

# SEQUENCE ANALYSIS OF FINITE POSITION MACHINE FPM

---

Jesus Trujillo, Enrique Baeyens  
*Universidad de Valladolid, Spain*  
*Jestru@eis.uva.es, enrbae@eis.uva.es*  
Zbigniew Pasek  
*University of Michigan*  
*zbigniewk@umich.edu*

*Logic control is an integral part of manufacturing systems. The creation and debugging of the logic control represents a significant amount of the effort needed to design a large manufacturing system. Today's rapidly shifting markets have greatly reduced the life of a product design and the manufacturing system to produce it. As a result reconfigurable manufacturing systems are being developed which are capable of producing different parts over its life. These systems will need control logic, which is capable of being easily reconfigured as the system changes. This work describes the analysis of Finite Position Machines (FPMs) for reconfigurable manufacturing systems thought VS Tree method. VS tree allows to create feasible Pattern structures, whose basically are systems of communicating finite position machines which represent safe reactive processes and allow behavior description.*

## 1. INTRODUCTION

Today a vast of industrial logic controllers are performed under computer named programmable logic controllers (PLCs). The PLCs are specially designed to respond to use as controllers on industrial processes. A recent research work [10] published on Control Engineering journal shows that 96% of those polled programs are using ladder diagrams. One of principal inconvenient of ladder diagrams is shown in the complexity of manufacturing systems. Several alternatives have been developed to lead diagrams to PLCs programming. In particular the standard IEC61131-3 publication and IEC61499 is directed to resolve some above problems [12]. The two formalisms more used for control manufacturing system are the finite states machine FSM and the Petri net [5,13]. However the real complex controller cases design though Petri net are few [6].

The statechart are an alternative framework that allows to describe the behavioral of a complex system in a compact form[9,8]. Similarly to a Petri nets they have a good concurrence. The complexity of semantic of execution does that the verification of control systems modeled with statecharts be a hard task [7]. On the other hand the supervisory control theory [14,15] resolves problem in specific cases, when a supervisor creates exactly the desire behavior in a close loop, even when the controller cannot control or observe all events [1,11]. SC theory could be not

adequate to be used in developing of complex manufacturing control systems reading [2,3]. The principal reason is that the controllers of SC theory are designed to prevent the effects by already events despite of to try controlling the reasons that they create. This paper introduces the analysis of finite position machine (FPM) for studying system by behavioral description [16]. A principal reason has been the behavioral description of logic control in complex manufacturing plants. The FSM framework is unavailable to represent adequately complex systems, since its required representation millions of states [4]. The finite position machine FPM has properties that allow developing a method thought VS tree, which identify all possible sequences in a control process. These sequences rapidly can be verified outline, and composed in a safe and fast form obtaining feasible pattern sequences [16]. Pattern sequences are capable to compose for a new control reconfiguration.

## 2. ANALYSIS OF A FPM

### 2.1 General

The majority of highly automated manufacturing process works cyclically, repeating manufacturing orders, in order to obtain products. The presence of non-repetitive lineal sequences is usually linked to the appearance of exceptions produced by conflicts, break downs or unexpected events. Therefore, in order to detect the presence of these conflicts, it is necessary to identify the appearance of sequences which break the operational cycle. With this in mind, it will be necessary to study the FPMs and to identify a priori the possible cycles that might appear. The appearance of cycles in an FPM is determined by their topological structure. Below, a condition is established which allows us to characterize the FPMs with cyclic processing.

*FPM definitions: Action:* An action consists of a tuple of five element formed of a set of a trigger,  $A = (\tau, \pi^-, \pi^+, T, S)$ . **FPM:** A deterministic FPM, denoted  $\Sigma$ , is a tuple of ten elements  $\Sigma = (P, A, T, \rho, \zeta, \tau, P_{G_{sa}}, P_{G_{ec}}, P_{G_{ic}}, P_{G_{vm}})$  produce the language  $L(G_v)$  and marks the language  $L_m(G_v)$ . Where:  $\rho : P \times A \rightarrow P$  or is the response of partial sequence  $\zeta$ : is the cyclic transition function or sequence of cyclic response  $G_v, \zeta : P \times A/T \rightarrow P, \tau_{G_{vo}}$  is the set of transition action triggered from  $G_v$  in position  $p \in P_{G_v}$ , as e.g., subset of  $AG_v$  for which  $\zeta_{G_{vo}}(p)$  is defined  $P_{G_{sa}}$  initial position of complete sequence in an acyclic processing,  $P_{G_{ec}}$  initial position of external cycle  $G_v$ ,  $P_{G_{ic}}$  initial position of internal cycle  $G_v$ ,  $P_m$  is the set of marked positions,  $P_m \subset P$ . All details is in [10].

