POLITECNICO DI MILANO
DEPARTMENT DIPARTIMENTO DI ELETTRONICA, INFORMAZIONE E BIOINGEGNERIA
DOCTORAL PROGRAMME IN INFORMATION TECHNOLOGY

MODEL PREDICTIVE CONTROL IN MANUFACTURING PLANTS

Doctoral Dissertation of:
Andrea Cataldo

Supervisor:
Prof. Riccardo Scattolini

Tutor:
Prof. Marco Lovera

Chair of the Doctoral Program:
Prof. Andrea Bonarini

2016 – XXVIII Cycle

to The Life

I would like to thank some people for this PhD thesis, in particular:

Professor Tullio Tolio,
as Director of the Institute of Industrial Technologies and Automation - National Research Council (ITIA-CNR),
for believing in my skills and then for allowing me to address these PhD studies

and

Professor Riccardo Scattolini,
the Supervisor of my PhD studies at DEIB-POLIMI,
for being able to turn this period of study and scientific specialization into an excellent working experience.

Moreover, I'm really grateful to the colleagues and friends of mine, Professor Marcello Farina and Dr. Bruno Picasso,
with whom I could have had discussions and dispelled doubts about the fascinating field of Model Predictive Control.

Abstract

Over the years manufacturing industries have become more and more demanding due to the complexity of production processes and to tighter rules and regulations. In the global market, enhancing the efficiency and productivity of manufacturing systems is mandatory to maintain high levels of competitiveness. Moreover the energy efficiency of manufacturing production systems is becoming a topic of paramount interest for many reasons, such as the need to minimize the energy consumption of industrial plants, to resize the factory energy supply infrastructures and to limit the CO_2 emissions.
Among the main issues in these fields, the development of advanced control strategies, such as Model Predictive Control (*MPC*), has an important role for the solution of many significant problems such as lotsizing, scheduling, packing, inventory, resource allocation, energy efficiency.
MPC is a control method nowadays widely used in the process industry in view of its capability of dealing with complex systems. This is due to the possibility with *MPC* to enable reformulation of control problems into optimisation ones, which gives the opportunity to explicitly add constraints on the control inputs and the controlled variables. Nevertheless, model predictive controllers are mostly used with continuously varying systems, while they are less frequently applied to discrete-event systems, typical of the manufacturing field. The scarce use of *MPC* for discrete-event systems can be explained by the fact that they are characterised by integer or Boolean decision variables, so that the use of *MPC* could lead to large combinatorial optimisation problems to be solved on-line, which is seen as a computational bottleneck.

In this thesis, in order to prove the applicability of *MPC* to control problems typical of manufacturing systems, different industrial applications have been taken into account and the adequacy of *MPC* in terms of *"easy to design and use"* and performances has been proven.

Firstly, in Chapter 2 the efficient routing of the pallets in networks made by machines and transportation lines is studied in order to avoid bottlenecks, starvation, congestion, and to maximize throughputs. However, the design of optimal routing strategies is difficult due to the combinatorial nature of the problem and the implemented control laws are often based on heuristic logic rules tuned by means of simulation studies. In this scenario, *MPC* has been applied to a de-manufacturing transport line in which a multi-pallet, dynamic multi-target problem has to be solved. The dynamic system of the transport line has been formulated as a Mixed Logical Dynamical (*MLD*) system. Then, a performance index has been defined and the optimal control sequence has been recursively computed and applied according to a receding horizon approach. The *MPC* algorithm developed has been used to control the transport line placed at the Institute of Industrial Technology and Automation - National Research Council ($ITIA-CNR$).
Secondly, in Chapter 3 the problem of optimizing on-line the production scheduling and the buffer management of a multiple-line production plant composed by L machines M_i, $i = 1,...,L$, which can operate at different speeds corresponding to different energy demands, has been considered. The path from a common source node, where the part to be processed is assumed to be always available, to each machine may differ in the number of buffer nodes and the energy required to move the part along these transportation lines must be suitably considered in the computation of the overall energy consumption. Therefore, the control problem consists of computing, at each sampling instant, the sequence of commands to be applied to the transportation lines and the processing speed of the machines in order to optimize the throughput of the system and to limit the overall energy consumption.
In Chapter 4 a laboratory stacker crane, a specific type of Automated Storage / Retrieval System (AS/RS), has been considered. The AS/RS system has been modeled in terms of *MLD* system and the control problem has been reformulated as an integer linear programming problem.
Finally, in order to be able to compute the energy consumption for a manufacturing plant, in Chapter 5 a specific new energy consumption computation method has been defined and validated, named aCtuatorS Methodology (*CSM*), based on the Discrete Event System (*DES*) approach for the computation of the energy consumption of discrete systems, i.e. systems where the energy consumption is mainly due to the on/off switching of the actuators governed by the control logic.

In order to evaluate the *MPC* performances in the considered applications,

many experiments have been performed.
Concerning the de-manufacturing transport line, the experimental results show the very satisfactory behavior of the proposed algorithm when applied both to the discrete event simulation model and to the real system.
In the problem related to the multiple-line production plant, the simulation results show that the algorithm is highly flexible and its performance can be easily adapted to obtain different behaviours by means of the tuning of simple and easy-to-understand parameters of the cost function. Moreover, the proposed method allows to cope with dynamic changes of the minimum production and maximum absorbed power and to choose the constraints to be violated in case of infeasibility, features that are very difficult to be achieved with standard scheduling techniques based on the solution of *MILP* problems or on heuristics.
Finally, the laboratory stacker crane example witness the potentialities of this control method also for this class of problems.

According to the considered manufacturing applications, future research activity could be respectively aimed at:

- customizing the algorithm described in Chapter 2 to production lines with operating machines whose working function settings can be dynamically changed in order to further optimize the production line efficiency;
- including constraints on the early production of parts or considering non deterministic behaviours of the machines described in Chapter 3;
- improving the modelling phase by reducing the complexity of the AS/RS model considered in Chapter 4 and decreasing the required computation time. The focus hereby should be on the reduction of the number of integer variables, since these determine the complexity of an integer linear programming problem. The current model can easily be extended, e.g. to a form with multiple final storage nodes. Next to these improvements and extensions it would be interesting to compare the *MPC* method to other control methods such as time instant optimisation *MPC* and heuristics.

Contents

1 Introduction **1**
- 1.1 MPC concepts . . . 1
- 1.2 MPC strategy . . . 3
- 1.3 MPC applications . . . 5
- 1.4 Contents of the work . . . 8
- 1.5 Thesis structure . . . 11
- 1.6 List of publications . . . 11

2 Dynamic pallet routing in a manufacturing transport line **13**
- 2.1 The plant layout . . . 18
- 2.2 The plant low level control system . . . 21
- 2.3 The plant LLCS implications . . . 23
- 2.4 The dynamic model . . . 25
  - 2.4.1 Model of the nodes . . . 25
  - 2.4.2 Model of the Buffer Zones . . . 26
  - 2.4.3 Model of the machines . . . 28
- 2.5 MPC formulation . . . 30
- 2.6 MPC validation . . . 33
  - 2.6.1 Simulation results . . . 34
  - 2.6.2 Experimental results . . . 38

3 Production scheduling of parallel machines **41**
- 3.1 Problem formulation . . . 46
  - 3.1.1 Node model . . . 46
  - 3.1.2 Simplified Machine Model (SMM) . . . 48
  - 3.1.3 Enhanced Machine Model (EMM) . . . 51
  - 3.1.4 Mixed logical dynamical model of the system . . . 55
  - 3.1.5 Overall plant model . . . 56
- 3.2 Model predictive control . . . 57
- 3.3 Simulation experiments . . . 61

3.3.1 Minimum production 62
3.3.2 Production maximization 64
3.3.3 Computational issues 66
3.3.4 Sensitivity to the cost function parameters 67
3.3.5 Comparisons with a non *RH* solution 67

4 Automated storage / retrieval system 79
4.1 The automated storage / retrieval system 83
4.2 Dynamic model 86
4.2.1 State variables 86
4.2.2 Decision variables 87
4.2.3 Dynamic equations 87
4.2.4 Deliveries and demands 88
4.2.5 Constraints 89
4.3 MPC formulation 91
4.4 Case study 93
4.4.1 Laboratory stacker crane 93
4.4.2 Weight factors 94
4.4.3 Simulation results 94

5 An energy consumption evaluation methodology for manufacturing plants 99
5.1 The CSM energy consumption methodology 103
5.2 The application of CSM to a de-manufacturing plant . . . 104
5.2.1 Control system structure 106
5.2.2 Dynamic control platform for industrial plants . . 107
5.2.3 Discrete event system simulator of the plant 111
5.3 Simulation and experiments results 113
5.3.1 CSM applied to a simple introductory example . . 113
5.3.2 CSM applied to the simulated model of the pallet transport line 115
5.3.3 CSM applied to the de-manufacturing pilot plant . 118

6 Conclusions 123

Appendices 127

A MLD model of the transport line 129

B Low level control system sequences **139**
B.1 Transport module sensors and actuator 140
B.2 Transport module control sequences 158
B.3 Finite state machine control sequences 173

C Low level control system sequence list **209**

Acronyms **213**

List of Figures **215**

List of Tables **219**

Bibliography **221**

Chapter 1

Introduction

Over the years manufacturing industries have become more and more demanding due to the complexity of production processes and to tighter rules and regulations. In the global market, enhancing the efficiency and productivity of manufacturing systems is mandatory to maintain high levels of competitiveness. Moreover the energy efficiency of manufacturing production systems is becoming a topic of paramount interest for many reasons, such as the need to minimize the energy consumption of industrial plants, to resize the factory energy supply infrastructures and to limit the CO_2 emissions.
Among the main issues in these fields, the development of advanced control strategies, such as *MPC*, has an important role for the solution of many significant problems such as lotsizing, scheduling, packing, inventory, resource allocation, energy efficiency, that in general can be described by means of discrete-event system models.

