COST BASED IMPLEMENTATION OF MODULAR SUPERVISORY CONTROL THEORY

Gilvan O. Costa, Eduardo A P Santos, Marco A Busetti

Pontifical Catholic University of Parana, Mechatronics Engineering, 80 215 901, Curitiba-PR, Brasil

Abstract: This paper presents a proposal for the implementation of the local modular supervisory control theory focused on the cost. From the approach based on the controllable languages, the paper explores the accomplishment of the theoretical model in low cost platforms, based on small-sized PLCs, micro-controllers and commercial electronic devices. A control structure is, then, proposed as the adequate solution for small and medium size manufacturing automated systems. In order to demonstrate the efficiency of the proposal, a didactic workbench is used to simulate the current manufacturing system. *Copyright © 2005 IFAC*

Keywords: automation, discrete event systems, control, manufacturing, programmable controllers.

1. INTRODUCTION

According to Erbe (2002), many companies have been finding difficult to adopt the automation as a competitiveness strategy. According to the author, the main reason for this assumption is related to the insufficient flexibility of the highly automated systems. The combination of the losses in consequence of the conversion, the equipments idle time and the high costs of specialized maintenance constraint the economical expected benefits. Thus, many companies have decreased the automation level in the shop floor, or they intend to do so.
Accordingly, since a large amount of the cost of automated systems involves the control system, a great effort has been made seeking approaches, which deal with the controllers' synthesis. Among the various approaches, one can name the Supervisory Control Theory (SCT), proposed by Ramadge and Wonham (1989), which presents the advantage of allowing the automatic synthesis of controllers for the Discrete Events Systems (DES). Thus, one can avoid the utilization of empirical and informal methods, based on the experience and inspiration of the designer (Chandra et al., 2003). One reaches, like this, a greater reliability, maintenance simplicity, errors' detection and the inclusion of additional specifications in a systematic form. In consequence, one can obtain a smaller implementation cost and an increased flexibility related to the control system.
An important issue emerging from the implementation in control industrial platforms is the compatibility of the proposed theoretical model (the SCT, based on the Languages and Automata Theory) with the characteristics and the limitations of such platforms, being the programmable logic controller (PLC), nowadays largely used in industrial environments. However, the technology surrounding this platform's use leads to a much too high investment cost, considering the applications performed in small and medium sized companies. One can designate the PLC cost (itself), the software licenses, the communication additional modules, among others.
Within this context, the current paper presents a control typology focused on the cost, having as theoretical grounds the SCT and the Local Modular Control (LMC) proposed by Queiroz and Cury (2000a,b), and the implementation is performed in micro-controllers and in low cost commercial electronic devices. The proposed control structure is based on previous works, which used PLC as platform for the control system realization (Queiroz and Cury, 2002).

2. NOTATION AND PRELIMINARIES

In the framework of Ramadge and Wonham (1989), the system spontaneously generates discrete events $\sigma \in \Sigma$ classified as controllable ($\sigma \in \Sigma_c$), when the event can be disabled by the control system, or uncontrollable ($\sigma \in \Sigma_u$). Let Σ^* be the set of all finite chains of elements in Σ, including the empty chain ε. A language is a subset of Σ^*. The behaviour of a

DES may be modelled by languages that, when regular, are represented by automata. An automata is a quintuple $G = (\Sigma, Q, \delta, q_0, Q_m)$, where Σ is the event set, Q is a set of states, $q_0 \in Q$ is the initial state, $Q_m \subseteq Q$ is the subset of marked states and δ: $\Sigma \times Q \rightarrow Q$, the transition function, is a partial function defined in each state of Q for a subset of Σ. Then, G is characterized by two languages: its closed behaviour $L(G)$ and its marked behaviour $L_m(G)$. Automata are illustrated by state transition diagrams that are directed graphs where nodes represent states and labeled arcs represent events. Marked state nodes are double lined and the initial state is pointed by an arrow. Intercepted arcs indicate controllable events.