### 2.2 Cyclicity Conditions

A finite position machine has cyclic processing if it satisfies the three propositions:

- 1) The number of actions  $\eta_A$  must be bigger or equal to the positions  $\eta_P$ .  
 $\eta_A \leq \eta_P$
- 2) all positions  $P_n$  must have at least one arrival action  $\pi_{An}^+$ ,

$$\forall \pi_{A_n}^+ \in P_n, \text{ when } \eta_{\pi_{A_n}^+} \geq 1, ; n = 1, 2, \dots$$

3) all positions  $P_n$  must have at least an exit action  $\pi_{A_n}^-$ ,

$$\forall \pi_{A_n}^- \in P_n, \text{ when } \eta_{\pi_{A_n}^-} \geq 1, ; n = 1, 2, \dots$$

Below, we will establish the two conditions which allow us to obtain the cycles present in an FPM. We will begin by defining some concepts associated with the topographical structure of the machine.

### 2.2 Structural Elements of an FPM

The associated structural elements are defined as follows.

**Node:** In any position  $P_n$  of the machine, where three or more actions concur

$\pi_{A_i}^-, \pi_{A_j}^+, \pi_{A_{i+1}}^-$  or  $\pi_{A_{j+1}}^+, \dots$ , where  $2 \leq \eta_A \in P_n$ . When, with respect to the node, only two control actions  $\pi_{A_i}^-, \pi_{A_j}^+$  at the same position, then this is named **junction position**.

**Branch:** is a set of actions which enclose two or more adjacent positions.

**Loop:** is a set of actions which forms a cycle. They could exist. **Unitary loops** formed by only one action whose starting and arrival positions are the same.

**Lattice:** is a loop which does not contain internal branches: i.e., actions neither leave nor arrive between the positions that form the loop, distinct from those included in the loop itself.

**Tree:** is all affinity set, where its branches contain all the nodes and junction positions, that does not contain any loop, there existing open branches, and consequently, it forms partial sequences of acyclic processing.

**Chain:** is all branch which does not form part of tree, but it has an affinity with it by one or other extremes.

**Basic-loop:** is a loop, which contains a single chain and all, or only part, of the tree branches are represented in it.

**Affinity Structure:** is that where its constitutive elements are formed by FPMs.

### 2.3 Example of an FPM plane

By means of the following example, the structural elements are shown corresponding to the FPM represented in Figure 1

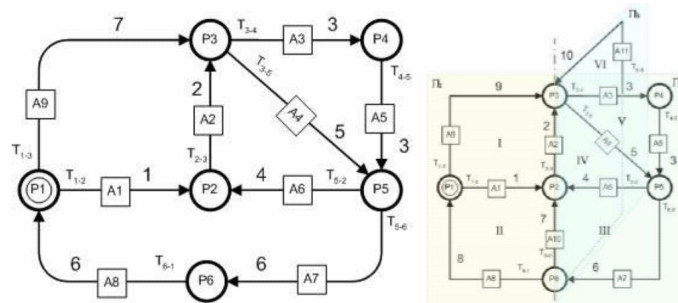


Figure 1 - Basic FPMs,

- 1) The number of branches  $\eta_r$  is seven, identified in Figure 1 by 1,...,7. The actions  $A_3, A_5$  in the positions  $P_3P_4$  and  $P_4P_5$  do not constitute different branches, but a single one, formed by the set  $P_3P_4P_5, A_3, A_5$ , where  $P_4$  is a position of union, since only two actions concur  $A_3^-, A_5^+$ ; similarly, it occurs for the position of union  $P_6$  where  $A_7^-, A_8^+$  concurs and where the branch is formed by the set  $P_5P_6P_7, A_7^-, A_8^+$ .
- 2) The number of nodes  $\eta_n$  is four and are represented by the positions  $P_1, P_2, P_3, P_5$ .
- 3) The number of loops  $\eta_l$  is twelve, as is shown in Figure 2
- 4) The number of loops  $\eta_b$  is four. These are shown in Figure 1 by (a),(i),(k) and (l). N.B. None of the loops contain actions in their interior.

