



PERFORMANCE AND AREA IMPROVEMENT OF PROGRAMMABLE LOGIC CONTROLLER (PLC) USING SYSTEM ON PROGRAMMABLE CHIP (SOPC)

Vilas V Deotare¹, Arjun J. Khetia², Dinesh V Padole³

^{1,2}Research Scholar, ³Professor, GHRCE,

^{1,2,3}Electronics and Telecommunication,
Nagpur, Maharashtra, INDIA.

Abstract: Programmable Logic Controller (PLC) is most important component in modern manufacturing automation systems. Processor and software have been bottlenecks for improving operational efficiency of PLC. To enhance the PLC performance and area, a new design called "Design And Implementation Of PLC Based on FPGA" has been proposed here. The design has been simulated using knowledge of FPGA design tools (such as HDL), to translate, integrate and implement the logic circuit in FPGA using condensed simultaneity graph theory (CSG) to optimize the design. PLC using system on programmable chip is based on parallel mechanism, so the proposed design works faster as compared to the conventional PLC. Implementing control logic with FPGA to replace a PLC, response time is improved while sustaining the flexibility of control. In FPGA side the 4 stage RISC pipeline based on RUNG- Architecture is designed for dynamically forming and controlling each rung. This design also provides increased flexibility, increased reliability, 60% less area utilization, better performance and 30% faster scanning time. The design also provides the GUI (Graphical User Interface) connectivity based controlling of designed PLC for the ease of monitoring and controlling the PLC and its applications from the control center. We can easily monitor and control different devices from Personal Computer (PCs). This discusses the implementation of FPGA based PLC that can be embedded into devices, machines & systems using suitable interface for industrial applications as well as for home applications. It provides Real-Time Speedy Operations and FPGA based Real-Time Control Systems with a capability to run multiple control tasks for reliable developing, monitoring and controlling of systems. The design results into highly cost-effective, compact and simple solution.

Keywords: Programmable Logic Controller (PLC), System on Programmable Chip (SOPC), Automation, Very Large Scale Integration System (VLSI), Condensed Simultaneity Graph theory (CSG).

I. Introduction

Before the introduction of programmable controllers, relays were commonly used to implement control systems. With the advent of low-cost microprocessors, Programmable Logic Controllers (PLCs) replaced relays as the predominant control element. As part of their heritage as relay replacements, PLCs are often programmed using a technique originally designed for relay networks - Ladder Diagrams (LD). Programmable logic controller (PLC) has been widely used for the logic control in the manufacturing systems, in commercial and industrial applications. It is difficult to adapt to the requirements of high-speed control in modern industry. Therefore, figuring out a way to realize high-speed control becomes more and more important. Field programmable gate array (FPGA) has characteristics of supporting high-speed parallel execution and hardware configuration. Realizing the function of PLC by it can greatly improve the execution speed of control logic and this is an important way to solve the current problems of PLC.

To describe the logic sequence, a ladder diagram (LD) is extensively used to program PLC applications. It is well known that PLC has a typical software-based implementation structure in which the microprocessor decodes and executes the LD program in a sequential and cyclic way. As a result, PLC performance is limited by the cyclic scan period which depends on the program length and the microprocessor's processing speed. The core of implementing PLC based on FPGA is how to convert IEC61131-3 standard language into hardware descriptive language. Ikeshita converted SFC to Verilog language [4]; Ichikawa implemented the research and conversion to the framework of instruction table translation into HDL [5]. But [4, 5] documents don't implement the conversion of the LD that is widely used, and also have indirect and hidden problems.

Here we will show how FPGA technology can make “rungs parallel” execution of ladder logic commercially feasible. Industry is reluctant to consider new technologies such as field programmable gate arrays (FPGAs) for industrial process control applications. The primary reason is that typically, manufacturing plant engineers have no training in digital logic design. Therefore, it is not feasible to require these applications to be programmed directly in Boolean algebra. Even if automatic translation from relay ladder logic to Boolean algebra is done [IO], implementation of relay ladder logic on a conventional FPGA would require the plant engineers to be able to use conventional tools for FPGA synthesis, and to interpret the intermediate steps of the synthesis.

Another considerable barrier to the use of conventional FPGA architecture for PLC logic is the need to guarantee correctness of a complex mapping. In FPGA design, parallel execution exists not only in one combinational logic operation but also in the multiple combinational logic operations. The design also provides the GUI (Graphical User Interface) based controlling of designed PLC for the ease of controlling the PLC and its applications from the control room also. We can easily monitor and control different devices from Personal Computer (PCs).