2.1 Local Modular Control

According to the LMC approach, the physical system must be separated in several subsystems asynchronous among themselves, thus constituting a Representation by Product System (RPS) (Ramadge and Wonham, 1989). An automaton G_i must represent each of the subsystems, so that it represents its behaviour in the most abstract way possible. Each of the automata will have the following structure $G_j = (\Sigma_j, Q_j, \delta_j, q_{0j}, Q_{mj})$, $j \in J = \{1..p\}$ where: p is the nº of physical subsystems, Σ_j is the sub-alphabet of events exclusive of each modelled subsystem, Q_j is the states set, δ_j is the function of the states transition according to $\delta_j{:}\Sigma_j \times Q_j \rightarrow Q_j$, $q_{0j} \in Q_j$ is the initial state and $Qmj \subseteq Q_j$ is the marked states set. The sub-alphabet Σ_j is divisible in a disjunctive way in controllable events $\Sigma cj \subseteq \Sigma j$ and non-controllable events $\Sigma uj \subseteq \Sigma_j$, and the physical system alphabet is given by the combination of the several sub-alphabets Σ_j.
Each one of the generic specifications, represented by an automaton $E_{gen,j}$ and defined in $\Sigma j \subseteq \Sigma$ for $j \in I = \{1..m\}$, constricts the behaviour of only a portion of the physical system. The composition of the subsystems whose behaviours are constrained by a stated specification will determine the local plant, and the composition of a generic specification with the correspondent local plant will define the local specification. By using this procedure, it is possible to effect the synthesis of a local supervisor for each one of the established specifications. Each of those local supervisors can be represented through an automaton $SupC(E_{loc,j}, G_{loc,j})$, whose stated language correspond to the maximum local plant controllable language within the local specification language (Queiroz and Cury, 2000a,b).

2.2 Implementation

Aiming to execute the local modular control approach in a readable structure, the control system is proposed to be programmed in a three level hierarchy (Queiroz and Cury, 2002). The set of local modular controllers is implemented in this level exactly as theoretically conceived by Ramadge and Wonham (1989). The program updates the active states according to the automata structures and the state changes in the Product System level. A feedback map associates the active states to a set of disabling signals that control the Product System. The main function of the Product System level is to execute the commands that are allowed by the plant and are not disabled by supervisors. The parallel evolution of the asynchronous subplants follows executed commands and responses from the Operational Sequences level, signalising state changes to the controllers. The Operational Sequences work as an interface between the theoretical Product System and the Real System. In this level, the program interprets the abstract commands as logical procedures that guide the operation of each particular subsystem. These low-level sequences generate the control system output signals and read the input signals, supplying the Product System with logical responses that reflect the occurrence of uncontrollable events (Queiroz and Cury, 2002).

3. THE MANUFACTURING CELL

The manufacturing cell used in the implementation of the control system, proposed by the current paper, is represented by Figure 1. The cell's objective is to perform a series of typical manufacturing operations: handling, drilling, transporting, and testing. Various control structures can be proposed, but the central issue is to operate it by using the maximum capacity and through the least restrictive manner. For example, the cell can operate with one, two, three or four pieces at once, depending on how the control system is built. Obviously, operating in the most restrictive manner, with only one piece at a time, it is the simplest way of implementing the control design, although it is the least efficient.

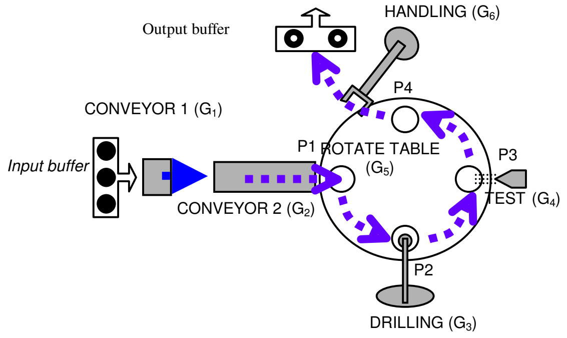


Fig. 1. Schematic of manufacturing cell.

