Increasing the efficiency of PLC Program Verification using a plant model

José J.B. Machado, Bruno Denis, Jean-Jacques Lesage, Jean-Marc Faure, Jaime C. L. Ferreira da Silva

Increasing the efficiency of PLC Program Verification using a plant model

José M. MACHADO [(1)] / Bruno DENIS [(2)] / Jean-Jacques LESAGE [(2)] / Jean-Marc FAURE [(2)] / Jaime C. L. FERREIRA DA SILVA [(1)]

[(1)] Mechanical Engineering Department, University of Minho, PORTUGAL
[(2)] LURPA, Ecole Normale Supérieure de Cachan, FRANCE
email: {jmachado, jaimefs}@dem.uminho.pt; {denis, lesage, faure}@lurpa.ens-cachan.fr

Abstract. More extensive work on formal methods is now available for checking PLC (Programmable Logic Controller) programs. Some approaches propose taking into account just the control system model, while others include a plant model as well. Accordingly, the level of detail of the model is very different from one approach to the next. In this paper, we explore the contribution of such a plant model within the context of formal verification by means of Symbolic Model-Checking. Our study is primarily experimental in nature and based on a case study. A set of properties to be checked and a detailed plant model are both proposed. We then analyze how a Symbolic Model-Checking tool (the NuSMV has been selected) ensures verification of these properties either with or without a plant model.

Key words: Discrete event systems, Plant model, Formal verification, Model-checking

Introduction

The work presented in this paper lies within the framework of a cooperative research program between the Mechanical Engineering Department of the University of Minho in Portugal (DEM) and the LURPA (Laboratoire Universitaire de Recherche en Production Automatisée) of ENS (École Normale Supérieure de Cachan) in France. This joint program focuses on the topic of "Dependable Control of Manufacturing Systems".

When designing and implementing the control of complex manufacturing systems, automation engineers are required to check that the behavioral models and controller programs they develop indeed fulfill all application requirements, especially those related to dependability. Formal verification methods, such as model-checking (Kowalewski *et al.*, 1999; Lampérière-Couffin *et al.*, 2000; Bornot *et al.*, 2000) or theorem-proving (Roussel and Denis, 2002) may be used to achieve this objective.

Nevertheless, the use of these methods in industrial models or programs requires considerable skill and can lead to combinatory explosion. In order to overcome this problem, several solutions may be envisaged: modular verification, introduction of constraints on variable states, or introduction of a plant model (model of the physical system) (Rausch and Krogh, 1998).

The work presented herein is intended to highlight the advantages of whether or not a plant model should be taken into account in formal verification methods for PLC (Programmable Logic

Controller) programs (see Figure 1). Does the introduction of a plant model allow verifying additional properties? Use of a plant model increases the size of models to be verified. Does this represent an obstacle to verification? The following assumptions will be made:

- The formal verification method is model-checking.
- Controller behavior is described in accordance with the IEC 61131-3 Standard.
- The plant behavior model is built using finite state automata.

