Int J Adv Manuf Technol (2004) 24: 279–287
DOI 10.1007/s00170-003-1722-y

ORIGINAL ARTICLE

J.-S. Lee · P.-L. Hsu

An improved evaluation of ladder logic diagrams and Petri nets for the sequence controller design in manufacturing systems

Received: 13 November 2002 / Accepted: 1 March 2003 / Published online: 22 June 2004

Abstract Sequence controller designs play a key role in advanced manufacturing systems. Traditionally, the ladder logic diagram (LLD) has been widely applied to programmable logic controllers (PLC), while recently the Petri net (PN) has emerged as an alternative tool for the sequence control of complex systems. The evaluation of both approaches has become crucial and has thus attracted attention.

The "basic element" approach was developed to evaluate the complexity and flexibility for LLD and PN design approaches [1, 2, 3, 4]. However, the basic elements of these two designs are inherently different and hence that approach may lead to unreliable results. Since sequence control here naturally implies the use of logic rules, a rule-based approach is proposed in this paper to provide unified measures for both LLD and PN designs. To illustrate the proposed approach, it is applied to five control sequences with increasing complexity for a stamping process. Results indicate that the proposed approach is more suitable for real applications and, furthermore, PN is increasingly superior to LLD as the control sequence becomes more complex.

Keywords Ladder logic diagrams · Petri nets · PLC · Sequence controllers · Manufacturing systems

1 Introduction

The ladder logic diagram (LLD) has been widely applied to manufacturing systems to design sequence controllers and it is generally implemented on a PLC, which has the advantages of reliability, robustness, and direct programmability. The I/O procedures of the PLC are specified by the LLD and industrial machines thus reliably perform repetitive operations. For most simple systems, it is easy to program the LLD with heuristic methods. However, as modern manufacturing systems have become increasingly complex and large-scale, the corresponding sequence controller design has become more difficult. Accordingly, LLD programming has also become more complicated and its application is thus limited. Moreover, qualitative analysis and performance characteristics of LLD-controlled processes are seldom discussed. In addition, as design specifications change, the LLD program usually needs to be modified significantly. Hence, researchers are pursuing systematic and efficient PLC programming approaches for system modelling, analysis, simulation, and evaluation [5, 6, 7, 8].

Recently, the Petri net (PN) approach has attracted much interest as a potential tool for designing sequence controllers in manufacturing systems [9, 10, 11, 12, 13, 14, 15]. However, the PN approach is not well known by most engineers. Although the sequential function chart (SFC), a PN-based representation tool, has been proposed as the IEC 1131-3 standard programmable language [16], in practice, PLC users still prefer to program LLD directly. Moreover, industrial practitioners are not clear whether the PN is superior to LLD for sequence control in different applications. Hence, realistic comparisons between the LLD and PN approaches are required, especially for large-scale and complex manufacturing systems.

In practice, only a limited amount of research comparing these approaches has been reported, because suitable comparison criteria are difficult to identify. Boucher et al. [17] studied the sequence control of a manufacturing system and reported that using PN makes the controller more tractable than LLD does. However, they did not formally quantify the comparison between LLD and PN to design sequence controllers. Venkatesh et al. [1, 2] proposed a number of "basic elements", which are nodes and links in the LLD and PN, as a quantified measure to compare their design

J.-S. Lee · P.-L. Hsu (✉)
Department of Electrical and Control Engineering,
National Chiao-Tung University, 1001 Ta-Hsueh Road,
Hsinchu, Taiwan
E-mail: plhsu@cc.nctu.edu.tw

complexity and response time. They claimed that PN offers a better solution than LLD, especially in adaptability as specifications change. Based on the basic element approach, Zhou and Twiss [3, 4] further compared the LLD and PN in terms of the comprehensibility, flexibility and the ability to perform correctness verification. They also reported that the PN displays better results. However, note that while basic elements in the LLD stand for push buttons, limited switches, relay coils, timers, counters, solenoids and lines, they are places, transitions and arcs in the PN. Since both nodes and links in the LLD and PN have different physical meaning, as shown in Table 1, analysis of LLDs and PNs simply by using the number of basic elements as the comparison measure may lead to an incoherent comparison.

In this paper, a new approach towards evaluating the LLD and PN methods is proposed via the IF-THEN transformation. By converting both the LLD and PN into the same IF-THEN formats [18], a unified comparison is then achieved based on the same measure, which is the sum of (1) the number of IF-THEN rules and (2) the number of logical operators for both LLD and PN. An example of five sequences with increasing complexity for a stamping process is provided to illustrate the proposed approach. It has been found that the proposed evaluation approach yields more reasonable results. Also, the realistic comparisons provided in this paper support the superiority of the PN approach.

The remainder of this paper is organised as follows. First, Sect. 2 introduces the rule-based comparison for LLD and PN. In Sect. 3, an application example of a stamping process is provided to illustrate the proposed approach. Then, Sect. 4 presents the discussions of the comparison results. Finally, conclusions are provided in Sect. 5.

Table 1 Basic elements in LLD and PN

Basic elements	LLD		PN	
Nodes	Push button	—⊥—	Place	→○→
	Normally open contact/switch	—||—	Transition	→|→
	Normally closed contact/switch	—|/|—		
	Relay coil	—(R)—		
	Timer	TIM		
	Counter	CNT		
	Solenoid	—/\/—		
Links	Line	——	Normal arc	——→
			Inhibitory arc	——○

1.1 Rule-based comparison

Two major factors for comparison of LLD and PN for sequence control are identified as design complexity and response time [1]. Design complexity is defined as the complexity associated with designing the control logic for a given specification. Response time is termed as the scan time in LLD or the execution time in PN. The major factor for design complexity is the physical size of the control logic model, whereas the response time is influenced not only by physical size, but also by the hardware of implementation. For simplicity, this paper focuses on comparison of the control logic models. The proposed approach includes two steps:

Step 1 Transform both the LLD and PN into the same IF-THEN format.

