DYNAMIC PETRI NET BASED DISCRETE EVENT CONTROLLER

Hussein H. Hussein
Benha High Institute of Technology
Benha, Egypt
s_hussein73@yahoo.com

Ahmed A. El-Betar
Benha High Institute of Technology
Benha, Egypt
dacbetar@yahoo.com

Magdy M. Abdel Hammed
Faculty of Engineering,
Ain Shams University
Cairo, Egypt
magdyma@gmx.net

Farid A. Tolbah
Faculty of Engineering,
Ain Shams University
Cairo, Egypt
ftolbah1@yahoo.com

Abstract

Previously, the Petri nets were used as modeling tools not a designing tool for **d**iscrete **e**vent **c**ontrollers (DEC's). In addition, there is a lacking to extract the mathematical model of the controller based on the Petri net. In this paper, a **D**ynamic **P**etri **N**et (DPN) is proposed to design a controller for the **d**iscrete **e**vent **s**ystems (DES's). The design procedure of the DPN based DEC is introduced. The proposed procedure has been investigated through designing controllers for many case studies, which represent real industrial discrete event flexible manufacturing systems. One of these controllers has been extensively tested experimentally. Both the theoretical and experimental results of the performance of this controller are introduced. Comparing and analyzing these results have been confirmed the validity of the proposed DPN based DEC and showed that the proposed technique offers many advantages such as its mathematical model base beside the graphical representation of the whole DES.

الملخص:

في المعتاد تستخدم شبكات بترى كأداة لنمذجة منظومات التحكم الرقمية وليس لتصميم متحكمات أنظمة التحكم الرقمية، هذا بالإضافة الى وجود قصور عند استخراج النموذج الرياضى للمتحكم اعتمادا على شبكة بترى بجانب عدم تتبعها للأحداث الانتقالية عند بعض الفترات (غير ديناميكية). وفى هذا البحث تم اقتراح شبكة بترى ديناميكية تستخدم لتصميم متحكم لمنظومات التحكم الرقمية, كذلك تم تقديم الخطوات التصميمية للطريقة المقترحة واستخدامها لتصميم متحكمات لبعض الحالات الدراسية والتى تمثل أنظمة تحكم رقمية صناعية حقيقية. وقد أجريت إختبارات معملية مكثفة على إحدى هذه المتحكمات. وتم تقديم النتائج النظرية والمعملية لأداء هذا المتحكم. وبدراسة وتحليل ومقارنة تلك النتائج فقد تأكدت صلاحية الطريقة المقترحة. كما أنها تتميز باعتمادها على النموذج الرياضى والتمثيل التخطيطى لمنظومة التحكم الرقمية والذى يمكننا من معرفة والتأكد من صلاحية المتحكم قبل تطبيقه عمليا بما لا يحتاج إلى طريقة التجربة والخطأ والتصحيح كبعض الطرق السابقة المستخدمة فى نفس المجال.

Keywords: Petri nets, discrete-event control systems, real-time implementation, controllers, automated manufacturing systems

1. Introduction

Discrete event control system (DECS) has been getting more attention in the last decade due to the need to automate and increase productivity of the manufacturing system. A discrete event manufacturing system consists of several units such as machines, robots, robotic assembly cells, flexible manufacturing systems, production lines, communication systems and computer based systems which function asynchronously to meet the dynamic changing needs [1-3]. In discrete event systems, the signals are transmitted in discrete time intervals in the form of ON/OFF signals either to the inputs or to the outputs of the discrete event system. The control of the DES is characterized by the control of processes, machines and equipments which are essentially ON/OFF in their behavior and the system operates according to the state of the machine to be controlled. Among the different types of discrete event control systems are those known as sequential control. Sequential control is considered as the heart of most control systems. Sequential control is characterized by current events being dependent on past events [4]. The other type of the discrete event control system is the combinational one, which is characterized by the current event being independent on past events [5-9].

In reviewing the efforts made to design the DEC's, it is apparent that various design methods for these controllers have been introduced. A drum timer [5] was one of the first methods to design a sequential controller. It is used in automatic washing machines, textile machines, semi-automatic cutting machines, engine camshafts and other applications. Ring counter and modular sequencer methods [6] are considered important methods used for designing sequential controllers. Their main shortcoming is attributed to the presence of redundant logic components in the derived sequential controller. The concepts of finite state automata and Boolean algebra [7,8] were used to design a sequential control of either synchronous or asynchronous type. According to these concepts, extraction of the equations representing the sequential controller requires long and tedious computations. The majority of sequential control applications currently use a programmable controller as a sequential controller. Programmable controllers commonly use a graphical language called ladder logic language. Difficulties of programmable controllers based on ladder diagrams were pointed out as follows [9]: (a) the way of implementing a sequential control protocol has an inherently slow response time, which increases linearly with program size because of continuous scanning of the entire program; (b) ladder logic language is not well-suited as a sequential control programming language. It is rather a combinational method and does not describe the sequential progression of the operation; (c) it depends on a trial and error technique to design a ladder language program.

Nowadays the attention is concentrated toward the flexible controllers, which are positively responding to change of the functions of the controlled system as Petri net [1-3, 10-21]. Petri nets, as graphical and mathematical tools, provide a uniform technique for modeling and analysis of DES's [1,10,17]. One of the major advantages of using Petri net models is that the same model is used for the analysis of behavioral properties and performance evaluation, as well as for systematic construction of discrete event simulators and controllers [1,10]. Caloini et. al. [11] presented a technique for designing software for robotic controllers based on Petri net. This technique depends on what they titled control nets. This work was deficient in a mathematical model of the control system. Another type of Petri net called automation Petri net (APN) is proposed by Uzam and Jones [12]. Who concluded that the APN is a suitable tool to design and implement the DECS. Also, they introduced a formal structure of APN which is able to read the state of sensors and write to the actuator outputs as well as converting the APN to a ladder diagram to be implemented on

a programmable logic controller. But it used various types of arcs with different weights. The main drawback of [11,12] is that a place may contain any positive integer number of token which makes the PN's model unbounded, unsafe, unreachable, and unlived. Feldmann et. al. [13] introduced in part I of their work the ordered color Petri nets in combination with some extensions as a modeling tool for logic controllers while they presented a technique to automatically generate code for a programmable logic controller from a validate textual description of this kind of Petri nets in part II [14]. Sun et. al [2] used the Petri net to model FMS through two sub models. One is the transportation model and the other is the process flow model. Tsuji et. al. [15] suggested a representation of ladder diagram components by Petri net components. Venkatech et. al [3] proposed a technique of Petri net based on the ladder logic diagram to design a sequence controller of a DES. In this technique, the intermediate states were not included. This was considered the main shortcoming of that work. Marranghello [16] described the digital system using Petri net and he used this description to synthesize the corresponding hardware.

From the previous survey regarding using Petri net as a designing tool for DEC, it can be concluded that: i. Most of these works used the ladder logic diagram as a basis for generating the Petri net, which mean that the Petri net was used as a molding tool, not a design tool, ii. Many types of arcs were used during constructing the Petri net model, which complicated these models, iii. There was a lack to extract the mathematical model of the controller based on the Petri net.

This paper presents a proposed technique of designing a DEC based on Petri net. The proposed technique will covers all the shortcomings of those reviewed before. The paper is organized as follows. Following section presents a discussion for the concept of a DECS and a formulation of the problem addressed in this paper. The proposed design of DEC by Dynamic Petri net is presented in section three. In section four, application of the proposed technique on different case studies are included, while the experimental work is introduced in section five, experimental results and discussion are introduced in section six, Finally the conclusions drawn from this work are presented.

2. Preliminaries and Problem Formulation

In this section, an overview of the Petri net description, a discrete event system definition and the formulation of the problem are introduced.

