Design of an automated railway crossing system with Verilog language in CPLD

M Zhilevski\(^1\) and V Hristov\(^1\)

\(^1\) Faculty of Automatics, Technical University of Sofia, Bulgaria
E-mail: mzhilevski@tu-sofia.bg

Abstract. In this paper the basic requirements for the automatic railways crossing devices are formulated. A block diagram, which consists of elements that make the automatic railways crossing systems are presented and their working principles are discussed. Based on the hardware description language - Verilog, a digital railway crossing control system is designed and is implemented with complex programmable logic device, taking into account the requirements. As an opportunity for practical application the controller CoolRunner II could be used. Simulation results with set input parameters are shown to represent the functionality of the designed system. The results obtained can be used for design, implementation, simulation and development of such types of systems in railway crossing automatics.

1. Introduction

The rail transport is one of the most energy efficient means of mechanized land transport, characterized by relatively high safety, comfort and cheapest modes of transportation. As a result of these benefits, rail transport is a major form of public transport in many countries [1, 2, 3].

But accidents related with railways are more dangerous than other transportation accidents. Therefore more efforts are necessary for improving safety. There are many railway crossings without automatic systems. Hence many accidents occur at such crossings since there is no one to take care of the functioning of the railway gate when a train approaches the crossing. That’s why different automated railway crossing systems are used [4, 5].

The railway crossing represents the passing through a railway line of a road for other road transport, tram and trolley lines, as well as with other railway lines. The safety of traffic through crossings for both rail and road transport is achieved by means of automatic railway crossing devices (ARCD). They provide maximum possibilities for the two modes of transport with a minimum time to close the crossing before the railroad vehicle arrives and open it immediately after the clearance zone of the last axle is released [1, 2].

The complex programmable logic devices (CPLD) like the controller Xilinx’s CoolRunner II are appropriate option in designing and implementing a control of many different types of electromechanical objects [6, 7, 8, 9]. These devices comprises of a group of programmable functional blocks. The inputs and the outputs of these functional blocks are connected together by a global interconnection matrix. This interconnection matrix is reconfigurable, so that we can modify the contacts between the functional blocks. There will be some input and output blocks that let us unite CPLD to the external world. Thanks to the hardware description language like Verilog, it is relatively easy to create digital schemes for control. The main advantage of Verilog is that it allows the system to be described (modeled) and verified (simulated) prior to implementation [10, 11, 12, 13].
In this paper the main requirements for the automatic railway crossing devices are formulated. The schematic of the elements that make the automatic railways crossing devices are presented. Based on the language Verilog, an automated railway crossing control is designed and implemented with complex programmable logic device. The CPLD controller is used like practical application for control of barriers, signaling to road vehicles, and so on. Some simulation results of railway crossing systems are shown and discussed.

2. Automated railway crossing devices
The automated railway crossing devices are the main elements used in designing and implementing the railway crossing system.

2.1. Basic requirements
The control system of the ARCD must be located in a cabin or other suitable room and protected from unauthorized access and inappropriate weather. ARCD apparatus executes the algorithm of action and performs control functions of all components. It can be implemented with electronic and/or relay elements [14].

The supply voltage required for the normal functioning of the ARCD is provided by two independent sources (industrial network and contact network) that switch automatically to ensure the safe functioning of the ARCD.

The basic requirements for the technical parameters of the ARCD can be formulated as follows [15]: to be connected with a special technical specification; power supply must be with the following parameters: 220V, 50Hz; to meet the safety level requirements - not lower than SIL3; the time of turning on the red flashing light of the cross railway traffic lights (CRTL) after an overlay closing command should be 1 second; the time of the closing command should be 6 ÷ 9 seconds after switching on CRTL; horizontal position of the beams should be 16 seconds; the ARCD contains a diagnostic system – an information board, which provides details for the position of the railways at one of the close stations.

The basic elements and the principle of operation of a railway crossing system must be taken into account when designing, implementing and simulating.