II. System Model

A. Hardware framework

The hardware framework of FPGA-based PLC is shown in Figure 1. This framework includes FPGA Spartan 3E-chip, I/O unit, A/D module, reset circuit, power supply circuit, program memory chip EPCS4, JTAC download port etc. ISA interface is used for communication with host computer, I/O unit and A/D module are used for acquisition of digital and analog signals.

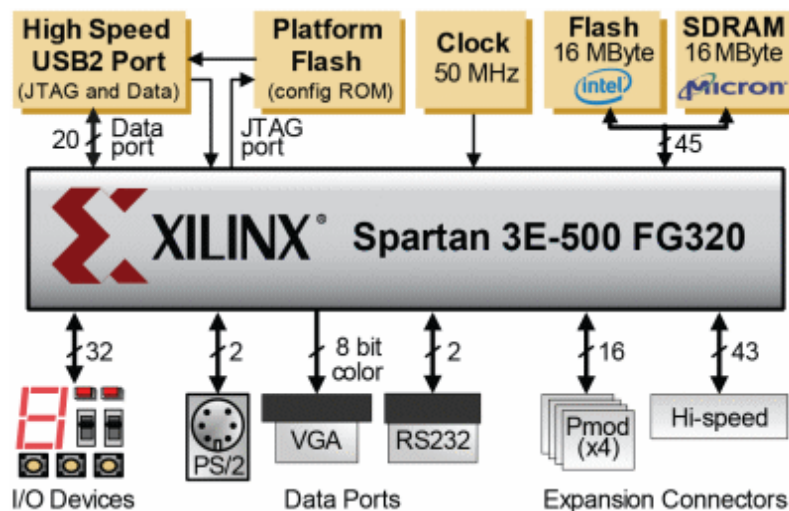


Figure 1 FPGA-based PLC Hardware Framework.

B. Software framework

This VHDL consists of two modules: One is ISA communication decoding module, which packages address decoding and latches data into the interface package. The other is generation module of VHDL program that has equivalent function with the ladder diagram [7-9]. The difficult and key technology is how to convert the ladder diagram into VHDL.

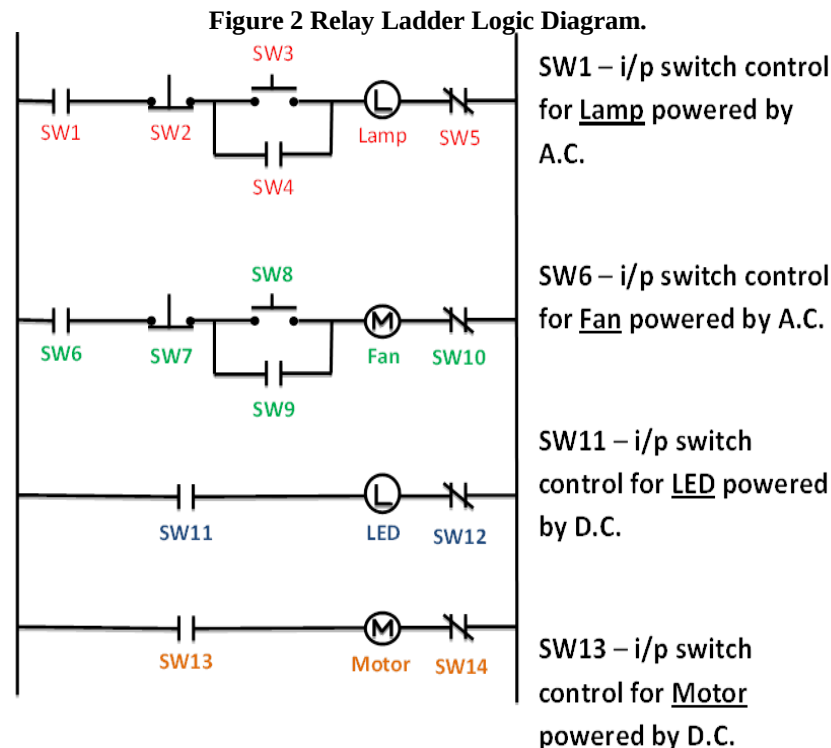
The Hardware/Software co-design Technology results in better solution providing following advantages:

- **Flexibility:** Design engineers (PLC manufacturers) can easily upgrade the FPGA-PLC designs. For example some features or instructions could be added in the existing design, by changing Hardware Description Language (HDL) and configuring the same FPGA chip for the modified design of PLC.
- **Accuracy:** Design being fast enables engineers to incorporate extremely time-critical functions to hardware such as limit and proximity sensor detection and sensor health monitoring resulting in more accurate solutions.
- **Short product development cycle:** Due to the use of standard HDLs and automated design tools design time is much reduced. Also the control code runs directly in silicon, engineers can try out various implementations.
- **Cost effective and compactness:** Due to above advantages designer can meet the market requirements by satisfying customer needs, increase the performance or functionality of the product. Hence result in high performance, low cost and compact designs as compared to the other available solutions.