2.1 Overview of ordinary Petri net

The ordinary Petri net is a graphical and mathematical modeling tool, which is able to model concurrent, asynchronous, distributed and parallel systems [3,17-20]. This Petri net consists of eight different components:

$$N = (P, T, I, C, M, D, X, Y) \tag{1}$$

where: P is m finite set of places, T is n finite set of transitions with $P \bigcup T \neq 0$ and $P \bigcap T = 0$, I: $P \times T \rightarrow N$, is an input function that defines the set of directed arcs from P to T, C: $P \times T \rightarrow N$, is an output function that defines the set of directed arcs from T to P, $M : P \rightarrow N$, is a marking matrix whose its components represent the number of tokens in each place. An initial marking is denoted by M_o, $D : T \rightarrow R^+$, is a firing time function where

R^+ is the set of non negative real numbers, $X : P \rightarrow \{-,0,1,2,.....K\}$ and $X(p_i) \neq X(p_j), i \neq j$, is an input signal function, where K is the maximum number of input signal channels, and "-" is the dummy attribute indicating no assigned channel to the place and $Y : T \rightarrow L$ is an output signal function, where L is a set of integers [1,10,14,19,20]. A token is a basic Petri net component and signifies that a particular place, or condition, is true.

Definition 2.1.1: Enabling of transitions

(i) Enabling Rule: A transition is said to be enabled if each input place *p* of transition *t* contains at least the number of tokens equals to the weight of the directed normal arc connecting *p* to *t*. Details of enabling rules exist in Uzam and Jones [12].

(ii) Firing Rule: A firing of an enabled transition *t* removes the number of tokens equal to the weight of the directed arc connecting *p* to *t* from each input place *p*. It also deposits a token with the weight of the directed arc connecting *t* to *p* in each output place. [1,3,12,17,18].

Definition 2.1.2: Incidence Matrix A

Let $P = (p_1,....., p_m)$ and $T = (t_1,....., t_n)$, the incidence matrix; denoted by A; can be calculated as follows [1,15,18]:

$$A = A^+ - A^- \tag{2}$$

where A^+ is an $n \times m$ matrix: $A^+(i, j) = W_N(t_j, p_i)$. W_N is the function determined the weights of the arcs. Similarly A^- is an $n \times m$ matrix and it is equal to $A^-(i, j) = W_N(p_i, t_j)$.

Definition 2.1.3: Petri net mathematical model

The mathematical model of Petri net can be viewed as follows:

$$M_k = M_{k-1} + A^T U_k \tag{3}$$

where M_k is the marking vector, A is an n×m matrix called the incidence matrix, U_k is the firing vector ($n \times 1$) with entries zero except for k^{th} row entry which equals to 1, k is a vector index =1, 2,.....n, and n is the number of transitions. M is an m×(n+1) matrix called the marking matrix.

2.2 Discrete Event Control System

Discrete event control systems may be combinational or sequential systems. A combinational system outputs at any time are determined directly from the present combination of input without regard to previous inputs while the outputs of sequential systems are a function of both the current and the previous input states. So, the sequential system is a combinational system with feedback states.

A sequential machine is a system that can be characterized by a quintuple [20],

$$DECS = (\sum, Q, Z, \delta, \lambda) \tag{4}$$

where: Σ= finite nonempty set of input symbols σ_1, σ_2,……, σ_l, Q = finite nonempty set of states q_1, q_2,……, q_n, Z = finite nonempty set of output symbols, z_1, z_2,…..z_m, δ = next-state function, which maps $Q \times \sum \to Q$, λ = output function which maps $Q \times \sum \to Z$

2.3 Problem Formulation

Figure 1 represents a block diagram of the proposed DECS. A typical discrete DECS consists of a collection of subsystems, which are driven by actuators and detected by sensors working in discrete nature. These subsystems are required to be controlled via a controller, which is discrete in nature. Petri nets will be adapted to design such controller. However, ordinary Petri nets can not deal with actuators or sensors [12]. So, a proposed technique based on Dynamic Petri net will be introduced in this work.

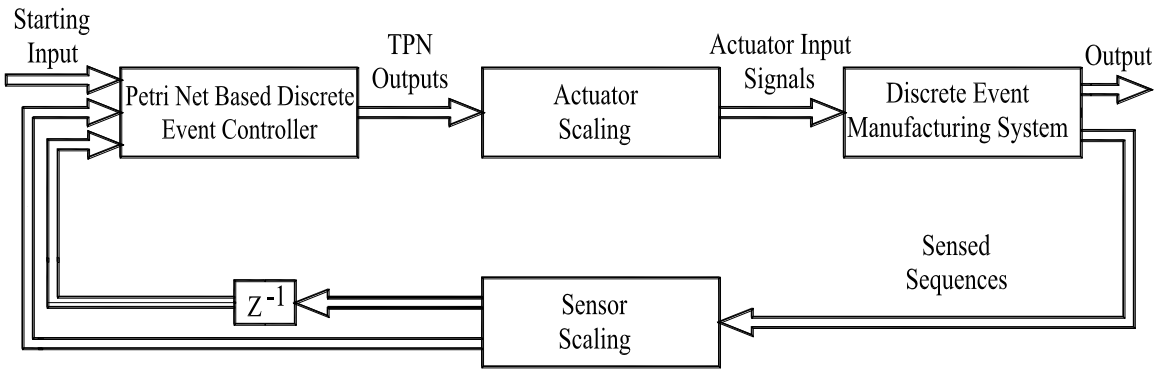


Fig. 1: Block diagram of the discrete event control system.

The objectives of this paper are to establish a proposed design procedure based on both mathematical model and graphical representation of a DEC, introduce a verification tool to validate the controller function, apply the proposed design procedure on different case studies with experimental investigation. The proposed design procedure is based on the DPN. The proposed DPN can deal with different actuators and sensors including the starting, closing and intermediate states of the DECS.

3. Design of Discrete Event Controller Based on Dynamic Petri Net

Nowadays, there is a need to design flexible, reusable, and maintainable control software as DEC. All these requirements can be achieved by using the DPN. Moreover, the DPN is promising as a very important tool to provide an integrated solution for modeling, analysis, simulation and control of the DES. So, this section is focused on the DPN design procedure for the DES.

3.1 The Proposed Dynamic Petri Net Based Controller

The proposed DPN based DEC has been introduced. The proposed DPN is seven components and can be presented as follows:

$$DPN = (P, T, S, E, O, A, M). \quad (5)$$

where:

1- P is a finite set of places. A place is used to represent the condition of the DES. These conditions include both the conditions of inputs and the output sequences of the DECS. So the places of a DPN model of a DES consist of m_i places which represent the number of input conditions and m_s places which represent the number of output sequences. Both m_i and m_s are determined from the DES. The total number of places (m) can be calculated from:

$$m = m_i + m_s \quad (6)$$

The place is presented by a circle in the DPN model.

2- T is a finite set of transitions where $P \bigcup T \neq 0, P \bigcap T = 0$. A transition is used to represent the instantaneous action of the actuator(s). The execution of an action or event is modeled by two transitions, starting and closing transitions. The starting transition represents the beginning of the action while the closing transition signifies the instantaneous termination of the action. The last event in the transition state table is not indicating any transition because it represents the home or idle position. Hence the total number of transitions denoted by n can be calculated from:

$$n = 2v - 2 \quad (7)$$

where v is the total number of events represented by rows in the transition state table of the DES. The transition is presented by rectangular in the DPN model. An output sequence place is put between the starting and closing transitions to model the progress of the actuator action.

3- S is a set of arcs; called starting arcs; connecting a set of places P to a set of transitions T. Each arc of S is coming from a place which has already contained a token. From this S set, one can define starting weights of these arcs, W_s. These starting weights W_s are defined by:

$$W_s : P \times T \rightarrow N \quad (8)$$

where $N \in \{0,1\}$.