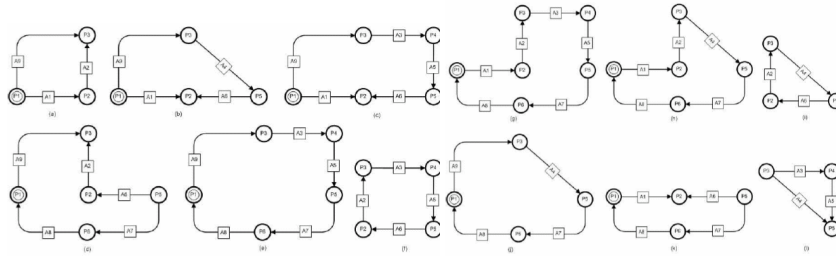


Figure 2 - ((a),..., (j)) Set of the twelve different loops which the FPM can build in Fig. 1.

- 5) Some trees of the FPM, to give an example, are (a) formed by the positions  $P_1P_3P_2P_5$ , (b)  $P_1P_3P_5P_2$ , (c)  $P_1P_3P_4P_5P_2$ , (d)  $P_1P_3P_4P_5P_2$ , (e)  $P_1P_2P_3P_5$  and (f)  $P_2P_3P_5P_6P_1$ .
- 6) The reference tree in figure 3, which chains are the branches between the positions  $P_2P_3, P_3P_5$  and  $P_5P_6P_1$ .
- 7) In a similar way, in figure 3 (f) is taken to be a reference tree, where a basic loop is formed, similarly, by the positions  $P_1P_3P_4P_5P_6P_1$ , since it only contains one chain between the positions  $P_5P_6P_1$ .

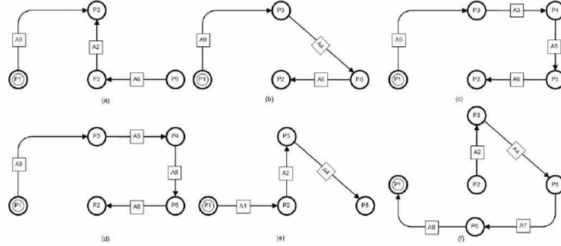


Figure 3 - Some trees corresponding to the FPM of figure 1

### 2.3 Basic relation in the analysis of FPMs

Grouping together the relations given in the previous example, we obtain:

$\eta_r$  = number of branches,  $\eta_e$  = number of chains,  $\eta_n$  = number of nodes,  $\eta_b$  = number of lattices,  $\eta_l$  = number of loops.

- During the formation of a specific tree of an FPM, two nodes are chosen which are united by a branch, which afterwards add new branches. It can be observed that for every new branch that we incorporate, a new node will be added. In fact, it can be shown that the number  $\eta_r$  of branches of a specific tree is:

$$\eta_r = \eta_n - 1 \tag{1}$$

- If, to the tree considered, chains are added, every new chain is a new branch. Furthermore, as the number of branches is  $\eta_r$ , where  $\eta_r = \eta_e$ .

Adding the two previous expressions element by element, we obtain:

$$e = \eta_e + \eta_n - 1 \tag{2}$$

- For any FPM the number of branches is the same as the lattices, plus the number of nodes minus one,

$$\eta_r = \eta_b + \eta_n - 1 \tag{3}$$

- Considering a global structure formed by a set of independent S branches, by applying the previous expression  $\eta_r = \eta_e + \eta_n - 1$  to each of them for a specified tree of each branch, we obtain:

$$\eta_r = \eta_e + \eta_n - 1 \tag{4}$$

This is the equation which relates the number of branches, the number of chains, relative to a specific tree of each portion independently, and the number of nodes of the global structure of all the set, with the number of independent portions.

Below, we clarify this idea by means of an example. In the tree (f) in Figure 1, it can be observed that the number of branches is  $\eta_r = 3$  and the number of node  $\eta_n = 4$ .

Therefore, the previous relation (1) is fulfilled where  $3=4-1$ .

In the tree considered, we include in it as many chains as considered necessary to form the primitive FPM calling  $\eta_r$  the number of branches added to the tree, equal to the number  $\eta_e$  of chains which have been placed in it. The result will be that the number of branches and chains added will be as indicated in Figure (1), shown in red. This chain allows the closing of the tree, forming a loop and the rest of the chains added will coincide with three, corresponding to chains 2,3 and 4.