2.2. Main blocks
Main blocks of the automatic railway crossing devices are presented in Fig.1. Basic notations are as follows: CE – cabin for equipment; CRS – cross railway signals; 1, 2, 3, 4 – traffic sensors (axle counters); BM – barrier mechanisms; BS1 and BS2 – block sections.

Figure 1. Main blocks of the automatic railway crossing devices.

The cabin for equipment is used for accommodation and installation of relays, fuses, rectifier, converters, axle counters, etc. related to ARCD.
The Cross Railway Signals are two pieces situated on the left and right side of the train. When a railway vehicle is crossing the traffic lights, they should be lightening consecutively with a red light of $60 \pm 5$ flashes per minute and the audible alarm should be activated.

Four traffic sensors are used and the railway vehicle should activate the ARCD for at least 30 seconds before entering the area of the railways. The axle counters are used to control the movement of railways before and after the passing of the vehicle.

One configuration of a railway crossing is shown in Fig. 2.

![Figure 2. Configuration of the railway crossing.](image)

2.3. Operation principle

Traffic information on the train is transmitted by four road sensors. The used Xilinx CoolRunner-II controller is used to address the operational-technical requirements for the automatic overlay devices.

2.3.1. Output state of the apparatus

In the non-active state of the ARCD when a railway vehicle is not in the area of the railways, the traffic of vehicles is allowed - road traffic lights blink in white - 30 pulses per minute at a pulse / pause ratio of $1:1 \pm 10\%$ [16], the barriers are in the vertical position ($85 \div 90^\circ$ to the horizontal). This informs the driver that there is no train in the area of the overturning zone.

The controller receives information about: free block sections BS1 and BS2; control of road traffic lights; superior control of the barrier mechanisms and integrity of the beams. It provides information that the traffic is permitted when there is a white flashing light on CRS.

2.3.2. Automatic closing of the level crossing

Upon the arrival of the railway vehicle over the first axle of the railway, a command from CoolRunner-II is generated to close the level crossing. The white traffic lights switch to red lights; the buzzers in CRTL turn on; after 6 seconds a command is generated to close the overlay and the barriers are lowered to horizontal position (maximum 16 seconds); after obtaining a lower control of the barrier mechanisms, the buzzer is turned off.

2.3.3. Open the railway crossing

When the railway vehicle leaves the crossing (passes a counting point) the overhead device opens for the road vehicle - the barrier beams are raised to the vertical position (max. 16 seconds) after releasing the area from the railroad vehicle. Upon receiving upper control of the barrier mechanisms, the red flashing lights of the CRTL switch to white ones.
3. Synthesis of the control program

The synthesized program in Verilog is divided into four main parts: defining inputs and outputs; determination of registers, parameters and clock generator; initialization, clock generators and axle counters; logic control.

Some of the used inputs and outputs are: “clk” – an input for generating a clock signal; “in1”, “in2”, “in3”, “in4” – 1-bit inputs for axle counters; “barriers_up” and “barriers_down” – 1-bit inputs for extreme top and bottom positions respectively; “lights_error” and “barriers_error” – 1-bit inputs showing that there is a fault with the lights and barriers respectively and others.

Some of the used registers, parameters and clock generator are: \texttt{reg [7:0] count1}, \texttt{reg [7:0] count2} – 8-bit registers for internal counters 1, 2 respectively; \texttt{reg r_barrier1}, \texttt{reg r_barrier2} – 1-bit registers related to the outputs „barrier1“ and „barrier2“; \texttt{reg r_white1}, \texttt{reg r_white2} – 1-bit registers related to the outputs „white1“ and „white2“.

The first part of the Verilog code development initializes the signals in such way that when the program starts it will be in an open state and without any errors:

```verilog
initial
begin
  count1<=0;
  count2<=0;
  clk_lights<=0;
  clk_lights_delay<=0;
  clk_sounds<=0;
  clk_sounds_delay<=0;
  r_barriers_alert<=0;
  r_lights_alert<=0;
  r_barrier1<=0;
  r_barrier2<=0;
end
```

The next part of the code generates a clock generator that is used to control the flashing of the lights from CRS:

```verilog
always @(posedge clk)
begin
  if(clk_lights_delay==149)
    begin
      clk_lights<=~clk_lights;
      clk_lights_delay=0;
    end
  else
    begin
      clk_lights_delay=clk_lights_delay+1;
    end
end
```

With each increase of the signal “clk”, the variable "clk_lights_delay" will increase by 1 until it reaches the set value in the “if” block, which in this case is 149. Then "clk_lights_delay" will reset and the clock generator "clk_lights" will change its status from logical "0" to logical "1" or inverse.