C. Relay Ladder Logic

Relay ladder logic emulates a synchronous electromechanical relay panel. It pictures control logic as a one-way flow from a left rail of the ladder across its rungs toward coil elements along the right rail. Flow along a rung is passed or cut off by a series of relays (contacts), effectively defining logical products. Vertical paths between rungs allow flows to merge, effectively summing the rung products. All ladder logic dialects describe this same basic form.

Figure 2 illustrates the graphical symbols used in programming relay ladder logic. Contacts are of two kinds. A normally open contact passes the flow when activated (i.e., closed), effectively inserting its variable into a product term. A normally closed contact passes the flow unless activated (i.e., opened), effectively inserting the complemented variable into the product. Coils are state variables. A set of adjacent rungs connected by vertical paths is called a network. On each cycle of operation, known as a scan, the flow for all networks is computed. Flow on an individual network implements a Boolean function for one state or output variable.

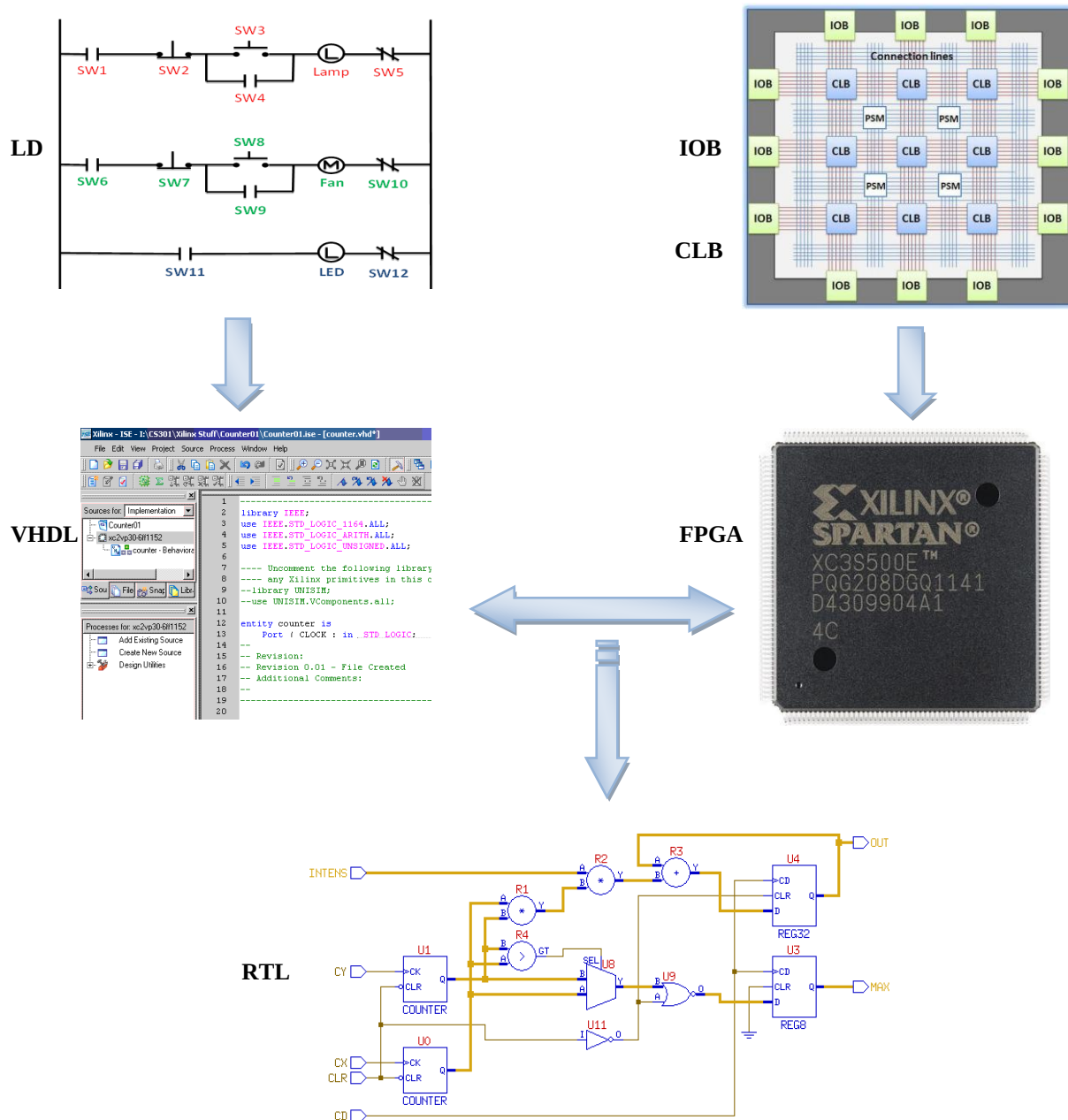


III. Implementation Of FPGA-Based PLC

Since the internal structure of FPGA is reconfigurable with input/output block and configurable logic block, the required circuit can be built as long as a ladder diagram is converted into RTL architecture and downloaded to the FPGA chip. Such an implementation will perform the same functions as the original PLC ladder diagram, but not in the traditional sequential cyclic scan manner. The new solution can respond to the input signals with parallel execution at hardware processing speed, which dramatically improve the PLC speed. Moreover, it can be reconfigured any times as a new PLC program is converted and downloaded. As for the RTL architecture of FPGA implementation, it is a low-level description of internal logic circuit of FPGA.

Several commercial software tools are available for FPGA development. The typical one is Xilinx ISE, which can establish RTL architecture by synthesizing a high-level program such as VHDL. Figure 1 shows the approach to implement a FPGA-based PLC from the ladder diagram. There are two steps to construct FPGA-based PLC: (a) convert ladder diagram into VHDL program with sequential and parallel design and (b) synthesize the VHDL design and implement it to FPGA device. With the programmable hardware design technology such as PLD, FPGA, and complex programmable logic device, it becomes possible to design the discrete event control system based the hardware platform. FPGA-based PLC has been generally discussed by many previous researchers [2–8].