4- E is a set of arcs; called ending arcs; connecting a set of transitions T to a set of places P. Each arc of E is entering to a place that will be activated when firing is occurred. From this E set, one can define the ending weights of these arcs, W_e. These ending weights *We* are defined by:

$$W_e : T \times P \rightarrow N. \quad (9)$$

From 3, 4, it is obviously clear that at any place S must be equal to E except at the place which represents the push button. Each arc is presented by a single line with arrow ended in the DPN model.

5- O is a token which is represented by a small filled circle denoted by "●" inside a place in the DPN model. This token indicates the ON state in the DES. The token considered in this work is featured by its weight which never exceeds one. The movement of token inside the complete DPN model is accomplished by to the enabling and firing rules.

6- A is the incidence matrix and it was explained in definition 2.1.2.
7- M is the marking matrix of dimensions $(m \times (n+1))$ and it is a function of the tokens O, W_s and W_e. The first column of this matrix is denoted by M_o which is the initial marking $(m \times 1)$ column vector representing the number of tokens in each place at the starting position of the DES. The other vectors of M are determined from:

$$Iff \quad M_{i-1}(p_j) = W_s(p_j, t_i) \; then \; M_i(P) = W_e(t_i, p_j) \quad \forall p_j \in P \tag{10}$$

where $i = 1 \rightarrow n,\ j = 1 \rightarrow m$ and $M_i(p_j)$ denotes a token in place p_j related to marking vector M_i while $M_i(P)$ imply the marking vector number i.

The DPN proposed in this work has the following features compared to other Petri nets considered earlier:

1. Only one type of places is considered to build the DPN model rather than multi types of places as given in [2].
2. All types of arcs should have a weight equal to one rather than different arc weights as specified in [3,10,12-15].
3. The flow of input and output signals are modeled by sets of arcs; S arcs and E arcs. The S arcs (P-T arcs) of the starting transitions represent the flow of the input signals of DES while the E arcs (T-P arcs) of the closing transitions represent the flow of the output signals of DEC. This is contrasting to earlier types of PN's [3].
4. The token inside a place in the DPN model never exceeds one rather than any positive integer number of token which may lead to unbounded, unsafe, unreachable, and unlive PN model [3,10,12-15].
5. The extraction of the DPN, model is based on the order of occurrence and the transition state table of the DECS instead of its dependence on the ladder diagram or Boolean algebra as in [3,15].

3.2 Basic considerations

This section provides some considerations regarding the design of the controller based on DPN. These considerations can be presented as follows:

1. Each actuator has at least two sensors, one to indicate the beginning of its action and the other to indicate the ending of its action.
2. Each row in the transition state table represents an event.
3. For each event in the transition state table, there should be at least an ON state of its input states. If this condition is not exist, that will lead to a deadlock situation. To avoid deadlock; "No enabled transitions", there is a need to add a dummy place(s) that represents the previous state of any actuators or sensors.
4. Either the sensor or the actuator has only two states (ON or OFF states).
5. The intermediate state represents an event in which there is no contact between the actuator and its sensors. In this case, there is no need to add a new output sequence place because the output sequence place which is responsible for generating this intermediate state was added before.
6. Extracting the DPN model of the DES begins from the starting condition.

7. There are no similar events inside the transition state table of the DES, but one criterion that defines the sequential DES is the similarity among rows or events in the transition state table. So, to solve this problem, the previous state of one or more limit switches is to be added in the transition state table. The number of these previous states is determined from this rule: $SR \equiv 2^{n1}$ where $n1$ represents the number of previous states input added, and SR represents the number of similar rows.

3.3 Design Procedure of DEC based on DPN

In the section, the proposed design procedure of a DEC based on the DPN has been introduced. The design procedure can be explained as follows; see Fig. 2:

1- According to the function of the DECS, one can define the number, type, function and states of each sensor and actuator and assign the proper output sequences according to the function of the DES.
2- Derive the corresponding transition state table, or master control sheet of the DECS. The transition state table (TST) or the master control sheet contains the starting state which indicates the state of the DES at the beginning its operation. This state is presented in the first raw of the TST. In addition, the idle or home state of the DES which represents the system at finishing its operation is included in the TST.
3- Determine the total number of places as defined in section 3-1 using the expression (6)
4- Build the Dynamic Petri net model based controller according to the following steps:
   a. Draw the input places m_i representing the input conditions of the DES
   b. Add a token in each input place which its state is ON in the starting state.
   c. Represent each event, except the last one, by two transitions one for starting this event (starting transition) and the second for closing of this event (closing transition) where the total number of transitions is n is given in expression (7).
   d. Draw an output sequence place between the starting and closing transitions of each event, considering that in dealing with the intermediate states, do not add an output sequence place between these transitions because it has been already added before.
   e. For i =1 to v –1, add arcs according to the following rules:
      i. Draw two arcs connecting the output sequence places with their starting and closing transitions. The first arc is directed from the starting transition and terminated at its output sequence place while the second arc is directed from the output sequence place and terminated at the closing transition
      ii. Draw a set of arcs, which are directed from input places already containing a token in the i^{th} event and terminated at the starting transition of the i^{th} event. This set of arcs defines the input function, where $S : (P \times T) \rightarrow N$.
      iii. Draw another set of arcs directed form the closing transition of the i^{th} event and terminated at input places which predicted to have a token in the $(i+1)^{th}$ event when firing is occurred. This set of arcs defines the output function, where $E : (T \times P) \rightarrow N$.
5- To ensure correct firing, check the equality of the S-set of arcs and E-set of arcs must be equal at all places except for the place that represents the start push button.
6- Determine the incidence matrix [A] according to Eq. (2).
7- Extract the M matrix by assigning the initial marking M_o from the DPN model. M_o is a column vector with dimension $m \times 1$ and represents the number of tokens in each place at the starting position, then identify the successive marking M_i (i=1 to n) by firing each transition according to its order in the DPN model.

8- Finally, the mathematical model of the DEC can be defined as given in Eq. (3). Noting that the marking matrix M combines both the output and input vectors; i.e. M_k represents the output vector while M_{k-1} represents the input vectors, k=1,2,…,n

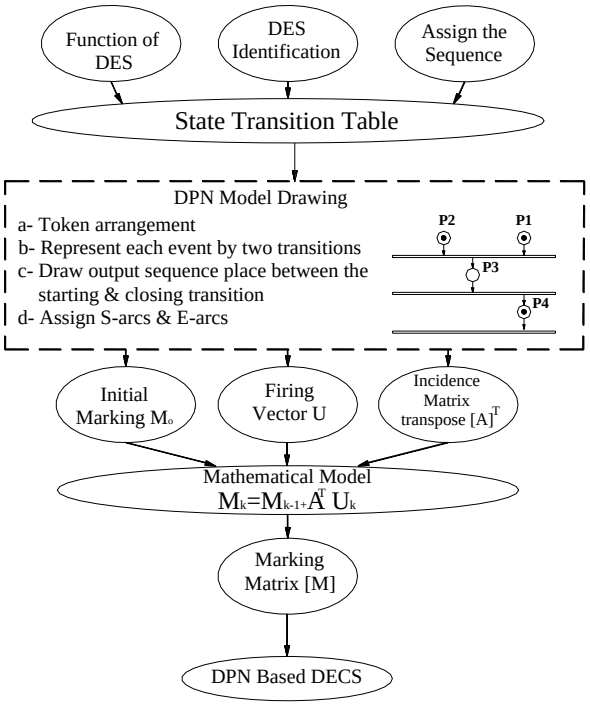


Fig. 2: Schematic diagram of the design procedures

4. Applications of the DPN Based DEC on Different Case Studies

To assure the validity of the proposed DPN based DES controllers, the design procedure considered in this work has been applied to design controllers for many industrial systems. In this section, we introduce only two case studies. These case studies are:

Case study 1: DPN based controller for two tanks filling.