In a similar way the creation of a clock generator is realized, which is used to control the signaling sound and axle counters in both directions.

The logic of operation of the railway crossing system is described by the following Verilog code:

```verilog
always @(posedge clk)
begin
end
```
if (lights_error==0 && r_barriers_alert==0)
begin

For the implementation of the opening of the railway crossing the following code is used:

if (count1==0 && count2==0)
begin
r_barrier1<=0;
 r_barrier2<=0;
 r_red11<=(barriers_up)?0:clk_lights;
 r_red12<=(barriers_up)?0:~clk_lights;
 r_red21<=(barriers_up)?0:clk_lights;
 r_red22<=(barriers_up)?0:~clk_lights;
 r_white1<=(barriers_up)?clk_lights:0;
 r_white2<=(barriers_up)?clk_lights:0;
 r_sound1<=0;
 r_sound2<=0;
end

When the signals "count1" and "count2" are in logical "0", this means that the railway crossing is free and the passage of vehicles is allowed. The signals "barrier1" and "barrier2" become in logical "0", which means that the barrier mechanisms have been signaled to be opened. Until logic 1 receives the signal "barriers_up", the lights "r_red11", "r_red12", "r_red21", "r_red22" flash with the clock frequency "clk_lights", and the lights "r_white1" and "r_white2" are off. When the signal "barriers_up" becomes in a logical "1", it means that the barriers are in the highest position and the vehicles can pass. Then the lights "r_red11", "r_red12", "r_red21", "r_red22" go out and the lights "r_white1" and "r_white2" flash with the frequency of the clock generator "clk_lights", which indicates to the vehicles that they can pass freely. The buzzers "sound1" and "sound2" are off in this process.

For the implementation of the closing of the railway crossing the developed code is similar like in the opening of the railway crossing.

Protection against damage to light signals and barrier mechanisms has been synthesized in Verilog code.

4. Simulation results
In the Verilog Test Bench Code a 4-axle passing locomotive is simulated in one direction. Some results from the computer simulations are presented in the next figures.

![Figure 3. Initial initialization.](image-url)
Fig. 3 presents the results obtained in the initiative initialization of the program. At this initialization of the program the railway crossing is open. Axle counters are reset. Barriers are lifted. Feedback is obtained that the barriers are in the extreme top position. The white lights blink with 30 pulses per minute at a pulse/pause ratio of 1:1 ± 10%. Red lights and buzzers are off. Lights and barriers as well as their alarms are turned off.

Fig. 4 shows the results obtained when the railway crossing passes from open to closed state. When the locomotive enters the section in one direction of the movement, the corresponding counter increases its value by as many as the number of wheels passed. The counter is different from 0. The white lights stop flashing. The red lights start flashing with 30 pulses per minute in a pulse/pause ratio of 1:1 ± 10%. After 6 seconds the barrier signal is switched. While the barriers are closed, the buzzer starts sounding. When feedback is received that the barriers are in the extreme lower position, the buzzer stops sounding.

Fig. 5 presents the results obtained during the passage of the crossing from the closed to the open position. Upon exiting the locomotive from the section on one direction of movement, the corresponding counter decreases its value by as many as the number of the wheels passed. When the counter is set to 0, a barrier signal is displayed. When the feedback is received that the barriers are in the extreme position, the red lights stop blinking. The white lights start flashing with 30 pulses per minute at a pulse/pause ratio of 1:1 ± 10%. The buzzers are turned off.
Fig. 6 shows the results obtained in case of failure in CRS.