3.1 Plant Models

According to SCT, the plant modelling to be controlled can be obtained as follows: Identify the subsystems or equipment set involved in the system to be controlled; Build the basic model as a DFA (Deterministic Finite Automata) Gi, for each equipment i involved in the system in the most synthetic manner; Calculate the most refined Representation by Product System, making the composition of the subsystems synchronous; Define the control structure Γ, through the identification of

controllable and non-controllable sets of events Σ_c e Σ_u, affecting the system.
Instead of having inapprehensible models of the manufacturing cell, and to make the modelling simpler, we develop smaller sized sub-models by partitioning the entire system into six sections according to figure 1, namely, conveyor 1 (G1), conveyor 2 (G2), drilling (G3), test (G4), rotate table (G5) and handling (G6). The DFA model of these subsystems is given in Fig. 2. This model represents the abstract behaviour of each subsystem composing the cell and it can be explained as follows: in the initial state g0 the subsystem G_i is inactive; when the event α_i occurs the subsystem evolves to the active state g1, meaning that it is operative; when the event β_i occurs the subsystem returns to the initial state, meaning the end of its operation.

3.2 Operational specifications models

The operational specifications are used to specify tasks, which the plant must perform in order to accomplish the manufacturing objectives. Regarding the studied cell, the specifications must impose the proper pieces' flow in the least restrictive manner, allowing the simultaneous processing of several pieces. Also, the safety aspects must be observed and respected.
In order to reach such objectives, ten operational specifications are built. Figure 2 presents three of these specifications. The specification E_b prevents the operation of the rotary table (G5) without rough piece in P1, without drilled piece in P2 or without tested piece in P3. The specifications E_{ci} (i=2,3,4,6) represent the mutual exclusion between the conveyor belt (G2), the drilling (G3), the testing (G4) and the manipulator (G6) with the rotary table (G5). The rotary table shall not be activated while such subsystems are working. The specification E_{d1} imposes the proper pieces' flow between the conveyor belt and the rotary table. With such objective, the pieces overlay in the table's position P1 is avoided, the drilling without the rough piece in position P2 is prevented and the activation of the rotary table in P2 with rough piece is avoided.

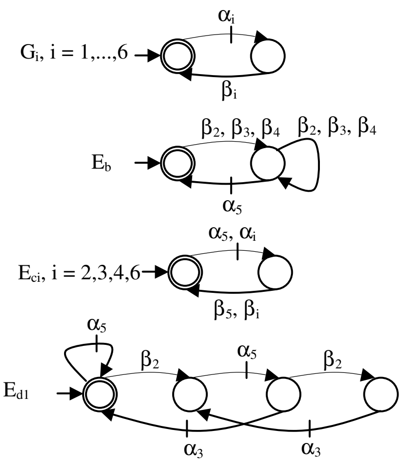


Fig. 2. Automata to subsystems and operational specifications.

3.3 Controllers synthesis

According to the LMC approach, it is necessary to obtain the local plant for each specification, composing the RPS subsystems, which have events in common with it. One calculates, then, the language of each local plant, addressing the specification through the synchronous product of each local plant with its correspondent specification. In continuation, one calculates the maximum controllable language within each local specification in order to, finally, verify the modularity of the resulting languages. The local modularity assures there is no loss of efficiency between this one and the best centralized solution (Queiroz and Cury, 2000a,b).
For example, considering the specification E_{d1} (figure 2), the local plant is obtained by the synchronous composition of the automata correspondent to the subsystems sharing events with this specification ($G_{loc,d1} = G_2 \parallel G_3 \parallel G_5$). The local specification is obtained through the composition of the generic specification E_{d1} with the correspondent local plant ($E_{loc,d1} = E_{d1} \parallel G_{loc,c1}$). One can, then, calculate the maximum controllable language in the local specification, that is SupC ($E_{loc,d1}$, $G_{loc,d1}$). Then, through a minimization algorithm for supervisors (Vaz and Wonham, 1986), one can obtain a supervisor with a diminished number of states, according to figure 3. The hatched lines indicate the supervisor action control, that is, the disablement of the plant controllable events. By its end, the presented local modular controller observes and acts the subsystems G2 (conveyor 2), G3 (drilling) and G5 (rotate table).