Figure 3 Implementation of FPGA-based PLC from ladder diagram.



IV. Optimization of ladder diagram program

The PLC performs its execution cyclically. It reads input status, executes logic and updates output status. It is called as a PLC scan. Scanning of ladder program occurs from left to right and top to bottom recurrently. Scanning of complete ladder program is ladder scan cycle. The condensed simultaneity graph (CSG) theory [3, 5] is used to realize the reorganization of the ladder diagram.

The process is as follows: The first step is to convert the LD into VHDL program. (1) Determine the basic elements of constructing the LD; (2) Implement the construction of LD and logical preservation; (3) Convert the LD into Boolean equivalence; (4) Distribute trigger signal and generate VHDL program. The second step is to compile and implement VHDL program in FPGA devices. (1) The VHDL program is taken as the input of design for functional simulation to complete the preliminary function detection; (2) Build logic net lists and take post simulation to check whether the results are consistent with the design; (3) Use the placement and routing function to configure the logic net lists and the bottom unit to the FPGA chip, and to correctly connect the components. (4) Feedback the delay information of placement and routing to the lists, and test whether there are timing violations by the timing simulation. (5) Generate data files, then the files are downloaded and tested in the FPGA.

	$\overline{A}\overline{B}$	$\overline{A}B$	AB	$A\overline{B}$
$\overline{C}$	0	1	1	0
C	0	0	1	1

$$\text{Load} = AB + BC + \overline{A}C$$

K-map for Software Notification Signal:

	$\overline{A}\overline{B}$	$\overline{A}B$	AB	$A\overline{B}$
$\overline{C}$	1	1	1	1
C	0	0	0	0

$$S_{\text{not}} = \overline{C}$$

K-map for Hardware Notification Signal:

	$\overline{A}\overline{B}$	$\overline{A}B$	AB	$A\overline{B}$
$\overline{C}$	0	0	0	0
C	1	1	1	1

$$H_{\text{not}} = C$$

There are two modes of operating a PLC

- (i) Program Mode: In this mode, the ladder program is loaded into the program memory of PLC.
- (ii) Run/Execution: In this mode, the PLC executes the ladder logic cyclically.

With the help of CSG optimization we get direct Boolean equation on which we can easily build the circuit diagram which is compatible to HDL programming. Figure 4 shows the translated and optimized circuit of PLC which can program on any FPGA with the help of Xilinx and controlled with user controlled software from personal computer.