Starting from these considerations, the scope of this thesis consists of applying the MPC technique to discrete-event systems, in particular to specific manufacturing industrial applications, in order to evaluate its adequacy in terms of "easy to design and use" and performances.

1.1 MPC concepts

Among the advanced control strategies nowadays available, *MPC* represents a control method widely used in the process industry in view of its capability of dealing with complex systems, due to the possibility to enable reformulation of control problems into optimisation ones, which gives the

opportunity to explicitly add constraints on the control inputs and the controlled variables.
MPC [17, 77, 95] does not designate a specific control strategy but rather a set of control methods which make use of a mathematical model of the system to be controlled in order to obtain the control signal, by minimizing a specific objective function.
The basic concepts are:

- use of a mathematical model to predict the process output at future time instants (prediction horizon);
- calculation of an optimal control sequence by minimizing a specific objective function;
- implementing the receding horizon strategy, that is at each time instant, the horizon is displaced towards the future and just only the first control signal of the calculated optimal sequence is applied at each step.

MPC presents different advantages over other control methods:

- it can be used to control a great variety of processes, from those with relatively simple dynamics to more complex ones [91], including systems with long time delay or nonminimum phase or unstable ones;
- the multivariable case can easily be dealt with;
- it intrinsically compensates for dead times;
- it introduces feed forward control in a natural way to compensate for measurable disturbances;
- its extension to the treatment of constraints and multiple objectives [77, 95] is conceptually simple and these can be systematically included during the design process;
- it is an open methodology based on basic principles which allow for future extensions.

However, it also has its drawbacks:

- if the process dynamics do not change and the system is unconstrained, the derivation of the controller can be done beforehand, but in other cases all the computations have to be carried out at every sampling time. In addition, when constraints are considered, the amount of computation required is even higher;

- an appropriate model of the process is required. The design algorithm is based on prior knowledge of the model and the benefits obtained will be affected by the discrepancies existing between the real process and the model used.

1.2 MPC strategy

The *MPC* methodology is characterized by the following strategy, represented in Figure 1.1:

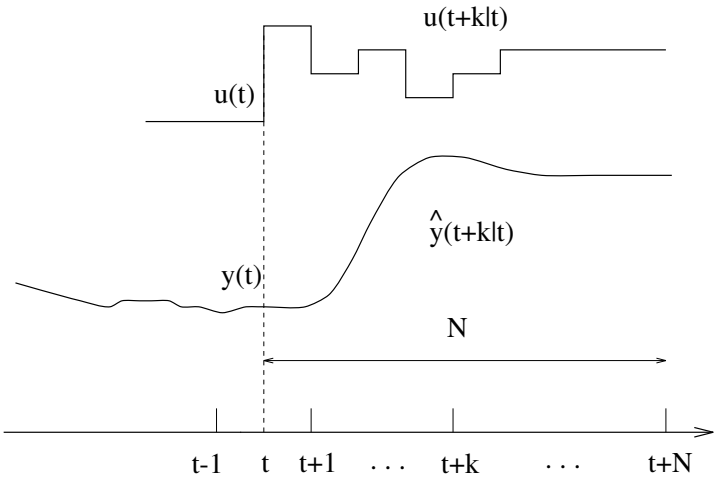


Figure 1.1: MPC strategy.

1. the future outputs in a determined prediction horizon N_{RH} are predicted at each discrete-time instant t using the process model. These predicted outputs $y(t+k|t)$ [1] for $k = 1 \ldots N$ depend on the known values up to instant t, past inputs and outputs, and on the future control signals $u(t+k|t)$, $k = 0 \ldots N-1$, which are those to be computed and sent to the system;

2. the set of future control signals is calculated by optimizing a given criterion to keep the process as close as possible to the reference trajectory $w(t+k)$, which can be the setpoint itself or a close approximation of it. This criterion usually takes the form of a quadratic function of the errors between the predicted output signal and the predicted reference trajectory. The control effort is included in the objective function in most

[1] The notation indicates the value of the variable at the instant $t+k$ calculated at instant t.

cases. An explicit solution can be obtained if the criterion is quadratic, the model is linear and there are no constraints; otherwise an iterative optimization method has to be used. Some assumptions about the structure of the future control law are also made in some cases to simplify the optimization problem, such as that it will be constant from a given instant onward;

3. the control signal $u(t|t)$ is sent to the process whilst the calculated control signals $u(t+i|t)$, $i>0$ are rejected because at the next sampling instant $y(t+1)$ is already known and step 1 is repeated with this new value and all the sequences are brought up to date. Thus the term $u(t+1|t+1)$ is calculated, which in principle will be different from the $u(t+1|t)$ because of the new information available, using the receding horizon concept.

In order to implement this strategy, the basic structure shown in Figure 1.2 is used.

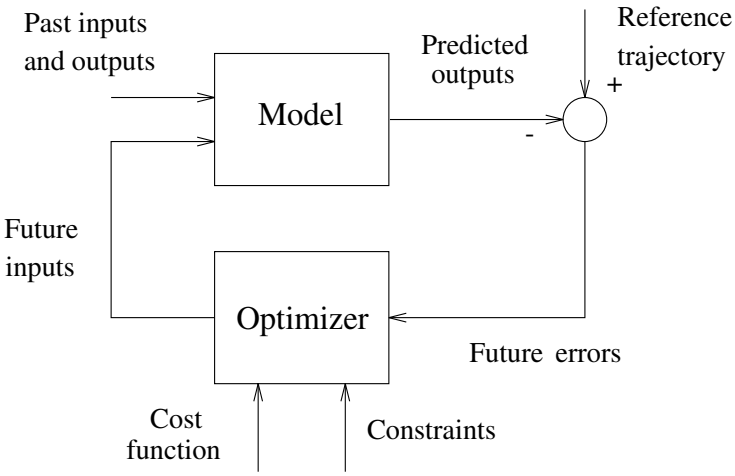


Figure 1.2: MPC schema.

A model is used to predict the future plant outputs, based on past and current values and on the proposed optimal future control actions. These actions are calculated by the optimizer taking into account the cost function, where the future tracking error is considered, as well as the constraints. The process model plays, in consequence, a decisive role in the controller. The chosen model must be able to capture the process dynamics to pre-

cisely predict the future outputs and must be simple to implement and understand. The optimizer is another fundamental part of the strategy as it provides the control actions. If there are no constraints and the cost function is quadratic, its minimum can be obtained as an explicit linear function of past inputs and outputs and the future reference trajectory. In the presence of constraints, the solution has to be obtained by more computationally demanding numerical algorithms. The size of the optimization problems depends on the number of variables and the prediction horizons used. However, the amount of time needed for constrained problems can be of various orders of magnitude higher than the one needed for the unconstrained case and the bandwidth of the process to which constrained *MPC* can be applied is considerably reduced.
For what concerns the stability properties of *MPC* implemented by means of the receding horizon idea, it might be possible that the resulting closed loop is unstable. To this regard, many *MPC* algorithms have been developed to guarantee stability in both nominal and perturbed conditions (robust *MPC*, see [95] for a thorough description of the main approaches). These methods have been extended to specific classes of hybrid systems [49], in which continuous and discrete variables are considered in the system model, see [12].