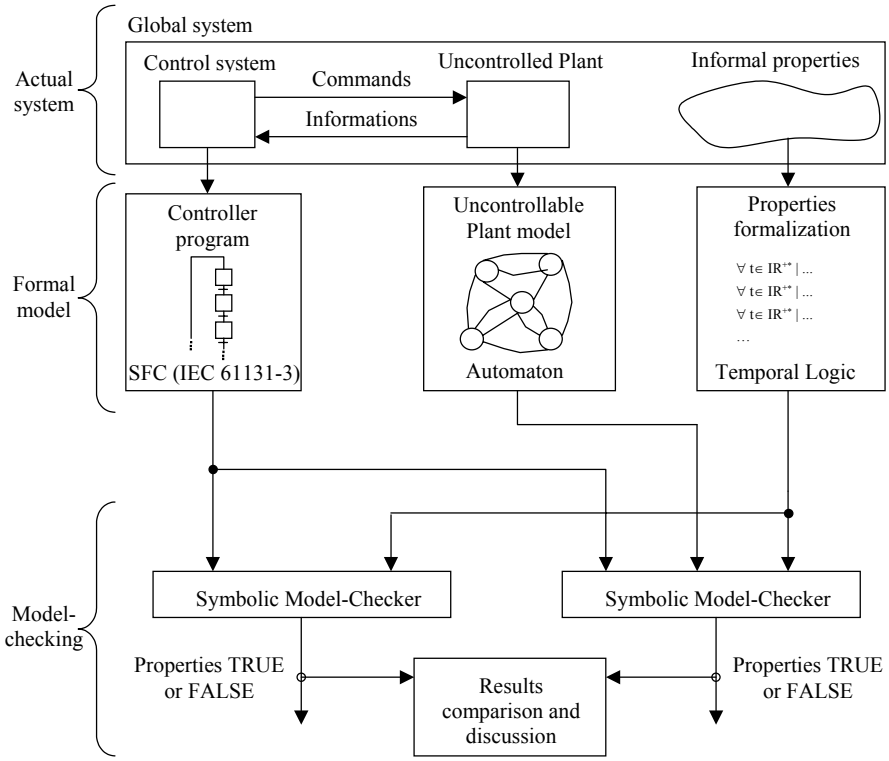


Figure 1: Verifying the global system

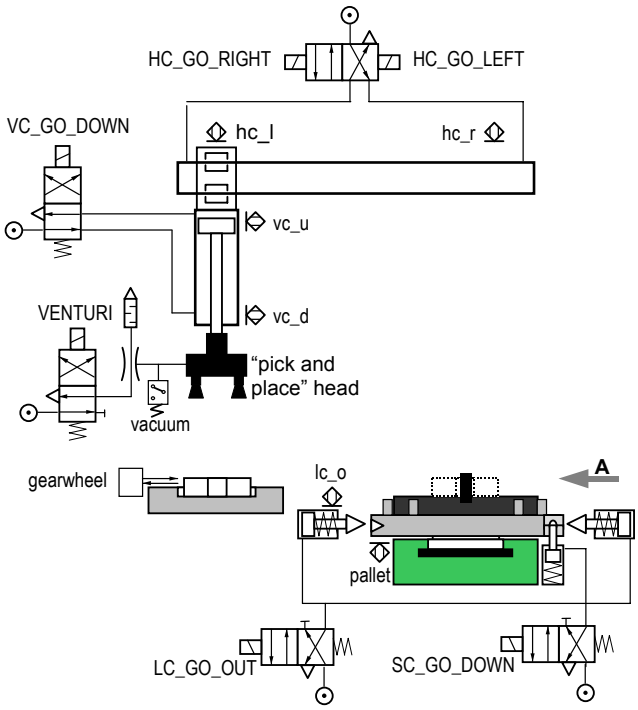
The paper has been organized as follows. Section 1 is devoted to the general presentation of a case study involving a "pick-and-place" workstation. We thus begin with a workstation already designed mechanically, as well as its PLC program in accordance with the IEC 61131-3 Standard. We next provide a list of the non-formal properties to be checked by the PLC program. Section 2 focuses on formalizing properties in temporal logic. Section 3 then describes a behavioral plant model of the uncontrolled workstation. The last section discusses model-checking results in order to determine, from this case study, the impact of the uncontrolled plant model within the formal verification method.

1. Case study

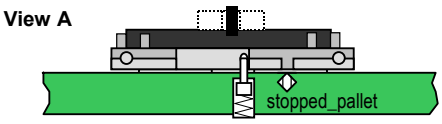
1.1 Aim and structure of the entire system

The case study presented is based on an assembly line that produces spur gears. A chain conveyor transfers gear housing from one workstation to the next. In this paper, we will focus on a pick-and-place workstation with the main operations consisting of: stop and locate incoming pallets, pick up gearwheels with suction cups, transfer gearwheels to gear housing using two pneumatic cylinders, and release pallets (gearwheel feeding lies beyond the scope of this study).

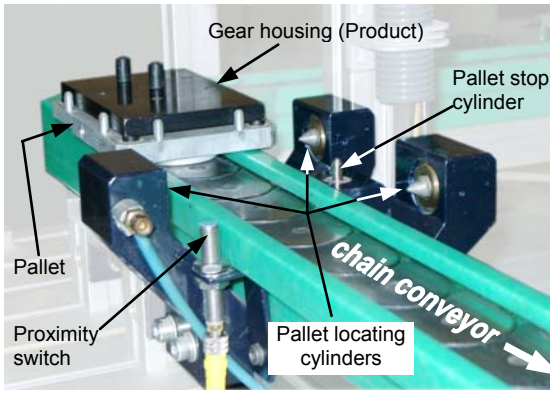
Figure 2 provides the various views of the workstation. It is important to note the diversity of the actuators technologies employed herein: double-acting cylinders, single-acting spring-loaded cylinders and single-acting spring-retracted cylinders. With respect to the pre-actuators, both single- and dual-solenoid valves are involved.



