. The 7-segments displays allow verifying the operation and inform, as well, about the state of the controller or the X10 devices. Finally, the JTAG communication port has been used for download and test the application in the FPGA In figure 2 the components that have been used are shown.

The FPGA development environment is the Xilinx Platform Studio (XPS), of which we have used the tools that allow covering the design flow of embedded systems. This flow consists of the hardware architecture description stage, the synthesis and implementation of the control circuit, the development of the software applications, the compilation of the above mentioned applications and the programming of the FPGA and the SRAM memory.

The FPGA board controls any X10 receiver module in the home by means of the computer interface module CM11A (figure 3a). The CM11A plugs into an outlet and connects to the RS232 serial port on the FPGA board

The lamp module, shown in figure 3b, is used to control a lamp through X10 protocol. It is controlled by the remote CM11A module. It connects the existing lamp in the socket of this module, which is connected in the socket of the wall. It is the interface module between the lamp and the socket of the wall without requiring any additional connection and any modification in the existing electrical circuit. It offers the possibility of switch on/off the lamp and varying its luminosity remotely by means of an X10 controller.

IV. System Design

A. Hardware architecture

Figure 4 illustrate the configuration scheme of the system. In this scheme the controller has been implemented on the FPGA development board described in the previous section. This board communicates with the X10 interface module by means of the RS232 port. This one transmits the address and commands through the electrical network. The X10 lamp module that has been address receives the commands and acts on the device that it has connected.

a) b)

Fig. 3. a) CM11A X10 computer interface module, b) lamp module

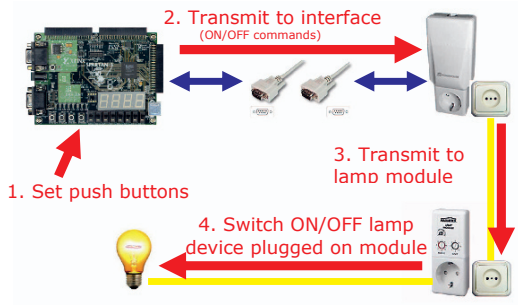

Fig. 4. Communications scheme

The design of the controller has been realized applying a hardware&software codesign methodology that has allowed defining an open and flexible architecture. The architecture is open in the sense that allows incorporating new elements and it is flexible because it allows to be reconfigured.

The system is constituted by two types of elements: the hardware components that constitute the circuit programmed in the FPGA, and the application software that runs on the hardware and is stored in the system memory. The partition of the functionality of the system into the hardware and software components has been realized having in account the already mentioned of open and flexible architecture. For it the circuit has been developed with IP (Intellectual Property) modules. These IP modules allow the reuse and easy the insertion of new functionality.

The architecture, which shows in figure 5, is based on the utilization of the MicroBlaze processor from Xilinx. The MicroBlaze processor is a soft-core 32 bits RISC architecture that can be include in Xilinx's FPGA as an IP module. MicroBlaze has Harvard architecture; this means separated data and addresses buses. It has several structures of buses of which we have used the LMB (Local Memory Bus) and the OPB (Peripheral On-chip Bus). Bus LMB allows accessing the internal FPGA memory in only a clock cycle. The disadvantage is in that the size of this memory (BRAM) is limited to the available resources in the FPGA device.

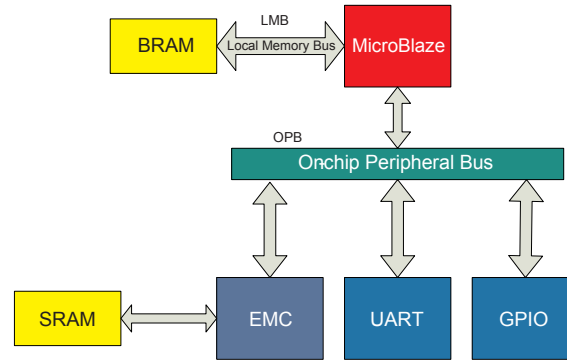

Fig. 5. Controller architecture

The processor communicates with the peripheral devices by an OPB (On-chip Peripheral Bus) IBM's standard bus. The peripheral ones are mapped in the memory address space. This means that the inclusion of new peripheral consists simply of connecting it to the OPB bus and including it within the memory address space of the processor. In our application only the three devices shown in figure 5 have been required. The peripheral devices correspond to a parallel port (*gpio*), a serial port (*uart*) and a memory controller (*emc*). The parallel port receives information of the push-buttons of the board and generates the signals to the 7-segment displays. The serial port allows realizing the transmission of information towards the X10 module. The external memory controller facilitates the transfer of information with the SRAM memory that contains the application software.

In figure 5 it is possible to observe that the system has two RAM memory blocks: BRAM and SRAM. BRAM is the FPGA internal RAM memory whereas the SRAM is the FPGA external memory module. The employment of both memories justifies itself due to the fact of the size of the software application requirements. BRAM module has better access time (one clock cycle), nevertheless its size is limited. Block SRAM can be expanded based on the system requirements. There are three expansion connectors in the development board to adapt any additional device such as a memory module.

B. Software application development

The software application has been implemented using C++ language. The XPS development environment provides a series of software packages developed in ANSI C that facilitate the codification of the user's applications (peripheral drivers, libraries and operating systems) [11].