1.3 MPC applications

There are many applications of predictive control successfully in use at the current time, not only in the process industry but also in other fields, ranging from robots [86] to clinical anaesthesia [73]. Applications in the cement industry, drying towers, and robot arms are described in [28] whilst developments for distillation columns, *PVC* plants, steam generators, or servos are presented in [96] and [97]. The good performance of these applications shows the capacity of the *MPC* to achieve highly efficient control systems able to operate during long periods of time with hardly any intervention. Although *MPC* is characterized by these features, its application in manufacturing industry is still limited, while notable exceptions refer to the following areas:

- *Baggage handling systems*, see [109–112]. The baggage handling system plays a decisive role in the airport's efficiency and comfort. It is successful if all the bags are transported to the corresponding lateral, a

lateral being the place where the bags are lined up, waiting to be loaded in containers, before the plane has to be loaded. Hence, the process is time critical. The faster the transportation is performed, the more efficient the baggage handling system is. In order to transport the bags in an automated way, a baggage handling system could incorporate technology such as scanners that scan the labels on each piece of luggage, baggage screen equipment for security scanning, networks of conveyors equipped with junctions that route the bags through the system, and Destination Coded Vehicles (*DCV*) which transport the bags at high speed on a "mini" railway network. The main control problems of a baggage handling system are coordination and synchronization of the processing units, route assignment of each bag, and implicitly the switch control of each junction, velocity control of each *DCV*, line balancing and prevention of buffer overflows. Synchronization and coordination are required when loading the bags onto the system in order to avoid damaging the bags and blocking the system, or when unloading them to the corresponding lateral. When using *DCV*, the route and the velocity profile of each vehicle have to be determined in order to assure the system optimum. The key in controlling the available capacity of the system consists in fact in assuring a balanced transport service. This problem has been named in [5] the "line balancing" problem. *MPC* can be adopted to determine the optimal route and velocity for each *DCV* in the network and then to optimize the baggage handling system efficiency;

- *Traffic networks*, see [7–9, 59, 70–72]. Urban areas are places where traffic congestion most likely happens, when people need to use the common infrastructures with limited capacity at the same time, especially during rush hours. Huge losses may be caused by traffic jams. Traffic delays grow because of the congestion, and the economic losses and the traffic pollution will also increase accordingly. Moreover, traffic congestion may also threaten the safety of the public transportation. Expanding the transportation infrastructure can alleviate the congestion to some extent, but it is too time and money consuming, and it is also limited by the existing geography of cities. Therefore, traffic control strategies are an attractive method to address congestion problems of urban areas. Since the emergence of traffic control, traffic control strategies have gone through various developments from isolated intersection control to coordinated control. Isolated intersection controllers have been well developed as local controllers. However, even though the local controller works properly, it still cannot guarantee that no congestion is caused in the other

regions within the traffic network. To avoid this phenomenon, it is necessary to have a coordinated control strategy for the whole traffic network. A number of coordinated urban network control strategies have already been developed. Fixed-time coordinated control strategies control and coordinate the control measures of the network based on the historical traffic data. But the fixed-time control strategy hardly respond to the real-time variations of the traffic demands and the disturbances. Traffic-responsive coordinated control strategies measure the traffic states in the network, and adapt real-time the control schemes according to the measured traffic states. Although the traffic-responsive control strategy can respond in real-time to the variations of traffic states, it is still a myopic control method, which does not look ahead. To overcome these disadvantages, *MPC* can be applied to control urban traffic networks;

- *Supply chain*, see [1, 20, 22, 63, 88, 118, 119]. Supply Chain Management (*SCM*) can be defined as a set of approaches used to efficiently integrate suppliers, manufacturers, warehouses and stores with the overall goal conceiving the maximization of the four customer service factors (the right products, in the right quantity, in the right place, at the right time) and the minimization of the four major costs that are materials, production, storage, transport. *SCM* is considered a competitive strategy for integrating supplies and customers with the objective of improving responsiveness and flexibility of manufacturing organizations and reducing their costs. This usually requires keeping safety stocks at very high levels, sometimes, as much as a whole year's worth of demand. Operating in this fashion is clearly undesirable. Therefore *MPC* strategies can be used to robustly manage inventory levels in supply chains despite inaccurate lead times and random disturbances. Recent work using *MPC* has shown it as an attractive method for inventory control and supply chain management. These approaches are conceptually different and require less detailed knowledge in comparison with cost-optimal stochastic programming solutions which require many "what-if" cases to be run and examined by highly skilled professionals. Yet *MPC* offers the same flexibility in terms of the information sharing, network topology, and constraints that can be handled. The appeal of *MPC* for dynamic inventory management in supply chains consists of minimizing or maximizing an objective function that represents a suitable measure for supply chain performance;

- *Manufacturing systems*, see [117, 118]. As already mentioned at the top of this chapter, the development of advanced control strategies has an important role for the solution of many significant problems such as lot-

sizing, scheduling, packing, inventory, resource allocation [90], energy efficiency [2, 15, 21, 32, 39, 46], and AS/RS management [65]. In particular the introduction of AS/RS improved inventory management and control, increased storage capacity and reliability, and reduced unnecessary labour costs. The complexity of the control algorithm of AS/RS is related to the number of depots and the storage policy of the system. Many papers use a random storage policy, which allows a pallet to be stored randomly on any available storage location. Due to the flexibility of such a policy, the number of possible solutions enlarges, which causes a higher complexity to optimally solve the storage assignment problem. Many researchers solve this problem with heuristics [3, 34, 47, 78]. A solution to problems with a dedicated storage policy is described by [44], where in contrary to a random storage policy, a dedicated storage policy predefines a unique storage location for each storage request. [44] show that this problem can be solved in polynomial time by using the fact that the crane always returns to a depot.

In general, one of the reason why *MPC* are mostly used with continuous systems and less frequently used in the manufacturing field regards the nature of manufacturing systems, which are discrete-event systems characterised by integer or Boolean decision variables [123]. For discrete-event systems, large combinatorial optimisation problems often need to be solved on-line, which is seen as a computational bottleneck. This aspect also depends on the mathematical model used to describe the discrete-event system, like for example Petri nets [66–68].

1.4 Contents of the work

In order to evaluate the *MPC* adequacy applied to discrete-event systems, different manufacturing industrial applications have been taken into account in this thesis.
The first application regards the efficient routing of the pallets in networks made by machines and transportation lines so as to avoid bottlenecks, starvation, congestion, and to maximize throughputs. The design of optimal routing strategies is difficult due to the combinatorial nature of the problem, and the implemented control laws are often based on heuristic logic rules tuned by means of simulation studies. In this scenario, *MPC* has been applied to a de-manufacturing transport line showed in Figure 1.3, see Pub-

blication (NJ1) and (NJ2), in which a multi-pallet, dynamic multi-target problem has to be solved.

Figure 1.3: The de-manufacturing plant.

The dynamic system of the transport line has been formulated as a *MLD* system, see Pubblication (IC2). Then, a performance index has been defined and the optimal control sequence has been recursively computed and applied according to a receding horizon approach. The developed *MPC* algorithm has been used for control of the transport line placed at $ITIA - CNR$, see Pubblication (IJ3).
The specificities of the problem here considered prevent the direct use of the algorithms described in baggage handling system research. In particular, [112] considers the management of a baggage handling system, which in some sense could be considered a similar problem but the loading and unloading stations are distinct and the allowed paths are substantially unidirectional, with just the possibility of intermediate loops, and the final destination of each baggage, i.e. the target unloading station, does not change during the operations. On the contrary, in the pilot plant of Figure 1.3, double directional paths among the machines must be followed and the target machines for the pallets on the transport line are dynamically changed in

a partially unpredictable way, which depends on the outcome of the testing machine. More in general, and to the best of the authors' knowledge, no systematic (not heuristic) algorithms are available for the problem here considered.

The second application regards the problem of optimizing on-line the production scheduling and buffer management of a multiple-line production plant composed by L machines M_i, $i = 1, ..., L$, which can operate at different speeds corresponding to different energy demands, see Pubblication (IJ1). The path from a common source node, where the part to be processed is assumed to be always available, to each machine may differ in the number of buffer nodes and the energy required to move the part along these transportation lines must be suitably considered in the computation of the overall energy consumption. Therefore, the control problem consists of computing, at each sampling instant, the sequence of commands to be applied to the transportation lines and the processing speed of the machines in order to optimize the throughput of the system and to limit the overall energy consumption.

The third application regards a specific type of AS/RS, see Pubblication (IC3). The complexity of controlling a AS/RS is related to the number of depots and the storage policy of the system. Due to the MPC advantages, the complexity coming from the application of MPC to AS/RS is overcome by describing the system by means of logical propositions and then by translating such model into a MLD system by means of the propositional calculus [12, 75, 79, 93, 122]. The transformation to an MLD system gives the advantage that the control problem can be expressed as a $MILP$ problem.

In order to be able to compute the energy consumption for a manufacturing plant, a specific new energy consumption computation method has been defined and validated, named CSM, based on the DES approach, according to which the energy consumption is mainly due to the on/off switching of the actuators governed by the control logic, see Pubblication (IJ2).

1.5 Thesis structure

The thesis is organized as follows.
Chapter 2 is aimed at investigating the potentialities of *MPC* to control the pallet routing in the de-manufacturing pilot plant.
Chapter 3 considers the problem of optimizing on-line the production scheduling of a multiple-line production plant composed of parallel equivalent machines which can operate at different speeds corresponding to different energy demands.
In Chapter 4 the *MPC* approach applied to a AS/RS is described together with the its dynamical model and constraints.
Chapter 5 proposes the *CSM* approach for modeling and predicting the energy behavior of discrete systems, i.e. systems where the energy consumption is mainly due to the on/off switching of the actuators governed by the control logic.
Chapter 6 draws some conclusions and hints for future works.
Finally, in the Appendices, details regarding the the transport line are presented, in particular the *MLD* model is discussed in the Appendix A while in the Appendix B and Appendix C the functionalities of the transport line Low Level Control System (*LLCS*) are drown.