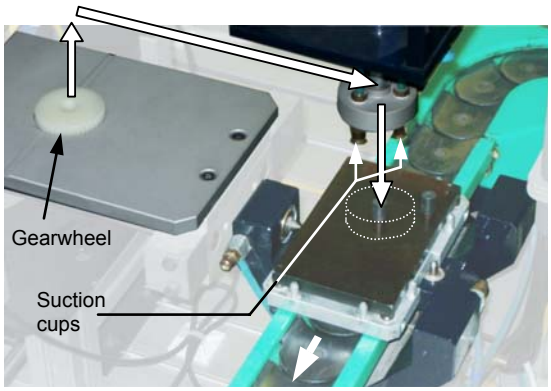
a: layout, actuators and pre-actuators



b: detail of the layout of stopped_pallet sensor



c: locating station



d: assembly sequence

Figure 2: Overall presentation of the workstation

1.2 Control system behavior

Control system behavior is expressed according to IEC 61131-3 and presented in Figure 3. The control system boundary is composed of a set of logical inputs and outputs, as follows:

Input	Output
pallet - Inductive proximity sensor indicating the presence of a pallet within the station-locating area. **stopped_pallet** - Inductive proximity sensor indicating the presence of a pallet at the exact station-locating place. **lc_o** - Magnetic sensor indicating whether or not the locating cylinder is out. **gearwheel** - Optical sensor indicating the presence of the gearwheel in place for holding prior to pallet transfer. **vacuum** - Vacuum sensor indicating whether or not the venturi is activated. **vc_d** - Magnetic sensor indicating the end of the stroke when the vertical cylinder is down (out). **vc_u** - Magnetic sensor indicating the end of the stroke when the vertical cylinder is up (in). **hc_r** - Magnetic sensor indicating the end of the stroke when the horizontal cylinder is on the pallet side. **hc_l** - Magnetic sensor indicating the end of the stroke when the horizontal cylinder is on the gearwheel side.	**SC_GO_DOWN** - Solenoid of the stop cylinder valve serving to free the pallet. **LC_GO_OUT** - Solenoid of the locating cylinder valve serving to locate the pallet. **VC_GO_DOWN** - Solenoid of the vertical cylinder valve. **HC_GO_RIGHT** - Solenoid of the horizontal cylinder valve that moves the "pick-and-place" head in the locating station direction (bistable function). **HC_GO_LEFT** - Solenoid of the horizontal cylinder valve that moves the "pick-and-place" head in the gearwheel-loading direction (bistable function). **VENTURI** - Solenoid of the vacuum system that enables gearwheel lifting.