Step 2 Evaluate the LLD and PN based on the number of (a) rules and (b) logical operators.

In general, control models use a smaller number of IF-THEN rules and logical operators are easier to understand, debug, check and maintain. Moreover, they may have a shorter response time. Thus, the proposed approach is based on the unified rule-based format to compare the corresponding design complexity and response time for different LLD and PN structures.

1.2 IF-THEN formats

Basically, compound IF-THEN rules, which involve both the conjunctive and disjunctive connectives in their antecedent or conclusion part, can be categorised into the following basic four types [18]:

Type 1 IF (A and B) THEN C, or expressed as $(A \cap B) \to C$,

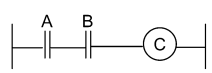
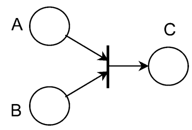
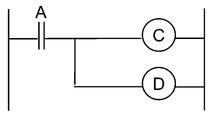
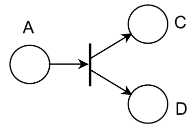
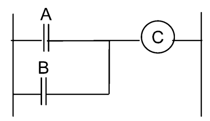
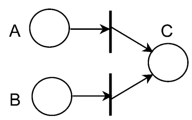
Type 2 IF A THEN (C and D), or expressed as $A \to (C \cap D)$,

Type 3 IF (A or B) THEN C, or expressed as $(A \cup B) \to C$,

Type 4 IF A THEN (C or D), or expressed as. $A \to (C \cup D)$.

The Type 2 rule can be broken into two simple rules, $A \to C$ and $A \to D$. Similarly, the Type 3 rule is equivalent to the two simple rules $A \to C$ and $B \to C$ because the truth of either A or B (or both) implies the truth of C. In practice, since the Type 4 rule does not achieve the specific implication and often causes conflict problems, it is generally not suitable for real applications in the sequence control. The IF-THEN rules excluding Type 4 for the LLD and PN transformations are shown in Table 2. Note that the timers and counters can also be expressed in the basic rules. For example, condition A may represent delaying the desired time units and the status C may express that a counter increases or decreases one unit.

Table 2 IF-THEN rules for LLD and PN

IF-THEN rules	LLD	PN
IF A and B, THEN C A∩ B → C		
IF A, THEN C and D A → C∩ D		
IF A or B, THEN C A∪ B → C		

1.3 Unified comparison measures

Based on the IF-THEN rules, two measures are proposed to evaluate PN and LLD as follows.

Measure 1 The number of IF-THEN rules.
Measure 2 The number of logical operators, including the conjunction (AND), disjunction (OR), block and implication.

The summation of measure 1 and measure 2 can be recognised as a new measure for evaluating different structures. By transforming both the LLD and PN to the same IF-THEN formats, comparisons with a unified measure can then be made. Basically, models use a smaller number of IF-THEN rules and logical operators are easier to understand, debug, check and maintain. Moreover, they often have a shorter response time. Therefore, the sum of measure 1 and measure 2 properly signifies the design complexity and response time for the process represented in either LLD or PN structures.

Table 3 Comparison of LLD and PN for the sequence: $A+$, $A-$

Comparison measure	LLD		PN	
Basic elements	Push button	1	Place	6
	No contact	7	Transition	5
	Nc contact	2	Normal arc	11
	Relay	2		
	Solenoid	2		
	Line	20		
	Total	34	Total	22
IF-THEN rules	Rule	4	Rule	5
	Operator	14	Operator	6
	Total	18	Total	11

