Realization of Timers, Counters and Shift Registers for Programmable Controller Using Ladder Diagram

Shobha S [1], S. Ramachandran [2]

[1] *Cambridge Institute of Technology, Bengaluru, India* , [2]*SJB Institute of Technology, Bangalore, India*

Abstract: Programmable Controller is the leading edge technology in control automation systems finding applications in modern factories. As Ladder Logic Diagram implementations pretenses a stumbling block in the design of complex and real-time Programmable Controllers, it has become the main modeling method of Programmable Controllers. Ladder diagram programming lets Programmable Controllers to perform different types of tasks, including Boolean logic, timing, counting, shifting, arithmetic and special functions. This paper presents the RTL Verilog design of timing, counting, drum counter and shift register operations using Ladder Diagram for Programmable Controller. The paper describes the architecture of timers, counters, drum counters and shift registers. The Ladder Logic Diagrams are targeted for implementation in Field Programmable Gate Arrays. Simulation results of the proposed architecture shows promising results.

Keywords: Bit Operators, Counters, Drum counters, Ladder Diagram, Programmable Logic Control, Shifters, and Timers.

1. Introduction

Ladder logic programs paradigm the electrical circuit diagrams used for wiring control systems in the industry. The basic element of a ladder diagram is a "Contact". A contact has only two states: open or closed. An open contact holds the current flow whereas a closed contact permits current to flow through it to the next element. Programmable Controller substitutes relays, timers and counters using Ladder diagram programming. The facility to accept programming in ladder diagram format envisioned the success of programmable controllers in the industry. The likenesses between the ladder diagrams for programmable controllers and the relay ladder logic once used to control industrial systems, improved the transition from hardwired relay systems to Programmable Controller-based systems in the industry. The impact of Programmable Controller lies in setting up timely events and preserving precise time between events. Programmable Controller timers and counters are internal instructions that provide the same functions as hardware timers and counters. They activate or deactivate a device after a time interval has expired or a preset count has reached. In the counter mode, these counters can be used to count the external events fed from the respective inputs.

2. Related Work

This section briefly outlines the relevant work done by researchers in the field of Programmable Controllers (PC). Angelika Mader et al. [1] suggested two different approaches for modelling timers. These approaches are applied in verification and testing techniques, which lead to two different timed automaton models. The timer models are: Timers as symbolic function block calls and Timers as parallel automata. The first model is close to the way how timers are realized in existing Programmable Logic Controllers (PLCs). The second model tries to capture the idea of timers and uses clocks instead of integers. Hai Wan et al. [2] addressed the formal validation of PLC systems with timers. The timer behavior is characterized formally. A refinement validation methodology is presented in terms of an abstract model and a concrete model. The refinement is calibrated by a mapping relation. T. Kalaiselvi et al. [3] had presented an application, namely, filling and capping bottles simultaneously using PLC. The filling and capping operation takes place in a synchronized manner. It also includes a user-defined volume selection menu through which the user can input the desired volume to be filled in the bottles. G. Krishna et al. [4] proposed the usage of shift registers for power reduction of the digital components by introducing a novel architecture. The proposed architecture eliminates the conducting path in between voltage rail to ground rail during state transitions by providing high resistance path, so that the short circuit current in between rails can be reduced, which in turn causes drastic reduction in power consumption. Phadnis et al. [5] described duty cycle control of single phase induction motor with braking using programmable controller. The timers are used to set the time for forward and reverse direction rotation of motor. Mallaradhya H M et al. [6] have presented an application that fills liquid automatically in to the bottles of different heights fully controlled by the PLC. The system sequence of operation is designed by ladder diagram and the PLC programming software. Sagar P Jain et al. [7] have described a bottle filling process using a timer. The timer is castoff to count for a predefined value of time. The timer is programmed to give two outputs, Enable output and Done output. The Enable output remains high while the timer is counting and the Done output goes high after the timer has finished counting. Savita et al. [8] described a method that involved placing of bottles onto a conveyor belt and filling one bottle at a time. The work focused on counting the number of filled bottles. The filling and counting operation takes place in a synchronized manner using PLC. Shashank

Lingappa M et al. [9] have proposed the advantages of using automated packing machine which can automatically pack various sized products and also to check its performance. D. V. Pushpa Latha et al. [10] present the Smart Street Lighting system, an approach to accomplish the demand for flexible public lighting systems using a PLC.

Kurapati Venkatesh et al. [11] proposed a methodology to evaluate the complexity of Petri Nets and Ladder diagrams for sequence controller design. Complexity of a design is characterized by the number of basic elements used to model the given control logic. Hyung Seok Kim et al. [12] have described detailed steps of the translation method that converts a Ladder Diagram directly to a native code for Programmable Logic Controllers used in most automation systems. Norimasa Shoji et al. [13] proposed a control sequence extraction method to translate ladder diagram to Sequential Flow Chart. A tentative Sequential Flow Chart is created by searching all possible states by using transitions in the ladder diagram. LI Dandan et al. [14] proposed a method for inter conversion between Ladder Diagram and Instruction List. This algorithm is based on forest and binary tree. At first the algorithm maps ladder diagram to forest, and then it establishes a binary tree to represent ladder diagram logical relationship with forest. The algorithm realizes the exchange of Programmable Logic Controller ladder diagram and instruction list by traversing binary tree. Zhiyuan Tang et al. [15] proposed a method of Compiling Ladder Diagram based on Node Method. This approach presents node method to compile it in the VC++ environment. The node method eliminates the traditional compilation process into the Instruction List. Using the node method in Ladder Diagram, compiling can make complex series-parallel relationship of Ladder Diagram into a concise relationship between the contacts and the nodes. Xuekun Chen et al. [16] proposed an algorithm for translating the Ladder Diagrams to ordinary Petri Nets. In this approach, Petri net theory is used to simulate and analyze Ladder Diagram programs to check whether Programmable Logic Controller systems are live, reversible and free of race and deadlocks. Qin Xingguo [17] presented a Model Integrated computing based tool prototype for designing the PLCs considering a printing control system as a case study. In this approach, non Ladder Diagram parts of the printing control system are translated into Structured Text. Yogesh Godhwani et al. [18] described an Automation technique for the protection of Induction Motor using Programmable Logic Controller to mitigate overvoltage, over current, over temperature, over speed and in-rush current problems.

S. V. Kartalopoulos [19] have presented a simple implementation of a fully programmable electronic circuit controller using memories of a matrix type. The controller consists of two memory parts and generates a set of binary valued waveforms. The instruction codes are stored in one memory and the program codes in the other. Any change in one or more of the waveforms is easily made by redefining the program and/or instruction codes. Gokhan Gelen et al. [20] proposed the concept of postponed event for introducing the time delay functions within untimed automata. By using this type of automata, supervisors can be synthesized easily and effectively for Discrete Event Systems that require time delay functions. The ladder logic code implementation of postponed event has been demonstrated by an example. Norashikin M. Thamrin et al. [21] developed an IDE to emulate the real Programmable Logic Controller (PLC) for educational purpose in technical institute, high learning institute and industrial organization. PLC IDE is portable and can be stored in any removal media storages. PLC IDE can be run without installing the software in computer. The software can be accessed offline anywhere at any time. Miroslav Nedved et al. [22] focused on the Ladder Diagram and intended to provide the control engineers with a tool enabling them to visually compare differences/similarities in the Ladder Diagram programs. The algorithms for visual comparison of Ladder Diagrams were proposed and a tool that works with projects from the RSLogix5000 development environment was implemented. It was confirmed that this tool detects and shows visual differences similar to what humans would do and that it does so with a good performance. Huijuan Ni et.al [23] proposed to design PLC for an automated guided vehicle system (AGVS) using ordinary Petri net (PN). It models an AGVS as a plant PN. The PN supervisor is designed to firm a closed-loop PN by augmenting the plant PN. It can translate a closed-loop PN into a ladder diagram that can be performed by a PLC. Ankur G. Gajjar et al. [24] discussed the importance and requirements of the bottle washer machine in the beverage industry. The design of the proposed bottle washer machine for Returnable Glass Bottles (RGB) has been created in the Creo software. The bottle washer machine has been automatized using the Siemens S7-317-2-PN/DP PLC and programmed using a ladder diagram in the SIMATIC Manager.