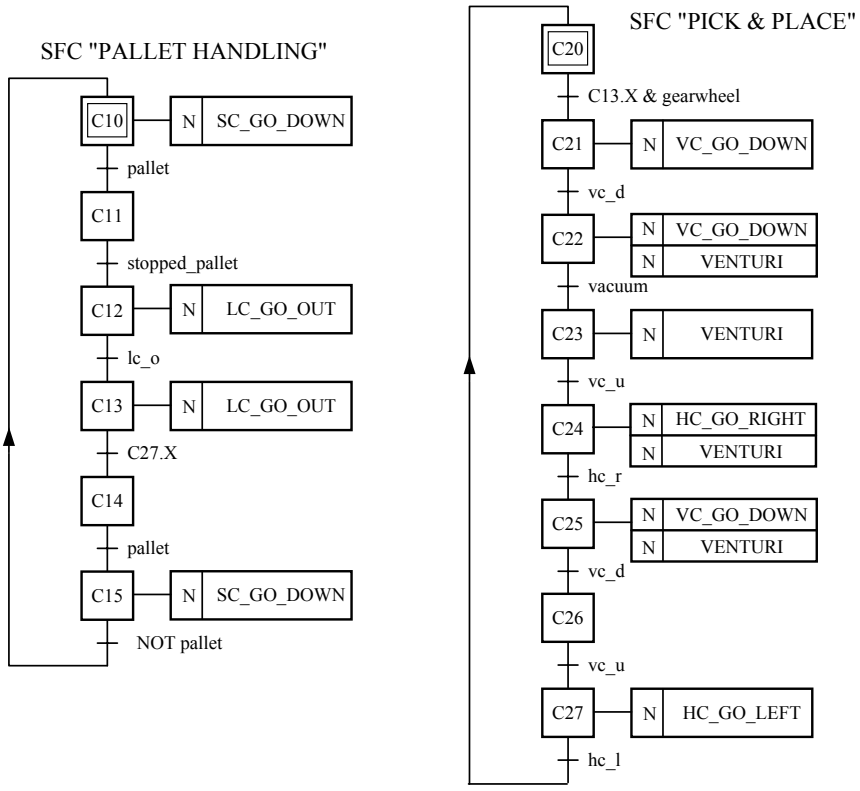


Figure 3: SFC specifications of the described APS control system

1.3 Expected informal properties

Considerable research has dealt with intrinsic PLC program properties, such as "*each step of our model must be reachable*" or "*there is no deadlock*". These properties have been checked in the current case study to ensure PLC program quality yet will not be developed any further in this paper. Interested readers seeking additional information are referred to Bornot *et al.*, 2000 and De Smet and Rossi, 2001. We will focus herein directly on the functional properties, such as mechanical collision avoidance. Table 1 lists selected properties that lead to various difficulties.

Table 1: List of expected properties

	Non-formal description	Relevant information			Remarks
		PLC Program inputs	PLC Program states	PLC Program outputs	
PROP1	It is forbidden to command the horizontal cylinder in two directions at the same time.			✓	Basic property involving only one actuator
PROP2	If the vertical cylinder is going out, then the horizontal cylinder must remain immobile.			✓	Property involving two actuators
PROP3	If the vertical cylinder is going in, then the horizontal cylinder must remain immobile.		✓	✓	Property involving a mono-stable function
PROP4	If the vertical cylinder is out, then the horizontal cylinder must remain immobile.	✓		✓	Basic property whereby an input implies output behavior
PROP5	The vertical cylinder must be retracted during movement of the horizontal cylinder.	✓		✓	Complex property mixing inputs and outputs
PROP6	If the vertical cylinder is going down to place the gearwheel in its housing, then the pallet must be located.	✓		✓	Basic property whereby an output implies input behavior

2. Property formalization

In Section 1, expected properties were presented in a non-formal manner. In order to input these properties into a model-checker, they must now be formulated formally. Classical Boolean operators are used for this purpose. Negation $\neg$, Boolean connectives $\wedge$ (conjunction, “and”), $\vee$ (disjunction, “or”), and $\Rightarrow$ (logical implication) where P$\Rightarrow$Q stands for “if P then Q” are all employed in this effort. Such operators allow constructing complex statements relating various atomic propositions.

Formalization of PROP1

$$\forall\, t \in \mathbb{R}^{+*} \mid \neg(\text{HC\_GO\_RIGHT} \wedge \text{HC\_GO\_LEFT}) \qquad \text{(PROP1)}$$

Formalization of PROP2

$$\forall\, t \in \mathbb{R}^{+*} \mid \text{VC\_GO\_DOWN} \Rightarrow \neg(\text{HC\_GO\_RIGHT} \vee \text{HC\_GO\_LEFT}) \qquad \text{(PROP2)}$$

Formalization of PROP3 Vertical cylinder goes up by lack of VC_GO_DOWN order. Then, SFC program step states are used to express the property:

$$\forall\, t \in \mathbb{R}^{+*} \mid (C23.X \vee C26.X) \Rightarrow \neg(\, HC\_GO\_RIGHT \vee HC\_GO\_LEFT) \qquad \text{(PROP3)}$$

Formalization of PROP4

$$\forall\, t \in \mathbb{R}^{+*} \mid vc\_d \Rightarrow \neg(HC\_GO\_RIGHT \vee HC\_GO\_LEFT) \qquad \text{(PROP4)}$$