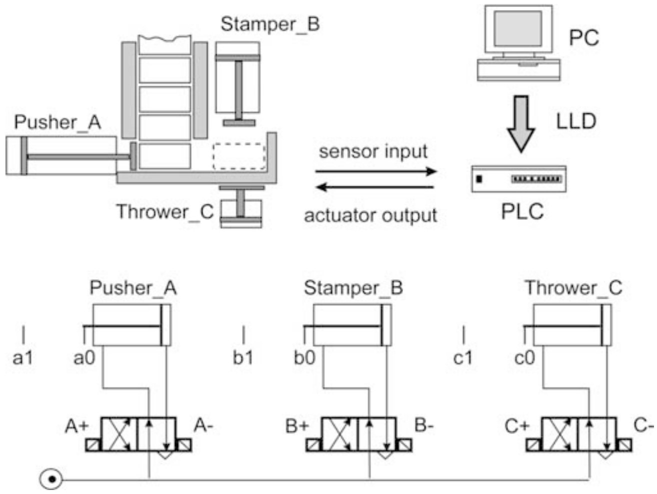


Fig. 2 The stamping system

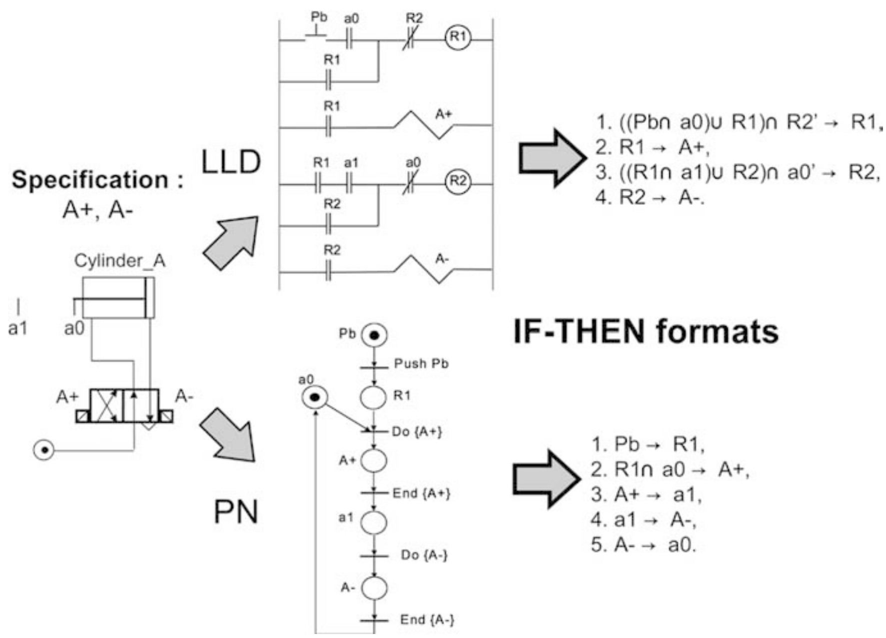
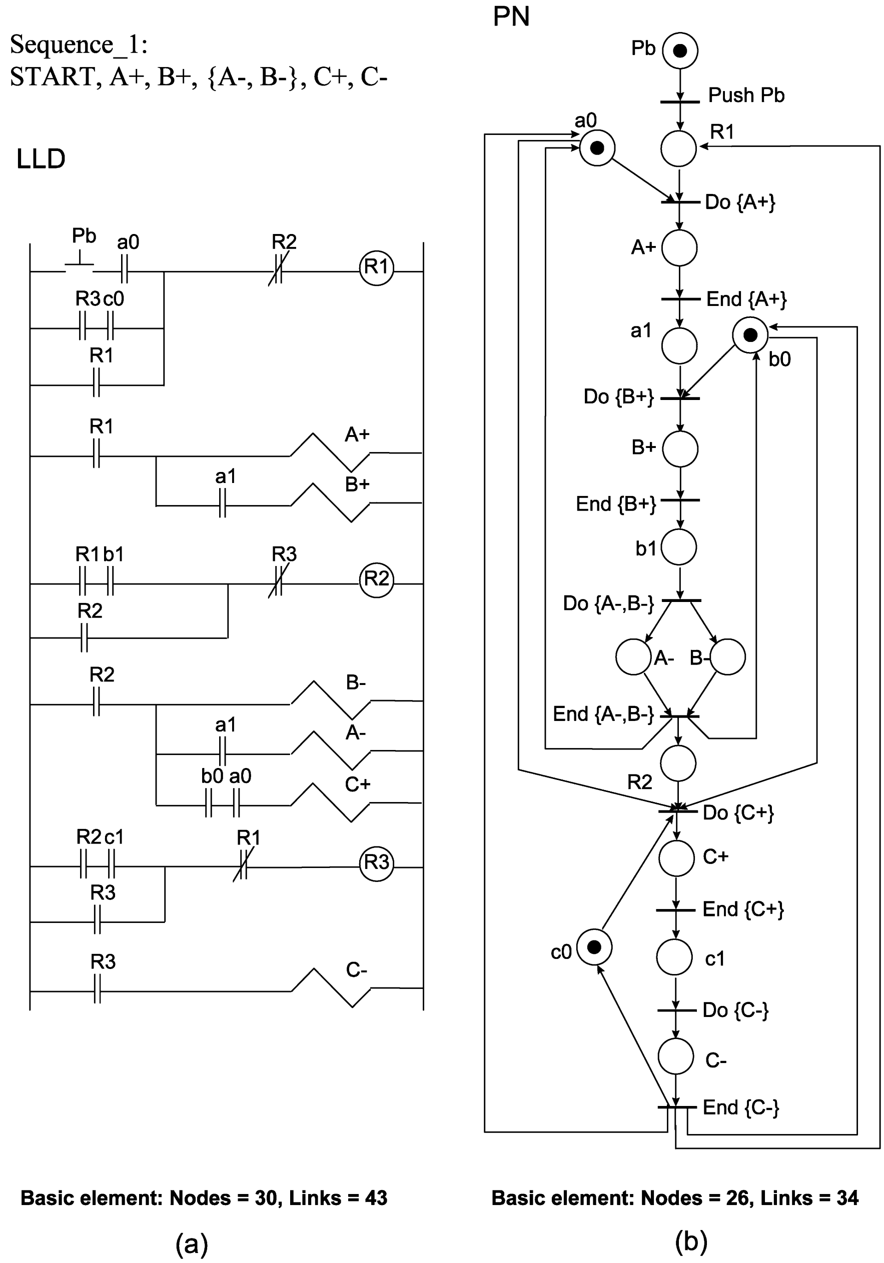


Fig. 1 The LLD and PN for the sequence: $A+$, $A-$

Fig. 3 LLD and PN for Sequence_1



1.4 A preliminary comparison

A simple example we use to illustrate the proposed approach is shown in Fig. 1, which a piston performs a forward stroke and then retracts. In this figure, the specification $A+$ indicates a forward stroke and $A-$ indicates return stroke sequentially. Both the LLD and PN controllers, as shown in Fig. 1, can be represented by either the basic elements or transformed into the same IF-THEN format, as listed in Table 3. Results show that the number of basic elements for the LLD and PN are 34 and 22, respectively. However, the basic elements in LLD and PN are physically different, as mentioned before, and the comparison based simply on the number of basic elements for different structures is apparently inappropriate. On the other hand, the results obtained from the IF-THEN transformation indicate that the LLD programming needs 4 IF-THEN rules and 14 logical operators, while the PN only needs 5 IF-THEN rules and 6 logic operators. Therefore, the number of IF-THEN rules and logical operators for LLD and PN is 18 and 11, respectively. Although the results of both approaches indicate that the PN offers a better solution than LLD, the present IF-THEN transformation provides more reasonable results when evaluating different structures in sequence controller design. Furthermore, the degree of programming flexibility can be analysed by observing the increase ratio of either the number of basic elements or the number of present rules/operators as sequences become more complex.