1.6 List of publications

The research activity developed during over the years of PhD studies and reported in this thesis has lead to the following publications:

International journals

(IJ1) Cataldo A., Perizzato A., Scattolini R. *Production scheduling of parallel machines with model predictive control.* Control Engineering Practice. Volume 42, September 2015, pp. 28-40.

(IJ2) Cataldo A., Scattolini R., Tolio T. *An energy consumption evaluation methodology for a manufacturing plant.* CIRP Journal of Manufacturing Science and Technology. Vol 11, November 2015, pp. 53-61.

(IJ3) Cataldo A., Scattolini R. *Dynamic pallet routing in a manufacturing transport line with Model Predictive Control.* IEEE Transactions on

Control Systems Technology, September 2016, Vol 24, Issue 5, pp. 1812-1819, ISSN 1063-6536.

National journals

(NJ1) Cataldo A., Scattolini R. *Progettazione del controllo e simulazione di un impianto di de-manufacturing.* ANIPLA, Automazione e strumentazione. N. 8 November-December 2014, pp. 78-83.

(NJ2) Cataldo A., Scattolini R. *Logic control design and discrete event simulation model implementation for a de-manufacturing plant.* Automazione-plus on-line journal, www.automazione-plus.it, November 2014, http://automazione-plus.it/logic-control-design-and-discrete-event-simulation-model-implementation/.

International conferences proceedings

(IC1) Cataldo A., Perizzato A., Scattolini R. *Management of a production cell lubrication system with model predictive control.* Proc. APMS international conference on Advances in production systems, APMS 2014, Ajaccio, France, 20-24 September 2014, Part III, IFIP AICT 440, pp. 131-138, 2014.

(IC2) Cataldo A., Scattolini R. *Modeling and model predictive control of a de-manufacturing plant.* 2014 IEEE Multi-conference on Systems and Control, October 8-10 October 2014, Antibes, France, pp. 1855-1860.

(IC3) Nativ D. J., Cataldo A., Scattolini R., De Schutter B. *Model Predictive Control of an Automated Storage / Retrieval System.* 8^{th} IFAC Conference on Manufacturing Modelling, Management & Control, June 28-30 2016, Troies, France, pp. 1393-1398.

Chapter 2

Dynamic pallet routing in a manufacturing transport line

In the global market, enhancing the efficiency and productivity of manufacturing systems is mandatory to maintain high levels of competitiveness [38]. Among the main issues in this field, the development of optimal routing strategies has an important role for the solution of many significant problems such as lotsizing, scheduling, packing, inventory, and resource allocation, see e.g. [42, 62, 101]. In particular, the efficient routing of the pallets in networks made by machines and transportation lines is required to avoid bottlenecks, starvation, congestion, and to maximize throughputs, see [64] and the papers quoted there. However, the design of optimal routing strategies is difficult due to the combinatorial nature of the problem, and the implemented control laws are often based on heuristic logic rules tuned by means of simulation studies, see [42].
In this scenario, *MPC* has been applied to control the pallet routing in a pilot plant, in particular a de-manufacturing transport line has been considered, in which a multi-pallet, dynamic multi-target problem has to be solved. The plant, designed for the testing, repair, or disruption of electronic boards is composed by a multi-path transport line and by loading/unloading, testing, repair, and discharge machines; its structure and behavior are presented in Section 2.1 and extensively described in [29, 30, 84]. The control system of the plant has been designed according to a multi-level, hierarchical structure [100]. At the higher level, a coordinator has to manage the movement of the pallets along the transportation line in order to optimize the plant performances and to fulfill a set of logical constraints imposed by the transport line structure. At the lower level, Programmable Logic Controllers (*PLC*), one for each transport module, acquire the sensor signals and drive

the actuators. In order to implement the pallet movement through the transport modules according to their structure and to the transport line topology, *LLCS* functionalities need to be specifically designed and implemented, as described in Section 2.2. According to this control architecture, the de-manufacturing plant can be represented as a mathematical model based on a directed graph with nodes and arcs, see Section 2.3 so to be able to apply the *MPC* technique. In this way, in Section 2.4 the dynamic model of the system is derived together with the constraints which must be considered to properly represent its physical limitations. Then the *MPC* control problem can be formulated, see Section 2.5, by firstly presenting the adopted *MLD* formulation of the plant model [12,49], see Appendix A, and then by defining the performance index used to state the optimization problem to be solved on-line according to a Receding Horizon *RH* strategy. The *MPC* algorithm has been implemented in the Dynamic Control Platform for Industrial Plants (*DCPIP*), a custom $C++$ control platform, which has allowed to implement the interface of the *MPC* controller developed in MATLAB. First, a discrete event simulation model of the de-manufacturing plant developed in SIMIO [58] has been used to test the software implementation, see [23], then the algorithm has been used to control the real system.
Many simulation and real experiments have been performed to assess the properties of the method and the experimental results clearly show the very satisfactory behavior of the proposed algorithm when applied both to the discrete event simulation model and to the real system.
Since the real de-manufacturing plant is still on set-up phase, only four pallets have been available for the experimental tests and the real machining operations of the machines $M_1 - M_4$ have been implemented in terms of plain delays, see Section 2.6. In the real experiments, all the tuning parameters of the *MPC* algorithm have been set equal to those used in the simulation experiment.
To evaluate the computational burden of the *MPC* algorithm, simulations have been carried on with different values of prediction horizon N_{RH} and with a different number of pallets on the transport line. The simulation experiments show that the computational burden of the algorithm is largely acceptable for the considered case (five pallets with $N_{RH} = 4$), with also the possibility to use more pallets and a larger N_{RH}. Obviously, as the number of pallets and/or the length of the prediction horizon increase, the computational issue can become relevant.
However, this problem could be partially solved by resorting to the distributed optimization methods already described e.g. in [10, 55].
Future research activity could be aimed at customizing the algorithm to

production lines with operating machines whose working function settings can be dynamically changed in order to further optimize the production line efficiency.

Nomenclature

M_1	load/unload robot cell
M_2	testing machine
M_3	reworking machine
M_4	discharge machine
T_n	$n-th$ transport module
$\Upsilon_{i,j}$	$j-th$ position of the $i-th$ transport module
S_n	$n-th$ control sequence
N_i	$i-th$ node
$I_{i,in}$	set of indices j for the commands $u_{j,i}$ which move a pallet from N_j to N_i
$I_{i,out}$	set of indices j for the commands $u_{i,j}$ which move a pallet from N_i to N_j
I_u	$= \bigcup_{i=1}^{35} I_{i,in} = \bigcup_{i=1}^{35} I_{i,out}$ set of pair of indices (i,j) associated with all the commands $u_{i,j}$
Ψ	set of indices (m,h,i,j) associated to specific nodes neighbors of M_1 - M_4
$u_{i,j}$	control input which moves a pallet from N_i to N_j
k	discrete-time index
$\Gamma_i(k)$	target state of the pallet in node N_i
$\bar{\Gamma}_i$	target associated to each M_i
$\gamma_i(k)$	minimal distance of the pallet to its final destination M_i
$\phi_{i,l}$	minimum distance from the node N_i to the $l-th$ target M_l
$\zeta_{i,s}(k)$	auxiliary binary variable for the γ_i function linearization
$\eta_i(k)$	counter of the time permanence of the pallet in N_i on the transport line
$\delta_i(k)$	auxiliary binary variable used for the counter associated with the pallet in N_i
$\vartheta_i(k)$	auxiliary binary variable for the identification of a pallet not free in N_i
x_{i1}	Extended Finite Automata (EFA) binary state related to the M_i idle state
x_{i2}	EFA binary state related to the M_i working state
x_{i3}	EFA binary state related to the M_i waiting state
n_i	time counter associated with M_i
$\bar{n}_i$	time parameter associated with the M_i machining operation
$\delta_{i,23}$	auxiliary binary variable for the state transitions of M_i
$x(k)$	state variable of the MLD system
$u(k)$	control actions of the MLD system
$\delta(k)$	auxiliary binary variables of the MLD system

$z(k)$	auxiliary continuous variables of the *MLD* system
J	total performance index
J_1	primary performance index term of the lexicographic optimization
J_2	secondary performance index term of the lexicographic optimization
N_{RH}	prediction horizon
q_{xi}	x_{i3} weight in the performance index
q_{Ni}	η_i weight in the performance index
σ_m	auxiliary binary variable for the off-limit zone performance index penalty
λ_m	weight on the off-limit zone in the performance index
$q_{ui,j}$	weight on the control action in the performance index

2.1 The plant layout

The de-manufacturing pilot plant is located in the laboratory of *ITIA – CNR*, Italy, see Figure 2.1.

Figure 2.1: The de-manufacturing plant.

The plant has been designed for the testing, repair, or discharge of electronic boards. Its scheme is sketched in Figure 2.2, and its main components are:

- machine M_1, load/unload robot cell: the electronic board is either loaded on a pallet, which is then placed on the adjacent transport module of the transport line, or unloaded from the pallet;
- machine M_2, testing machine: the board is tested and its failure mode is identified;
- machine M_3, reworking machine: the board is machined in order to be repaired;
- machine M_4, discharge machine: the non-repairable boards are discharged from the pallet and destroyed;

- fifteen transport modules T_n, $n = 1, \ldots, 15$, connected together according to a specific meshed configuration, and composing a modular and flexible transport line (the detailed description of these transport modules is reported in the following).

