Vol. 2, No. 3, 2012, 29–34

Shahrood University of Technology

Journal of Solid and Fluid Mechanics

A rule-based evaluation of ladder logic diagram and timed petri nets for programmable logic controllers

A. A. Pouyan[1,*], Z. Mazinanian[2] and R. Shams[2]

[1] **Assist. Prof., School of Computer Engineering, Shahrood University of Technology, Shahrood, Iran**
[2] **M.Sc. Student in Computer Engineering, Shahrood University of Technology, Shahrood, Iran**

Abstract

This paper describes an evaluation through a case study by measuring a rule-based approach for designing sequence controllers. The increasing complexity and functionality of modern discrete event manufacturing systems have challenged the traditional tools and design methods based on ladder logic diagrams (LLDs) for designing programmable logic controllers (PLC). Furthermore, while the complexity and functionality of manufacturing systems increases, designing flexible, reusable, and maintainable control software becomes more difficult. Petri nets as a high level specification language are an emerging technique in designing control software systems for complex manufacturing systems. It is used for modeling, analysis and simulation of industrial automated systems. The proposed rule based approach tends to a unified measurement. This is performed in three levels. System complexity increases in a step by step and incremental fashion level by level. We have shown that, when the levels are more complex, Petri nets are more tractable and more verifiable. Moreover, it can be concluded that Petri nets are more efficient than ladder logic diagrams in designing control software systems.

Keywords: Ladder logic diagram; Timed Petri nets; Programmable logic controllers; Manufacturing systems.

1. Introduction

Programmable logic controllers (PLC) apply automation in control systems. PLC let users to program simply and directly. Ladder logic diagram (LLD) is the most used language for programming PLCs.

Heuristic methods, apply to program LLD for simple systems easily. Design of controller is more difficult, when systems are more complex. So, the implementation of LLD is more complicated. Because of graphical representation in LLD program designs, they are difficult to debug and modify [5]. Hence, to achieve superior approaches for system modeling, analysis, simulation and evaluation, researchers must be pursuing systematic and efficient PLC programming.

Recently, Petri nets (PN) for designing sequential controllers attracted attentions and became a popular tool in manufacturing systems by representing a powerful graphical and mathematical modeling. But in reality, the PN approach is not well known by most engineers. In fact, most industrial PLC users still prefer to program in LLDs. Thus, realistic comparisons between the advantages of using LLD and PN are required, especially for complex manufacturing systems [1-4].

Boucher et al. [6] reported that controllers would be more tractable if they use PN instead of LLD. However, they were not quantifying comparison between PN and LLD to design sequential controllers, formally. Venkatesh et al. [7, 8] expressed a measure to compare complexity and response time of PN and LLD by proposing a number of nodes and links in them, called “basic elements”. Zhou and Twiss [9, 10] also used basic elements of PN and LLD to compare their flexibility and verify the correctness. Both of the last researches showed that PN is more efficient than LLD. It is significant that basic elements in LLD and PN have different physical meaning. Thus, these comparisons may terminate unacceptable results. Lee and Hsu [1] proposed a method for evaluating LLD and PN by the IF-THEN transformation. They converted both LLD and PN in to similar IF-THEN format to

* Corresponding author, Phone: (+98)273-3393116
Email: apouyan@shahroodut.ac.ir

achieve a unified comparison via a simple example of five sequences with increasing complexity.

In this paper, an approach is proposed to compare LLD and Time Petri nets (TPN) using IF-THEN transformation. An integrated comparison is obtained by measuring the sum of the number of IF-THEN rules and number of operators for both LLD and TPN. The proposed approach is explained by a three level example for an assembly process system and its complexity increases level by level.

The reminder of this paper is organized as follows. First, LLD is defined in section 2. Then, section 3 describes TPNs that can be used for analysis and model production systems. Section 4 introduces IF-THEN formats. Afterwards, section 5 provided an application of an assembly process system that produces desks to achieve the proposed approach and conclusions are presented in section 6.

2. Ladder logic diagram (LLD)

LLD is the most common used language to develop software for PLCs. It is a graphical language and represents programs by graphical diagrams [5]. LLD is based on physical relayed diagrams, where power flows from the left to the right and top to bottom. The diagram includes a rectangular box with two vertical rails and a set of horizontal rungs, while their inputs are on the left and their outputs are on the right side. Thus, the power passes through inputs to active outputs. Each rung and its connected elements indicate a Boolean equation. Series variables represent AND function while parallel variables represent OR function.