2 An application example

To illustrate the proposed approach, an industrial process for automatic mark stamping is used and the way the specifications change is examined as five increasingly complex sequences are considered.

2.1 System description

As shown in Fig. 2, a mark stamping system consists of three cylinders which are operated by four-port and two-way solenoid valves. Each cylinder has two normally open limit switches. For example, when the end of *pusher_A* contacts the limit switch *a*0, then *a*0 is closed, meaning that *pusher_A* is at the end of its return stoke. The whole system includes seven input sensors corresponding to six limit switches and one push button for starting the system, and six output actuators corresponding to six solenoid valves. In the stamping process, *pusher_A* moves the workpiece from a store onto the worktable. Then, the workpiece is stamped by *stamper_B* and afterwards is ejected by *thrower_C*. The logical sequence of the stamping system is $A+$, $B+$, $\{A-, B-\}$, $C+$, and $C-$, where $\{A-, B-\}$ represents two concurrent actions as the pistons of both *pusher_A* and *stamper_B* perform return stokes simultaneously. Five sequences with increasing complexity are considered here as follows:

Sequence_1	START, $A+$, $B+$, $\{A-, B-\}$, $C+$, $C-$
Sequence_2	START, $A+$, $B+$, 10 s, $\{A-, B-\}$, $C+$, $C-$(Sequence_1 with one 10 s timer added)
Sequence_3	START, 3 [$A+$, $B+$, 10 s, $\{A-, B-\}$, $C+$, $C-$] (Sequence_2 with one 3-time counter added)
Sequence_4	START, 3 [$A+$, $B+$, 10 s, $\{A-, B-\}$, $C+$, $C-$], 30 s, 2 [$A+$, $B+$, 10 s, $\{A-, B-\}$, $C+$, $C-$] (Sequence_3 with one 30 s timer and one 2-time counter added)
Sequence_5	Sequence_4 with one emergency stop added.

The complexity of these five sequences increases as specified above.

2.2 Sequence controller design

In order to solve the interlock problem, the LLD programs are usually developed with the assistance of the cascaded method which divides the required sequence into groups [6]. Possible contradictory solenoid signals can be thus avoided. On the other hand, since PN is a