Case study 2: DPN based controller for a pick and place sequences.

4.1 Case study 1: DPN Based Controller for Two Tanks Filling

This case study is inspired from Rene [21]. In this case study, it is required to design a DPN based sequential controller for two tanks; tank1 and tank 2; filling as shown in Fig. 3. For each tank, there are two sensors b_i and h_i (i =1, 2) to indicate the minimum and maximum level respectively. Also each tank has two valves V_i and W_i (i=1, 2) for flow rate in and flow rate out respectively. Both tanks are used in a similar way. Tank 1 is empty when the level is less than b_1, i.e., b_1=0, and is full when the level is greater than h_1, i.e., h_1=1. At the initial state, both tanks are empty. If push button (m) is pressed, both tanks are filled by opening valvesV_1 (V_1=1 means that valve V_1 is opened) and V_2. When one tank is full, e.g., tank 1, filling stops (by closing valve V_1) and its contents start to be used (by opening valve W_1). When tank 1 is empty, valve W_1 is closed. For this system, five inputs are used; one as start button m and two level sensors (b_i, h_i) for each tank. The system has four outputs corresponding to four valves.

The two tanks are to be controlled according to the following sequences:

a) Begin with two tanks filling by pushing the start button SW1.
b) Fill tank 1 and tank 2 simultaneously; designated by V_1 and V_2; until h_1=1, h_2=1.
c) Start tank 1 discharge; designated by W_1; until b_1=0.
d) Finally, discharge of tank 2; designated by W_2; until b_2=0.

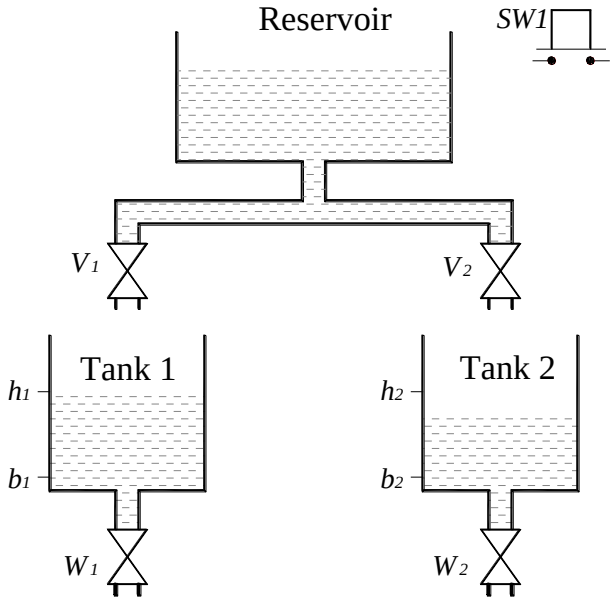


Fig. 3: Two tanks filling

Derivation of the TST: By tracing the series of events that take place, according to the required sequences, the TST can be derived as given in table 1. In table 1, the input combinations A_i *(i=1 to 5)* of the proposed controller and their outputs Y_j *(j=1 to 4)* are arranged in columns while the sequence steps to be accomplished by the controller are defined in rows. The entries of the table should be ones and zeros. It is one for an ON state input switch while it is zero for OFF state switch. In addition, for outputs, the entry of the table is one when a valve is opened while it is zero when a valve is closed.

Table 1: The TST of case study 1.

Step order	*Input states*					*Actuator states*			
	SW1	b_1	H_1	b_2	h_2	v_1	w_1	v_2	w_2
	A_1	A_2	A_3	A_4	A_5	Y_1	Y_2	Y_3	Y_4
1	1	0	0	0	0	1	0	1	0
2	0	0	0	0	0	1	0	1	0
3	0	1	0	0	0	1	0	1	0
4	0	1	0	1	0	1	0	1	0
5	0	1	1	1	0	0	1	1	0
6	0	1	0	1	0	0	1	1	0
7	0	0	0	1	0	0	0	1	0
8	0	0	0	1	1	0	0	0	1
9	0	0	0	1	0	0	0	0	1
10	0	0	0	0	0	0	0	0	0

To avoid similarity between the fourth and the sixth rows and between the seventh and ninth rows, according to the item 7 in section 3.2, the previous state of the V_1 and V_2 are added to Table 1. In addition, a deadlock appears in the second row. This deadlock; and according to the item 3 in section 3.2; is avoided by the previous addition of the previous states of the V_1 and V_2. Table 2 shows the TST of the modified control actions. Fig. 4 depicts the theoretical timing diagram of case study 1.

Table 2: The modified TST of case study 1

Step order	*Input states*							*Actuator states*			
	SW1	b_1	H_1	b_2	h_2	$v_{1(n-1)}$	$v_{2(n-1)}$	v_1	w_1	v_2	w_2
	A_1	A_2	A_3	A_4	A_5	A_6	A_7	Y_1	Y_2	Y_3	Y_4
1	1	0	0	0	0	0	0	1	0	1	0
2	0	0	0	0	0	1	1	1	0	1	0
3	0	1	0	0	0	1	1	1	0	1	0
4	0	1	0	1	0	1	1	1	0	1	0
5	0	1	1	1	0	1	1	0	1	1	0
6	0	1	0	1	0	0	1	0	1	1	0
7	0	0	0	1	0	0	1	0	0	1	0
8	0	0	0	1	1	0	1	0	0	0	1
9	0	0	0	1	0	0	0	0	0	0	1
10	0	0	0	0	0	0	0	0	0	0	0

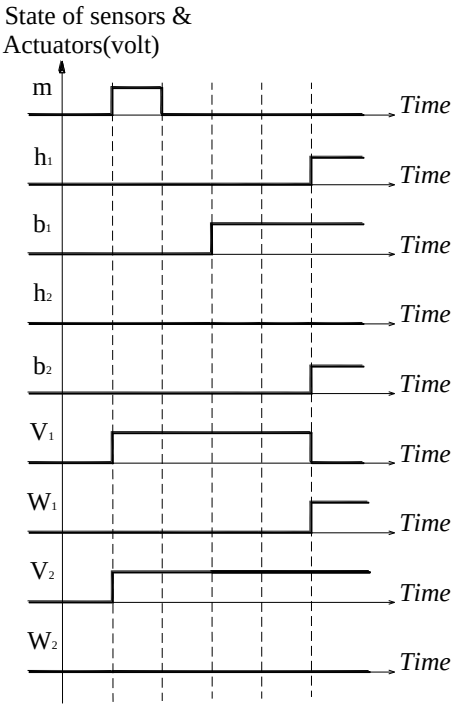


Fig. 4: Theoretical timing diagram of case study 1

Structure of the DPN model: As given in table 2, seven input nodes A_i *(i=1 to 7)*; represent seven input state places; $m_i = 7$. The number of required sequences are four where there is a four outputs Y_j *(j=1 to 4)* having two changes; from 0 to 1 and from 1 to 0; $m_s = 4$. So the total number of places denoted by m can be calculated from Eq. (6) to be *m*= 11. The number of transition is calculated from Eq. (7) and referrs to the number of events in Table 2 where ν= 10, the total number of transitions *n*= 18. The DPN model of case study 1 can be extracted according to step 4 of the design procedure. The DPN model of this case study is shown in Fig. 5.