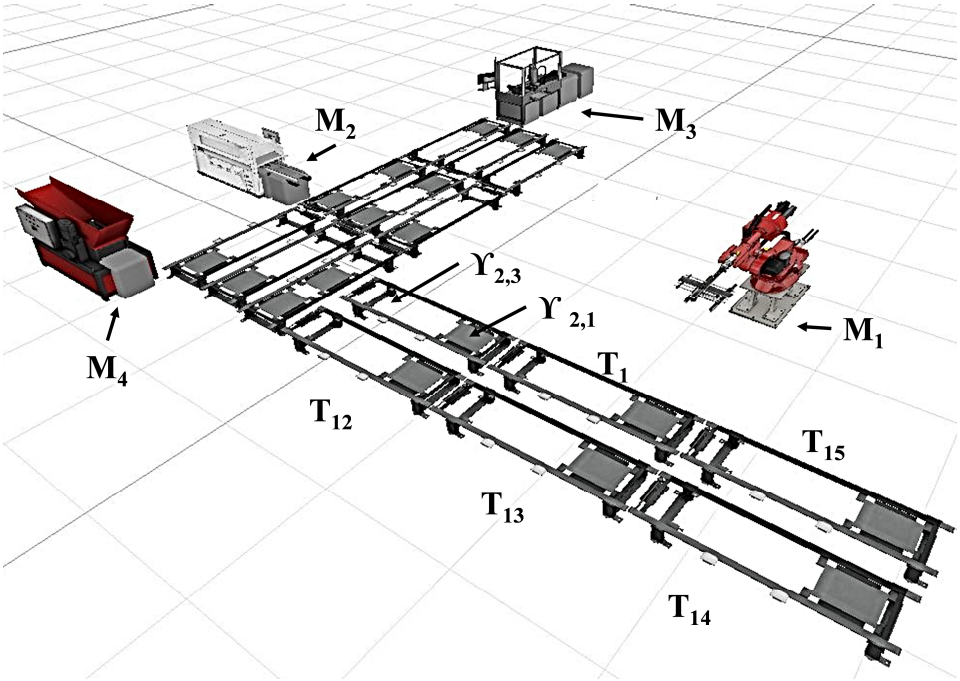


Figure 2.2: The de-manufacturing plant structure.

The sequence of operations to be performed on each board is the following:

- the board is loaded on the pallet by M_1;
- the transport line moves the pallet to M_2 where the board is tested and its failure mode is identified;
- the pallet with the board is moved to M_3 where the failure is repaired, if possible;
- the pallet is moved back to M_2 and the test is repeated. If the board is properly working, it is sent back to M_1 where it is unloaded from the pallet and stored in the warehouse, otherwise it is sent to M_4 where it is discharged from the pallet and destroyed. Then, the pallet is ready to load a new board to be tested and must be sent to M_1.

In this work, attention has been focused on the management and control of the transport line, with the goal to compute, at any time instant, the routing strategy to move the pallets from one machine ($M_1 - M_4$) to the other, so as to optimize the overall system's behavior. For this reason, it is mandatory to describe the modules of the transport line with some detail.
The basic element used to move a pallet is the transport module. It allows the pallet laying in specific zones of it or moving towards the possible directions which are imposed by the transport line topology. Each transport module has a specific configuration in terms of mechanical structure and automation system instrumentation. In order to deeply explain the transport module working function, the T_8, one of the most flexible transport module in the de-manufacturing transport line, has been taken into account as example. In order to move a pallet, a transport module can use up to four different kind of devices with regarding sensors and actuators, see Figure 2.3, in which all the possible pallet movements are also indicated by means of continuous arrows. They are:

(*a*) the main track which is used to move forward or backward (M_Tr_F/B) the pallet along the main direction of the transport module;

(*b*) the block pallet Piston (Ev_P) which is used to stop the pallet during its movement onto the main track;

(*c*) the stacker crane which is used to laterally move the pallet by means of the up and down commands ($Ev\_Sc1/2\_U/D$) and the regarding left and right motors activation ($M\_Sc1/2\_Tr\_L/R$);

(*d*) the setback which allows to stop the pallet in the right position during its lateral movement, by means of the up and down commands ($Ev\_SL/R\_U/D$).

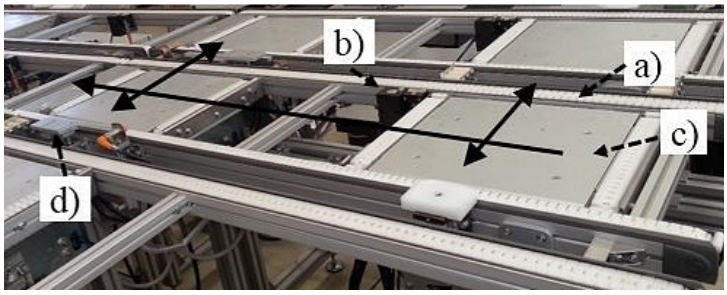


Figure 2.3: The transport module T_8 configuration.

The deeply analysis of the transport line mechanical structure, has allowed to deduce that a pallet can lay or move to/from different transport module T_i areas; according to its specific configuration, up to three pallets can lay in three adjacent positions, called Buffer Zones (*BZ*) so that in the sequel $\Upsilon_{i,j}$ will denote the $j-th$ position ($j = 1,2,3$) of the $i-th$ transport module ($i = 1,\ldots,15$). The actual number of *BZ* available on each transport module depends on its specific mechanical configuration, this means that there are transport modules with only one *BZ*, others with two *BZ* or three *BZ*, see Appendix B.1.
The pallet placed in a *BZ* can be moved forward, in some cases backward, or to the lateral positions, so allowing for different possible paths along the transport line. In Figure 2.4 a sketched representation of the T_8 has been given in which the three *BZ* $\Upsilon_{8,1}$, $\Upsilon_{8,2}$ and $\Upsilon_{8,3}$ have been indicated.

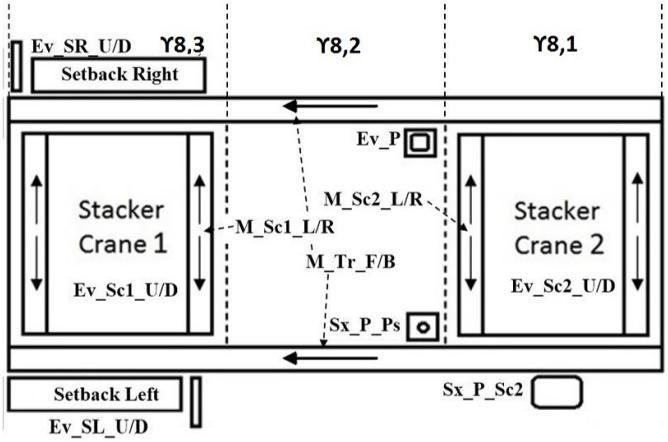


Figure 2.4: Schematic representation of the transport module actuators.

2.2 The plant low level control system

The control system of the plant has been designed according to a multi-level, hierarchical structure [100]. At the higher level, a coordinator has to manage the movement of the pallets along the transportation line in order to optimize the plant performances and to fulfill a set of logical constraints imposed by the transport line structure. At the lower level, a set of *PLC*, one for each transport module, acquire the sensor signals and drive the actuators. In order to implement the pallet movement through the different *BZ*

according to the transport line topology and the transport modules structure, *LLCS* functionalities need to be specifically designed and implemented, see Appendix B.3.
The *BZ* definition and the regarding actuators and sensors involved into the pallet movement have led to identify a set of thirty-six control sequences able to perform all possible pallet movements on the transport line, see Table C.1 in Appendix C. This means that it has not been necessary to develop specific control software for each transport module *PLC* but just only download those control sequences necessary to implement the pallet movement for the specific transport module configuration.
As an example, in Figure 2.5 the transport module T_8 with three *BZ* is sketched, together with the possible movements of the pallet, named control sequences S_n, $n = 1,\ldots,36$, and represented by arrows, which are implemented by a specific low level logic control system [23]. As shown in Figure 2.5, the pallet placed in a *BZ* can be moved forward, or in some cases to the lateral positions, so allowing for different possible paths along the transport line. In Appendix B.2 the different control sequences applied to the different T_i are showed.