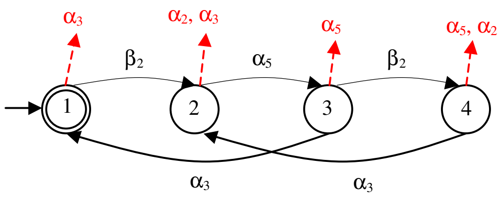


Fig. 3. Reduced controller SupC($E_{loc,d1}$, $G_{loc,d1}$).

4. RESULTS

After the modelling of subsystems and local controllers' synthesis phase, one begins implementing the control system. One proposes through the current work the implementation according to the proposal by Queiroz and Cury (2002), with the specific modifications to attend the objective of a low cost system. Figure 4 represents the proposed control structure, which is divided in four levels: the modular controllers level, the product system level, the communication interface level and the operational sequences level.
According to the structure shown in Figure 4, the levels correspondent to the modular controllers and the product system (1) are implemented in a low cost hardware, a micro-controller (Microchip, 1999). The communication interface (2) is realized through the

commercial electronic components. The operational sequences level (3) is implemented in small sized PLCs, being one for each plant subsystem G_i.

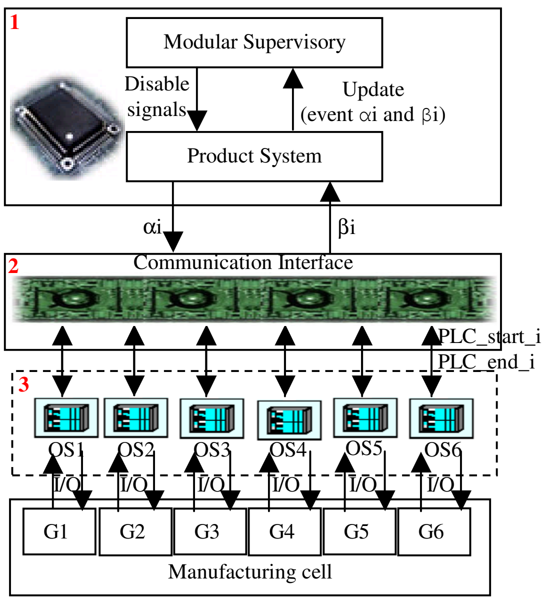


Fig. 4. Control structure proposed to manufacturing cell.

4.1 Local modular controllers and product system

The local controllers set is implemented in the highest level of the control structure. The evolution of the local controller states is performed through the events generated in the product system level (through the events α_i and β_i). A feedback map associates the active states of such supervisors to a set of disablement signals for the controllable events, which control the subsystems' evolution in the product system. In the SCT it is assumed that the physical system performs a spontaneous behaviour, so generating events which will determine its evolution. Because this is not observed in industrial automation systems, it is necessary to implement the product system in the program, which represents the physical system to be controlled in its most abstract way.
The state evolution related to the subsystems G_i of the product system is determined by the set of disablement signals for the controllable events and by signals from the operational sequences (conclusion of the activities performed by the subsystem G_i and implemented in the operational sequence). Also, in the product system level are generated the necessary signals for the control of the correspondent operational sequences (PLC_start_i – activation of the initial step related to the operational sequences).
The implementation of this level was performed through a micro-controller, and the behaviour described in the previous paragraphs was implemented through the flow chart systematisation shown in Figure 5, according to the specific rules of such micro-controller. The micro-controller ports B and C are configured as inputs and outputs for the events β_i and α_i, respectively. During the scanning, all events α_i are enabled, i.e., they receive logic value 1. After the verification of the occurrence of the events βi, one makes the local controllers updating, then, one loads which events α_i must be disabled. Those allowed to occur are released by the micro-controller port C, thus operating in specific PLC. The event's α_i occurrence is also verified, updating correspondent local controllers. Figure 6 presents the source code of the local controllers and the product system in language C.

