International Journal of Computer and Information Technology (ISSN: 2279 – 0764)
Volume 04 – Issue 02, March 2015

“Catching LED” Game: An FPGA Developing Board Implementation

Horacio Fernandez-Jimenez
Electrics and Electronics Department
Veracruz Institute of Technology
Veracruz, Mexico

Ricardo F Martinez-Gonzalez
Electrics and Electronics Department
Veracruz Institute of Technology
Veracruz, Mexico
Email: imag [AT] itver.edu.mx

Paz V Garcia-Hernandez
Electrics and Electronics Department
Veracruz Institute of Technology
Veracruz, Mexico

Antonio Dominguez-Cedillo
Electrics and Electronics Department
Veracruz Institute of Technology
Veracruz, Mexico

Abstract—**This work is about designing and implementation of a game kind “Simon says”. Game system was divided, division provides advantages. Finding solutions for minor problems is easier than for mayor ones; giving as result that debugging process is faster than full-attack methods. Moreover, FPGA usage was a great improvement for digital designing, it has modular capability and VHDL programming is concise and clear. Finally, system was physically implemented and then it was tested by several first-time users, giving some unexpected conclusions. First, dexterity games depend not only of sophistication to be amusing but playability and well-done ideas. Second, it does not matter how good you could have planned a game, it can always be improved.**

Keywords—Game, FPGA developing board, VHDL

I. INTRODUCTION

One of the primordial decisions at electronic product designing time is to choice correct device. Some considerations for device selecting are: cost, features and designing method. The choice must be taken at first designing process stage; however, it has a great impact on covering range of project.

Field programmable gate array (FPGA) developing board has become an interesting option for rapid digital system development [1,2]. Developing board make easier debugging process, because reconfiguration just requires a few clicks.

FPGA is not only alternative for digital system designing; there are also Complex Programmable Logic Device (CPLD) [3,4] and microcontrollers [5,6]. Both alternatives have either advantages or disadvantages.

On next, some FPGA advantages over CPLD are presented [7]:

- A FPGA has a larger logic density than CPLD.
- A FPGA is cheaper by logic unity than CPLD, even though FPGA integrated circuit is more expensive.
- FPGA are able to implement more complex designs than CPLD, owing to larger logic density.
- FPGA has optimized embedded circuits for arithmetic functions, like counters, adders, multipliers, et cetera.

Now, some FPGA advantages over microcontrollers (µCs) are presented [8]:

- FPGA are able to make more operations by duty cycle than µC.
- FPGA has ability for parallel operation execution, speeding its data processing up.
- FPGA can be reprogrammed to change its functionality; otherwise, µC has a static architecture and instruction set.

Among FPGA developing boards, Altera ones have good features with low costs [9]. Altera Inc. releases online a completely free developing suite named Quartus II. It allows compiling, programming, simulating, even intellectual property (IP) elaborating [10] and also it has online help.

Altera offers three FPGA families; each family has its own specifications. Stratix is highest gamma family, then Arria family comes, and finally simplest one, Cyclone.

For project developing, DE0 Nano developing board was selected. It has a Cyclone IV EP4CE22F17C6N FPGA; this board is ideal for portable prototypes or projects, because of its light-weight, reduced size and no external power necessity.

There are several forms to determinate FPGA behavior; nonetheless, hardware description languages (HDLs) are most common and used option. From HDLs, VHDL and Verilog HDL are the most extensive [11,12,13]. Both languages are open and standardized by Institute of Electrical and Electronics Engineers

(IEEE); VHDL on 1076 and 1164 IEEE standards and Verilog on 1364 one.

This project uses VHDL, due to its versatility; it allows determinate proceedings and functions location for any design unit availability, also VHDL has libraries, those concepts do not exist on Verilog. Furthermore, VHDL is a very concise and robust language, making to write its code simple and objective; on the contrary, Verilog compiler uses to omit errors; and later, such errors show up [14].

II. Game development

"Catching LED" game is kind "Simon says". The main objective of it is to direct a joystick where LED is ON; in other words, catching LED is ON. If user catches LED, so user's score rises. LEDs are distributed surrounding joystick on distribution presented in Figure 1.

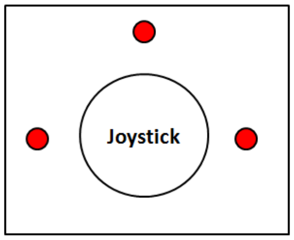

Figure 1. LEDs distribution

Following game explanation, LED will be on just for an established time, and then another one will and so forth. Game has a scoreboard implemented by two 7 segment displays. Moreover, game has two duration times, 10 and 30 seconds; also it has difficulty, establishing LED-ON frequency.