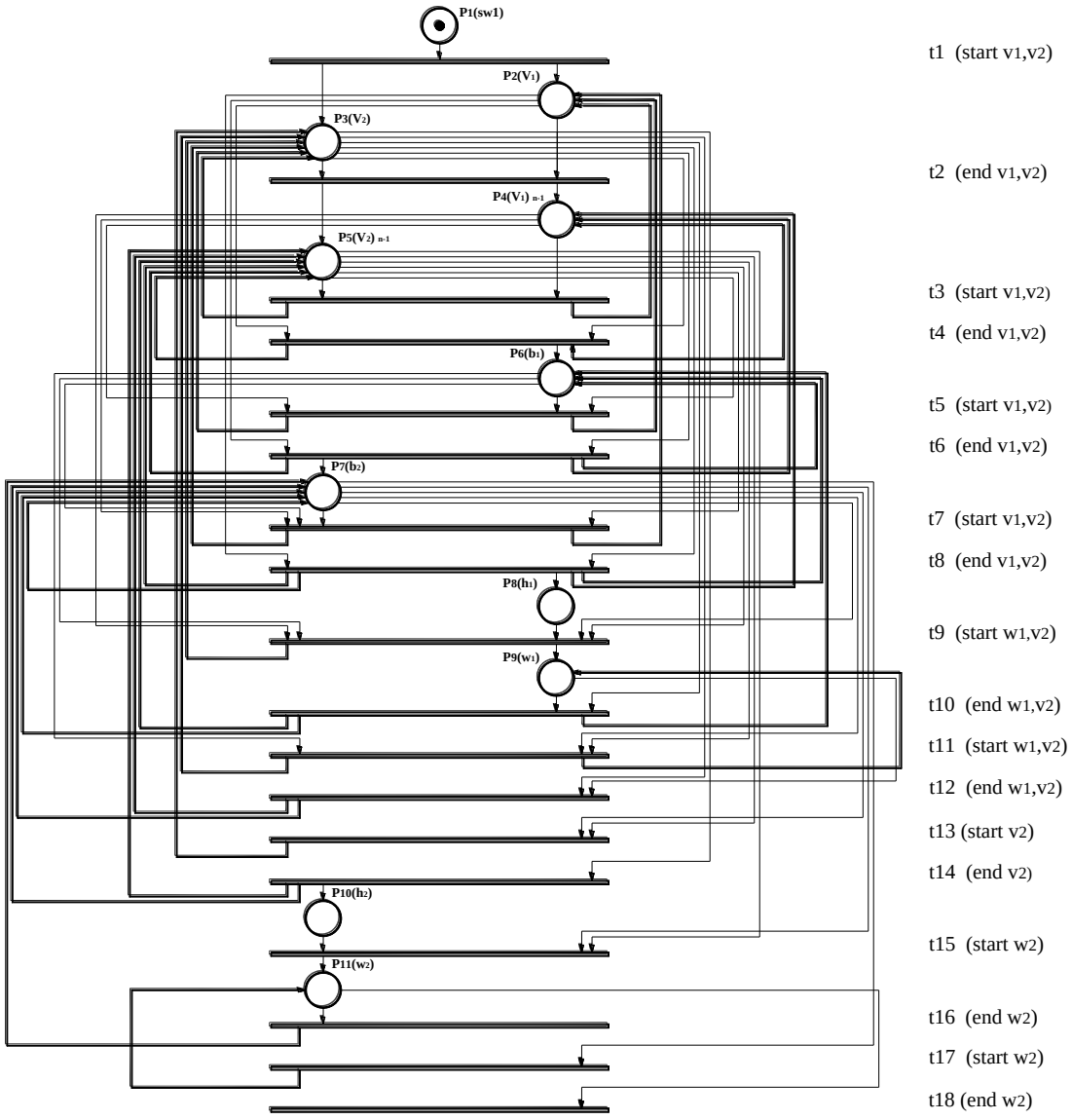


Fig. 5: DPN Model of case study 1

The Mathematical model of the DEC: From the model shown in Fig. 5. The incidence matrix transpose A^T can be determined and given in the form of Eq. (11). The initial marking denoted by M_o is equal to $\{1,0,0,0,0,0,0,0,0,0,0\}^T$. The Marking matrix *M* of this case study can be obtained by successive transitions firing according to Eq. (10) and it is given in Eq. (12).

$$[A]^T = \begin{array}{c} \begin{matrix} t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & t_8 & t_9 & t_{10} & t_{11} & t_{12} & t_{13} & t_{14} & t_{15} & t_{16} & t_{17} & t_{18} \end{matrix} \\ \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \end{bmatrix} \end{array} \begin{matrix} p_1(SW1) \\ p_2(V_1) \\ p_3(V_2) \\ p_4(v_1)_{n-1} \\ p_5(v_2)_{n-1} \\ p_6(b_1) \\ p_7(b_2) \\ p_8(h_1) \\ p_9(W_1) \\ p_{10}(h_2) \\ p_{11}(W_2) \end{matrix} \tag{11}$$

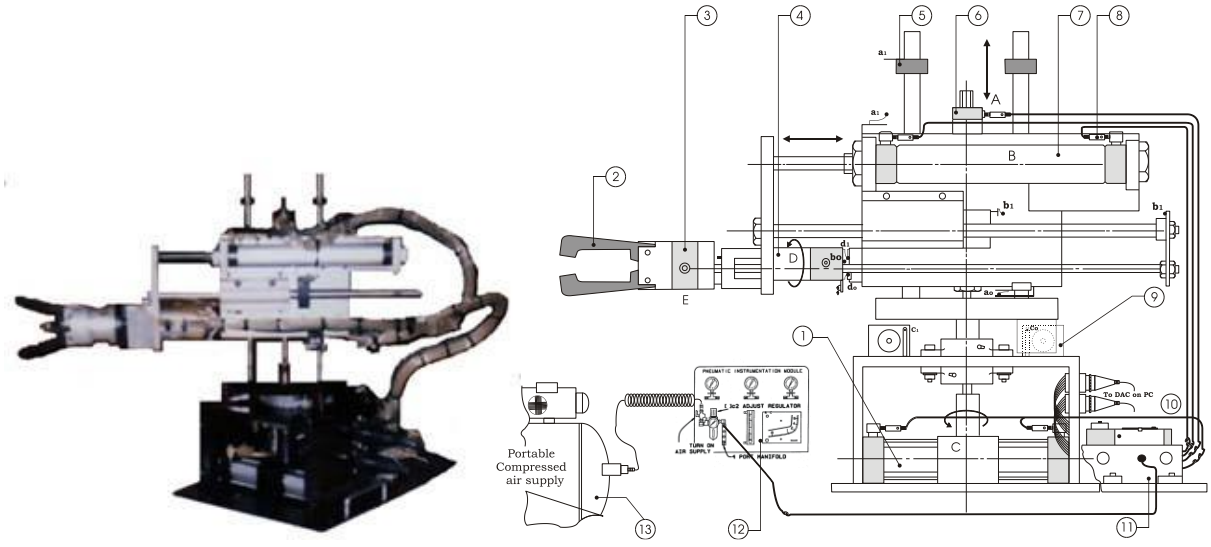
$$[M] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{matrix} p_1(SW1) \\ p_2(V_1) \\ p_3(V_2) \\ p_4(v_1)_{n-1} \\ p_5(v_2)_{n-1} \\ p_6(b_1) \\ p_7(b_2) \\ p_8(h_1) \\ p_9(W_1) \\ p_{10}(h_2) \\ p_{11}(W_2) \end{matrix} \tag{12}$$

Referring to the M matrix determined in Eq. (12), this matrix contains all the input and the output states of the DECS. The formation of this matrix depends on the successive transitions firing of the DES according to the DPN model based DEC.

4.2 Case study 2: Pick and place sequences

The industrial system considered in this case study is a Cartesian pneumatic robot; the Twinstar Robot TR; as shown in Fig.6a and b. The TR is to be controlled to pick and place a certain object in a flexible manufacturing systems. It consists of five double acting pneumatic cylinders; A, B, C, D and E. Each cylinder is driven by a 5/2 directional control valve, solenoid operated- spring in return. The TR has nine inputs corresponding to eight normally open limit switches; a_o, a_1, b_o, b_1, c_o, c_1, d_o, d_1; two limit switches; for cylinder A, cylinder B, cylinder C and cylinder D respectively; and one push button SW1 for the starting of the TR operation. The driving signals to this TR are five 110 AC voltage signals, one signal for each solenoid.