The previous work [25] presented the design and simulation of the attributes of a Ladder diagram. Each attribute of a Ladder Diagram was represented as a part of the user defined 16-bit Code. An algorithm for various attributes for efficient realization of Ladder Diagrams has been developed. RTL code has also been designed for these attributes using the developed algorithm. Researchers have worked on various ways of using Programmable Controller. But none of the Programmable Controller discussed in the above work has timers, counters, a special feature called drum counter and cascaded registers for shift operations defined in it. This paper proposes a novel method in which the algorithms are developed for various sequential operations like Timers, Counters, drum counters, Bit operations and Shift operations. This work supports 64 timers/counters, 16 drum counters and 32 registers cascaded for shift operations. Based on the algorithm developed, the RTL Verilog code for these operations are designed and simulated. The rest of the paper is organized as follows. Section 3 describes the design of Register array used to store and retrieve data and also the development of the algorithm. Section 4 briefs the

Architectures of timers, counters, bit operations, shift register and drum counter operations. Section 5 describes the Ladder Diagram realization of sequential circuits. Section 6 presents the Simulation results. Conclusion is presented in the last section.

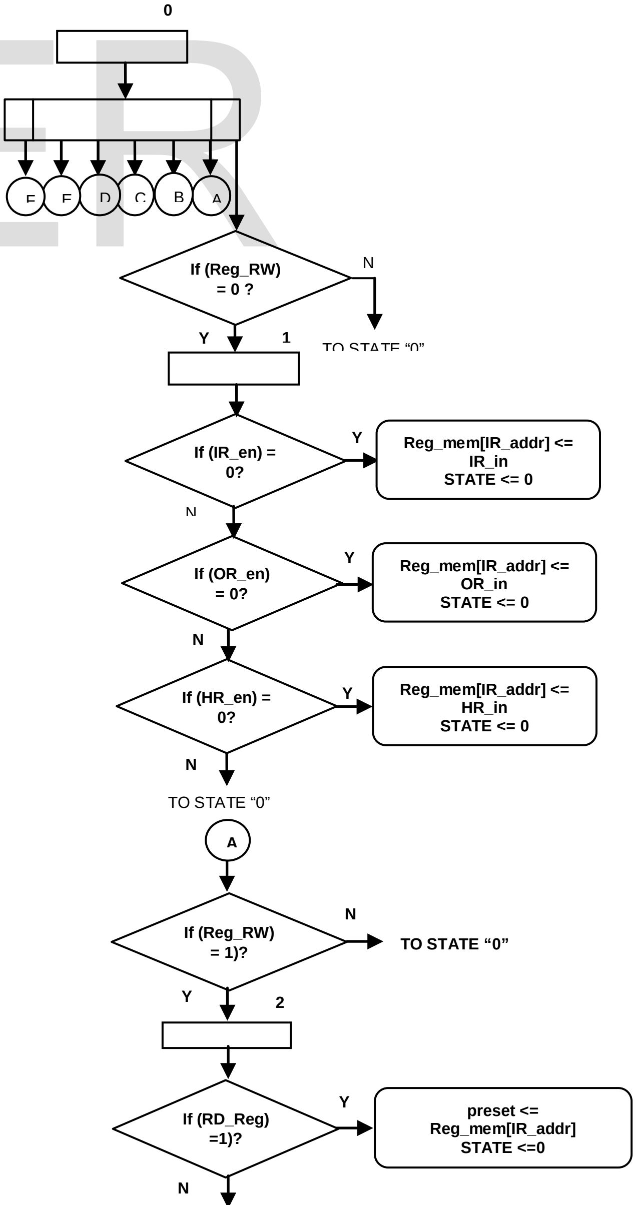
3. Design of Register Array Module – Development of ASM Charts

Register arrays are used to preset data for the timers and counters. The final value of the accumulated timer count is also stored in it. It is designed for a size of 2047 x 16 bits, although the size can be amended to suit new applications. This array supports three registers: Input, Output and Holding registers. In this work, Input Register is used to fetch preset value, Output Register for storing accumulated count of timer/counter and Holding Register is meant for drum counters. The data of Register array can also be used for shift operations and bit set/clear operations. Since our goal is hardware implementation using Verilog, ASM charts offer a convenient and optimum means of implementation. Accordingly, ASM charts for operating the Register array is presented in Fig. 1 (a) and Fig. 1 (b). In state "0" the status of the signal "Reg_RW" is read which indicates Register array write or read operation. The Write operation is activated in state "1". Enable signal for each register type is defined for storing the data into Register array. Read operation is performed in state "2". "RD_Reg" signal is used to read preset value for timer and counter operations from the Register array. The final count of counter operation is stored back into Register array using "actual" signal in state "3". The "actual" signal represents the data at the location of Output register or Holding register. The state "4" and "5" are used to read data for Bit SET and Bit CLEAR operations and to store the results. Shift operations are performed in states "6" and "7". In this work, the codes framed to recognize the type of operation for programming the Ladder Diagram are illustrated in Table 1. These operations are stored in the Code memory in the form of 16-bit user defined code, subsequently fetched and executed. The data required for these operations are stored in Register array.