Figure 4 Design Architecture of PLC optimized circuit.

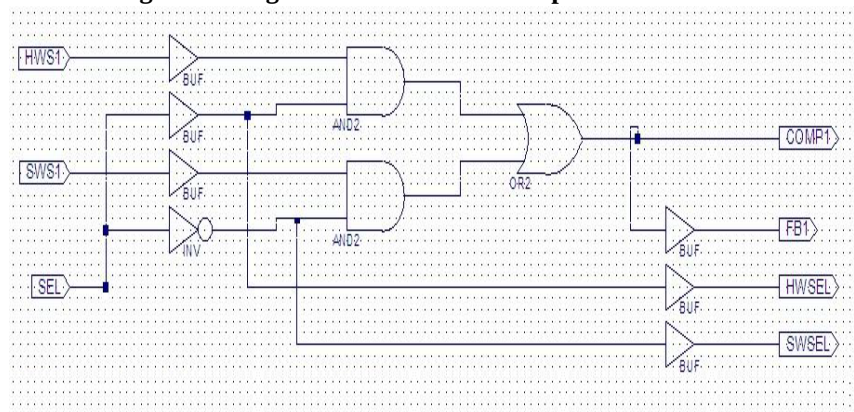


Table 1 shows the final optimized output table which is predicted before designing the PLC program to perform. Table represents the H(sw) as Hardware Enable Switch, S(sw) as Software Enable Switch, Sel as Selection line for selecting different output, O/P as final output pin, FB as feedback from each component, Hnot as Hardware Notification and Snot as Software Notification.

Table I PLC based on FPGA optimized output table.

H(sw)	S(sw)	Sel	O/P	FB	Hnot	Snot
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	1	1	0	1
0	1	1	0	0	1	0
1	0	0	0	0	0	1
1	0	1	1	1	1	0
1	1	0	1	1	0	1
1	1	1	1	1	1	0

V. HARDWARE OPTIMIZATION

It is observed that a large proportion of the FPGA hardware (more than 80%) was consumed by timers. Each timer requires several 32-bit registers plus associated logic. By comparison, each logic rung may require just a few single-bit logic gates. Therefore, with this co-design approach all the timer based instruction is directly given to GUI based software for handling. This way we are able to save near about 60 to 65% of FPGA hardware. The numbers of clock cycles taken by each implementation in evaluating the 4-rung application logic. These times exclude external communications overheads which are relatively constant across implementations. In most cases, the execution time varies with the input values. For some combinations of inputs, the software is able to optimize execution and thus evaluate the logic more quickly. The Interpreted HW implementation takes varying amounts of time due to communications interference. The execution time variation was not large – the minimum execution time was around 85-90% of the worst-case time.

VI. MATHEMATICAL MODEL OF PLC

PLC programs can be developed in a number of different forms. Relay ladder logic (RLL) is a common representation used to document PLC programs. Grafcet, Function Chart, Boolean and other high level programming languages are also available for PLC program development [3], [6]. These representations of a PLC program can be converted into equivalent logical expressions, for example, Petri nets or Finite State Machines [5]. We select a popular language, RLL, and apply the automatic model checking method to verify PLC software. This section illustrates the modeling of RLL as an input to the model checker. Relative issues are PLC scanning cycles, basic elements of RLL and the conversion rules from a RLL to an equivalent model expression. When controlling a system, a PLC repeatedly executes a scanning cycle. One scanning cycle consists of three steps:

- Reading input values from sensors and storing them in an internal bit table,
- Executing the programmed logic and altering the internal bit table, and
- Writing new outputs to actuators based on the values in the bit table.

Use the following formula to calculate the amount of time required for the remote I/O scan update:

TRS — Remote Master scan time

TCS — CPU scan time

TIN and TOUT — Module response delay time

TRS = Time for scan 1 line + (2ms x No. of Slaves)

The CPU scan (TCS) is estimated to be 25ms,

TIN = Maximum response input module time = 8ms

TOUT = Maximum response output module time = 10ms

So,

TRS = 15.34ms + (2ms x 3) = 21.34ms < TCS

Total Delay for Configuration = TIN + 6(TCS) + TOUT
 = 8ms + 6(25ms) + 10ms
 = 158.01ms

VII. DESIGN CONSIDERATIONS

A. Size: Size of the design increases with increase in number of instructions/functions, which a PLC can handle, number of rungs that can be connected in parallel (p). Hence designer can use an instruction set containing optimum instructions.