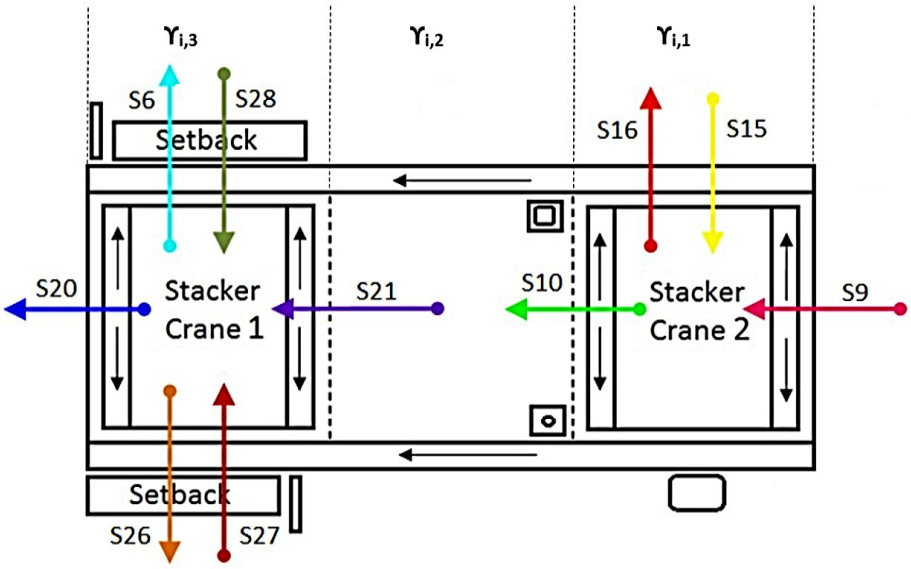


Figure 2.5: Schematic representation of the transport module.

The selection of the proper path for each pallet on the transport line is exactly the control problem to be dealt with.
For example S_{21} is used to move the pallet from $\Upsilon_{8,2}$ to $\Upsilon_{8,3}$ while S_{28} is activated to laterally move the pallet from the adjacent transport module to $\Upsilon_{8,3}$.
Some control sequences are similar in terms of implemented pallet movement implementation but they are different regarding the devices or sensors or actuators involved. Then both S_{15} and S_{28} are used to laterally move a pallet from an adjacent transport module to T_8 but the control sequence S_{15} do not have to manage any setback while the control sequence S_{28} has to control the setback right.
The *LLCS* design approach allows in general to download a specific control sequences into more *PLC*. This possibility allows to reuse the control sequences by simplifying the design, the development and the maintenance of the plant control software. In order to highlight such concept, in Table C.2, Appendix C, the sets of control sequences related to each transport module are listed while in Table C.3, Appendix C, the transport modules which uses a specific control sequences are grouped.
By representing the *BZ* and the control sequences related to all the fifteen transport modules, it is possible to represent the whole transport line as showed in Figure 2.6. From the schema it is possible to deduce how to combine the control sequences in order to obtain the pallet movements. For example in case a pallet lay in $\Upsilon_{15,3}$ and has to move to $\Upsilon_{1,1}$ then S_3 must be applied to T_{15} and, at the same time, S_{19} must be activated to T_1.

2.3 The plant LLCS implications

The *BZ* and control sequences transport line model showed in Figure 2.6 is strictly dependent from the specific *LLCS* design approach that means that, in case of any control sequence redefinition, then the transport line representation will change and then the High Level Control System (*HLCS*) design too. Moreover, from the *HLCS* point of view, it is useless to know what control sequences has to be actuated in order to move a pallet; on the contrary it is necessary to know what are the two *BZ* involved in the pallet movement. In this way it is possible to define specific variables uniquely associated to the pallet movement between two specific *BZ*. In practice these defined variables could be considered the control actions of the *HLCS* which have to be sent to the *LLCS* in order to manage the field plant.

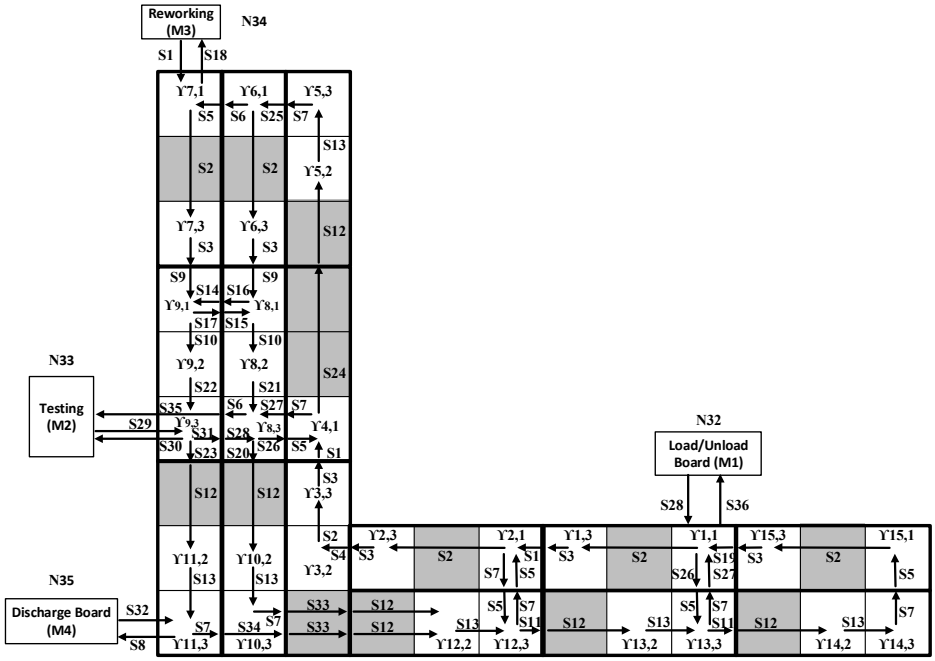


Figure 2.6: Buffer Zone and Control Sequence transport line representation.

Hence the system functionality related to the implementation of the control actions moving the pallet from a *BZ* to an adjacent one allows to represent the transport line as a directed graph, see Figure 2.7, where the nodes represent the *BZ* (circles) and the machines M_i (rectangles) where the pallet can lay.

For convenience, these nodes are progressively numbered and labeled $N_1 - N_{35}$. Specifically the *BZ* are labelled $N_1 - N_{31}$ while the machines $M_1 - M_4$ are labelled $N_{32} - N_{35}$. This distinction is necessary since the *BZ* and the machines have some different functionalities with respect to the pallet management, so that they will be characterized by slightly different dynamic models. For this reason, the *BZ* and the machines are firstly defined in terms of generic nodes to describe the common parts of their dynamics; then their model is detailed to describe their different behavior. Associated with the arcs in the graph representation there are the variables $u_{i,j},\ i,j = 1,\ldots,35$, which are the available binary commands allowing to move the pallet from node N_i to node N_j. Note that many of these commands do not exist because of the specific transport line topology (and the corresponding $u_{i,j}$ can be formally set to zero). In practice, only fifty commands (also shown in Figure 2.7) are available.

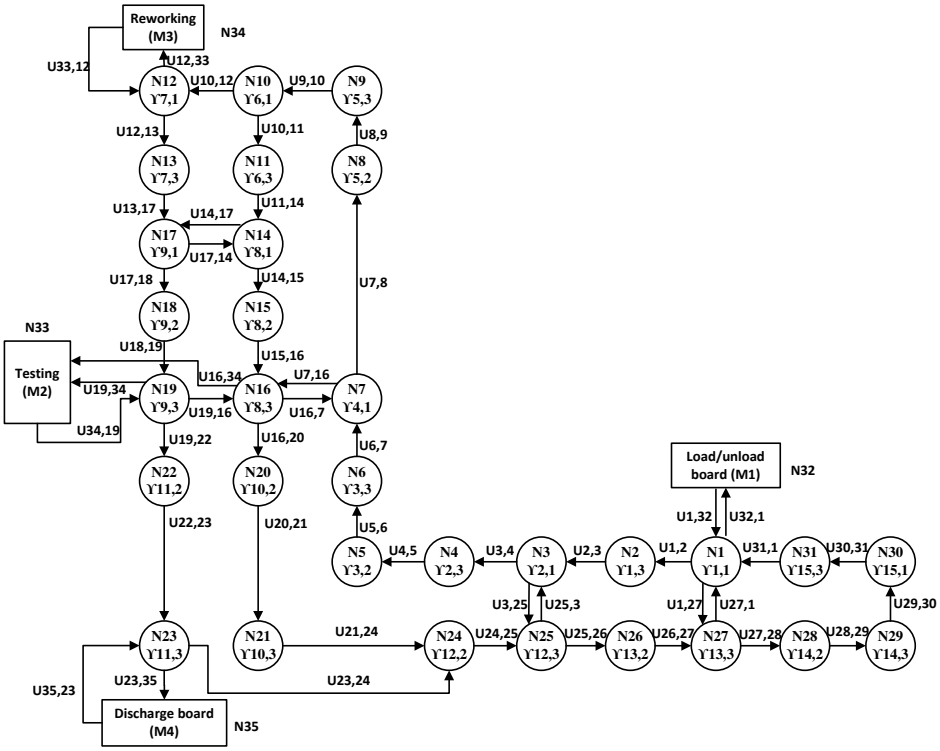


Figure 2.7: Direct graph representation of the plant.