4. Architecture of Timers/Counters, Drum Counters and Shift Registers

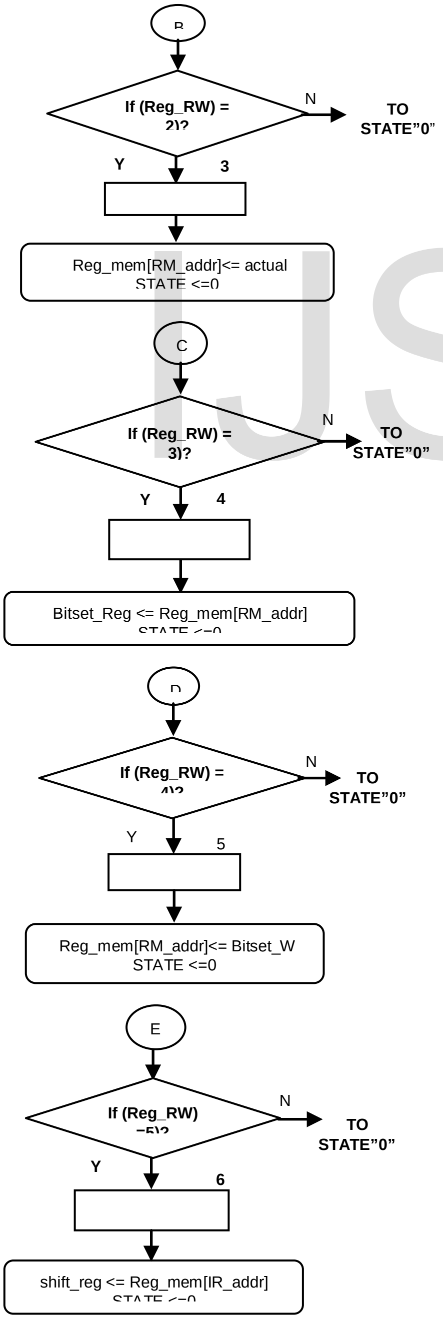
Programmable Timers

The power of Programmable Controller lies in scheduling timely events and maintaining precise time between events. These features are possible due to the timers and counters embedded in the Programmable Controller. The timing for these functions is usually derived from the crystal clock. Timing events can be scheduled at periods that are multiples of the clock periods. Basically timers are used to delay an action, to run an operation for a predetermined period of time and to record the total accumulated time of continuous or intermediate events. Depending upon the application, timers are of different types. The most popular types used in industries are On-delay Timer, Off-delay Timer and Interval-delay Timers. With On-delay timers, the output associated with the timing instruction is turned ON some specified time after the input is ON, but it turns OFF immediately when the input is turned OFF as shown in Fig. 2. With off-delay timers, the output is turned ON immediately when the input is turned ON. However, it remains ON for a specified period of set time after the input has been turned OFF as shown in Fig. 3. Interval-delay Timer starts the timing with the rising edge of a trigger input as shown in Fig. 4. Programmable Controller timer resolution can be tenths of a second (0.1 S) 'TT' or seconds 'TS' derived from the system clock. The time duration for which a timer has been set is termed the preset value and is programmed in multiples of time base used. The time base may be regarded as the resolution or the accuracy of the timer. The preset value for the timer is stored in the Register array.



Sl. No.	Operation	Codes
1	Timer in Seconds (TS)	100000
2	Timer in 0.1 Seconds (TT)	100001
3	UP Counter (UC)	100010
4	Down Counter (DC)	100011
5	Bit Set	100100
6	Bit Clear	100101
7	Shift Right	100111
8	Shift Left	101000

Figure 1 ASM Chart for Access of Register Memory


B
If (Reg_RW) = 2)?
N
TO STATE"0"
Y
3
Reg_mem[RM_addr]<= actual STATE <=0
C
If (Reg_RW) = 3)?
N
TO STATE"0"
Y
4
Bitset_Reg <= Reg_mem[RM_addr] STATE <=0
D
If (Reg_RW) = 4)?
N
TO STATE"0"
Y
5
Reg_mem[RM_addr]<= Bitset_W STATE <=0
E
If (Reg_RW) =5)?
N
TO STATE"0"
Y
6
shift_reg <= Reg_mem[IR_addr] STATE <=0

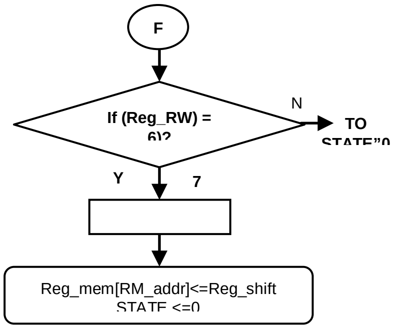


Table 1 Operations of Ladder Diagram

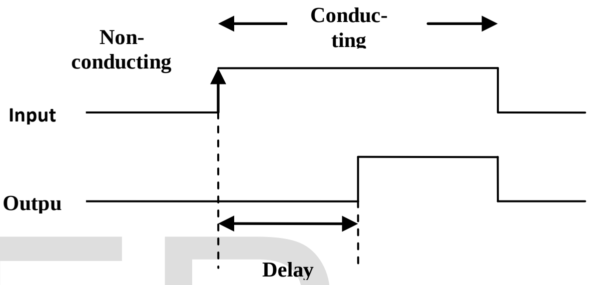


Figure 2 Timing Diagram of On Delay Timer

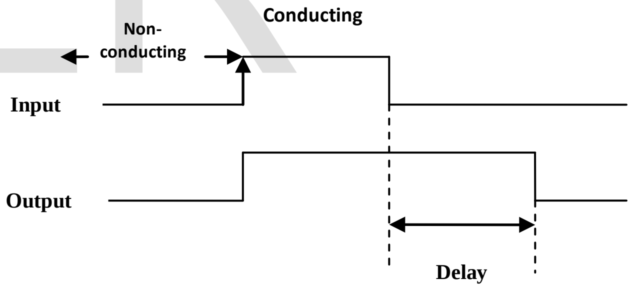


Figure 3 Timing Diagram of Off Delay Timer

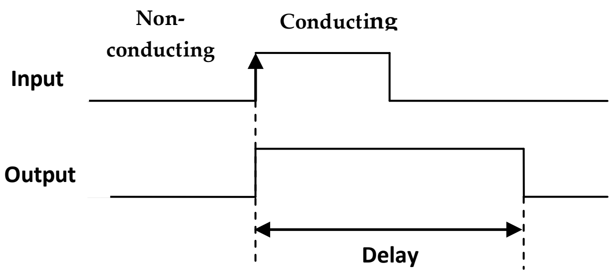


Figure 4 Timing Diagram of Interval Delay Timer

Programmable Counters

Counters are provided as built in components in Programmable Controllers and allow the number of occurrences of input signals or events to be counted. This might be where manufactured items have to be counted as they pass along a conveyor belt, or the number of revolution of shaft or perhaps the number of people passing through a door or number of vehicles entering or leaving the parking area. A counter performs counting for predetermined value called “Preset” value. Coil is energized once the preset value is reached. There are basically two count operations performed, namely, count-up and count-down. The count-up instruction simply increases the value of a counter by 1 each time an upward transition is detected in the input. A count-down instruction does the opposite, decreasing the count value by 1 on each upward transition of the input. Counting instructions are executed in a similar manner as timing instructions. The preset value for the counter is stored in the Register array.

Basic Architecture of Timers/Counters

The main blocks of Programmable Controller Timers/Counters is shown in Fig. 5. Active low signal ‘reset_n’ clears timers/counters number register ‘TCN [5:0]’. When code pertaining to timer or counter like TT/TS/UC/DC is decoded, the enable signal called ‘TCEN’ and timing/counting signal called ‘TRIG’ activated from the output of Ladder will be input to Timer/Counter module. ‘TCRR [63:0]’ is a timer/counter result register, which stores the timer/counter output. The signal ‘Coil’ indicates that the output Control Relay coil of the ladder is energized once the preset count of timer/counter is reached. The signal ‘timer_type’ represents the type of timer, whether it is ON delay, OFF delay or Interval delay timer. Register array module is a set of memory locations of user memory used as registers for temporary storage of data. Three types of registers are considered pertaining to this module, namely, Input Register [IR], Output Register [OR] and Holding Register [HR]. The signals ‘RM_in’ and ‘RM_out’ are used to write data into and read it from these register arrays. The address of these array locations are indicated by the signal ‘RM_addr’. The signal ‘RM_rw’ is used to write and read the preset values for timers/counters into IR register. The signal ‘H_O_write’ represents write operation to store accumulated count values of timers and counters into the register OR.