For a better developing control, project was divided in several parts.

A. Game time selector

This component is in charge for establishing game time. There are two options, 10 and 30 seconds; so two entities are needed. A clock pulse counter determinate time, developing board has a 50 MHz quartz crystal, which is used for clock pulse generation.

Calculating for period determination,

$$\frac{1}{50\ MHz} = 2\ ns$$

now to obtain 10 and 30 seconds, next calculations were made

$$\frac{10\ s}{20\ ns} = 500 \times 10^6 \quad \frac{30\ s}{20\ ns} = 1.5 \times 10^9$$

Both obtained quotients are used by accumulator entities, which start with a logic one in their outputs, and when their accounts reach corresponding quotient, entity toggle its output and account is reset.

Physically, time selection is made by a single action ON/OFF switch; and its position determines which entity is in charge, 10 or 30. Complete description is shown in Figure 2.

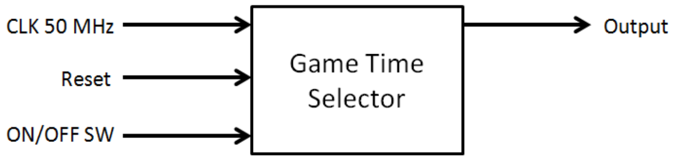

Figure 2. Make time selector diagram

B. LED-ON frequency selector

Selector controls LED time ON, for it a counter is used; when counter reaches certain value, selector puts a logic one on its output. For ON-time obtaining, a 32 bit counter is created and its positions 22 and 23 determinate function, just one at time, so two frequencies are possible. To calculate period next calculations were made

$$\frac{2^{23}}{50\ MHz} = 167\ ms \quad \frac{2^{22}}{50\ MHz} = 83\ ms$$

Once again, physical selection is made by a single action ON/OFF switch, and depending on it, a counter position takes control. Complete function of selector appears on Figure 3.

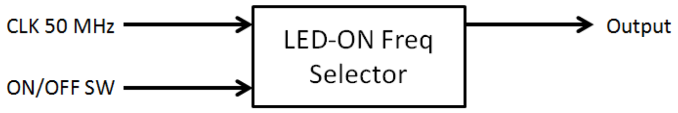

Figure 3. LED-ON frequency selector

C. Pseudo random sequence generator

Generator produces 3 bits sequences, it is implemented using a linear feedback shift registers (LFSR), this project has a 30 bit LFSR. First at all, a seed vector is planted on LFSR; seed is the first value and it determinates some generator features. Each 50 MHZ clock pulse, LFSR right shifts one position and XORs bit 1 and bit 30. XOR result is reintroduced in least position bit in LFSR. The entity output is three least significant bits from LFSR; The LFSR used for generator implementation is shown in Figure 4.

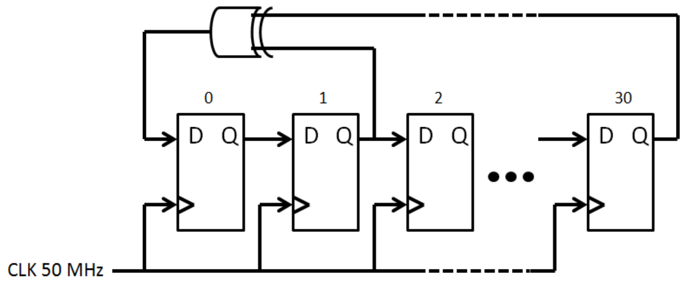

Figure 4. LFSR implemented for pseudo random sequence generator

The final part of the entity is an adjustment to synchronize this entity and frequency selector. Generator produces sequences each 20 ns, so when selector gives signal a sequence is caught.

D. Encoder

Pseudo random sequence is three bit long; however, just one LED must be ON at same time. In order to determinate which one, an encoder is implemented. Table 1 is truth table for the implemented encoder.

TABLE I. ENCODER FUNCTION

Input	Output
000	000
001	000
010	001
011	001
100	010
101	010
110	100
111	100

E. Joystick command catcher

Joystick actions switches, three in total. This input can occur at any time, so a register captures user's command. After it, command is delivered, for score determination.

F. Comparator

This component makes comparison between joystick command and encoder data; if both are equal, then comparator rises its output up, otherwise it falls its output down.

G. Scoreboard controller

Controller sums correct answers and controls scoreboard. It has two entities: "goods counter" and "display encoder". When comparator's output is high then counter increases its account. This counter is not binary but decade.

Making a decade counter eases next entity, display encoder; which encodes account for a pair of 7 segment displays. Both entities are connected as shown on Figure 5.