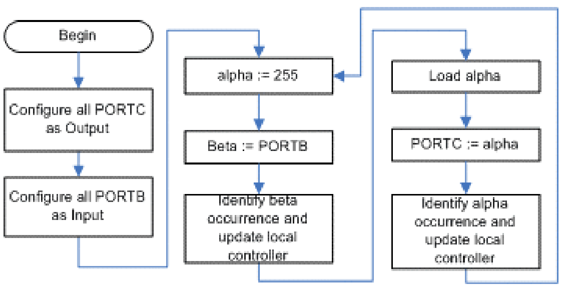
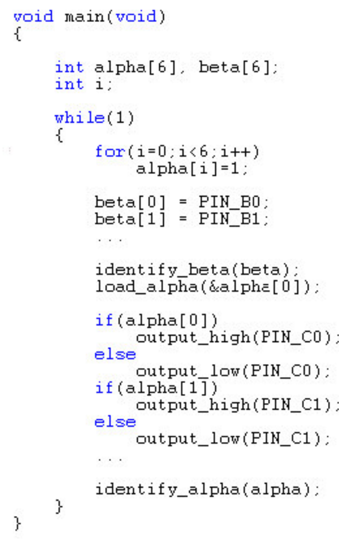

Begin
Configure all PORTC as Output
Configure all PORTB as Input
alpha := 255
Beta := PORTB
Identify beta occurrence and update local controller
Load alpha
PORTC := alpha
Identify alpha occurrence and update local controller

Fig. 5. Flow chart of implemented code.

The function "*identify beta occurrence*" consists in the scanning of the six positions vector beta (each position is correspondent to a subsystem G_i, with i=1,..,6). The occurrence of the activity conclusion of a subsystem G_i, marked by the events β_i , is detected continuously by this function, which also generates the updating for the local controllers affected in a given moment by some event β_i. The accomplishment of this function is shown in Figure 7. The function "*identify alpha occurrence*" also presents the same systematisation rationale.



```
void main(void)
{

    int alpha[6], beta[6];
    int i;

    while(1)
    {
        for(i=0;i<6;i++)
            alpha[i]=1;

        beta[0] = PIN_B0;
        beta[1] = PIN_B1;
        ...

        identify_beta(beta);
        load_alpha(&alpha[0]);

        if(alpha[0])
            output_high(PIN_C0);
        else
            output_low(PIN_C0);
        if(alpha[1])
            output_high(PIN_C1);
        else
            output_low(PIN_C1);
        ...

        identify_alpha(alpha);
    }
}
```

Fig. 6. Code source implemented.

The states evolution for a local controller, according to the occurrence verification of events α_i and β_i, is illustrated through a code segment shown by Figure 8, correspondent to the controller presented by Figure 3. The variable "*Controller_SLD1*" identify the controller, which in the modelling and synthesis step is designed SupC ($E_{loc,d1}$, $G_{loc,d1}$). Observing the controller structure, the occurrence of event β_2 makes the same evolution from the state 1 to the state 2 or from the state 3 to the state 4.

```
void identify_beta(int *beta)
{
    if(beta[0])
        beta_1();

    if(beta[1])
        beta_2();

    if(beta[2])
        beta_3();

    if(beta[3])
        beta_4();

    if(beta[4])
        beta_5();

    if(beta[5])
        beta_6();
}
```

Fig. 7. Code for “identify beta occurrence” function.