FPM shown in Figure 1 though equations (3) and (4) is easier to check the result of number of branches, nodes, chains and lattices. Let  $A\{SP\}+(P_j)$  be the set of all the actions which starts from position  $P_j$  and arrive at any position of sequence  $SP$  and let  $A-(P_j)$  be the set of all the actions that begin from any position of the sequence  $SP$  arrive at the position  $P_j$ . For each position  $P_j$  and for each action  $A_k$  of an FPM, a numerical value can be associated with them, as was done in section~}, when position sequences were represented by recurrence rules. Let  $x_P(P_j)$  and  $x_A(A_k)$  be the numerical values associated with position  $P_j$  and with the action  $A_k$ . The associated numerical value can be associated with a weight or a dynamic characteristic that could temporally affect each action or position, such as for example, a specific priority in a process, when the action represents that process. This characteristic will be of great importance in those cases in which the FPMs are used for simulating processes, as could be the case of the processes and operations which intervene in the productive process scheduling.

1) **Rule of position cycles:** For any sequence of positions  $S_p$  of a FPM that forms a cycle, the sum of the numerical values associated with the starting actions minus the sum of the numerical values associated with arrival actions of all the positions of the sequence is zero.

$$\sum_{P_k \in S_p} \left( \sum_{A_j \in \mathcal{A}^-(S_p)} x_A(A_j) - \sum_{A_j \in \mathcal{A}^+(S_p)} x_A(A_j) \right) = 0$$

2) **Rule of actions cycles:** For any sequence of actions  $S_A$  of an FPM that forms a cycle, the sum of the numerical values associated with the starting positions minus the sum of the numerical values associated with arrival positions of all the actions of the sequence, is zero.

$$\sum_{A_k \in S_A} (x_P(\pi^+(A_k)) - x_P(\pi^-(A_k))) = 0$$

These two rules are dual, although the action rule is more simplified when the FPMs are defined in such a way that all the actions are different. If this supposition were not made several positions could have the same starting or arrival action and the duality would be complete.

Making use of these two rules, it is relatively simple to program an algorithm search which would allow us to obtain all the cyclic sequences on an FPM. Let us consider the machine in Fig.4, where the numerical values have been defined of the positions and actions indicated in the following table:

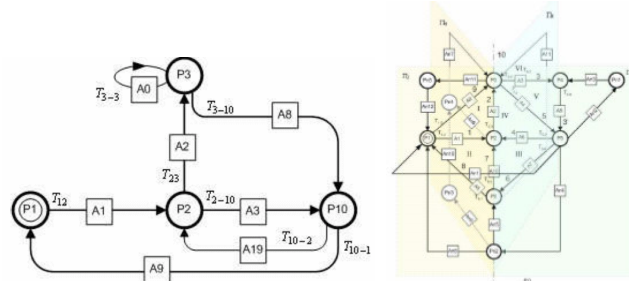


Figure 4 - Graphic of two FPM with cyclic processing

Let the following sequences of positions be valid:  $S\{P1\}=\{P1, P2, P10\}$ ,  $S\{P2\}=\{P_2, P10\}$ ,  $SP3=\{P2, P3, P10\}$  and actions:  $S\{A1\}=\{A0\}$ ,  $\{A3,A19\}$ ,  $S\{A2\}=\{A1,A3,A9\}$ ,  $S\{A3\}=\{A2,A8\}$ . In order to see which of these sequences are cyclic we apply the cyclic rules~\ref{le0401}. On applying the rule of position cycles we obtain:  $S\{P1\}=0$ ,  $S\{P2\}=0$ ,  $S\{P3\}=-11$ .

In order to deduced that all sequences form cycles except  $S\{P3\}$ . Now applying the rule of cycles of actions~\ref{le0402} can then be obtained,  $S\{A1\}=0$ ,  $S\{A2\}=0$ ,  $S\{A3\}=0$ ,  $S\{A4\}=2$ .

We are able to deduce that all the action sequences form cycles except  $S\{A4\}$ . These two rules are important because the behavior of a modeling system with an FPM is given by the sequences that it creates. Furthermore, the sequences can be composed amongst themselves, giving rise to more complex sequences. A modeling system with an FPM has a finite set of elementary, lineal and cyclic sequences whose composition completely characterizes the machine and which is denominated as elementary reference patterns of the FPM. The two rules studied in this section allow us to obtain and characterize simply the elementary working patterns of an FPM.

#### 4. GENERATION OF VS TREE\ AND FUTURE WORKS

The VS tree is formed by three basic levels: the axis level, the plane level and the order loop, lattice and closed loop level. The VS tree has the function of creating the basic structure, where all the possible loops that an FPM might contain are to be found.

##### 3.1 Definitions in the generation of the VS tree

**Axis:** is the set of actions and positions which concur, forming a line which contains the positions of greatest confluence.

**Plane:** This contains the concatenate loops via common branches and which is initiated from a single axis, where other parallel loops cannot be given to those which exist.