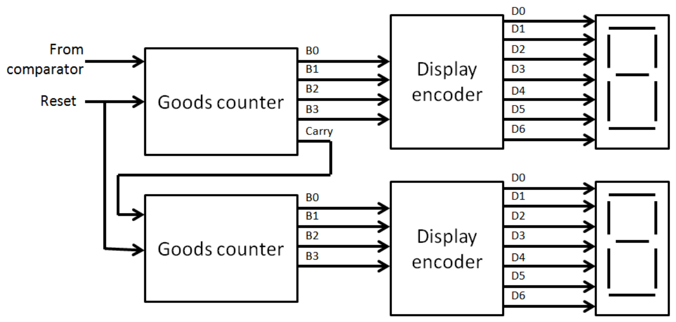

Figure 5. Scoreboard controller architecture

Putting every compound together, whole system is presented in Figure 6, for obvious reason multiple lines are replaced with a bus line.

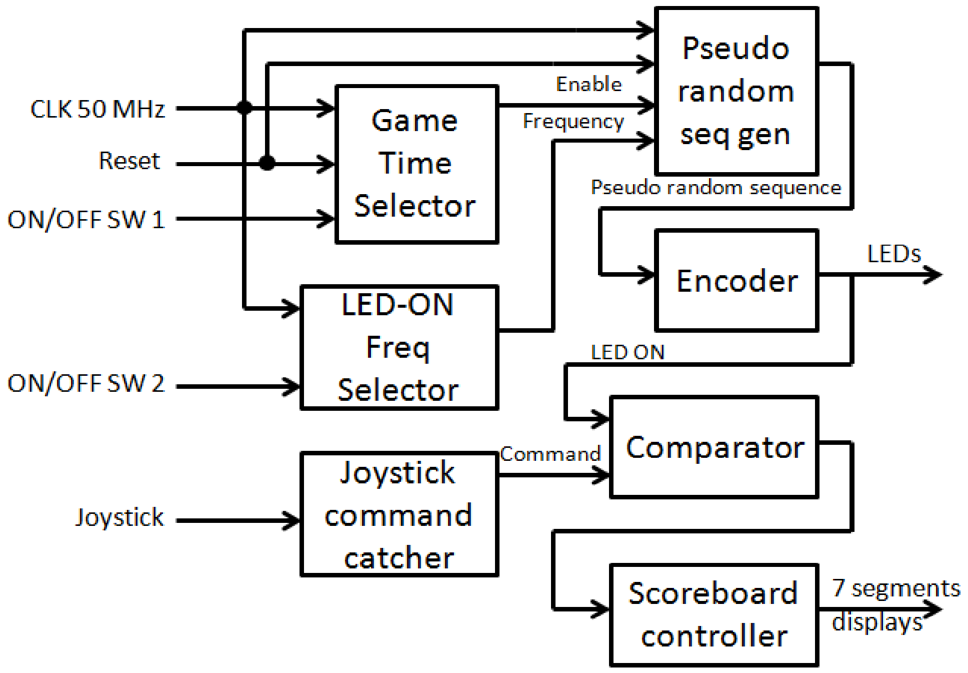

Figure 6. Complete system architecture

III. Obtained Results

Using the LFSR pseudo random sequence generator successfully works; some sequences are presented at Figure 7. When reset is high, seed vector is reintroduced to LFSR; and when it is low, LFSR starts working. Its good function can be evaluated after output does not have symmetry.

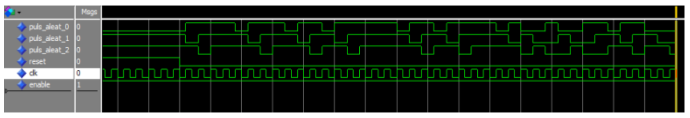

Figure 7. Pseudo random sequence generator output signals

Encoder was also simulated, and Figure 8 shows how it only has one bit in high level, and other one is low.

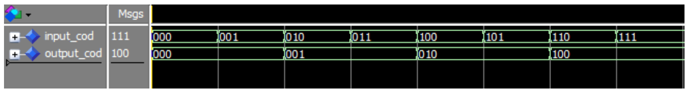

Figure 8. Output encoder

Goods counter entity counts from zero to nine; it also generates its corresponding carry signal. With its good functioning, display encoder would not have problems. Test outputs of good counter entity appear in Figure 9.

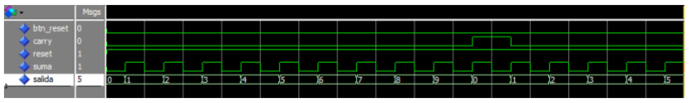

Figure 9. Good counter entity's outputs

Timers were tested using a chronometer and they were exact. Finally, a game does not work if it does not entertain, so it was physically implemented, see figure 10, and tested. During an Institutional Expo, prototype was tested by various persons, giving some results. First, prototype works as it was expected. Second, difficulty selector needs improvements; easy is too easy and hard is very hard; some intermediate levels need to be developed. Third, timer needs more levels. As last, user experienced an amusement moment and they were very interested in take part of test.