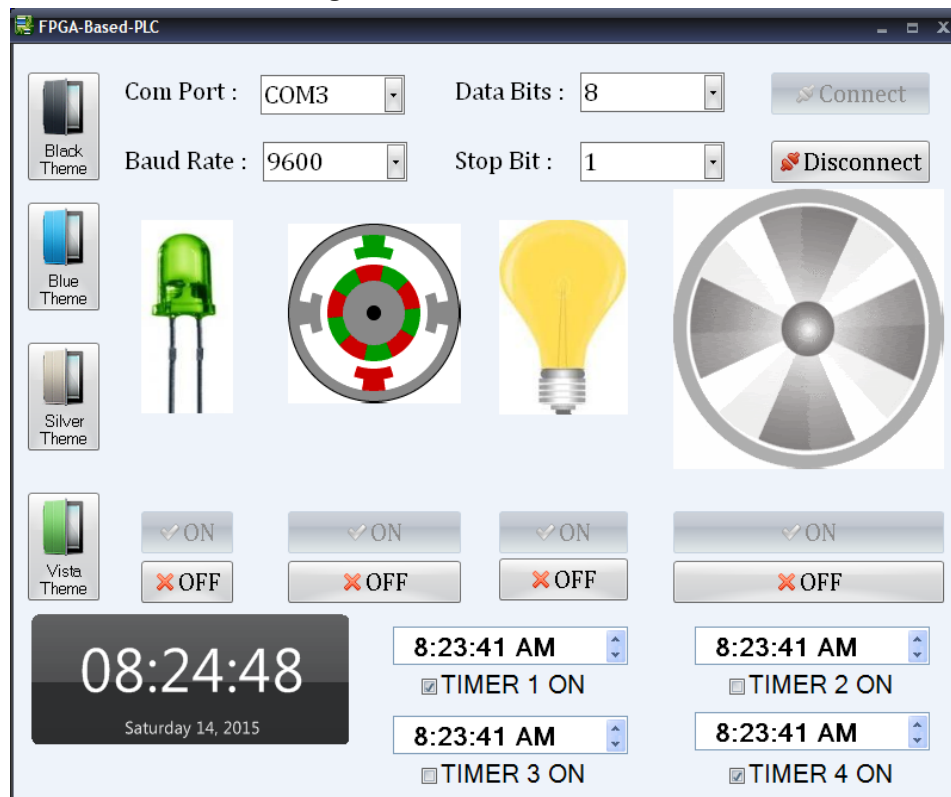
B. Scan Time: An important characteristic of PLC is the time required for one complete I/O scan and execution [11]. This depends on how many input and output channels are involved and length of ladder program. The speed of PLC execution depends on clock frequency of the processor. Higher the frequency less is the scan time. With the proposed design one can achieve very less scan times as every rung execution requires '2m' clock cycles. Thus, if ladder program comprises of 'n' rungs it will require '2mn' clock cycles for PLC scan. Scan time can be further reduced by targeting faster, efficient architectures of FPGA thus achieving faster PLC. The demonstrated design could achieve 2.24-microsecond scan time at 100MHz clock for largest possible ladder logic in the design.

C. Memory: PLC requires memory to store program and temporary data. In conventional PLCs separate memory chips are used for this purpose. With the proposed design one can use the memory available inside same FPGA chip reducing extra circuitry and delays involved in read/write operations in conventional designs. This results in a faster and compact solution. The demonstrated design used block memory as user memory and distributed memory as data memory. Furthermore extra huge memory is available with hardware – software co-design by using the system's memory to store and read data from ROM of computer.

VIII. SOFTWARE (GUI)

Field Programmable Gate Array (FPGA) based Programmable Logic Controller (PLC) GUI is developed by using Visual Studio 2010 software in VB.Net language with serial (RS232) as USB connectivity. As shown in figure 5 the GUI (Graphical User Interface) screen, with monitoring as well as controlling and timer configuration module is also shown.