Figure 1 shows a simple LLD. According to what was illustrated, the Boolean equation will be

$$N = ((A \wedge B) \vee N) \wedge \neg M \quad (1)$$

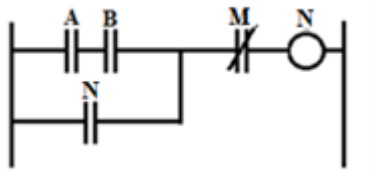


Fig. 1. Simple ladder logic diagram

We can consider, LLD programs having difficulty in detecting faults, visualize and unreadable even for small programs and as control specifications change, LLD program needs to be modified significantly [1, 11].

3. Timed Petri nets (TPN)

Petri nets are tools for modeling approach. They can be indicated as a bipartite graph with two node types, called places and transition. Directed arc connect the nodes together. Connections between two nodes of the same type are not allowed. A Place is symbolized with a circle and a Transition with a rectangle. Every Place contains an integer number of Tokens. Distribution of tokens can indicate state of system [12].

There are many definitions for Petri net. Formally, a Petri net is a five-tuple PN= (P, T, I, O, m0) where [12]:

- P = {p1, p2, …, pk} , k > 0 is a finite set of places,
- T = {t1, t2, …, tl} , l > 0 is a finite set of transition (considering and),
- I: P × T → N is an input function that specifies weights of arcs directed from places to transitions,
- O: P × T → N is an output function that specifies weights of arcs directed from transitions to places,
- M: P → {0, 1, 2, …} is a marking, m0 is the initial marking.

There are some extended Petri nets which solve certain problems, and color Petri nets, timed Petri nets also prioritized Petri nets are among them. In the previous section, PN model was described without time dimension. Time was not considered when a transition occurred.

Timed Petri net is alike Petri net with the addition of time structure for each time transition. In this model we consider time when a transition is being fire. The firing rules in this model are that a transition fire when it is enabled and firing transition occurred in finite time [13].

Formal representation of the introduced TPN is [2]: TPN = (P, T, I, O, s0, f), where:

- P, T, I, O are the same as above,
- s0 is the initial state of TPN,
- f: T → R0+ is the function that assigns a non-negative time-delay to every tj T.

4. IF-THEN transformation

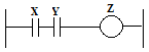
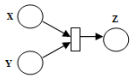
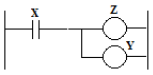
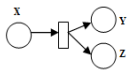
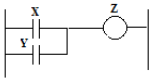
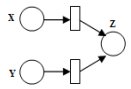
Complexity and response time are the most important factors for the comparison of LLD and PN. Complexity is measured by the physical size of the model, while response time is influenced by both physical size and the hardware of implementation.

PLC programs are logical and sequential. In other words, if an event occurred, then the next event can be executed. Hence, to achieve a unified approach, both LLD and TPN transform into IF-THEN format and then, they can be compared based on the number of rules and logical operators.

The IF-THEN categories for LLD and PN are shown in Table 1. Models use smaller numbers of rules and operators are easier to understand, debug, check

and maintain. Therefore, the summation of rules and operators can signify the complexity and response time for both LLD and PN structures [1].

Table 1. IF-THEN rules for LLD and PN [1]

IF-THEN rules	LLD	PN
IF X and Y THEN Z X∩Y → Z		
IF X ,THEN Y and Z X → Y∩Z		
IF X or Y THEN Z X∪Y → Z		

5. An application of assembly process system

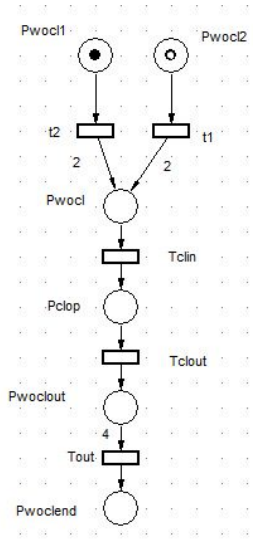
Our approach will be explained on a model of an assembly system, where desks are produced. The scheduling to manage several departments in production system can be performed using TPNs. The process starts with a punching machine. Then, assembly lines, galvanization lines and paint line will complete the product. In our model, only a simple part of the production process will be evaluated. As shown in Table 2 there are four types of different products, which have been planned to be produced. The request of how many products should be produced and their starting time are also shown [2]. In this approach, only producing the left corner clamp is considered. Now, the hierarchical sequences can be applied to our approach in three levels with increasing complexity.