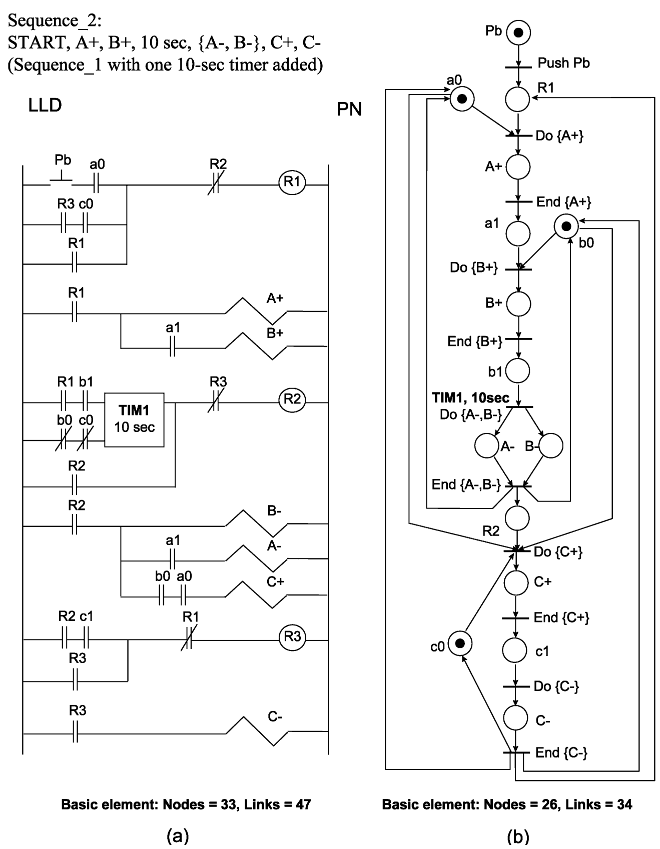


Fig. 4 LLD and PN for Sequence_2

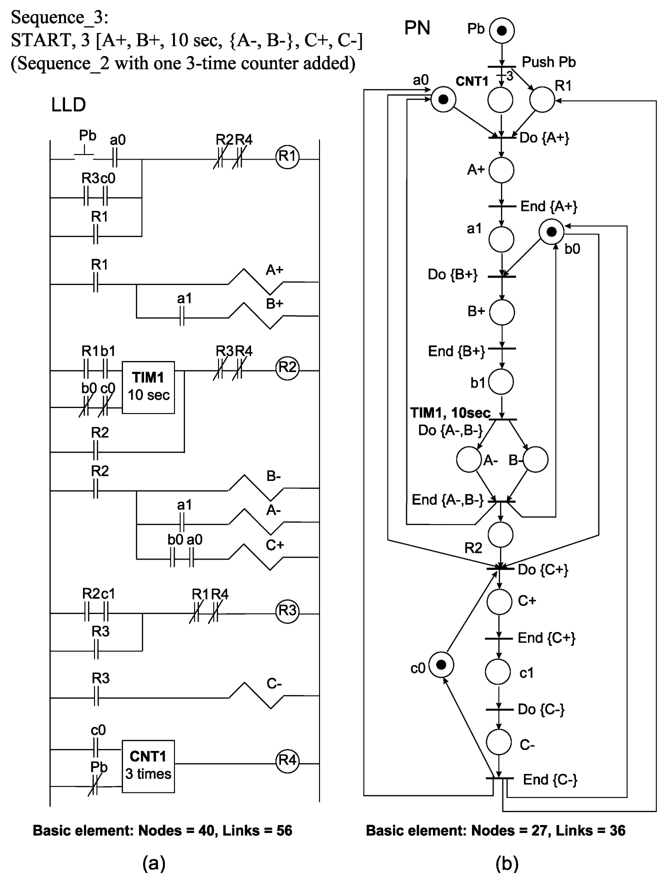


Fig. 5 LLD and PN for Sequence_3

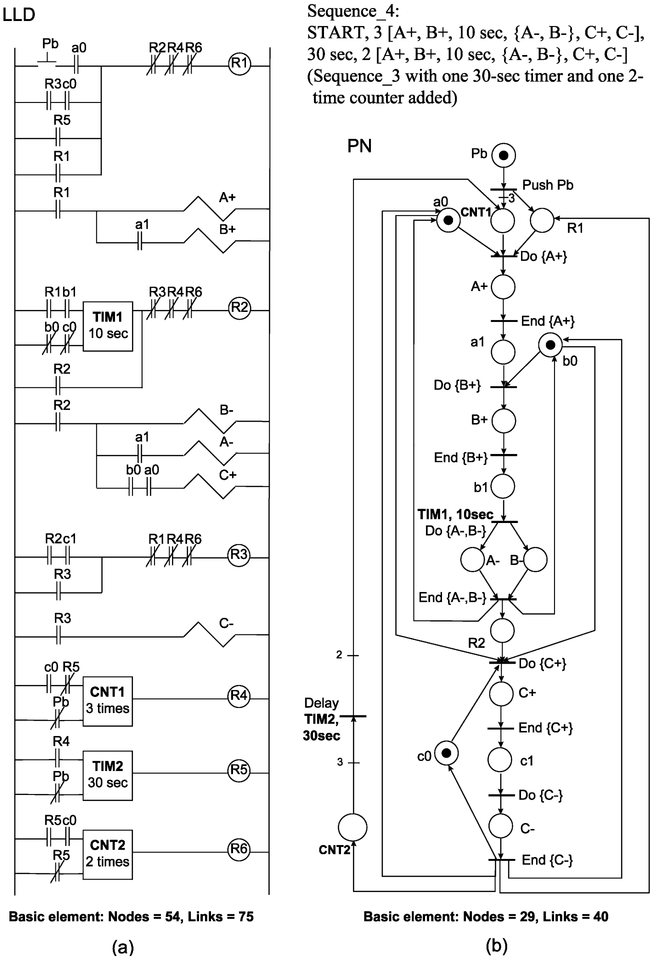


Fig. 6 LLD and PN for Sequence_4

concurrent operation, it can be verified to avoid the interlock logic problem via the simulation [8]. The LLD and PN for the Sequence_1–Sequence_5 are shown in Figs. 3, 4, 5, 6 and 7. Although the sequences compared here only consider a typical cylinder-actuating system, a similar analysis can be extended to general industrial applications such as motors, pumps, heaters and conveyors.

2.3 Comparison of LLD and PN

Table 4 shows the IF-THEN formats of the LLD and PN in Figs. 3, 4, 5, 6 and 7. The required basic elements in the basic element approach, and the required rules and logical operators in the IF-THEN transformation for the five sequences are shown separately in Figs. 8 and 9. For these five sequences, the increase ratio, which is the normalised measure based on Sequence_1 corresponding to the increasing sequence complexity, is also shown in Figs. 10 and 11 for the two approaches. In general, a larger ratio indicates that the design is less flexible when subjected to changes in sequence control. All results indicate that the PN is superior to LLD in terms of design simplicity, response time and flexibility responding to the specification changes.

3 Discussions

This paper presents a novel and unified approach to evaluating the computational burden and complexity subject of sequence programming for different structures. Because the basic elements for LLD and PN structures posses different physical meanings, results using the basic element approach are not adequate to conclude which design structure is more efficient. By applying the proposed IF-THEN transformation approach, the same IF-THEN rules and logical operators are obtained for both LLD and PN structures and thus the results in Fig. 9 show conclusively that the PN structure design is more efficient.

Furthermore, by applying the IF-THEN transformation, results indicate that the PN structure also leads to a lower increase ratio than the LLD structure, as shown in Fig. 11. Thus, design via the PN structure is more flexible when the specification changes. A similar trend can also be observed using the basic element approach as shown in Fig. 10. Therefore, the PN structure for sequence control design will become more valid for large-scale processes.

Although both the basic element approach and the IF-THEN transformation present similar results in terms of increase ratios for given sequence changes as shown in Figs. 10and 11, a comparison indicates that the basic element approach overestimates the complexity of LLD, and underestimates that of PN. For example, comparing Sequence_1 with Sequence_2, which adds a timer to Sequence_1, results of the basic element approach indicate that both sequences require the same number of basic elements by using the PN, as shown in Fig. 8. This is obviously misleading. On the other hand, evaluation results with the present IF-THEN transformation properly indicate that the complexity of PN increases from 34 to 35, as shown in Fig. 9. Therefore, the proposed IF-THEN transformation is more realistic for evaluating sequence control design than the basic element approach.

4 Conclusions

In this paper, a unified comparison approach has been proposed to adequately evaluate the LLD and PN using the IF-THEN transformation. Thus, more realistic and reasonable results can be obtained to analyse the design complexity and flexibility to specification changes for different structures. Results of the given example show that the PN is simpler and more flexible than LLD in realisation of sequence controllers. Hence, PN is a promising solution for modern industrial control systems.