In this case study, only three cylinders A, B and C with their attachments are used for the pick and place sequences as shown in Fig. 7. Initially, the piston rods of these cylinders are at home positions. This means that all piston rods are fully retracted. Hence the limit switches a_o, b_o, and c_o are closed. The schematic layout of the pick and place sequences is shown in Fig. 7.



a **b**

Fig. 6: a. Industrial robot TR.

b. Detailed schematic layout front view of the TR.

The test rig (TWINSTAR) consists of: ① Pneumatic cylinder (Rack and pinion) C – for swing robot arm (rotation of robot arm),② Gripper (end effector), ③ Pneumatic cylinder E – opening and closing the gripper, ④ Pneumatic cylinder D – turning the gripper,⑤ Vertical stopper,⑥ Pneumatic cylinder A – up and down robot arm, ⑦ Pneumatic cylinder B – robot arm movement in and out,⑧ Flow control valve,⑨ Shock absorber, ⑩ solenoids, ⑪ direction control valve, ⑫ filter / Regulator, and ⑬ Portable compressed air supply

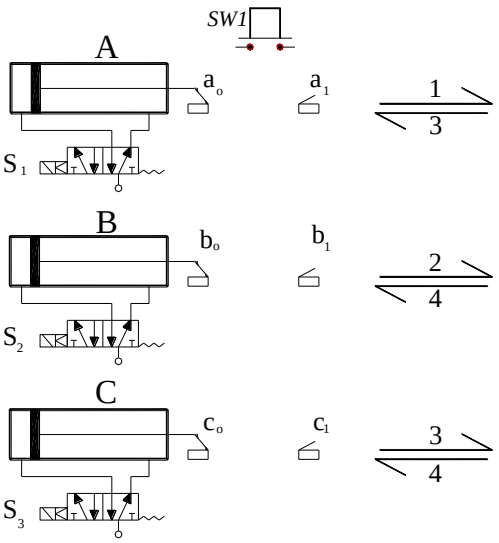


Fig. 7: Schematic layout of a pick and place sequences

The pick and place sequences are defined as follows:

a) Begin the TR running by pushing the *SW1* switch.
b) Extract the piston rod of cylinder A until it picks the object (designated by A^+). This rod remains fully extended.
c) The piston rod of cylinder B begins to advance (designated by B^+).
d) After complete extension of rod B, the rod C begins to advance while rod A begins to retract (designated by (C^+, A^-)).

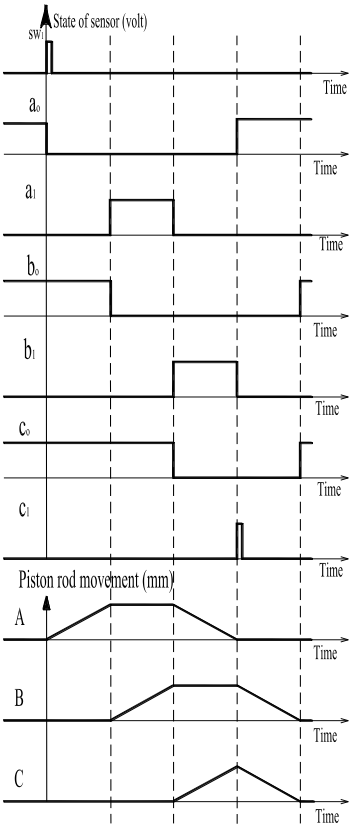
e) Finally, both the piston rods of B and C begin to retract simultaneously (designated by (B^-, C^-)).

This means that the required output sequences can be summarized to the following sequence: A^+, B^+, (C^+, A^-) and (B^-, C^-) where (C^+, A^-) and (B^-, C^-) represent two concurrent actions taking place simultaneously.

Derivation of the TST: As given in case study 1, Table 3 depicts the TST of case 2. In table 3, there are seven inputs A_i *(i=1 to 7)* and three outputs Y_j *(j=1 to 3)* of the proposed controller Fig. 8 shows the theoretical timing diagram of this case study.

Table 3: The TST of case study 2

Step order	*Input states*							*Actuator states*		
	SW1	a_o	A_1	b_o	b_1	c_o	c_1	S_1	S_2	S_3
	A_1	A_2	A_3	A_4	A_5	A_6	A_7	Y_1	Y_2	Y_3
1	1	1	0	1	0	1	0	1	0	0
2	1	0	0	1	0	1	0	1	0	0
3	0	0	0	1	0	1	0	1	0	0
4	0	0	1	1	0	1	0	1	1	0
5	0	0	1	0	0	1	0	1	1	0
6	0	0	1	0	1	1	0	0	1	1
7	0	0	0	0	1	0	0	0	1	1
8	0	1	0	0	1	0	1	0	0	0
9	0	1	0	0	0	0	0	0	0	0
10	0	1	0	1	0	1	0	0	0	0



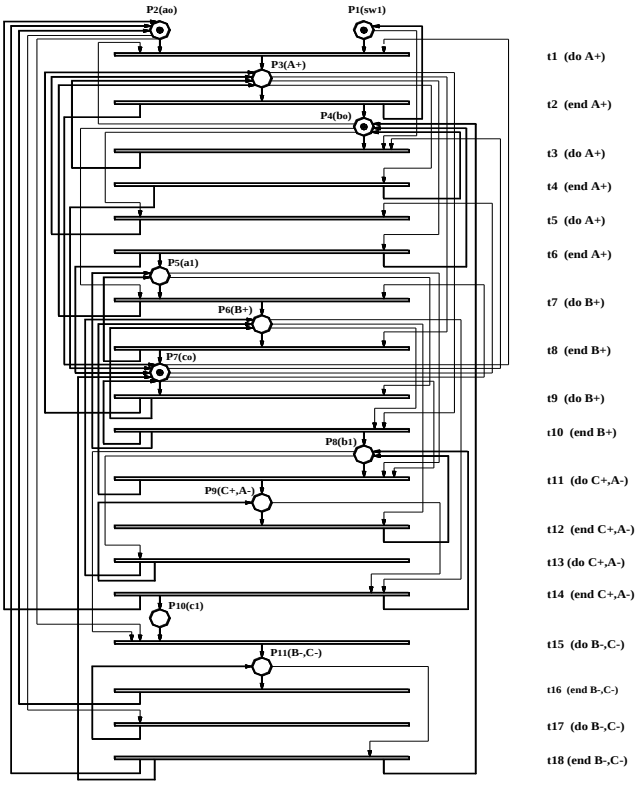


Fig. 8: Timing Diagram of case study 2 **Fig. 9: DPN model of case study 2**

Structure of the DPN model: In case study 2, the inputs conditions are seven, one condition for each sensor and SW1 switch, $m_i = 7$. The number of the required output sequences is four; $m_s = 4$. So the total number of places denoted by *m* can be calculated from Eq. (6) to be *m=11*. Referring to Table 3, there are ten events (*v*=10) which implies that the number of transitions according to Eq. 7 is: *n*=18. The DPN model of this case study is constructed according to the design procedure introduced in section 3-3 and shown in Fig. 9.