Table 2. List of required products [2]

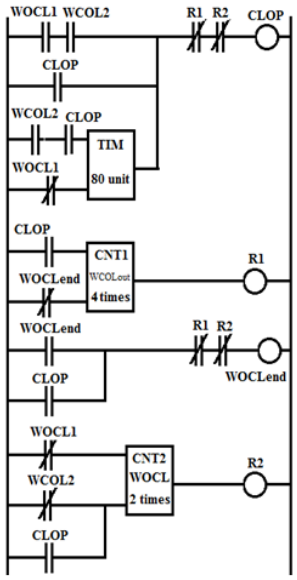
Product	Quantity	Start Time
Corner clamp L	2	0
Corner clamp L	2	80
Corner clamp R	2	0
Clamp Holder	2	0
Angle-bar	2	0

5.1. Sequence_1

In this level, the left corner clamp just needs one time slice to be produced. As it is shown in Table 2, there are two left corner clamps with different start times: the first one has to start immediately with its production and the second one has to start 80 time units later. The LLD and TPN of this level are shown in Figure 2. Pwocl1 and Pwocl2 are starting places; Pclop is the state of corner clamp operation and the weight of each arc shows the quantity of places. Starting times are defined with the tokens in the starting places, which transition t2 will be enable to fire in the beginning of production and transition t1 fires 80 time units later.



(a)



(b)

Fig. 2. TPN (a) [2] and LLD (b) for sequence_1

As we can see in Figure 2 (a), operation will be fulfilling in place Pclop. However, in next levels, this place will be extending. In the following level and according to the LLD and TPN diagrams, the IF-THEN rule should be applied to achieve a unified evaluation. Both transformation are being calculated and gathered in Table 3.

Table 3. IF-THEN formats of LLD and TPN in Fig 2

LLD	TPN
1) ((WOCL1∩WOCL2)∪(WOCL2∩CLOP ∩TIM)∪CLOP)∩ R1' ∩ R2' → CLOP	1) WOCL1 → (SET) WOCL
2) WOCL1' → (RST) TIM	2) WOCL2 ∩ t2 → (SET) WOCL
3) CLOP ∩ CNT1→ R1	3) WOCL → CLOP
4) WOCLend' → (RST)CNT1	4) CLOP → (SET) WOCLout
5) (WOCLend ∪ CLOP) ∩ R1' ∩ R2' → WOCLend	5) WOCLout → WOCLend
6) CLOP ∩ CNT2→ R2	
7) WOCL1' ∩ WOCL2' → (RST)CNT2	
Rules: 7, Operators: 24	Rules: 5, Operators: 6

However, the complexity of the first sequence was simple and it had a simple TPN as well, its LLD diagram and LLD IF-THEN rules are rather complicated. Thus, in the coming levels only TPN will be considered.

5.2. Sequence_2

In this sequence, four different sub-items are needed to perform in producing the left corner clamp ('Angle-bar with holder L', 'Clamp', 'Nut' and 'Screw') [2]. When those sub-items are produced, the place named Pclop in the previous TPN, changes into four parallel lines and after the production completed, transition TCL2in will be enable to fire and tokens in place PCL2op shows that, sub-items are produced. The situation described is shown with a TPN structure in Fig 3. The Number of rules depends on the number of transitions. Thus, in this level number of TPN rules increases but it is still readable. IF-THEN rules are shown in Table 4.

5.3. Sequence_3

In the final sequence, routing data are needed to build the sub-items in more detail, with lower level of abstraction. Therefore, one operation is needed to produce the sub-item 'screw', two operations to produce 'Nut' and 'Clamp' and three operations to produce the 'Angle-bar with holder L'[2]. That is in each operation a higher level has been extended to more operations with sufficient detail. This process can be extended to lower levels of abstraction until attaining the satisfactory level of detail. Figure 4 is the TPN model of corner clamp L with all details mentioned in previous sequences. The right line in Figure 4, shows 'Angle-bar with holder L', which needed three sub-operations to be completed. First sub-operation depends on two parallel lines and when transition TCL2inA is enabled, it is fired (performing the job). The second sub-operation can be completed when transition TCL3inA fires. The third transition will be fulfilled after place PCL3Aop gets enough tokens and etc. IF-THEN formats are depicted and considered in Table 5.