**order loop:** These are formed by lattices and are the result of all the possible combinations between the above lattices. Firstly, the FPM has to be ordered in order

to obtain all the possible loops resulting from all the possible interactions which could take place in the FPM. For this, an ordering subroutine is used.

### 3.2 Algorithm of FPM ordering

For cases of an FPM macro-structure of great complexity, it could be possible to begin the process by identifying all the axes and obtaining all the relational combinatorial between them, and then to continue with the steps already taken.

The VS tree includes three fundamental concepts: on the one hand, the determinism of feasible structures, and on the other hand, the possibility of increasing or reducing elements corresponding to a structure, which determines an action that verifies the new extension adopted and, finally, it allows the rapid composition of structures for the creation of patterns.

## 4. CONCLUSIONS AND FUTURE WORKS

In this paper has been introduced the sequence analysis of finite position machine FPM. This analysis includes method to obtain all sequence possible easier and feasible form thought VS tree. VS pattern are safe structures that allow to compose faster control sequences for use and implementation on-line.

FPM is a new available framework to represent complex manufacturing system by faster and safer manner. This method allow to achieve that the patterns comply ones confidence margins, which this can determine that obtained properties in- $\backslash$ cite{TRU04}. They can warrant a feasible use of these reference patterns under optimal conditions.

### 4.1 Acknowledgments

This work has been supported in part by the Programa de Apoyo a Proyectos de Investigación del Ministerio de Educación y Ciencia,() and University of Michigan NSF Engineering Research Center for Reconfigurable Manufacturing Systems (NSF grant EEC-9529125)

## 5. REFERENCES

1. Cassandras C.G. and Lafortune S.. *Introduction to Discrete Event Systems*. Kluwer Academic publishers, MA., 1999.
2. Charbonnier F., Alla H., and R. David. The supervised control of discrete-event dynamic systems. *IEEE Transactions on Control Systems Technology*, 7(2):175–187, March 1999.
3. Dietrich P., Malik R., Wonham W.M., Brandin B.A.. Implementation consideration in supervisory control. In Caillaud B., Danrondeau P., Lavagno L. and X. Xie, editors, *Synthesis and Control of Discrete Event Systems*, pages 185–201. Kluwer, 2001.
4. Endsley EW. Tilbury DM. Modular verification of modular finite state machines. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, volume 1-5, Nassau, Bahamas, December 2004.
5. Genc S. and. Lafortune S. Distributed diagnosis of discrete-event systems using petri nets. In *Proceedings of the 24th International Conference on Applications and Theory of Petri Nets*, pages 316–336, Oulu, Finland, June 2003.

6. Gollapudi C., Tilbury D.M. Logic control design and implementation for machining line tested using Petri nets. In *Proceedings of the ASME-IMECE Dynamic Systems and Control Division*, November 2001.
7. Gruer P., Koukam A., Mazigh B. Modeling and quantitative analysis of DES: A statecharts based approach. *Simulation Practice and Theory*, 6:397–411, 1998.
8. Harel D, Naamad.A. The statechart semantics of statecharts. *ACM Transaction on Software Engineering and Methodology*, 5(4):239–333, 1996. [
9. Harel D.I, Pnueli A., Schmidt J.P., Sherman R.. On the formal semantics of statecharts. In *Proceedings of the Second Annual Symposium on Logic in Computer Science*, pages 54–64, Ithaca, NY, June, 22-25 1987. Computer Society Press.
- 10 Johnson D. Nano devices lead assault on traditional PLC applications. *Control Engineering*, 49(8):43–44, August 2002.
11. Kumar R. and Garg V.K. *Modeling and Control of Logical Discrete Event Systems*. Kluwer Academic Publishers, Norwell, MA, 1995.
12. Lewis R.W.. *Modeling control systems using IEC 61499*. The Institution of Electrical Engineers, 2001.
13. Park E, Tilbury D.M., and Khargonekar. P.P. Modular logic controller for machining systems: Formal representation and performance analysis using Petri nets. *IEEE Transactions on Robotics and Automation*, 15(6):1046–1061, 1999.
14. Ramadge P.J.G. and Wonham.W.M. Supervisory control of a class of discrete event processes. *SIAM Journal of Control and Optimization*, pages 206–230, January 1987.
15. Ramadge P.J.G. and W.M. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77(1):81–98, January 1989.
16. Trujillo J.. *Finite Position Machine for Logic Control: Composition of Patterns in Reconfigurable Manufacturing System*. PhD thesis, University of Valladolid, Spain, 2004. DET '06 ,3<sup>rd</sup> International CIRP Conference on Digital Enterprises Technologies, 2006.