Formalization of PROP5. To express this property, all the combinations of the relevant inputs and outputs must be considered (see Figure 4). Karnaugh map allows us to be exhaustive to obtain variable combinations (truth table could also be used). We can define the "authorization for the vertical cylinder to go out" (AVCGO) as a combination of some variables like is shown in presented Karnaugh map. Then, it comes the following expression to AVCGO:

$$AVCGO = (hc\_r \wedge \neg hc\_l \wedge \neg HC\_GO\_LEFT) \vee (\neg hc\_r \wedge hc\_l \wedge \neg HC\_GO\_RIGHT)$$

The property that must be verified can be enounced as follows: "The vertical cylinder goes out if it is "authorized" to do that. In a point of view of the input and output signals we have:

HC_GO_RIGHT, HC_GO_LEFT \ hc_r, hc_l	00	01	11	10
00	0	1	0	1
01	0	1	0	0
11	0	0	0	0
10	0	0	0	1

Figure 4 Karnaugh map of AVCGO

$$\forall\, t \in \mathbb{R}^{+*} \mid VC\_GO\_DOWN \Rightarrow AVCGO\ ;$$

$$\forall\, t \in \mathbb{R}^{+*} \mid VC\_GO\_DOWN \Rightarrow ((hc\_r \wedge \neg hc\_l \wedge \neg HC\_GO\_LEFT) \vee (\neg hc\_r \wedge hc\_l \wedge \neg HC\_GO\_RIGHT)) \qquad \text{(PROP5)}$$

Formalization of PROP6

$$\forall\, t \in \mathbb{R}+* \mid (VC\_GO\_DOWN \wedge hc\_r) \Rightarrow lc\_o \qquad \text{(PROP6)}$$

3. Design of plant behavior model

Most research work pertaining to formal verification merely targets the PLC program and fails to take plant behavior into account. Incorporating plant behavior (see Frey and Litz, 2000) leads to the potential use of either constrained-based or model-based approaches. Constrained-based approaches only include simple knowledge about the plant (e.g. two position sensor signals always being disjointed), whereas the model-based approach relies upon a detailed complex model (e.g. the dynamic discrete event model).

In our approach, we only consider model-based approaches and are excluding:

- models other than logical ones,
- plant failures (we assume that the plant is in an automatic operating mode),
- any non-automatic operating modes.