The Mathematical model of the DEC: Referring to case study 1, the incidence matrix transpose A^T is given in Eq. (13). The initial marking denoted by M_o is equal to $\{1,1,0,1,0,0,1,0,0,0,0\}^T$ .The marking matrix M of this case study can be obtained by successive transitions firing according to Eq. (3) and it is given in Eq (14).

$$[A]^T = \begin{array}{c} \begin{matrix} t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & t_8 & t_9 & t_{10} & t_{11} & t_{12} & t_{13} & t_{14} & t_{15} & t_{16} & t_{17} & t_{18} \end{matrix} \\ \begin{bmatrix} -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \end{bmatrix} \end{array} \begin{matrix} p_1(sw_1) \\ p_2(a_o) \\ p_3(A^+) \\ p_4(b_o) \\ p_5(a_1) \\ p_6(B^+) \\ p_7(c_o) \\ p_8(b_1) \\ p_9(C^+, A^-) \\ p_{10}(c_1) \\ p_{11}(B^-, C^-) \end{matrix} \quad (13)$$

$$[M] = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{matrix} p_1(sw_1) \\ p_2(a_o) \\ p_3(A^+) \\ p_4(b_o) \\ p_5(a_1) \\ p_6(B^+) \\ p_7(c_o) \\ p_8(b_1) \\ p_9(C^+, A^-) \\ p_{10}(c_1) \\ p_{11}(B^-, C^-) \end{matrix} \quad (14)$$

The marking matrix M determined in Eq. (14) represents both the input and the output vectors of the DEC.

5. EXPERIMENT

A test rig was designed and constructed in order to assess the applicability of the proposed DPN based DEC of case study 2 experimentally. Fig. 10 shows a block diagram of the layout of the test rig. A photograph of the setup including the control system is shown in Fig. 11. The experimental test rig consists mainly of the TR with its attachments, limit switches and the real time controller with its attachments.

The control of the industrial TR is carried out via intensive experiments using the proposed DPN based discrete event controller. This controller of case study 2 is tested and analyzed experimentally.

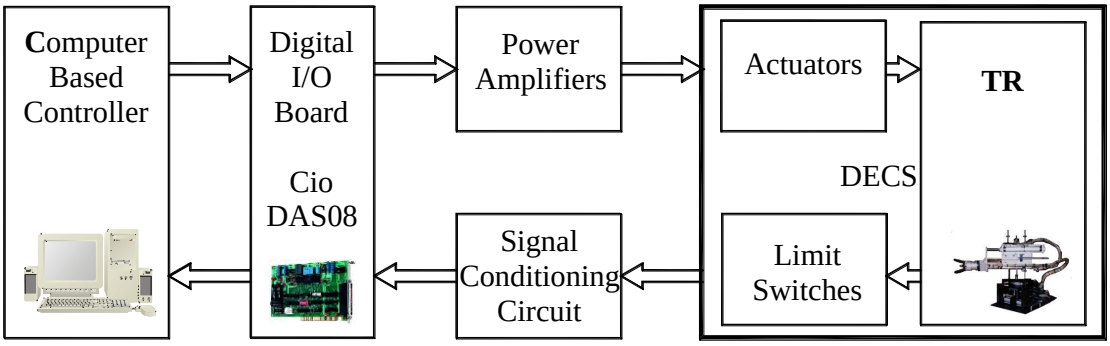


Fig. 10: Block diagram of the experimental test rig.

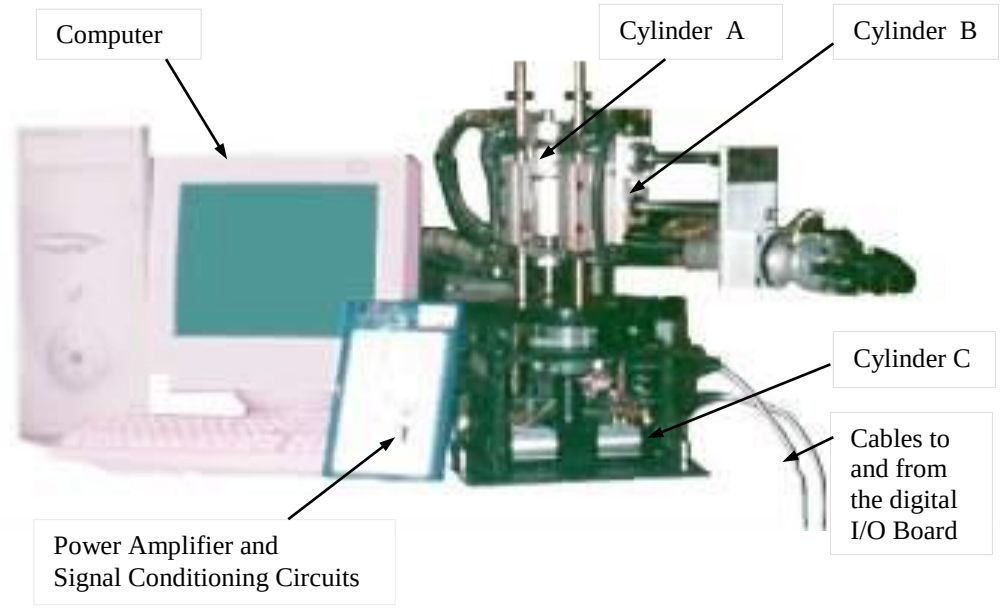


Fig. 11: Photograph of the test rig

The DPN based DEC has been programmed using Turbo Pascal 6 and implemented on a Pentium III, 700 MHz personal computer. The control interface analogue to digital and digital to analogue ADDA card; Cio-DAS 08; is *16* digital input/output channels and *8* analogue to digital channels. In addition, it has two independent digital to analogue channels. The resolution of this card is 12 bits. The control actions of the controller (*0– 5 volts*) are fed to the power amplifier. This power amplifier contains three channels with 2 amperes and 110 AC output volts. This amplifier drives the three solenoids of the directional control valves of the cylinders A, B and C. The limit switch signals of the TR are fed to the signal condition

circuit and then fed to the ADDA card. Finally, the limit switch signals are monitored to indicate the sequential operating cycle of the pick and place sequences.

6. Experimental Results and discussion

Figure 12 shows a hard copy of the control sequence diagrams of the experimental work. Fig. 12 contains ten diagrams. The logic state of the six limit switches, the state of the SW1 and the three output signals of the three solenoids are monitored in this figure. The state of the sensors or the solenoids when it is ON is represented by 1 (high voltage) while their OFF state is represented by zero volt (low voltage). Comparing these experimental results with the theoretical results shown in Fig. 8, it can be verified that the experimental sequence: A^+, B^+, $\{C^+,A^-\}$, and $\{B^-,C^-\}$ is matched with theoretical results. This indicates that the proposed designed DPN based DEC behaves exactly according to its desired function. This assures the effectiveness of the proposed design procedure of DPN-based DEC.