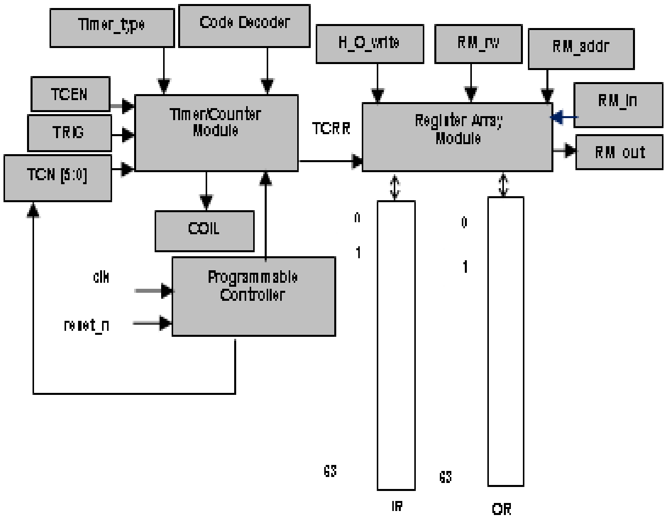


Figure 5 Processing Blocks of Timers/Counters

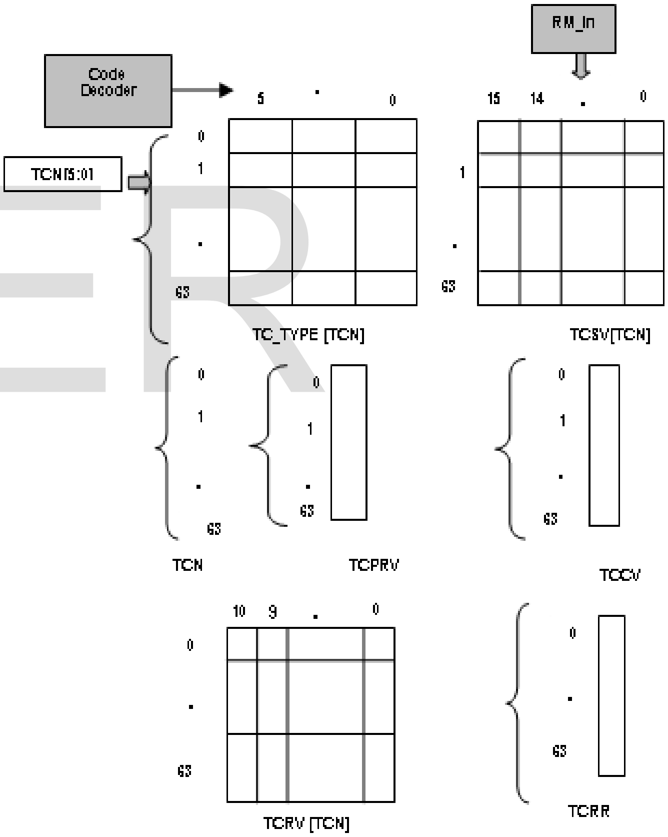


Figure 6 Detailed Architecture of Timers/Counters

Drum Counters

Drum counters are basically Up counters, except that they have a number of ‘tap points’. When these drum counters are enabled, the drum counter starts counting from zero to a maximum value of 2047, advancing the count by one at the rising edge of an input event. Whenever the running counter value equals or exceeds the set 'tap value', then the corresponding output/flag is activated. In this work, there are 16 drum counters of size 11 bits that has the capacity to count from 0 to 2047. Using this drum counter together with a timer, one can switch on or off any number of outputs or flags at different points of time. Although this feature can also be realized using a number of counters and timers, the usage of drum counters saves considerable number of counters and timers.

Architecture of Drum Counter

Fig. 7 shows the architecture of Drum Counter. The registers drc_no [3:0] is used to store the drum counter number. Active low signal 'reset_n' clears the result register 'actual[15:0]' that represents the final count of tap value of drum counter. The signal 'DC_taps [3:0]' that indicates the number of taps used per drum counter is input to the drum counter module. In this work, 16 drum counters are defined. The signal drc_no[3:0] represents the drum counter number. When the code pertaining to drum counter is decoded, the enable signal called 'EN' and counting signal called 'step_ckt' activated from the output of Ladder will be input to drum counter module to enable up counting. The signal 'Coil' indicates that the output Control Relay coil of the ladder is energized once the preset count of drum counter is reached. The signal 'H_O_write' represents write operation to store accumulated count values of drum counter into the HR register.

Shift Registers

In digital circuits, a shift register is a cascade of flip flops, sharing the same clock in which the output of each flip-flop is connected to the data input of the next flip-flop in a chain resulting in a circuit that shifts the bit array stored in it by one position. At each transition of the clock input, the data at the input as well as the output shifts by one position. A digital signal or waveform with discrete delay can be produced using shift registers. In Serial data transmission, for the conversion of parallel form to serial form, shift registers are used. Many slow parallel data lines are replaced with a single serial high speed circuit in serial data communications. Numerous peripherals, including analog to digital converters, digital to analog converters, display drivers, and memory uses shift registers to reduce the amount of wiring on PCB's. The shift operations like SHIFT LEFT and SHIFT RIGHT are implemented in this work. The data to be shifted is the content of Register array. There are two enable signals in shift register. They are 'EN' and 'shift_ckt'. Fig. 8 shows the architecture of Shift Register. The registers used in the architecture are 'shiftreg_num [4:0]' and numofshift [3:0]. The register 'shiftreg_num [4:0]' represents the number of shift registers to be cascaded. The register 'numofshift[3:0]' represents the number of times shift register data are to be shifted. When code pertaining to shift register is decoded, the shift enable signal called 'EN' and shift circuit signal called 'shift_ckt' activated from the output of a Ladder circuit will be input to the shift register module to enable shift operation. The signal 'serialdata_ckt' is used to fill the vacant MSB or LSB during the shift right and shift left operation respectively. The register 'actual [15:0]' is a result register, which stores the shifted value. The signal 'Coil' indicates that the output Control Relay coil of the ladder is energized/de-energized based on the LSB/MSB during shift left/shift right operation. For shift operations, the registers OR and HR are used. The address of these array locations are indicated by the signal 'RM_addr'. The signal 'RM_out' stores the shifted value.

5. Ladder Diagram for Timers/Counters

Ladder Diagram for Timer

Timing circuit is activated from the output "56" of the ladder diagram shown in Fig. 9. Block diagram for the Timer operation is shown in Fig. 10. There are two signals to activate the timer operation. The signals are Timing circuit (TRIG) and Enable circuit (TCEN). Enable circuit is activated from the input. The timer is allowed to run (accumulate time) only when the Enable circuit is conducting. When the Enable circuit is not conducting, the timer is reset and held at zero. The timer will run only when the Timing circuit is conducting. If the Timing circuit stops conducting, the timer will retain the accumulated value as long as the Enable circuit is conducting. Accumulated time is stored in an assigned output register (OR) called the actual register. The associated coil is energized and its contacts are operated when the actual value equals a preset value. The preset value may come from an input register (IR).

Ladder Diagram for Counter

Both UP and Down counters may be programmed from the counter circuit shown in Fig. 11. Up counters begin counting at 0000 and count up to a maximum of 9999. The counter coil is energized upon reaching a preset value. Two contact circuits control the counting operation of the counter circuit: the Counting circuit (TRIG) and the Enable circuit (TCEN). Counting is allowed only when the Enable circuit is conducting. When the Enable circuit is not conducting, the counter's accumulated value is reset and held at zero. The UP counter will increase the accumulated count by one only when the Counting circuit is triggered from the output "56" of the ladder diagram shown in Fig. 9. The accumulated count is retained as long as the Enable circuit is conducting. The accumulated count is stored in an assigned Output Register (OR) called actual register. Then coil is energized and its contacts are operated when the actual value equals or exceeds a preset value. The preset value may come from Input Register (IR). The counter will continue to count past the preset value until 9999 is reached. The count remains at 9999 until the Enable circuit stops conducting. The Down counters begin counting at a preset value (from 0000 to 9999) and count down to 0000. The counter coil is energized upon reaching 0000. Counting is allowed only when the Enable circuit is conducting. When the Enable circuit is not conducting, the counter's accumulated value is reset and held at the preset value. The Down counter will decrease the count by one only when the Counting circuit is activated from the output of the ladder diagram shown in Fig. 9. The count is retained as long as the Enable circuit conducts. The coil is energized and its contacts are operated when the count reaches 0000. Down counters stop and remain at 0000 until the Enable circuit stops conducting.