2.4 The dynamic model

2.4.1 Model of the nodes

A fundamental information associated with the pallet concerns the destination of the loaded board, called Target, which is represented by an integer number. To recover and propagate this information, letting k be the discrete-event index, define by $\Gamma_i(k)$ the Target state of the pallet loaded in node N_i, $i = 1,\ldots,35$, which can take the following values:

- $\Gamma_i(k) = 0$ if the *BZ* or the machine corresponding to node N_i is empty at k;
- $\Gamma_i(k) = j,\ j = 1,2,3,4$ if the *BZ* or the machine corresponding to the

node N_i contains a board to be sent respectively to the machine M_1, M_2, M_3, M_4 (or, equivalently to the node $N_{32}, N_{33}, N_{34}, N_{35}$);

- $\Gamma_i(k) = 5$ if the *BZ* or the machine corresponding to node N_i contains a pallet without any Target to be reached (this happens when a pallet exits M_4 until the de-manufacturing system sets $\Gamma_i = 1$ so as to allow loading a new board on the pallet).

For not structurally null commands $u_{i,j}$'s, let

- $u_{i,j}(k) = \begin{cases} 0 & \text{if the command is not active at } k \\ 1 & \text{if the command is active at } k \end{cases}$
- $I_{i,in}$ be the set of indices j associated with the commands $u_{j,i}$ not structurally null which allow to move a pallet to the node N_i from an adjacent node N_j;
- $I_{i,out}$ be the set of indices j associated with the commands $u_{i,j}$ not structurally null which allow to move a pallet from the node N_i to an adjacent node N_j;
- $I_u = \bigcup_{i=1}^{35} I_{i,in} = \bigcup_{i=1}^{35} I_{i,out}$ be the set of pair of indices (i, j) associated with all the commands $u_{i,j}$ not structurally null.

Some constraints must be considered to guarantee the feasibility of the model. First, for each node N_i and at any k, only one control input and/or output can be allowed

$$\sum_{j \in I_{i,in}} u_{j,i}(k) \leq 1, \qquad i = 1, \ldots, 35 \tag{2.1}$$

$$\sum_{j \in I_{i,out}} u_{i,j}(k) \leq 1, \qquad i = 1, \ldots, 35 \tag{2.2}$$

If a node does not contain a pallet, no commands of output can be given. Denoting by $\rightarrow$ the implication operator, this can be expressed by

$$\Gamma_i(k) = 0 \rightarrow \sum_{j \in I_{i,out}} u_{i,j}(k) = 0, \qquad i = 1, \ldots, 35 \tag{2.3}$$

2.4.2 Model of the Buffer Zones

The specific dynamic equation for the generic *BZ* which describes the pallet movement and the Target propagation over the transport line is

$$\Gamma_i(k+1) = \Gamma_i(k) + \sum_{j \in I_{i,in}} \Gamma_j(k) u_{j,i}(k) - \sum_{j \in I_{i,out}} \Gamma_i(k) u_{i,j}(k), \quad i = 1, \ldots, 31 \tag{2.4}$$

Conditions guaranteeing that a node contains at most one pallet must also be considered. In fact, if the node N_i contains a pallet, it is possible to activate a control action $u_{j,i} \in I_{i,in}$ which moves another pallet into N_i only if, at the same time, a control action $u_{i,j} \in I_{i,out}$ is activated, so as to move out of N_i the loaded pallet. This constraint can be stated by imposing

$$\Gamma_i(k) > 0 \wedge \sum_{j \in I_{i,out}} u_{i,j}(k) = 0 \rightarrow \sum_{j \in I_{i,in}} u_{j,i}(k) = 0, \quad i = 1, \ldots, 31 \tag{2.5}$$

Associated with the $i-th$ node it is also possible to define the minimal distance $\gamma_i(\Gamma_i(k))$ of the corresponding pallet (if any) to its final destination represented by one of the machines M_i, $i = 1, \ldots, 4$. In view of this definition, $\gamma_i(\Gamma_i(k)) = 0$ if $\Gamma_i(k) = 0$ or $\Gamma_i(k) = 5$, i.e. if the node does not contain any pallet or the pallet is empty. In all the other cases, $\gamma_i(\Gamma_i(k))$ is equal to the length of the minimal path from node i to the destination node of the pallet specified by the actual value of $\Gamma_i(k)$. More formally, for any node i let $\zeta_{i,s}$, $s = 0, ..., 5$, be binary variables with $\zeta_{i,s} = 1$ if the index s is associated with the target of the pallet ($s = 0$ for empty nodes, $s = 1, 2, 3, 4$ for M_1, M_2, M_3, M_4, and $s = 5$ for empty pallets), while $\zeta_{i,s} = 0$ otherwise. Moreover, define by $\phi_{i,j}$ the minimum distance between the node i and the target machine in node j ($j = 0, 32, 33, 34, 35$), with $\phi_{i,0} = 0$ for any i. Then, it is possible to write

$$\begin{aligned} \gamma_i(\Gamma_i(k)) = {} & \zeta_{i,1}(k)\phi_{i,32} + \zeta_{i,2}(k)\phi_{i,33} + \zeta_{i,3}(k)\phi_{i,34} + \\ & + \zeta_{i,4}(k)\phi_{i,35} + (\zeta_{i,0}(k) + \zeta_{i,5}(k))\phi_{i,0}, \quad i = 1, \ldots, 35 \end{aligned} \tag{2.6}$$

In the design of the control system, it is also worth penalizing the permanence of the pallet on the transport line, so as to force its movement towards the target machine and avoid deadlocks. To this end, a counter η_i, $(i = 1, \ldots, 31)$ for each BZ is defined, and at each time instant it's value is set equal to the number of instants in which the corresponding pallet has been on the line. When the pallet enters a machine, the counter is reset, since its Target has been reached. The dynamic model of the counter is obtained by first defining for each node N_i the boolean variables $\delta_i(k)$, which is set equal to one if there are no inputs and outputs from/to node i, and

$\vartheta_i(k)$ which is set equal to one if N_i contains a pallet with a board to be sent to one of the machines M_1, M_2, M_3, M_4. Specifically,

$$\delta_i(k) = 1 \leftrightarrow \sum_{j \in I_{i,in}} u_{j,i}(k) = 0 \wedge \sum_{j \in I_{i,out}} u_{i,j}(k) = 0, \qquad i = 1, \ldots, 31 \tag{2.7}$$

and

$$\vartheta_i(k) = 1 \leftrightarrow (\Gamma_i(k) \geq 1 \wedge \Gamma_i(k) \leq 4), \qquad i = 1, \ldots, 31 \tag{2.8}$$

Then, the dynamic equation of the counter for the $i-th$ BZ is given by the equation

$$\eta_i(k+1) = \eta_i(k) + \delta_i(k)\vartheta_i(k) + \sum_{j \in I_{i,in}} [\eta_j(k)+1]\vartheta_j(k)u_{j,i}(k) + - \sum_{j \in I_{i,out}} \eta_i(k)\vartheta_i(k)u_{i,j}(k), \qquad i = 1, \ldots, 31 \tag{2.9}$$

2.4.3 Model of the machines

A specific model must be developed for the four machines M_i, $i = 1, \ldots, 4$ (nodes $N_{32} - N_{35}$). In particular, the generic M_i is described by a *EFA* [80, 105], see Figure 2.8, with the following three Boolean states: x_{i1} (idle, empty machine); x_{i2} (manufacturing); x_{i3} (end manufacturing with pallet still loaded).

The transitions among the states of the *EFA* are governed by the following

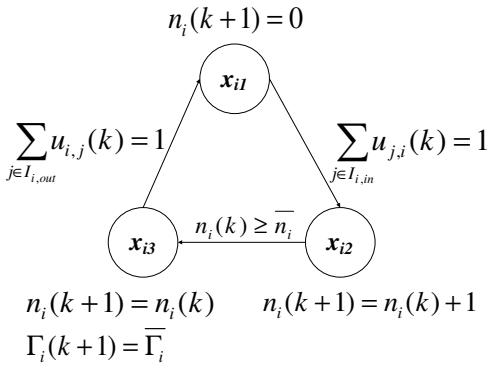


Figure 2.8: EFA model of the machines.

conditions:

- in x_{i1}, the machine is at idle, and a counter n_i is set to zero. If a pallet is in the *BZ* (N_j) adjacent to M_i and the control action $u_{j,i}$ is activated, then the *EFA* state switches from x_{i1} to x_{i2};
- when the state x_{i2} becomes active, n_i is increased at every step to count the progress of the manufacturing process. When this counter reaches a given threshold $\bar{n}_i$, which corresponds to the end of the working phase, the *EFA* state switches from x_{i2} to x_{i3};
- in x_{i3} the counter is kept constant at the maximum reached value $\bar{n}_i$, and a new target value $\bar{\Gamma}_i$ is assigned to the pallet. In particular $\bar{\Gamma}_{32} = 2, \bar{\Gamma}_{33} = 3$ or 4 or 5, $\bar{\Gamma}_{34} = 2$ and $\bar{\Gamma}_{35} = 0$. The values of $\bar{n}_i$ for the four machines are: $\bar{n}_{32} = 11, \bar{n}_{33} = 10, \bar{n}_{34} = 11, \bar{n}_{35} = 9$;
- as soon as the control action $u_{i,j}$ in $I_{i,out}$ is activated, the pallet is moved from the machine to the adjacent *BZ* of the transport line, the *EFA* state switches from x_{i3} to x_{i1}, and the counter $n_i(k)$ is reset.