Table 4 IF-THEN formats of LLD and PN in Figs. 3, 4, 5, 6, 7

	LLD	PN
Seq._1	1. ((Pb∩ a0)∪ (R3∩ c0)∪ R1)∩ R2' → R1, 2. R1 → A+, 3. R1∩ a1 → B+, 4. ((R1∩ b1)∪ R2)∩ R3' → R2, 5. R2 → B-, 6. R2∩ a1 → A-, 7. R2∩ b0∩ a0 → C+, 8. ((R2∩ c1)∪ R3)∩ R1' → R3, 9. R3 → C-. **Rules = 9, Operators = 31**	1. Pb → R1, 2. a0∩ R1 → A+, 3. A+ → a1, 4. a0∩ b0 → B+, 5. B+ → b1, 6. b1 → A-∩ B-, 7. A-∩ B- → a0∩ b0∩ R2, 8. R2∩ a0∩ b0∩ c0 → C+, 9. C+ → c1, 10. c1 → C-, 11. C- → a0∩ b0∩ c0∩ R1. **Rules = 11, Operators = 23**
Seq._2 (Seq._1 with one timer added)	1. ((Pb∩ a0)∪ (R3∩ c0)∪ R1)∩ R2' → R1, 2. R1 → A+, 3. R1∩ a1 → B+, 4. ((R1∩ b1∩ **TIM1**)∪ R2)∩ R3' → R2, **5. b0'∩ c0' → (RST)TIM1,** 6. R2 → B-, 7. R2∩ a1 → A-, 8. R2∩ b0∩ a0 → C+, 9. ((R2∩ c1)∪ R3)∩ R1' → R3, 10. R3 → C-. **Rules = 10, Operators = 34**	1. Pb → R1, 2. a0∩ R1 → A+, 3. A+ → a1, 4. a0∩ b0 → B+, 5. B+ → b1, 6. b1∩ **TIM1** → A-∩ B-, 7. A-∩ B- → a0∩ b0∩ R2, 8. R2∩ a0∩ b0∩ c0 → C+, 9. C+ → c1, 10. c1 → C-, 11. C- → a0∩ b0∩ c0∩ R1. **Rules = 11, Operators = 24**
Seq._3 (Seq._2 with one counter added)	1. ((Pb∩ a0)∪ (R3∩ c0)∪ R1)∩ R2'∩ **R4'** → R1, 2. R1 → A+, 3. R1∩ a1 → B+, 4. ((R1∩ b1∩ TIM1)∪ R2)∩ R3'∩ **R4'** → R2, 5. b0'∩ c0' → (RST)TIM1, 6. R2 → B-, 7. R2∩ a1 → A-, 8. R2∩ b0∩ a0 → C+, 9. ((R2∩ c1)∪ R3)∩ R1'∩ **R4'** → R3, 10. R3 → C-, **11. c0∩ CNT1 → R4,** **12. Pb' → (RST)CNT1.** **Rules = 12, Operators = 40**	1. Pb → R1∩ **(SET)CNT1** , 2. a0∩ R1∩ **CNT1** → A+, 3. A+ → a1, 4. a0∩ b0 → B+, 5. B+ → b1, 6. b1∩ TIM1 → A-∩ B-, 7. A-∩ B- → a0∩ b0∩ R2, 8. R2∩ a0∩ b0∩ c0 → C+, 9. C+ → c1, 10. c1 → C-, 11. C- → a0∩ b0∩ c0∩ R1. **Rules = 11, Operators = 26**
Seq._4 (Seq._3 with one timer and one counter added)	1. ((Pb∩ a0)∪ (R3∩ c0)∪ R1)∩ R2'∩ R4'∩ **R6'** → R1, 2. R1 → A+, 3. R1∩ a1 → B+, 4. ((R1∩ b1∩ TIM1)∪ R2)∩ R3'∩ R4'∩ **R6'** → R2, 5. b0'∩ c0' → (RST)TIM1, 6. R2 → B-, 7. R2∩ a1 → A-, 8. R2∩ b0∩ a0 → C+, 9. ((R2∩ c1)∪ R3)∩ R1'∩ R4'∩ **R6'** → R3, 10. R3 → C-, 11. c0∩ **R5**∩ CNT1 → R4, 12. Pb' → (RST)CNT1, **13. R4∩ TIM2 → R5,** **14. Pb' → (RST)TIM2,** **15. R5∩ c0∩ CNT2 → R6,** **16. R5' → (RST)CNT2.** **Rules = 16, Operators = 52**	1. Pb → R1∩ (SET)CNT1 , 2. a0∩ R1∩ CNT1 → A+, 3. A+ → a1, 4. a0∩ b0 → B+, 5. B+ → b1, 6. b1∩ TIM1 → A-∩ B-, 7. A-∩ B- → a0∩ b0∩ R2, 8. R2∩ a0∩ b0∩ c0 → C+, 9. C+ → c1, 10. c1 → C-, 11. C- → a0∩ b0∩ c0∩ R1∩ **(SET)CNT2,** **12. CNT2∩ TIM2 → (SET)CNT1.** **Rules = 12, Operators = 29**
Seq._5 (Seq._4 with one emergency stop added)	1. ((Pb∩ a0)∪ (R3∩ c0)∪ R1)∩ R2'∩ R4'∩ R6'∩ **R7'** → R1, 2. R1 → A+, 3. R1∩ a1 → B+, 4. ((R1∩ b1∩ TIM1)∪ R2)∩ R3'∩ R4'∩ R6'∩ **R7'** → R2, 5. b0'∩ c0' → (RST)TIM1, 6. R2 → B-, 7. R2∩ a1 → A-, 8. R2∩ b0∩ a0 → C+, 9. ((R2∩ c1)∪ R3)∩ R1'∩ R4'∩ R6'∩ **R7'** → R3, 10. R3 → C-, 11. c0∩ R5∩ CNT1 → R4, 12. Pb' → (RST)CNT1, 13. R4∩ TIM2 → R5, 14. Pb' → (RST)TIM2, 15. R5∩ c0∩ CNT2 → R6, 16. R5' → (RST)CNT2, **17. ES → R7.** **Rules = 17, Operators = 56**	1. Pb → R1∩ (SET)CNT1 , 2. a0∩ R1∩ CNT1∩ **ES'**→ A+, 3. A+∩ **ES'** → a1, 4. a0∩ b0∩ **ES'** → B+, 5. B+∩ **ES'** → b1, 6. b1∩ TIM1∩ **ES'** → A-∩ B-, 7. A-∩ B-∩ **ES'** → a0∩ b0∩ R2, 8. R2∩ a0∩ b0∩ c0∩ **ES'** → C+, 9. C+∩ **ES'** → c1, 10. c1∩ **ES'** → C-, 11. C-∩ **ES'** → a0∩ b0∩ c0∩ R1∩ (SET)CNT2, 12. CNT2∩ TIM2 → (SET)CNT1. **Rules = 12, Operators = 39**