Ladder Diagram for Drum Counter

Drum counter is similar to UP counter. It is programmed from the counter circuit shown in Fig. 12. In a Drum counter, up counting operation begins at 0000 and counts up to a maximum of 2047. The Drum counter coil is energized upon reaching a preset value. Two contact circuits control the counting operation of the counter circuit: the Step circuit (step_ckt) and the Enable circuit (EN). Counting is allowed only when the Enable circuit is conducting. When the Enable circuit is not conducting, the counter's accumulated value is reset and held at zero. The Drum counter will increase the accumulated count by one only when the Step circuit is triggered from the output of the ladder diagram shown in Fig. 9. The accumulated count is retained as long as the Enable circuit is conducting. The accumulated count is stored in an assigned Holding Register (HR) called actual register. Then coil is energized and its contacts are operated when the actual value equals or exceeds a preset value. The preset value may come from Input Register (IR). The counter will continue to count past the preset value until 2047 is reached. The count remains at 2047 until the Enable circuit stops conducting. Drum counter is designed with 'N' number of taps. When the preset value of each tap is reached, corresponding output is used to activate other circuits.

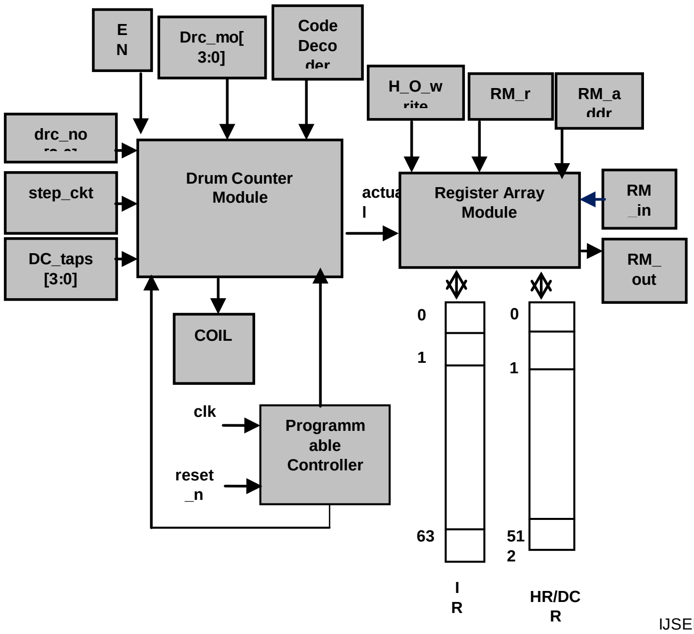


Figure 7 Processing Blocks of Drum Counters

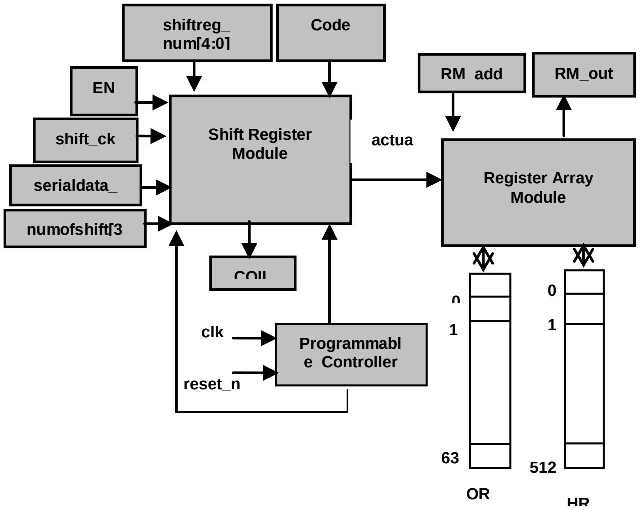


Figure 8 Processing Blocks of Shift Registers

Ladder Diagram for Shift Register

Shift functions are seldom used in conventional relay control systems because of the complexity of the associated circuitry. When required in such installations, the task is normally handled by some type of mechanical device. With electronic control systems like Programmable Controllers, however, these special functions can be designed into the control scheme easily and at a considerable cost reduction. In the proposed technique, shift functions operate on designated registers or groups of registers. When a shift command is received, each bit is moved to the next position: either all bits move to the left, or all bits move to the right.

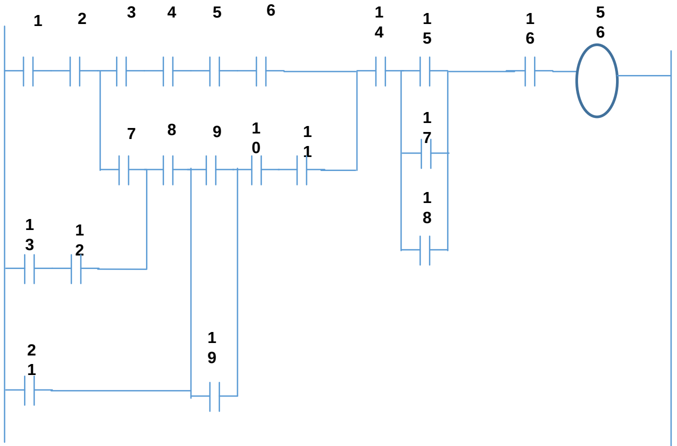


Figure 9 Ladder Diagram Logic, an Example for Triggering Timer, Counter, Drum counter and Shift Register circuits