The disablement of the controllable events α_i is performed through the function “*load alpha*”. According to the controllers evolution, a new subset of disabled controllable events is identified by this function, in order to allow the product system evolution according to the control objectives previously designed. Figure 9 presents a segment of the function “*load alpha*” correspondent to the controller shown by Figure 3. For instance, in state 2 the events α_2 (alpha [1]) and α_3 (alpha [2]) are disabled.

```
void beta_2(void)
{
    ...
    if(Controller_SLD1 == 1)
        Controller_SLD1=2;

    if(Controller_SLD1 == 3)
        Controller_SLD1=4;
    ...
}
```

Fig. 8. Code for update state with β_2.

```
void load_alpha(int **alpha)
{
    ...
    if(Controller_SLD1 == 1)
        alpha[2] = 0;

    if(Controller_SLD1 == 2)
    {
        alpha[1] = 0;
        alpha[2] = 0;
    }

    if(Controller_SLD1 == 3)
        alpha[4] = 0;

    if(Controller_SLD1 == 4)
    {
        alpha[1] = 0;
        alpha[4] = 0;
    }
    ...
}
```

Fig. 9. Code for send signal α_i.

4.2 Communication interface

Because the SCT does not assume the simultaneous occurrence of events, it is vital that the local controllers state is kept updated before the occurrence of a new evolution in the product system. If the events α_i or β_i occurs (generated during the state evolution of a subsystem G_i of the product system), it is necessary the correspondent occurrence of the updating of the local controllers states and the subset of the disabled controllable events, before the possibility of occurrence of the state evolution, related to the remaining subsystems G_i. If it is observed, there is the possibility of occurrence of an irregular state evolution related to one or more subsystems, providing the violation related to the synthesized control rule and the violation related to one or more operational specifications.
Thus, in each of the controllers the state evolution is allowed only through a sole event per updating cycle (assuring the proper controllers evolution for the current implementation model and according to the SCT, which does not assume the simultaneous occurrence of events). It is allowed, however, the simultaneous occurrence of subsystems G_i distinct events as long as their behaviour are not restricted by a common controller.
However, the combined operation between the micro-controller and the PLCs can cause non-evolutions or erroneous evolutions related to the local controllers. A vast part of these problems is due to the way such platforms work. The main problem is the existing disparity between the scanning of the micro-controller and the PLC. In the current described case, the first has a much smaller cycle than the latter. Thus, the following problems may occur:
- The micro-controller sends a signal α_i to a specific PLC, but this one, because it has a much longer cycle, will not receive the signal;
- A specific PLC sends a conclusion signal β_i during the scanning cycle, but the micro-controller will receive the signal during various cycles.
With this purpose, it is proposed a communication interface which generates forwarding and reception rules for the events α_i and β_i. Such interface is implemented through the sequential circuits rationale using flip-flop D type and more commercial electronic components. For each event α_i and β_i, an electronic circuit is created which performs the specified communication rules. Figure 10 and 11 present such circuits, respectively. From the forwarding and reception simulation of events α_i and β_i, one can validate the built circuits.

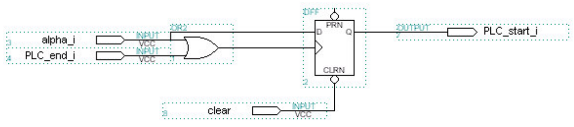


Fig. 10. Inteface to send event αi.

Figure 12 presents the simulation results for the event α_i. One verifies that, when the signal α_i (pulse) is generated by the micro-controller (A), the signal PLC_start_i is maintained in this circuit’s output (B). This signal initiates the operation of a specific PLC, staying in the logic level 1 for the necessary time for the signal recognition by the PLC. Then, the PLC_start_i signal shall return to the logic level 0 as soon as the PLC concludes the correspondent instruction to the Gi module, through the signal PLC_end_i (C).

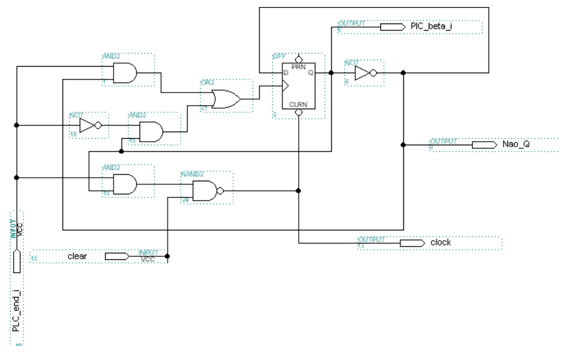


Fig. 11. Interface to receive event βi.