![Table](image)

**Figure 6.** CRS failure.

The RTL Schematic has been generated using Xilinx ISE Version 14.6. Fig. 7 shows RTL schematic of the design railway system. This schematic shows the hardware generated inside the complex programmable logic device. The design is successfully implemented on CoolRunner II board, after testing the design by simulation.

![Diagram](image)

**Figure 7.** RTL schematic of the design railway crossing system.

5. Conclusion

Complex programmable logic devices are used for high performance and different control applications. They can be applied in digital designs to perform the functions of boot loader. CPLD are used: for loading the configuration data of a field programmable gate array from non-volatile memory;
many applications like in cost sensitive, battery operated portable devices due to its low size and usage of low power.

In this paper an automated railway crossing system is designed and implemented through a complex programmable logic device-CPLD based on a hardware programming language - Verilog HDL. The controller of Xilinx company is used to industrial control barrier mechanisms, vehicle alarms, etc. Some results from the simulation are shown and discussed. The results obtained can be used for the design, implementation, simulation and development of such types of systems in railway crossing automatics.

Acknowledgments
The authors would like to thank the Research and Development Sector at the Technical University-Sofia for the financial support in this conference.

References
[1]. Enokela, J., K. L. Oladejo, An Automated Railway Traffic Station Control System, Nigerian Journal of Technology (NJORTECH), Volume 36, No. 1, pp. 138 – 147, 2017.
[2]. Leena G., C. Vidawat, N. Jha, Automatic Railway System, International Journal of Computer Applications, Volume 159, No 8, pp. 30-33, 2017.
[3]. Yordanov, R., Miletev, R., Damyanov, Il., Ionchev, E., Measurement and analysis of the road traffic, Proc. XXVII International Scientific Conference Electronics - ET2018, 2018.
[4]. Al-Zuhairi, A. S. M., Automatic Railway Gate and Crossing Control based Sensors & Microcontroller, International Journal of Computer Trends and Technology (IJCCT), Volume 4, Issue 7, pp. 2135-2140, 2013.
[5]. Krishna, S. Yadav, Nidhi, Automatic Railway Gate Control Using Microcontroller, Oriental Journal Of Computer Science & Technology, Volume 6, Issue 4, pp. 435-440, 2013.
[6]. Marella, V., CPLD Based Design and Implementation of Pipelined 32-bit RISC Processor with Floating Point Unit, International Journal of Emerging Technology and Advanced Engineering, Volume 5, Issue 12, 2015.
[7]. Hristov, V., M. Zhilevski, Extension of Programming Possibilities for Programmable Logical Devices through Matlab, Proceedings of Technical University of Sofia, Volume 68, Issue 2, pp. 79-88, 2018.
[8]. https://www.xilinx.com/publications/products/cpld/coolrunnerii-product-brief.pdf.
[9]. https://www.elprocus.com/complex-programmable-logic-device-cpld-architecture-applications/.
[10]. Samir Palnikar, “Verilog HDL – A Guide to Digital Design and Synthesis”, 2nd Edition, IEEE 1364-2001 Compliant, published by Pearson Edition.
[11]. Pavlitov, C., Blocks and Elements of Electrical Drives Implemented by Programmable Logic Devices, Monograph, Sofia, 2013.
[12]. Hinova, A., Synthesis of logic circuits with programmable logic controller, KNOWLEDGE – International Journal, Natural and Biotechnical Sciences, Volume 31.3, pp. 747-752, 2019.
[13]. Zhang Yi, Guo Xiaojuan, Development of intelligent traffic control system based on FPGA and single chip microcomputer technology, Journal of Chemical and Pharmaceutical Research, 2014, 6(6):1156-1159p.
[14]. Damyanov, Il., Investigation and analysis of transit road traffic through specific residential area with the use of an integrated system of mobile lpr cameras, Academic Journal of Mechanics, Transport and Communications, Volume 17, Issue 1, 2019.
[15]. Regulation No. 4 for the rail crossings, Railway Administration Executive Agency, Mart 1997.
[16]. Technical specification – “002-2005-Automatic level crossing”, National Railway Infrastructure Company.