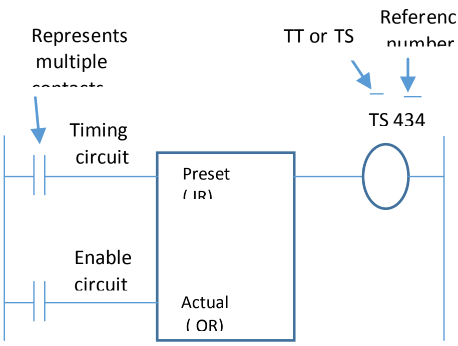


Figure 10 Block Diagram of Timer Circuit

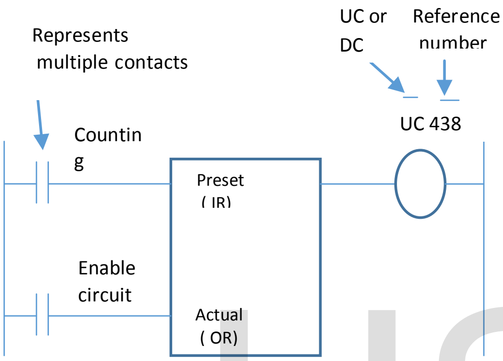


Figure 11 Block Diagram of Counter Circuit

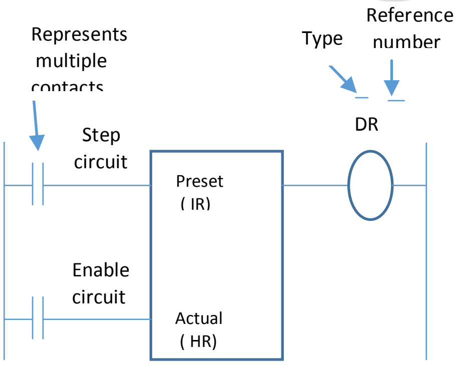


Figure 12 Block Diagram of Drum Counter Circuit

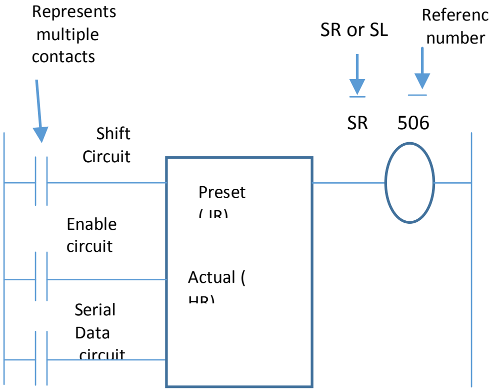


Figure 13 Block Diagram of Shift Register Circuit

Bits are loaded into the register at one end and leave the register at the other end. Both Shift Left and Shift Right operations can be performed from the diagram shown in Fig. 13. The shift Left function loads bits into LSB (0), shifts existing bits one position to the left, and provides bits from MSB (15). The Shift Right function loads bits into bit MSB (15), shifts existing bits one position to the right, and provides bits from bit LSB (0). In the proposed technique, the registers may be chained to form shift registers up to 512 bit positions, which can be met by using 32 registers of 16 bits each. Three contact circuits control shift functions: the Shift circuit, the Serial data circuit and the Enable circuit. Shift operations are allowed only when the Enable circuit is conducting. When the Enable circuit is not conducting, all associated registers are cleared and the coil is turned off. The Shift circuit is activated from the output of the ladder diagram shown in Fig. 9. The exit bit in the register shifts out of its position and is used to determine the state of the coil. A logic 1 will energize the coil and logic 0 will de-energize it. The entry position in the register is filled depending upon the condition of the Serial data circuit. If the serial data circuit is conducting, logic 1 will be entered into the entry position. If the serial data circuit is not conducting, logic 0 will be entered.

6. Simulation Results and Discussions

Architectures for bit operations, timer operations, counter operations and shift register operations were coded in Verilog as per RTL coding guidelines. In order to verify their functionality, a test bench was also written in Verilog. The RTL design was simulated using Model Sim. An example Ladder Logic is used for demonstrating the working of the design and is shown in Fig. 14. There are six paths to energize the output in a Ladder diagram. Those paths are the contacts "9" "1" and "2","3", "4","1" and "5", "6", "1" and "5","7", "8" and the reverse contact paths are "9","6", "7","8" and "2","3","4","6","7","8". Fig. 15 to Fig. 24 present the simulation waveforms. The 'reset_n' signal is the power on reset signal and 'clk' is the system clock signal. The signal 'ladder_output' represents the output of the ladder diagram shown in Fig. 14, CR316 in this case. The MSB 6 bits of code_in represents the type of operation to be performed and remaining 10 bits represents the Coil number indicated by the signal "O_addr". Fig. 15 and Fig. 16 shows the simulation results of bit operations. In these figures, the signals "BC_ckt" or "BS_ckt" are activated from the output of Ladder circuit shown in Fig. 14. The signal "BC" indicates "Bit clear" operation and "BS" indicates "Bit Set" operation. The signal "bit_num" indicates the bit to be cleared/set. The signal "Bitset_R" holds the data before and after bit operations. Coil is energized when the bit indicated by "bit_num" is set/cleared.

Shift register operation results are shown in Fig. 17 and Fig. 18. The signal "shift_ckt" is activated from the output of Ladder circuit shown in Fig. 14. The signal "SR" indicates "Shift Right" operation and "SL" indicates "Shift Left" operation. When the signal "shift_en_ckt" and "shift_ckt" are activated, shift left or shift right operation is performed. The signal "shiftreg_num" indicates the number of shift register cascaded for shift operation. The signal "actualdata" holds the data of the cascaded shift registers before the shift operation. The signal "Reg_shift" holds the value after each shift operation. Coil is energized or de-energized based on the LSB or MSB of the last cascaded register corresponding to the shift right or shift left operation. The LSB of Shift left operation and MSB of shift right operation are filled with "0" since the signal "serialdata_ckt" is assigned with "0". In this example, three registers are cascaded for shift Left and two registers for shift Right operation. For the shift left operation, the data of three registers considered are data 1 (LSB register) as "1111111111111111", data 2 as "0000111100001111" and data 3 (MSB register) as "1111000011110000". As these three data are shifted left, the vacated LSB of data1 is loaded with bit "0". Hence after the first shift left operation, data 1 becomes "1111111111111110", data 2 becomes "0001111000011111" and data 3 becomes "1110000111100000". Since the MSB of data 3 is "1", the coil is energized. Fig. 17 illustrates the data after each shift left operation for the count of five shifts. The signal "shiftcnt" represents the count of each shift operation and the signal "numofshift" represents total num of shifts considered in the example. Similarly, the Shift Right operation is performed as shown in Fig. 18 with two registers cascaded.

The simulation results of counter operations are shown in Fig. 19 and Fig. 20. The signal "counting_ckt" is activated from the output of the Ladder circuit shown in Fig. 14. The signal "UP" indicates the "UP Counter" operation and "DN" indicates the "DOWN Counter" operation. The corresponding operation will be performed when these signal gets "1". When the signals "counting_ckt" and "cnt_en_ckt" are activated, counter starts its function. UP counter starts counting from zero to the preset value indicated by the "preset" signal and the DOWN counter counts from preset value to zero. The signal "cnt_value" holds the current value of counting operation. Coil is energized once the counter completes its operation. Fig. 19 shows the DOWN counter results and Fig. 20 illustrates the UP counter results. The results of Drum counter which performs the up count from zero to 2047 range with four taps is shown Fig. 21. When the signal "drumcnt_en_ckt" and "step_ckt" are activated, the drum counter starts performing up counting from zero to 2047. The signal "cnt_value" holds the current count of the operation. The signals "tap0_out", "tap1_out", "tap2_out", and "tap3_out" are asserted once their corresponding count value is reached. The signal "tapnum" indicates the number of taps used in the drum counter operation. The signal "Coil" is energized once the count reaches 2047 value.

The simulation results of Timer operations are shown in Fig. 22 to Fig. 24. In these figures, the signal "timing_ckt" is activated from the output of the Ladder circuit as shown in Fig. 14. The signal "TT" indicates "Tenth of a Second" operation and "TS" indicates "Seconds" operation. The corresponding operation will be performed when these signal gets the logic value "1". When the signals "enable_ckt" and "timing_ckt" are activated, the timer count starts from zero to the preset value indicated by the "preset" signal. The signal "actual_tt" holds the current count of the "TT" operation and the signal "actual_ts" holds "TS" operation count. The signal "timer_type" is set to a binary value "00" for an OFF delay Timer, "01" for ON delay timer and "10" or "11" for INTERVAL delay timer operation. "Timer_off_coil" or "Timer_on_coil" or "Timer_interval_coil" is energized once the timer count reaches the preset value.