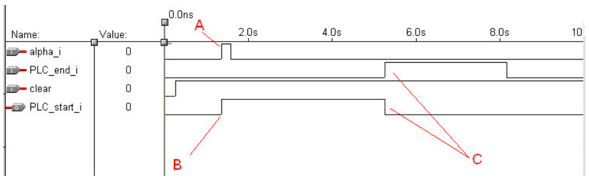


Fig. 12. Simulation of transmission of event αi.

4.3 Operational sequences

In the current paper, section 3, for the procedure of the local controllers synthesis, a functional abstraction was proposed for the physical subsystems to be controlled, abstraction necessary to enable the designed procedure. Through the implementation of the operational sequences, the compatibility of the obtained results above is reached with the technological solution used for the physical system realization. Thus, the operational sequences work as an interface between the theoretical product system level and the current system, as shown by Figure 4.
According to the current paper's proposal, each operational sequence is implemented in a sole PLC, reaching a structure distributed in this level. Because the considered plant was divided into six subsystems, one has six small sized PLCs. To illustrate the implementation of the sequences level, Figure 13 presents the subsystem G_3 program in a Sequential Function Chart. Two pneumatic actuators controlled by electro-valves and an electric motor for the drilling machine activation compose this module.

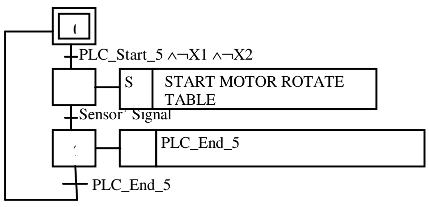


Fig. 13. Operational Sequence for G_5.

5. CONCLUSIONS

Implementation of SCT and LMC has been successfully applied to a manufacturing cell commanded by a micro-controller and PLCs. A cost-oriented generic structure has been proposed to the control system physical implementation. The final code readability and flexibility indicate the quality of project.
The model proposed by Ramadge and Wonham (1989) and Queiroz e Cury (2000a,b) is used in order to provide an automated process for the supervisor synthesis, instead of the usual manual or heuristic procedures. Besides this advantage, the controllers synthesis procedure is greatly appropriate, considering it is based on the open-loop system dynamics model and on the particular behaviour specification.

REFERENCES

Cassandras, C. G., S. Lafortune (1999). *Introduction to discrete event systems*, Kluwer Academic Publishers.
Chandra, et al. (2003). Automated Control Synthesis for an Assembly Line Using Discrete Event System Control Theory. In: *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 33, no. 2, pp. 284-289.
Erbe, H. –H. (2002). Low Cost Intelligent Automation in Manufacturing. In: Proceedings of the 15th Triennial IFAC World Congress, Barcelona, Spain.
Microchip (1999). PIC16F87X 28/40-pin 8-Bit Cmos Flash Microcontrollers, Microchip Technology Inc.
Queiroz, M. H. and J. E. R. Cury (2000a). Modular control of composed systems. In: *Proc. Of the American Control Conference*, Chicago, USA.
Queiroz, M. H and J. E. R. Cury (2000b). *Modular supervisory control of large scale discrete-event systems*. Discrete event systems: analysis and control, Kluwer Academic Publishers (proc. WODES, Ghent, Belgium).
Queiroz, M. H. and J.E.R. Cury (2002). *Synthesis and implementation of local modular supervisory control for a manufacturing cell.* Discrete Event Systems: Analysis and Control, Kluwer Academic Publishers, pp. 103-110. (*Proc. WODES 2002*).
Ramadge, P. J., and W. M. Wonham (1989). The control of discrete event systems. In: *Proceedings IEEE, Special Issue on Discrete Event Dynamic Sys tems,* vol. 77, pp. 1202-1218.
Vaz, A. F. and Wonham, W. M. (1986). On supervisor reduction in discrete-event systems. *InternationalJournal of Control,* 44(2):475-491.