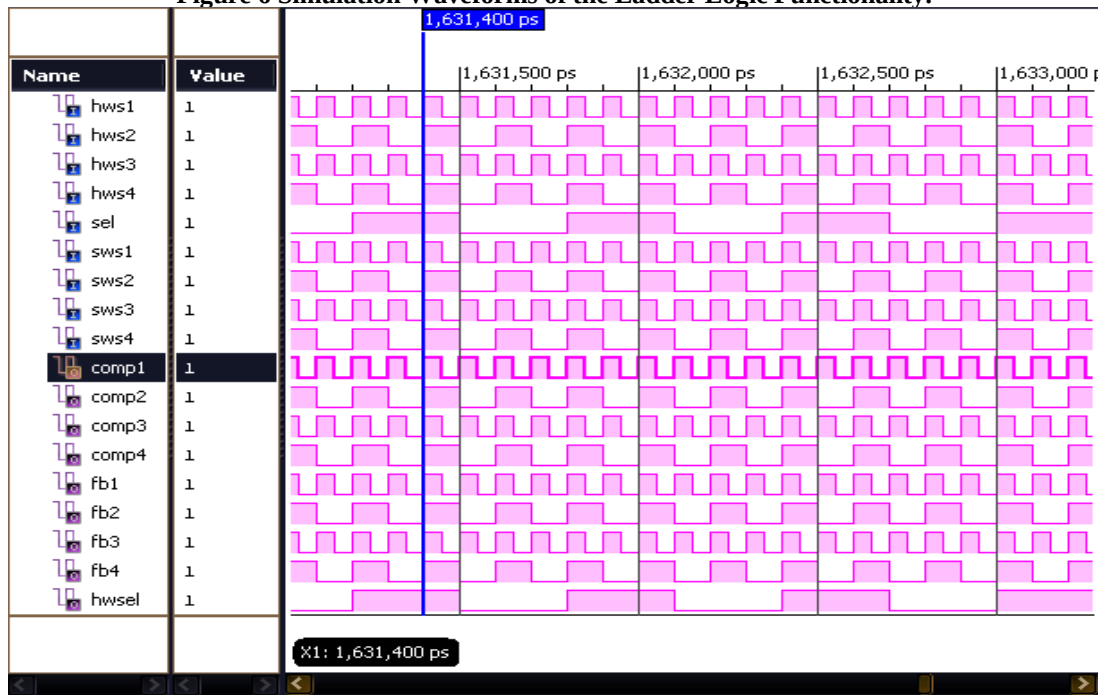
Figure 5 Software GUI Screen.



IX. DESIGN TESTING AND RESULTS

The design results into highly cost-effective, compact and simple solution. As discussed in previous chapters, many applications can be developed and can be parallel executed with the help of the designed PLC. With the ModelSIM, the VHDL program can be simulated and the result is shown in Figure 6.

Figure 6 Simulation Waveforms of the Ladder Logic Functionality.

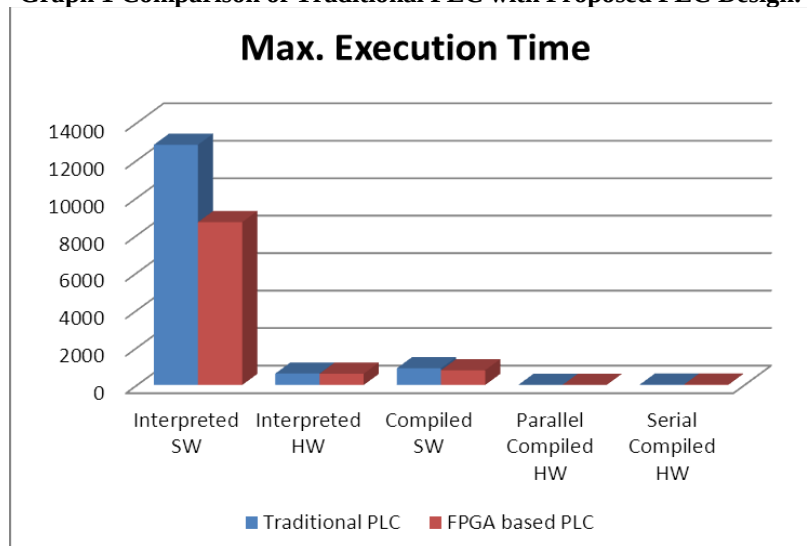


The results of simulation verify that the converted VHDL program can perform the same function as the ladder diagram does. Synthesize this VHDL program and download it to the FPGA, the control system will be implemented inside FPGA. The numbers of clock cycles taken by each implementation in evaluating the 4-rung application logic are shown in Table 2.

Table III Execution times and resources used.