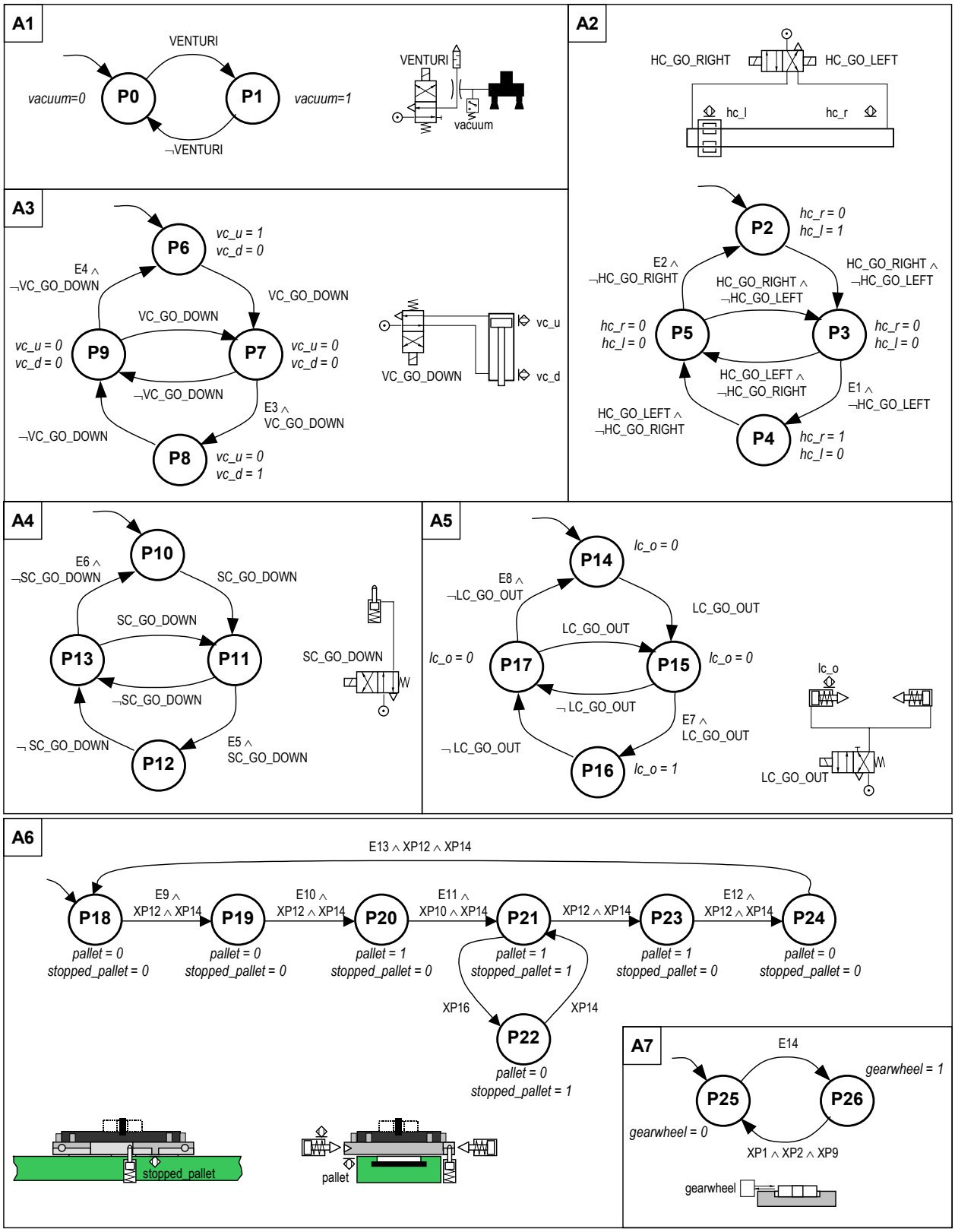


Figure 5: Uncontrolled plant model

In order to facilitate design of the uncontrolled plant model, a modular approach is proposed whereby each module can be identified as part of the entire plant. The plant model for the present case study has been divided into seven modules (see Figure 5). Five of these modules are composed of a generic automaton structure given that they describe the behavior of a generic component structure:

- (A1) suction cups with a venturi grip system driven by a single solenoid valve (VENTURI) and observed by a vacuum sensor (vacuum);
- (A2) a double-acting horizontal cylinder driven by a dual-solenoid valve (HC_GO_LEFT, HC_GO_RIGHT) and observed by two end-stroke sensors (hc_l, hc_r);
- (A3) a double-acting vertical cylinder driven by a single solenoid valve (VC_GO_DOWN) and observed by two end-stroke sensors (vc_u, vc_d);
- (A4) a single-acting and spring-loaded stop cylinder driven by a single solenoid valve (SC_GO_DOWN) and not observed;
- (A5) a single-acting and spring-retracted locating cylinder driven by a single solenoid valve (LC_GO_OUT) and observed by an extended end-stroke sensor (lc_o).

The three remaining modules are composed of specific automata given that they are related to the specific workstation layout:

- (A6) pallet-locating system - this automaton yields the behavior for both the “pallet” and “stopped-pallet” sensors;
- (A7) gearwheel-loading area - this automaton yields the behavior for the “gearwheel” sensor.

In this plant model, uncontrollable variables (Ei) have been introduced to handle unspecified behavior. Synchronization between automata is archived using state status (active/inactive) on transition labels. For example, XP1 stands for “State P1 is active”.

4. Verification results

Using the PLC program, coupled with a model of the uncontrolled plant and the formal properties, it is now possible to start the model-checking. The Symbolic Model-Checking tool chosen is NuSMV, Version 2.1.2; this tool has been designed for an automaton as behavior input specification. The uncontrolled plant model is then easily translated into NuSMV code. For the PLC program translation, algebraic equations have been introduced (Marcé and Le Parc, 1993; Lampérière *et al.*, 2000). All properties have been checked in less than two seconds on an Intel P4 architecture running on Windows XP operating system with 256 MB of memory and the following NuSMV options: -reorder -dynamic. Table 2 lists the results of each property check for each model, i.e. either with or without the plant model.