Table 4. IF-THEN formats of TPN in Fig 3

TPN
1) WOCL1 → (SET) WOCL
2) WOCL2 ∩ t2 → (SET) WOCL
3) WOCL → CLinA ∩ CLinC ∩ CLinN ∩ CLinS
4) CLinA → CL1Aop
5) CL1Aop → CLoutA
6) CLinC → CL1Cop
7) CL1Cop → CLoutC
8) CLinN → CL1Nop
9) CL1Nop → CLoutN
10) CLinS → CL1Sop
11) CL1Sop → CLoutS
12) CLoutA ∩ CLoutC ∩ CLoutN ∩ CLoutS ∩ Desk1 → CL2op
13) CL2op → CL2 ∩ Desk1
14) Cl2 → (SET)WOCLout
15) WOCLout → WOCLend
Rules: 15, Operators: 20

Table 5. IF-THEN formats of TPN in Figure 4

TPN	
1) WOCL1 → (SET) WOCL	12) CL1Nop ∩ RG2→ R1 ∩ CL2Nop
2) WOCL2 ∩ t2 → (SET) WOCL	13) CLinS ∩ RG2 → CL2Sop
3) WOCL → CLinA ∩ CLinC ∩ CLinN ∩ CLinS	14) CL1AoutB ∩ RDesk2 ∩ CL1AoutH→ CL2Aop
4) CLinA → CL1AinH ∩ CL1AinB	15) CL2Aop ∩ RG1 → RDesk2 ∩ CL3Aop
5) CL1AinB → CL1ABop	16) CL3Aop → CLoutA ∩ RG1
6) CL1ABop → CL1AoutB ∩ Pr1 ∩ R2	17) CL2Cop → RG2 ∩ CLoutC
7) Pr1 ∩ Cl1AinB ∩ R1 → CL1AHop	18) CL2Nop → RG2 ∩ CLoutN
8) CL1AHop → CL1AoutH ∩ R1	19) CL2Sop → RG2 ∩ CLoutS
9) CLinC ∩ R1 → CL1Cop	20) CLoutA ∩ CLoutC ∩ CLoutN ∩ CLoutS ∩ RDesk1 → CL2op
10) CL1Cop ∩ RG2 → R1 ∩ CL2Cop	21) CL2op → CLout ∩ RDesk1
11) CLinN ∩ R1 → CL1Nop	
Rules: 21, Operators: 43	

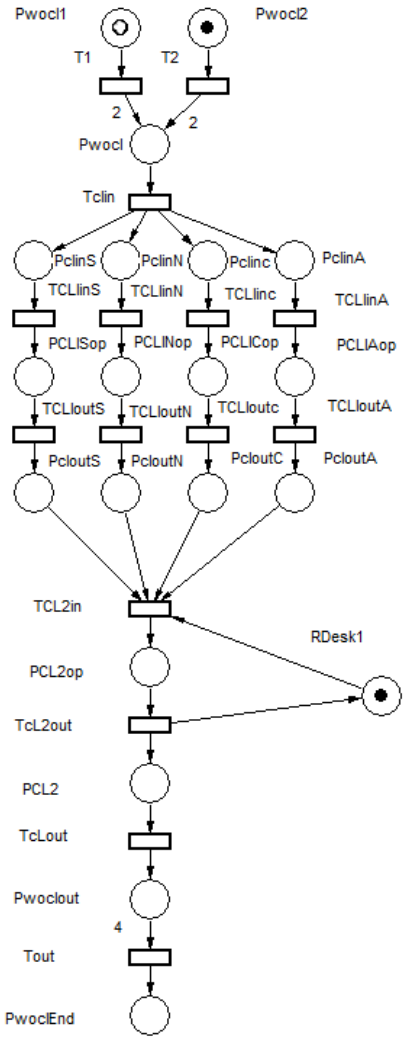


Fig. 3. TPN for sequence_2 [2]