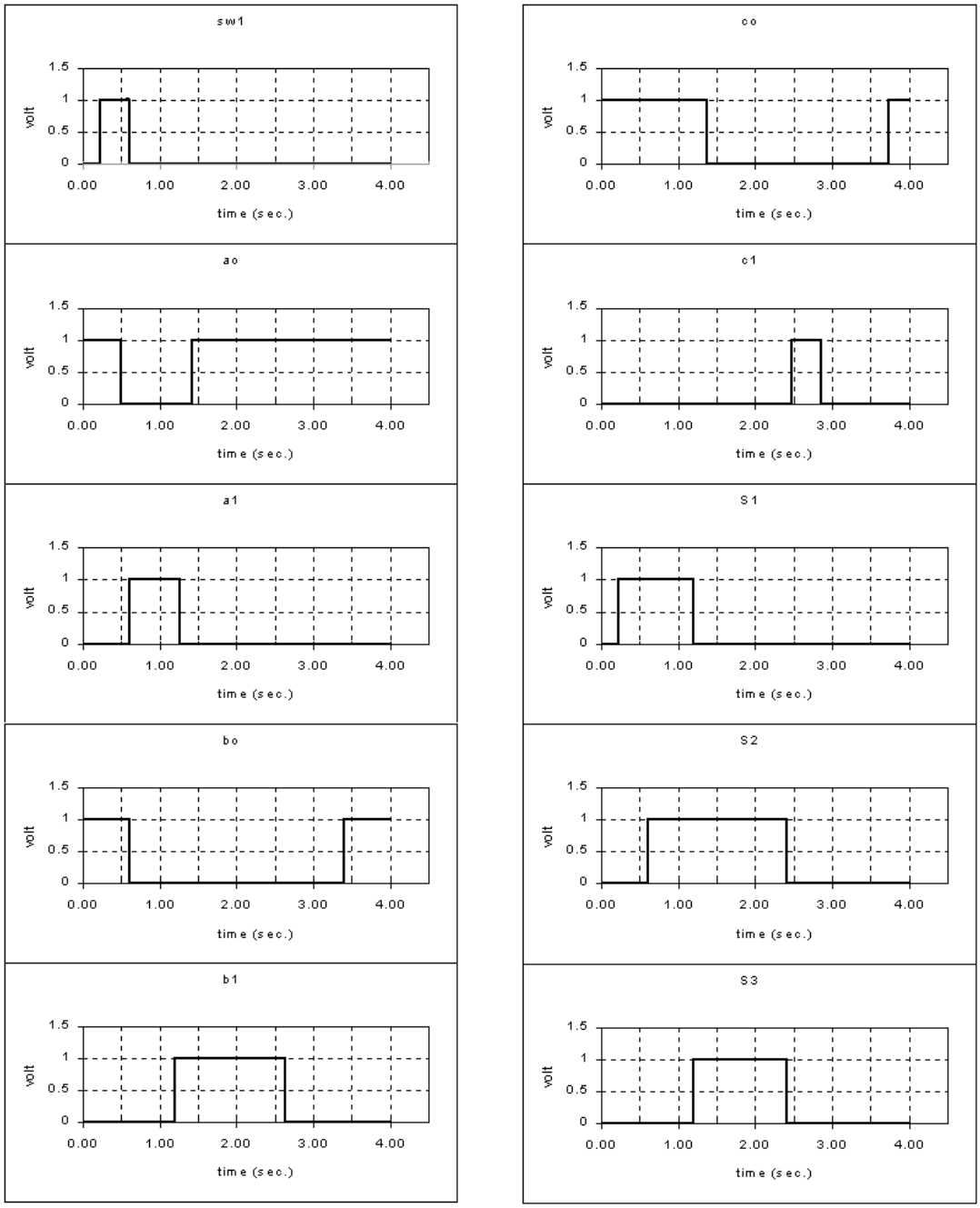


Fig. 12: Control Sequence diagram of the experimental work

7. CONCOLUSION

This work introduces a DPN-based discrete event controller for a manufacturing system. The controller design procedure is based on Dynamic Petri net. The proposed design procedure is applied to design a discrete event controller for two case studies, simple hydraulic system, and industrial robot system. The designed DPN- based controller of the industrial robot system is tested experimentally. Both theoretical and experimental results have been presented and discussed. The results indicate the following conclusions.

1. The proposed design procedure of DPN- based controller is effective in designing both a sequential controller and a combinational controller for different types of manufacturing systems.
2. The proposed design procedure is more flexible than finite state automata, where the restrictions of finite state automata theory have been overcame.
3. DPN based discrete event controller for the industrial robot system has been tested experimentally, which proves the effectiveness of the proposed design approach.
4. The proposed design procedure of the DPN based discrete event controller is a useful tool for control engineers due to:
    - Its capability of describing concurrent events and real- time systems, high-level DECS's and low-level implementation,
    - An error check can be mechanized by observing a marking procedures,
    - Flexibility and low cost, and
    - It is based on a mathematical approach instead of a trial and error approach.

REFERENCES

1. R. Zurawski, and M. Zhou, ”Petri Nets and Industrial Applications: A tutorial,” IEEE Trans. On Industrial Electronics, Vol. 41, No. 6, pp.567-581, 1994.

2. T. H. Sun, C. W. Cheng, and L. Chen Fu, ”A Petri Net Based Approach to Modeling and Scheduling for an FMS and a Case Study,” IEEE Trans. On Industrial Electronics, Vol. 41, No. 6, pp.593-600, December 1994.

3. K. Venkatesh, M. Zhou, and R. J. Caudill, “Comparing Ladder Logic Diagrams and Petri Nets for Sequence Controller Design Through a Discrete Manufacturing System,” IEEE Trans. On Industrial Electronics, Vol. 41, No. 6, pp.611-619, 1994.

4. M. Abdelhameed, and F.A. Tolbah, ”Design and Implementation of a Flexible manufacturing control system using neural network,” International Journal of Flexible Manufacturing System, Vol.14, pp.263-279, Feb. 2002

5. Lentz, K.W., “Design of Automatic Machinery,” Van Nostrand Reinhold Comp. Ltd., New York, NY, 1995.

6. McCord, F., “Designing Pneumatic Control Circuits,” Van Nostrand Reinhold Comp. Ltd., First edition, 1988.

7. Kohavi, Z., Hamming, R.W. and Feigenbaum, E.A., “Switching and Finite Automata Theory,” McGraw Hill College Div., 2nd edition, October 1986.

8. Sasao, T., “Switching Theory for Logic Synthesis,” Kluwer Academic Publishers (February 1999).

9. Webb, J.W., “Programmable Logic Controller, Principles and Applications,” Maxwell Macmillan International Editions, Singapore, 1992.

10. R. S. Brink "A Petri Net Design, simulation, and Verification Tool," Thesis release Permission Form, Rochester Inst. of Technology, College of Engineering, 1996.
11. A. Caloini, G. Magnani, and M. Pezze, "A Technique for Designing Robotic Control Systems Based On Petri Nets," IEEE Trans. On Control Systems Technology, Vol. 6, No. 1, pp.72-87, 1998.
12. M. Uzam, and A. H. Jones, "Discrete Event Control System Design Using Automated Petri Nets and their Ladder Diagram Implementation," Int. Journal of Adv. Manuf. Technology Vol. 14, pp.716-728, 1998.
13. K. Feldman, A. W. Colombo, C. Schnur, and T. Stockel, "Specification, Design, and Implementation of Logic Controllers based on Coloured Petri Net Models and the Standard IEC 1131 Part I," IEEE Trans. on Control Systems Technology, Vol. 7, No. 6, pp. 657-665, 1999.
14. K. Feldman, A. W. Colombo, C. Schnur, and T. Stockel, "Specification, Design, and Implementation of Logic Controllers based on Coloured Petri Net Models and the Standard IEC 1131 Part II," IEEE Trans. on Control Systems Technology, Vol. 7, No. 6, pp.666-674, 1999.
15. K. Tsuji, S. Kumagai, S. Kodama, and T. Yamada, "Modeling and Verification of Sequential Control Systems by Petri Nets," Proc. of IEEE inter. Symposium on Circuits and Systems, pp.988-991, 1986.
16. N. Marranghello, "Digital Systems Synthesis from Petri Net Descriptions," Technical Report DAIMI/PB-530, 1998.
17. J. S. Sagoo and J. T. Boardman, "Towards the formalization of soft systems models using Petri Net Theory," IEE proc-control theory Appl., Vol. 145, No. 5, pp.463-471, 1998.
18. T. Murata, "Petri Nets: Properties, Analysis, and Applications," Proc. IEEE, Vol. 77(4), pp.541-580, 1989.
19. G. Stremersch, and R. K. Boel, "Decomposition of the supervisory Control Problem for Petri Nets Under Preservation of maximal Permissiveness," IEEE Transaction on Automatic Control, Vol. 46, No. 9, pp. 1490-1496, 2001.
20. S. C. Lee, "Digital Circuits And Logic Design," Prentice Hall of India Private Limited, New Delhi, 1981.
21. D. Rene, "Grafcet: A powerful Tool for Specification of Logic Controllers," IEEE Transaction on Control System Technology, Vol. 3, No. 3, pp. 253-268, September 1995.