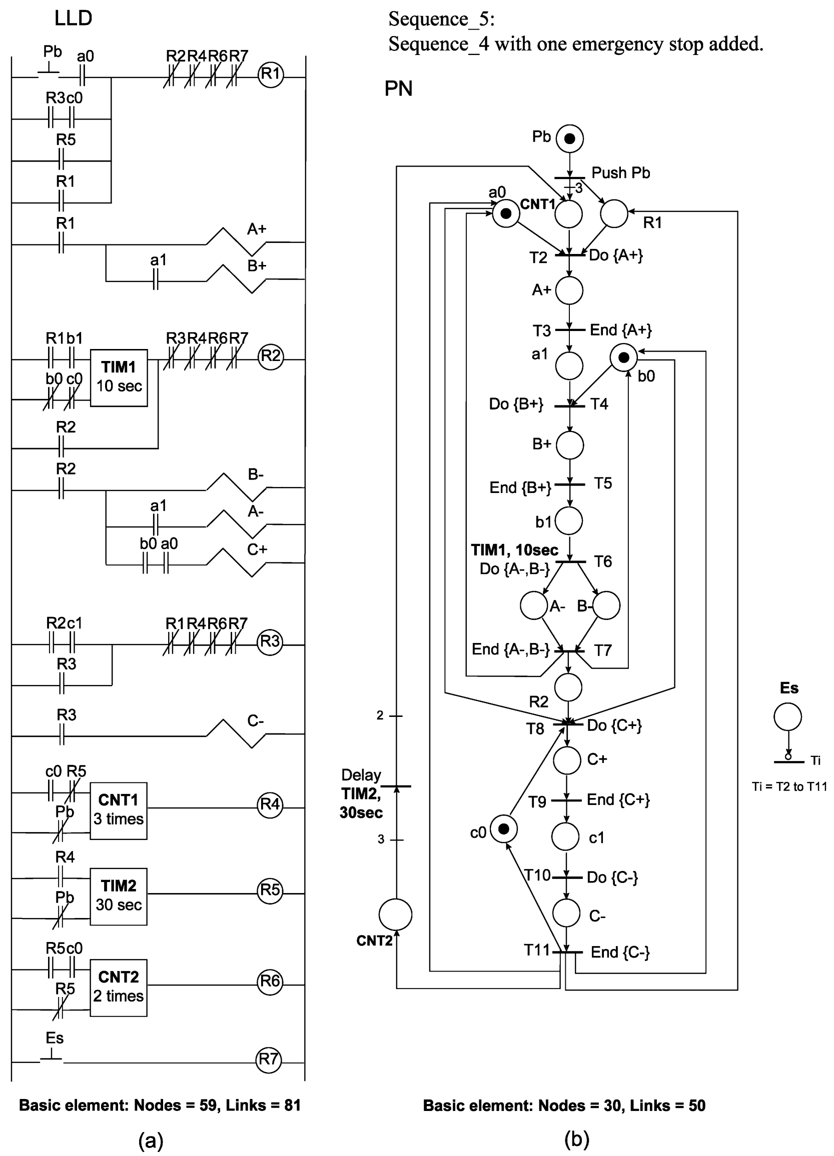


Fig. 7 LLD and PN for Sequence_5

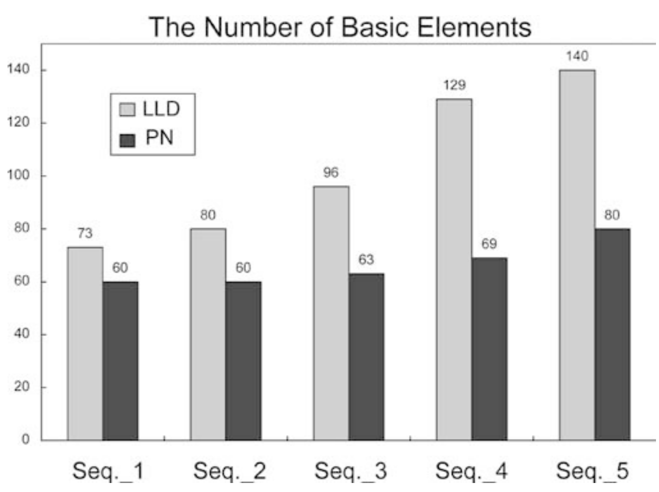


Fig. 8 Required basic elements in the basic element approach

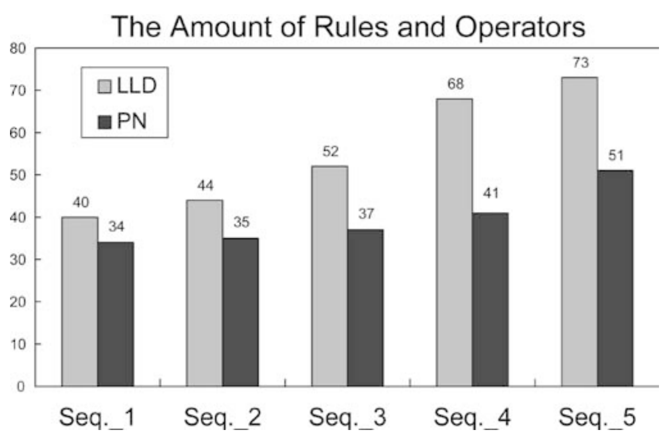


Fig. 9 Required rules and logical operators in the IF-THEN transformation

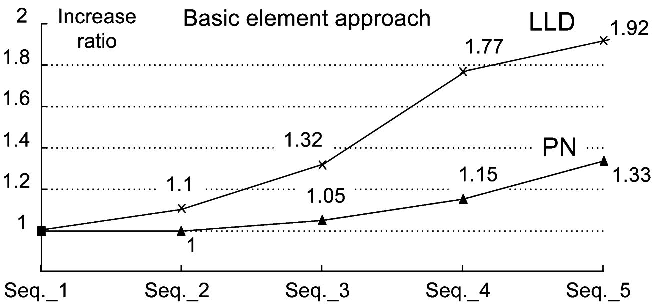


Fig. 10 The increase ratio for the basic element approach

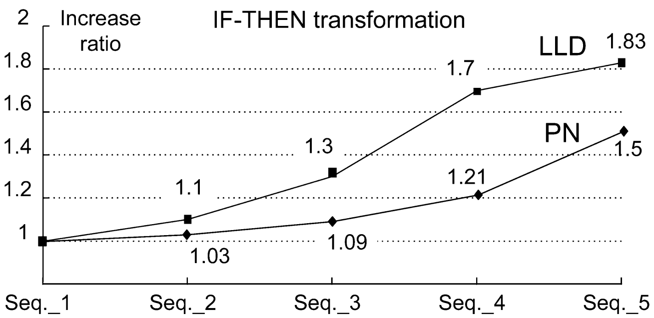


Fig. 11 The increase ratio for the IF-THEN transformation

Acknowledgements The authors would like to thank NASA system engineer Peter Graube for his valuable suggestions. This work was supported by the National Science Council, R.O.C., under Grant NSC 89-2218-E-009-035.

References

1. Venkatesh K, Zhou MC, Caudill RJ (1994) Comparing ladder logic diagrams and Petri nets for sequence controller design through a discrete manufacturing system. IEEE Trans Ind Electron (special section on Petri nets in manufacturing) 41(6):611–619
2. Venkatesh K, Zhou MC, Caudill R (1994) Evaluating the complexity of Petri nets and ladder logic diagrams and for sequence controllers design in flexible automation. In: Proceedings of IEEE symposium on emerging technology and factory automation, pp 428–435
3. Zhou MC, Twiss E (1995) A comparison of relay ladder logic programming and Petri net approach for sequential industrial control systems. In: Proceedings of IEEE international conference on control applications, pp 748–753
4. Zhou MC, Twiss E (1998) Design of industrial automated systems via relay ladder logic programming and Petri nets. IEEE Trans Syst Man Cyber, part C 28(1):137–150
5. Frey G, Litz L (2000) Formal methods in PLC programming. In: Proceedings of IEEE international conference on systems, man, and cybernetics, Nashville, TN, October 2000, pp 2431–2436
6. Pessen DW (1989) Ladder-diagram design for programmable controllers. Automatica 25(3):407–412