Figure 6 shows the model-view-controller (MVC) of the software application describing the classes' graph. The pattern has been used in order to separate data and user interface. In that way it is possible to improve and expand the application in an easy manner. This model divides the application in three layers:

- View: it is the layer that transmits and receives information of the end user. In order to include new services it is necessary to add more classes that implement the interconnection between the final user and the devices. Two classes belong to this layer:
  DisplayDriverClass: it is the class that shows the results and information of interest in the 7-segment display.
  ButtonsDriverClass: it is the class that receives the push-buttons signals and transmits the information to the controller.

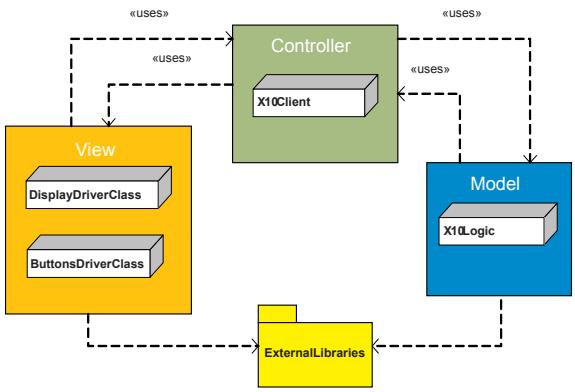

Fig. 6. Model-view-controller.

- Controller: in this layer the information is processed and transmitted up to the other two layers. For example, it receives the information of the pressed button and transmits the orders to the model layer, which will transmit the command to the target device through the serial port.
  X10Client: this class works as an adapter, reducing responsibility to the classes of the other two layers and improving the model of the system. This way a major independence is obtained between all the classes facilitating later modifications.
- Model: in this layer the whole application logic is carried out. It is the one that establish the communication with the X10 devices through the serial port.
  X10Logic: it is the class that contains all the set of necessary methods to establish the communication with the X10 interface through the serial port, among other functionality, there is the communication protocol of this technology. Figure 7 shows the code of the typical interaction between X10 interface and the development board.

CONCLUSIONS

This communication describes a system capable of controlling the devices of X10 technology distributed in a domotics space using an FPGA device. One of the objectives has been that the system is expandable and modifiable, so that it is possible in the future to add new functionalities and services to the end user. This will allow to incorporate new external devices in order to increase the functionality and improving the user interface (numeric keypads, LCD screens, tactile screens, etc). The system presented in this communication constitutes a development platform focused on economic solutions and easy managing in the world of the home automation

```
…
//next transmission, it send the address A1
outputBuffer = _HEADER_;//header
while (!XUartLite_Send  (&uart, &outputBuffer, 1));

outputBuffer = _DEVICE_ADDRESS;//device address
while (!XUartLite_Send  (&uart, &outputBuffer, 1));

while (!XUartLite_Recv  (&uart, &inputBuffer, 1));
//check the checksum
if (inputBuffer == ((_HEADER_ + _DEVICE_ADDRESS)&0xff))
{
   //checksum ok
   //transmit ACK to the board
   outputBuffer = _OK_;
   while (!XUartLite_Send (&uart, &outputBuffer, 1));

   while (!XUartLite_Recv  (&uart, &inputBuffer, 1));
   for (j=0; j<20000000; j++);//wait for synchronize

   if (inputBuffer == 0x55)
   {
      //interface ready for the next transmission
      …
```

Fig. 7. Code example of the X10 transmission

ACKNOWLEDGEMENT

This work was supported in part by the Spanish Education and Research Council under grants no. TEC2005-04359/MIC and no. TSI2006-13390-C02-02, and by the Andalusia Regional Government under grants no. TIC2006-635 and no. TIC2141.

REFERENCES

[1] Renato Nunes, "Implementing Tiny Embedded Systems with Networking Capabilities", IADIS International Conference on Applied Computing 2005, Algarve, Portugal, February 2005.
[2] LonWorks (ANSI/EIA 709.1-A), http://www.echelon.com.
[3] KNX, http://www.knx.org/.
[4] HomePlug, http://www.homeplug.org/home.
[5] R. N. Bucceri, "The Latest Technology in Automated Home Control - Book System Design Manual Using X-10 & Hardwired Protocols", Silent Servant, Inc, 2003.
[6] X10, http://www.x10.com
[7] Universal Powerline Bus, http://www.pcslighting.com/upb/overview.html.
[8] F. Mateos, V. M. González , R. Poo, M. García, R. Olaiz, "Design and Development of an Automatic Small-Scale House for Teaching Domotics", 31st ASEE/IEEE Frontiers in Education Conference, Reno, NV-USA, 2001.
[9] X10, CM11A Interface Communication Protocol, ftp://ftp.x10.com/pub/manuals/cm11a_protocol.txt.
[10] Xilinx, Inc. Spartan-3 Starter Kit Board User Guide, http://www.xilinx.com/bvdocs/userguides/ug130.pdf.
[11] P. Anderson. "Xilinx Platform Studio Tutorial". Embedded Systems Design- Advanced Course Homepage. http://www.cs.lth.se/EDA385/. 2006.