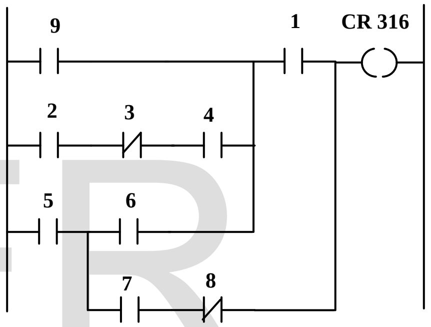


Figure 14 An Example Ladder Diagram Logic for Simulation

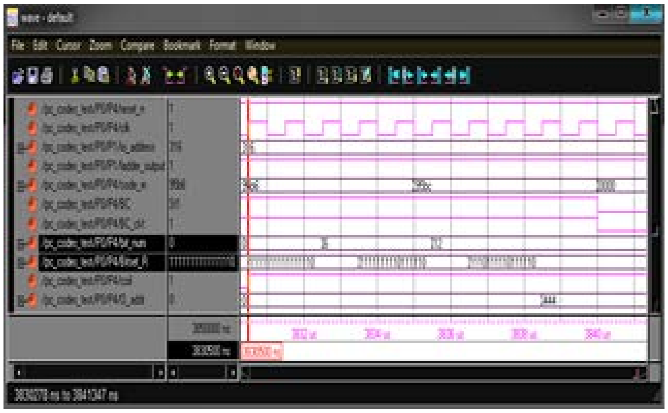


Figure 15 Bit Clear Operation

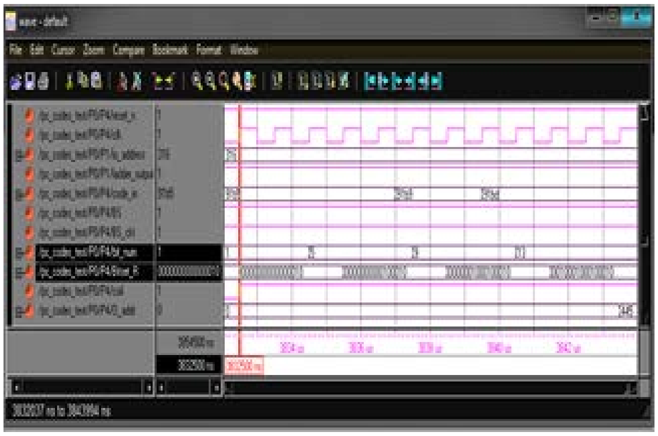


Figure 16 Bit Set Operation

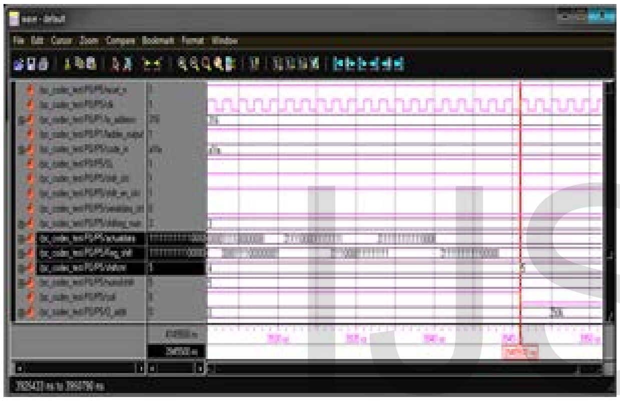


Figure 17 Shift Left Operation

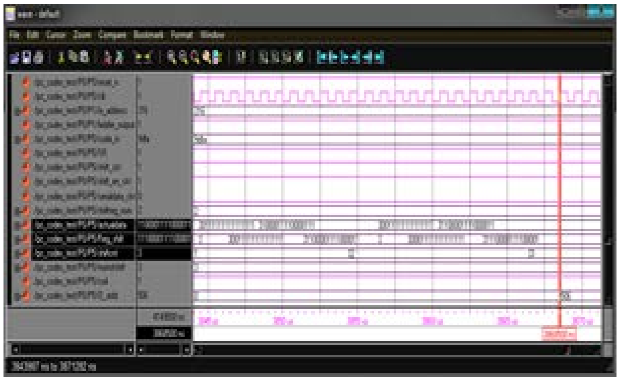


Figure 18 Shift Right Operation

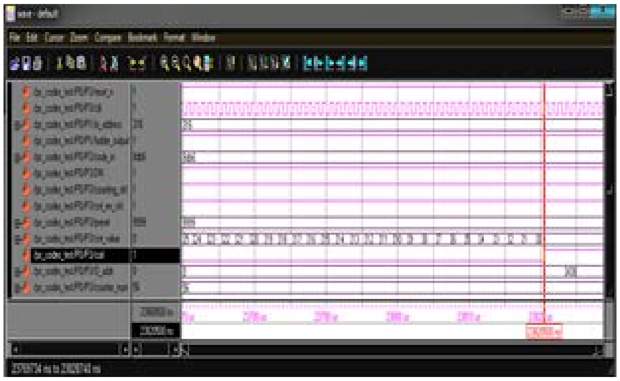


Figure 19 DOWN Counter Operation

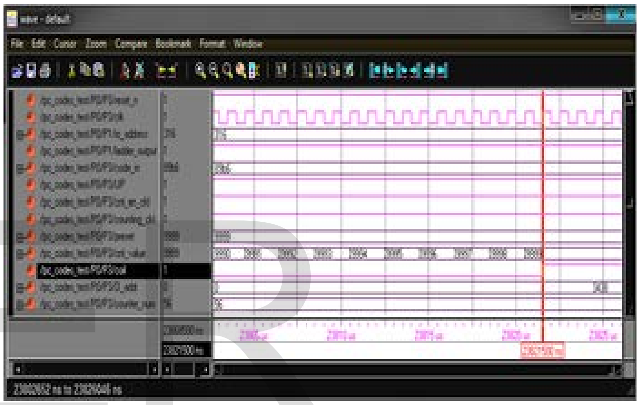


Figure 20 UP Counter Operation

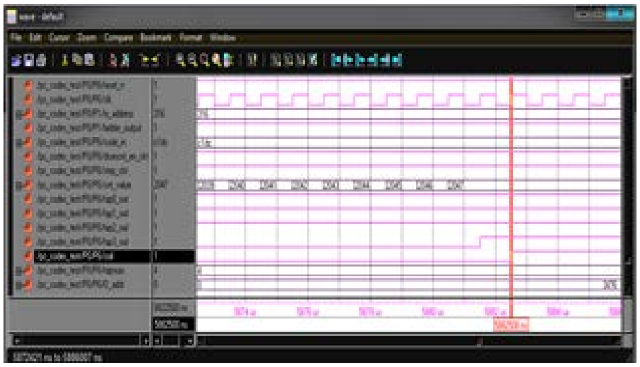


Figure 21 Drum Counter Operation with Output "tap3_out"