7. Cheng FT, Yang HC, Kuo TL, Feng C, Jeng M (2000) Modeling and analysis of managers in manufacturing execution systems for semiconductor packaging. IEEE Trans Syst Man Cyber, part C (special issue on discrete systems and control) 30(5):772–782
8. Zhou MC, Venkatesh K (1998) Modeling, simulation and control of flexible manufacturing systems: a Petri net approach. World Scientific, Singapore
9. David R, Alla H (1994) Petri nets for modeling of dynamic systems—a survey. Automatica 30(2):175–202
10. Uzam M, Jones AH (1998) Discrete event control system design using automation Petri nets and their ladder diagram implementation. Int J Adv Manuf Technol (special issue on Petri nets applications in manufacturing system) 14(10):716–728
11. Janneck JW, Naedele M (1998) Modeling a die bonder with Petri nets: a case study. IEEE Trans Semiconduct Manuf (special section on Petri nets in semiconductor manufacturing) 11(3):404–409
12. Zimmermann A, Hommel G (1999) Modeling and evaluation of manufacturing systems using dedicated Petri nets. Int J Adv Manuf Technol 15(2):132–138
13. Zuberek WM (2001) Timed Petri nets in modeling and analysis of cluster tools. IEEE Trans Robotic Autom (special section on semiconductor manufacturing systems: modeling, analysis, and control) 17(5):562–575
14. Lee JS, Hsu PL (2002) Design of remote environmental monitoring systems. In: Proceedings of IEEE international conference on control applications, Glasgow, Scotland, September 2002, pp 856–861
15. Lee JS, Hsu PL (2002) A systematic approach for the sequence controller design in manufacturing systems. Int J Adv Manuf Technol (in press)
16. International Electrotechnical Commission (1993) Programmable controllers part 3: programming languages. IEC 1131-3, IEC Geneva
17. Boucher TO, Jafari MA, Meredith GA (1990) Petri net control of an automated manufacturing cell. Adv Manuf Eng 2(3):151–157
18. Looney CG, Alfize AR (1987) Logic control via Boolean rule matrix transformations. IEEE Trans Syst Man Cyber 17(6):1077–1082