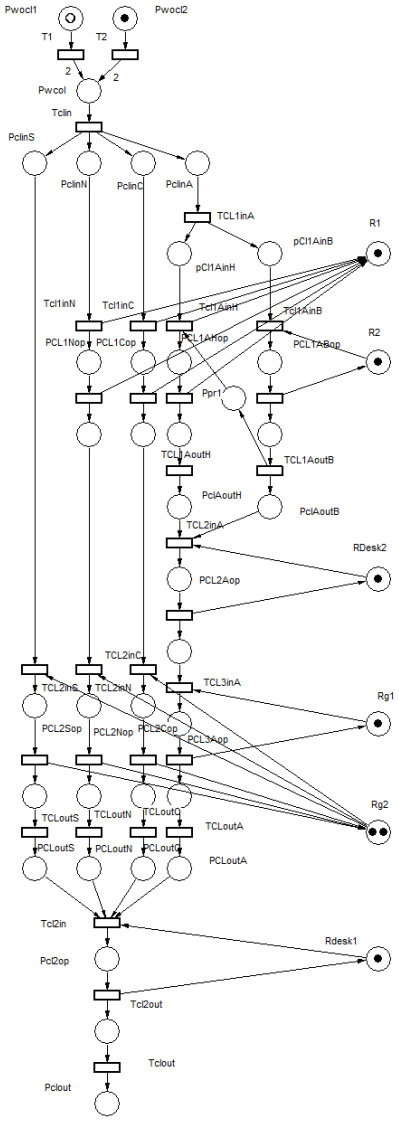


Fig. 4. TPN for sequence_3 [2]

6. Conclusions

This paper presents a unified approach to evaluate the LLD and TPN by IF-THEN transformation. Results show that when systems become more complex, TPN structures (in fact, PN structures), have lower increase rate than LLD structures. Thus, they are more stable. If the complexity of a system increases, the number of rungs in LLD is obviously increased but the number of transitions is more stable. PNs are also more readable than LLDs. Thus, they can be more tractable. System complexity increases in a step by step and incremental fashion, level by level. Based on the proposed approach, we have shown that, when the levels are more complex, Petri nets are more tractable and more verifiable and reliable. Moreover, it can be concluded that Petri nets are more efficient than ladder logic diagrams for designing control software systems in terms of flexibility and reliability.

References

[1] Lee JS, Hsu PL (2004) An improved evaluation of ladder logic diagrams and Petri nets for the sequence controller design in manufacturing systems. Int J Adv Manuf Technol 24: 279–287.

[2] Gradisar D, Music G (2006) Petri-net modelling of an assembly process system based on production

management data Modelling, Simulation, Verification and Validation of Enterprise Information Systems, Proceedings of the 3rd International Workshop on Modelling, Simulation, Verification and Validation of Enterprise Information Systems, MSVVEIS 2005, In conjunction with ICEIS 2005, Miami, FL, USA, May 2005; 01/2005

[3] Pouyan AA, Toossian Shandiz H, Arastehfar S (2011) Synthesis a Petri net based control model for a FMS cell. Computers in Industry 62: 501–508.

[4] Lee JS, Hsu PL (2005) A systematic approach for the sequence controller design in manufacturing systems. Int J Adv Manuf Technol 25: 754–760.

[5] Bender DF, Combemale B, Crégut X, Farines JM, Berthomieu B, Vernadat F (2008) Ladder metamodeling & PLC program validation through time Petri Nets. ECMDA-FA: 121–136.

[6] Boucher TO, Jafari MA, Meredith GA (1990) Petri net control of an automated manufacturing cell. Adv Manuf Eng 2: 151–157.

[7] 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: 611–619.

[8] Venkatesh K, Zhou MC, Caudill RJ (1994) Evaluating the complexity of Petri nets and ladder logic diagrams and for sequence controllers design in flexible automation. Proceedings of IEEE symposium on emerging technology and factory automation: 428–435.

[9] Zhou MC, Twiss E (1995) A comparison of relay ladder logic programming and Petri net approach for sequential industrial control systems. Proceedings of IEEE international conference on control applications: 748–753.

[10] 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: 137–150.

[11] Liu Y (2009) A Petri Net-Based ladder logic diagram design method for the logic and sequence control of photo mask transport. ICIC: 774–783.

[12] Uzam M, Jones AH (2002) A new Petri-net-based synthesis technique for supervisory control of discrete event systems. Turk J Elec Engin 10: 85–109.

[13] Guo J, Li Z (2010) A deadlock prevention approach for a class of timed Petri nets using elementary siphons. Asian Journal of Control 12: 347–363.