In reviewing the experimental results, our first remark is that certain properties can be true or false depending on whether the plant model is being implemented or not. Does this finding suggest that PLC programs are sometimes correct and sometimes incorrect? Obviously not, because the same program is always being targeted. The reason for this change in behavior of certain properties stems from the fact that the boundary of the system checked by NuSMV is changing. Even if our aim still remains checking the PLC program, in one case NuSMV checks behavior of the unconstrained PLC

program, and in the other case, it checks the synchronized behavior of the PLC program with the uncontrolled plant model.

Table 2: Model-checking results

	Without plant model	**With plant model**	**Nature of the property**
PROP1	true	true	$\forall t>0 \mid Q_1.Q_2=0$
PROP2	true	true	$\forall t>0 \mid f(Q_k)=1$
PROP3	true	true	$\forall t>0 \mid f(X_k) \Rightarrow g(Q_j)$
PROP4	false	true	$\forall t>0 \mid I_k \Rightarrow f(Q_k)$
PROP5	false	true	$\forall t>0 \mid Q_1 \Rightarrow f(I_k, Q_j)$
PROP6	false	true	$\forall t>0 \mid I_1 \Rightarrow f(I_k, Q_j)$
Reachable states	18 out of 16384	962 out of 1.17441e+08	

Q resp. X and I stands for Output resp. step state and input of PLC program

When we set out to check properties of the PLC program that only involve variables controlled by the PLC (i.e. outputs and step states), no contribution from the uncontrolled plant model is present (see PROP1 through PROP3). Conversely, once a program input variable is involved, the Model-Checker can no longer make any positive conclusions on the properties (see PROP4 through PROP6). In this case, only the addition of a plant model enables the engineer to pursue PLC program property-checking.

While the required plant modeling effort gives rise to an increase in the number of proven properties, the price to pay in terms of model size warrants further examination. As expected, the number of reachable states increases only moderately as model size increases (i.e. from 18 to 303); on the other hand, the number of potential model states increases most dramatically, from 16,384 to 1.69114e+10. Our case study did not entail any major increase in computing time, which always remained less than 2 seconds; for larger case studies however, this factor could become penalizing.

Conclusion and outlook

This study has shown that the impacts of using a plant model within the formal verification process are indeed sizable. Certain properties cannot be verified by the PLC program model alone. An uncontrolled plant model provides a means for enhancing the engineer's capacity to prove new properties. Adding a plant model to the control system model however increases the number of reachable states.

In order to reconcile these two aspects, subsequent work should consider both a partial plant model and different levels of model detail. We will also be exploring failure models for the uncontrolled plant so as to determine the robustness of a PLC program when confronted with plant component failure.

References

Bornot, S., R. Huuck, Y. Lakhnech, and B. Lukoschus (2000). Verification of sequential function charts using SMV. In: *PDPTA 2000 special session on Formal Validation*. Las Vegas.

De Smet, O. and O. Rossi (2002). Verification of a controller for a flexible manufacturing line written in a Ladder Diagram via model-checking. In: *21th American Control Conference*. CDRom paper Nº734, pp. 4147-4152, Anchorage.

Frey, G. and L. Litz (2000). Formal Methods in PLC programming. In: *2000 IEEE International Conference on Systems, Man & Cybernetics*. pp. 2431-2436, Nashville.

IEC 61131-3 (1998). Programmable Controllers – Programming languages, 1998.

Lampérière-Couffin, S., and J.-J. Lesage (2000). Formal verification of the sequential part of PLC programs. In: *Wodes2000 - 5th Workshop on Discrete Event Systems*. pp. 247-254, Ghent.

Kowalewski, S., S. Engell, J. Preußig and O. Stursberg (1999). Verification of Logic Controllers for Continuous Plants Using Timed Condition/Event-System Models. *Automatica*, 35, 505-518.

Marcé, L. and P. Le Parc (1993). Defining the semantics of languages for programmable Controllers with synchronous processes. *Control Engineering Practice*, 1, 79-84.

Rausch, M. and B. H. Krogh (1998). Formal Verification of PLC Programs, *American Control Conference*. Philadelphia.

Roussel, J-M., B. Denis (2002). Safety properties verification of ladder diagram programs. *Journal Européen des Systèmes Automatisés*. 36, 905-917.