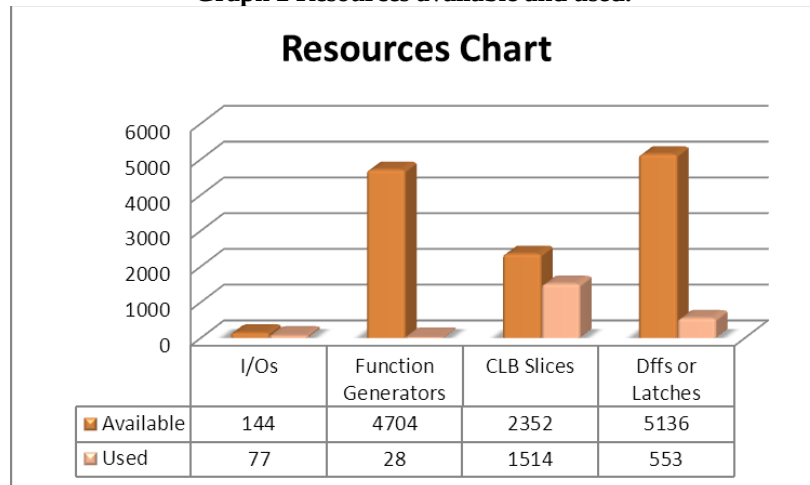
Implementation	Max. Execution Time (Traditional PLC)	Max. Execution Time (FPGA based PLC)	FPGA Resource Use (Slices)
Interpreted SW	12819	8693	2629
Interpreted HW	611	600	3024
Compiled SW	891	784	2629
Parallel Compiled HW	2	2	2731
Serial Compiled HW	19	19	2740

Graph 1 Comparison of Traditional PLC with Proposed PLC Design.



Synthesis Report: Device Utilization for Spartan III, XC3S500E is shown in graph 2. The synthesis report shows that the evaluated maximum operating clock frequency is 170 MHz

Graph 2 Resources available and used.



X. CONCLUSION

In order to overcome the performance limitation of the traditional microprocessor-based PLC and theatrically improve the PLC performance, in this paper FPGA-based PLC is proposed. However the idea has been implemented and demonstrated at smaller scale for typical application. Also the design is limited to digital I/O channels, it can be further extended for Analog I/O channels.

Furthermore, as a programmable hardware solution, FPGA device is reconfigurable, which make it easier for the application system to be modified and maintained. The design method is more flexible. From a practical perspective, this research is still in a nascent stage of development. There are several mathematical issues that must be examined. With the rapidly increasing extension of instruction of the LD program and increasing requirements of PLC, much more research needs to be done to improve this work.

VI. References

- [1] Neil W. Bergmann, Peter Waldeck & Sunil K. Shukla, "FPGA Implementations of Ladder Diagrams", 2013 Canadian Center of Science and Education.
- [2] Zhu Huabing*, Liang Benlei, Dong Bolin, Feng Xiao, "Research on FPGA-based Programmable Logic Controllers' Technology", December 2013 Telkomnika.
- [3] C. Economakos and G. Economakos, "FPGA Implementation of PLC Programs Using Automated High-Level Synthesis Tools", IEEE International Symposium on Industrial Electronics (ISIE 08), 2011, pp. 1908-1913.
- [4] C. Economakos and G. Economakos, "Optimized FPGA Implementations of Demanding PLC Programs Based on Hardware High-Level Synthesis", 13th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA '08), 2011, pp. 1002-1009.
- [5] Daoshan Du & Xiaodong Xu & Kazuo Yamazaki, "A study on the generation of silicon-based hardware Plc by means of the direct conversion of the ladder diagram to circuit design language", May 2010 Springer.
- [6] Dhanashri Gawali & V.K.Sharma, "FPGA BASED MICRO-PLC DESIGN APPROACH", 2009 IEEE.
- [7] Daoshan Du & Yadong Liu & Xingui Guo & Kazuo Yamazaki & Makoto Fujishima, "Study on LD-VHDL conversion for FPGA-based PLC implementation", 2009 Springer.
- [8] S. Ichikawa, M. Akinaka, R. Ikeda, H. Yamamoto, "Converting PLC Instruction Sequence into Logic Circuit: a Preliminary Study", IEEE International Symposium on Industrial Electronics (ISIE 2006), 2009, pp. 2930-2935.
- [9] John T. Welch and Joan Carletta, "A Direct Mapping FPGA Architecture for Industrial Process Control Applications", June 2000 IEEE.

XI. Acknowledgments

The authors would like to acknowledge their gratitude and vote of thanks to all the faculty members of Sinhgad Institute of Technology, especially to the Head of Department of Electronics and Telecommunication for providing the needed help and motivation. Also to their parents and friends for providing co-operation, necessary resources and support forever.