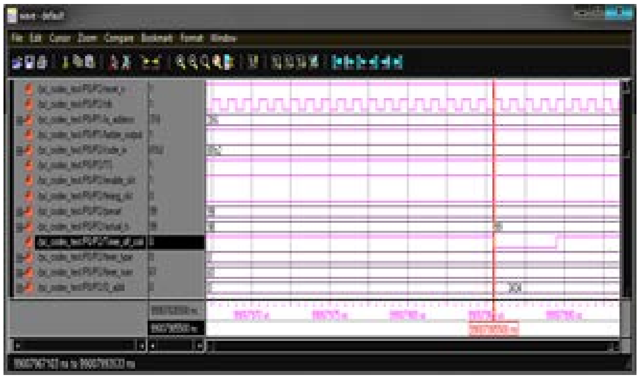


Figure 22 TS Timer Operation for OFF Delay

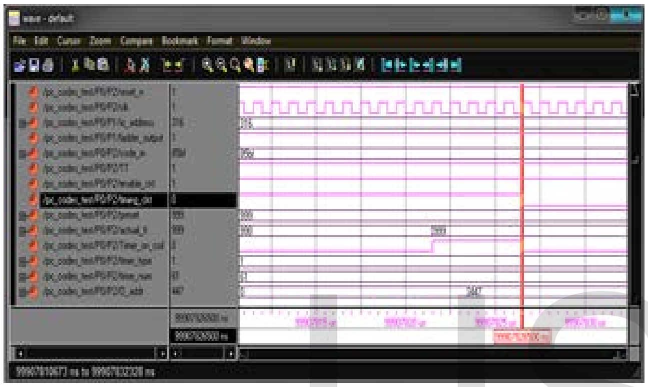


Figure 23 TT Timer Operation for ON Delay

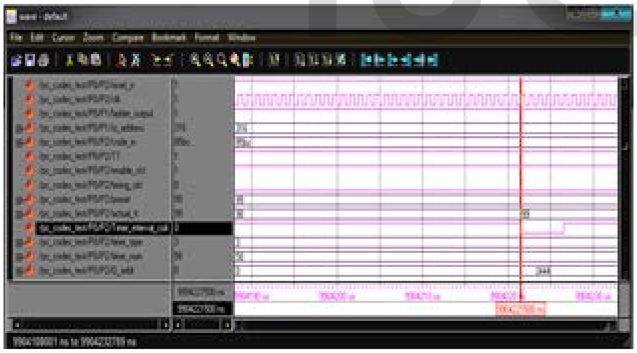


Figure 24 TT Timer Operation for INTERVAL Delay

7. Conclusions

New algorithm has been developed for timer, counter, drum counter, bit and shift operations for realizing control logic of a Ladder diagram for a Programmable Controller. The design is realized using RTL Verilog. The preset value for these operations are stored in Register array. Test bench has been developed in Verilog and the design has been successfully simulated using ModelSim. XILINX synthesis tool is used for synthesizing proposed design. Currently, arithmetic operations and conversion operations of Ladder Diagram based Programmable Controller are under progress.

References

[1] Angelika Mader, Hanno Wupper, "Timed Automaton Models for Simple Programmable Logic Controllers", Real-Time Systems, Proceedings of the 11th Euromicro Conference on 09 Jun, YORK, pages 106 – 113, IEEE, 1999.

[2] Hai Wan, Xiaoyu Song, Gang Chen, Ming Gu, "A Refinement-Based Validation Method for Programmable Logic Controllers", 10th International Conference on Quality Software, IEEE Computer Society, 2010.

[3] T.Kalaiselvi , R.Praveena, Aakanksha.R, Dhanya.S "PLC Based Automatic Bottle Filling and Capping System with User Defined Volume Selection", International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, Volume 2, Issue 8, August 2012.

[4] G.V.V.S.R.Krishna, S.Sindu Bhavani, Y.Jyothi, V. Sai Ram "A Novel Implementation of Low Power Universal Shift Register", International Journal of Advanced Research in Computer Science and Software Engineering, Volume Issue 8, ISSN: 2277 128X, August 2013.

[5] Nidhi Phadnis, Shruti Kharait, Devesh Tamrakar and Sudhir Pulambrikar, "PLC and its Application to Control Single Phase Induction Motor", International Journal of Engineering Sciences & Management, pages 36 – 41, July-Sep, 2013.

[6] Mallaradhya H M, K R Prakash, "Automatic Liquid Filling to Bottles of Different Height Using Programmable Logic Controller", International Journal of Mechanical and Production Engineering, ISSN: 2320-2092, Volume- 1, Issue-4, Oct-2013.

[7] Sagar P. Jain, Dr. Sanjay L. Haridas "Energy Efficient Automized Bottling Plant Using PLC And Scada With Speed Variable Conveyor Assembly", IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) ISSN: 2278-8735, Volume 9, Issue 1, Ver.II (Jan. 2014), PP 09-14, 2014.

[8] Savita, Lokeshwar, "Implementation and Performance Analysis of Bottle Filling Plant Using Ladder Language", International Journal of Science and Research, ISSN: 2378-2834, Volume 3 Issue 7, July 2014.

[9] Shashank Lingappa M., Vijayavithal Bongale, Sreerajendra, "PLC Controlled Low Cost Automatic Packing Machine", International Journal of Advanced Mechanical Engineering, ISSN 2250-3234, Volume 4, pp. 803-811, 2014.

[10] Dr. D.V.Pushpa Latha, Dr. K.R.Sudha, Swati Devabhaktuni "PLC Based Smart Street Lighting Control" Intelligent Systems and Applications, Volume 01, Pages 64-72, 2014.

[11] Kurapati Venkatesh, MengChu Zhou, Reggie Caudill, "Evaluating the Complexity of Petri Nets and Ladder Logic Diagrams for Sequence Controllers

Design in Flexible Automation", IEEE Symposium on Emerging Technologies and Factory Automation, 1994.

[12] Hyung Seok Kim, Wook Hyun Kwon, Naehyuck Chang, "A Translation Method for Ladder Diagram with Application to a Manufacturing Process", Proceedings of the IEEE International Conference on Robotics and Automation, Detroit, Michigan, May 1999.

[13] Norimasa Shoji, Yoshitomo Ikkai, Norihisa Komoda, "An Extraction Method of Control Flow from Ladder Diagram by Judgment of Transfer Frequency", Proceedings of Seventh IEEE International Conference on Emerging Technologies and Factory Automation, Barcelona, Vol. II, Oct. 18-21, pp. 1315-1319, 1999.

[14] LI Dandan, CAI Qixian, PEI Feng, "The New Inter-conversion Algorithm Between Ladder Diagram and Instruction List", Proceedings of IEEE International Conference on Intelligent Computing and Intelligent Systems, Xiamen, Oct. 29-31, pp. 820- 823, 2010.

[15] Zhiyuan Tang, Taiyong Wang, Jingchuan Dong, Zhifeng Qiao, Xianggui Ling, Yanyu Ding, "Compiling Ladder Diagram Based on Node Method", Proceedings of IEEE International Conference on Control, Automation and Systems Engineering, Singapore, July 30-31, pp. 1-4, 2011.

[16] Xuekun Chen, Jiliang Luo, Pengfei Qi, "Method for Translating Ladder Diagrams to Ordinary Petri nets", Proceedings of IEEE 51st Annual Conference on Decision and Control, Maui, HI, pp. 6716 - 6721, Dec.10- 13, 2012.

[17] Qin, Xingguo, "A Model-Integrated Computing Based Tool Prototype for Designing Programmable Logic Controllers", Conference Anthology, IEEE, China, Jan. 1-8, 2013.

[18] Yogesh Godhwani, Akshay Deshmukh, Rajendra Bhombe, "Programmable Logic Controller - An Automation Technique for Protection of Induction Motor", International Research Journal Of Sustainable Science and Engineering, Volume 2, March 2014.

[19] S. V. Kartalopoulos "A Memory Implementation of a Fully Programmable Digital Controller", The Bell System Technical Journal, Vol. 59, No. 9, Printed in USA, November 1980.

[20] Gokhan GELEN, Murat UZAM, Recep DALCI "The Concept of Postponed Event in Timed Discrete Event Systems and Its PLC Implementation", IEEE Journal, 2010.

[21] Norashikin M. Thamrin, Mohd. Mukhlis Ismail "Development of Virtual Machine for Programmable Logic Controller (PLC) by Using STEPS™ Programming Method", IEEE International Conference on System Engineering and Technology (ICSET), 2011.

[22] Miroslav Nedved, Pavel Vrba, Marek Obitko "Tool for Visual Difference Display of Programs in IEC 61131-3 Ladder Diagrams" , IEEE journal, 2015.

[23] Huijuan Ni, Jiliang Luo "Sequence Controller Synthesis for Automated Guided Vehicle Systems Using Ordinary Petri Nets", IEEE journal, 2014.

[24] Ankur G. Gajjar Alpesh I. Patel Raviprakash G. Singh "Design and Development of Bottle Washer Machine for Small Scale Beverage Industry", International Conference on Advances in Computer Engineering and Applications (ICACEA) IMS Engineering College, Ghaziabad, India, 978-1-4673-6911-4, 2015.

[25] Shobha S, S. Ramachandran "RTL Logic Realization Using Ladder Diagram For Programmable Controller", International Journal of Computer Engineering and Applications, volume VII, Issue I, Part I, July 2014.