IV. Conclusions

It is possible to design a digital system dividing it in subsystems or components. Each component is simpler than computer system so developing and testing is easier.

LFSR works as pseudo random sequence generator algorithm, and in conjunction of encoder entity, makes an apparently random system. LEDs ON did not follow any pattern, making the game funnier.

Despite of its simplicity, the game is very amusing. Even though its design considerations, the most qualified opinion, is which comes from players. Some updates needs to be done; however it was a good first shot.

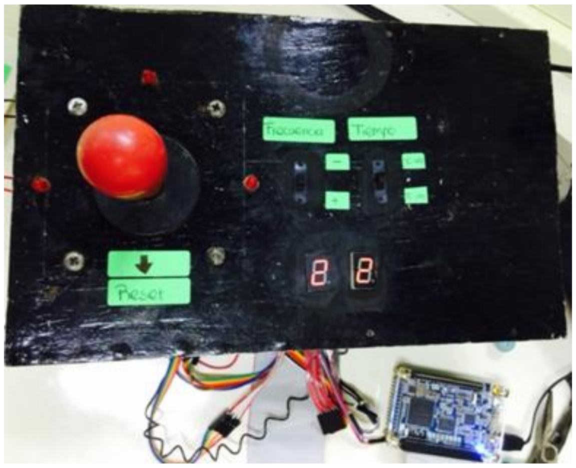

Figure 10. Physical implementation of project

REFERENCES

[1] Kao, C. (2005). Benefits of partial reconfiguration. *Xcell journal*, *55*, 65-67.

[2] Dongale, T., Magdum, S., Goilkar, K., Chougale, N., & Ghatage, S. R. (2012). FPGA Implementation of a PID controller for dc motor controller application. *proc. IJAIR*.

[3] Lakeou, S., Ososanya, E., Latigo, B. O., Mahmoud, W., Karanja, G., & Oshumare, W. (2006, September). Design of a low-cost solar tracking photo-voltaic (PV) module and wind turbine combination system. In *21st European Photovoltaic Solar Energy Conference* (pp. 4-8).

[4] Zeller, J., Zhu, M., Stimac, T., & Gao, Z. (2001, July). Nonlinear digital control implementation for a dc-to-dc power converter. In *INTERSOCIETY ENERGY CONVERSION ENGINEERING CONFERENCE* (Vol. 1, pp. 205-210). SAE; 1999.

[5] Zwavashe, T., & Duri, R. (2014). Integrating GSM and Zigbee Wireless Networks for Smart A2 farming Enterprises in Zimbabwe. *International Journal of Science and Research, Vol3, Issue6*.

[6] Laizans, K., Sünter, I., Zalite, K., Kuuste, H., Valgur, M., & Tarbe, K. (2014). Design of the fault tolerant command and data handling subsystem for ESTCube-1. In *Proc. Estonian Acad. Sci* (Vol. 63, pp. 222-231).

[7] Brown, S., & Rose, J. (1996). Architecture of FPGAs and CPLDs: A tutorial. *IEEE Design and Test of Computers*, *13*(2), 42-57.

[8] Parnell, K., & Bryner, R. (2004). Comparing and contrasting FPGA and microprocessor system design and development. *White Paper, XILINX Corporation*.

[9] Leventis, P., Chan, M., Chan, M., Lewis, D., Nouban, B., Powell, G., ... & Costello, J. (2003, November). Cyclone™: a low-cost, high-performance FPGA. In *Proceedings of the IEEE Custom Integrated Circuits Conference* (pp. 49-52). IEEE; 1999.

[10] Strell, S. (2006). Increasing Productivity with Altera Quartus II to I/O Designer/DxDesigner Interface.

[11] Swan, S. (2001). An introduction to system level modeling in SystemC 2.0. *Cadence Design Systems, Inc., draft report*.

[12] Golson, S. (1993, March). One-hot state machine design for FPGAs. In *Proc. 3rd Annual PLD Design Conference & Exhibit* (Vol. 1, No. 3).

[13] Storaasli, O., Yu, W., Strenski, D., & Maltby, J. (2007). Performance evaluation of FPGA-based biological applications. *Cray User Group*.

[14] Smith, D.J., (1996). VHDL and Verilog compared and contrasted-plus modeled example written in VHDL, Verilog and C. Design Automation Conference Proceedings 1996, 33rd, vol.3, no.7, pp.771-776.