In order to model this dynamic behavior, the following implications must be set

- switch from x_{i1} to x_{i2}:

$$x_{i1}(k) \wedge \left(\sum_{j \in I_{i,in}} u_{j,i}(k) = 1 \right) \rightarrow \begin{cases} x_{i1}(k+1) = 0 \\ x_{i2}(k+1) = 1 \end{cases} \tag{2.10}$$

- switch from x_{i2} to x_{i3}:

$$x_{i2}(k) \wedge (n_i(k) \geq \bar{n}_i) \rightarrow \begin{cases} x_{i2}(k+1) = 0 \\ x_{i3}(k+1) = 1 \end{cases} \tag{2.11}$$

- switch from x_{i3} to x_{i1}:

$$x_{i3}(k) \wedge \left(\sum_{j \in I_{i,out}} u_{i,j}(k) = 1 \right) \rightarrow \begin{cases} x_{i3}(k+1) = 0 \\ x_{i1}(k+1) = 1 \end{cases} \tag{2.12}$$

Moreover, the following additional constraints must be imposed to properly represent the operations of the machines.
Since a pallet cannot enter the machine if it is processing or still loaded, i.e.

if the state x_{i1} is non active, the control actions $u_{j,i}$ in $I_{i,in}$ must be deactivated

$$x_{i1}(k) = 0 \rightarrow \sum_{j \in I_{i,in}} u_{j,i}(k) = 0, \qquad i = 32, \ldots, 35 \tag{2.13}$$

Since a pallet cannot exit the machine if it is at idle or manufacturing, i.e. if the state x_{i3} is non active, the control actions $u_{i,j}$ in $I_{i,out}$ must be deactivated

$$x_{i3}(k) = 0 \rightarrow \sum_{j \in I_{i,out}} u_{i,j}(k) = 0, \qquad i = 32, \ldots, 35 \tag{2.14}$$

Finally, letting $\delta_{i,23}(k)$ be a Boolean variable which represents the logic condition associated to the transition from x_{i2} to x_{i3}

$$x_{i2}(k) \wedge (n_i(k) \geq \bar{n}_i) \leftrightarrow \delta_{i,23}(k) \tag{2.15}$$

the dynamic equations related to $\Gamma_i(k)$ and to the counter $n_i(k)$ associated with M_i, $i = 32, \ldots, 35$, are

$$\begin{aligned} \Gamma_i(k+1) = \Gamma_i(k) + \sum_{j \in I_{i,in}} \Gamma_j(k) u_{j,i}(k) + \\ - \sum_{j \in I_{i,out}} \Gamma_i(k) u_{i,j}(k) + \delta_{i,23}(k)[\bar{\Gamma}_i - \Gamma_i(k)] \end{aligned} \tag{2.16}$$

$$n_i(k+1) = [n_i(k) + x_{i2}(k)][1 - x_{i1}(k)] \tag{2.17}$$

2.5 MPC formulation

The dynamic model of the transport line has been translated into the *MLD* formulation by linearizing the non-linear terms and the logic propositions [12], see for details the Appendix A with the derivation of the *MLD* model. Then the HYSDEL tool [114] has been used to generate the *MLD* model described by

$$x(k+1) = Ax(k) + B_u u(k) + B_\delta \delta(k) + B_z z(k) \tag{2.18a}$$

$$y(k) = Cx(k) + D_u u(k) + D_\delta \delta(k) + D_z z(k) \tag{2.18b}$$

$$E_\delta \delta(k) + E_z z(k) \leq E_u u(k) E_x x(k) + E \tag{2.18c}$$

where x is the vector of the state variables, u is the vector of the control actions, with elements $u_{i,j}$, δ is the vector of Boolean auxiliary variables, and z is a vector of continuous auxiliary variables.
For system (2.18a)-(2.18c), the optimization problem consists of minimizing a linear performance index J with respect to the future control actions defined over the prediction horizon specified by the positive integer N_{RH}, so leading to a Mixed Integer Linear Programming (*MILP*) system. In J the following terms are weighted (see (2.20)):

(*a*) the distance of the pallets from their target machines in terms of number of steps to be performed in order to get the specific target machine, see for example Figure 2.9 which shows distance of a pallet placed in the N_1 from the different target machines;

(*b*) the permanence of a manufactured pallets into the states x_{3i}, $i = 1,...,4$ of the machines;

(*c*) the counters η_i associated with the permanence of the pallets on the transport line;

(*d*) the control actions;

(*e*) the permanence of a pallet in the nodes adjacent to $M_1,...,M_4$, so as to allow the manufactured pallets to exit the machines and move towards their new target.

Concerning the term (e), it is worth defining the following boolean variables

$$\sigma_{32}(k) = 1 \leftrightarrow (\Gamma_1(k) = 1 \vee \Gamma_{32}(k) = 5 \vee \vartheta_{32}(k) = 1) \tag{2.19a}$$
$$\sigma_{33}(k) = 1 \leftrightarrow (\Gamma_{19}(k) = 2 \vee \vartheta_{33}(k) = 1) \tag{2.19b}$$
$$\sigma_{34}(k) = 1 \leftrightarrow (\Gamma_{12}(k) = 3 \vee \vartheta_{34}(k) = 1) \tag{2.19c}$$
$$\sigma_{35}(k) = 1 \leftrightarrow (\Gamma_{23}(k) = 4 \vee \Gamma_{35}(k) = 5 \vee \vartheta_{35}(k) = 1) \tag{2.19d}$$

where, according to the previous definitions, $\Gamma_{32}(k) = 5$ and $\Gamma_{35}(k) = 5$ represent the presence of pallets without any Target to be reached, in M_1 and M_4 respectively.

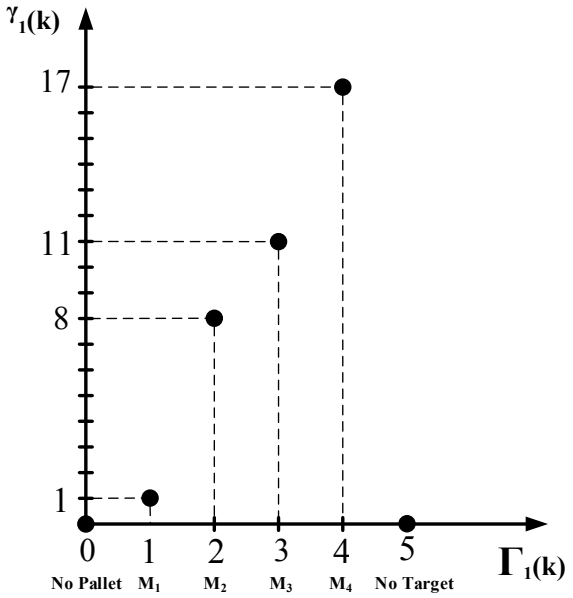


Figure 2.9: Distance of a pallet placed in the node N_1 from the different target machines.

Then, the adopted performance index J is given by

$$J = \sum_{h=1}^{N_{RH}} \Bigg\{ \underbrace{\sum_{i=1}^{35} \gamma_i(\Gamma_i(k+h))}_{(a)} + \underbrace{\sum_{i=32}^{35} q_{xi} x_{i3}(k+h)}_{(b)} + \underbrace{\sum_{i=1}^{31} q_{\eta_i} \eta_i(k+h)}_{(c)} + \\ + \underbrace{\sum_{(i,j)\in I_u} q_{ui,j} u_{i,j}(k+h-1)}_{(d)} + \\ + \underbrace{\sum_{(m,r,i,j)\in\Psi} \lambda_{m,r} \sigma_m(k+h-1) u_{i,j}(k+h-1)}_{(e)} \Bigg\} \tag{2.20}$$

with

$$\Psi(m,r,i,j) = \{(32,1,27,1),(32,2,31,1),(33,1,7,16),(33,2,15,16), \\ (33,3,18,19),(34,1,10,12),(35,1,22,23)\}.$$

Some guidelines for the selection of the weights q_{xi}, q_{η_i}, $q_{ui,j}$, $\lambda_{m,r}$ can be given. In particular, large values of q_{xi} are recommended to force the pal-

lets to exit from the corresponding machines M_i, while quite small positive values of the weights q_{η_i} can be used to include in the cost function the integral effect on the permanence of the pallets on the transport line (incidentally, this term has been proven to be very useful to avoid deadlocks and to allow for the use short prediction horizons). As for the control weights $q_{ui,j}$, even very small values can be used, since the goal of the term (d) in J is just to avoid useless commands to the actuators. Finally, the values of the coefficients $\lambda_{32,1}$, $\lambda_{32,2}$, $\lambda_{33,1}$, $\lambda_{33,2}$, $\lambda_{33,3}$, $\lambda_{34,1}$ and $\lambda_{35,1}$, have to be large enough to penalize the presence of a pallet in the nodes adjacent to $M_1,\ldots,M_4$. These general rules of thumb must then be refined by means of a proper tuning in simulation (or real) experiments.
The prediction horizon N_{RH} must be selected large enough to avoid possible deadlocks due to conflicting paths of the pallets. In particular the most critical deadlock situation could happen if in the node N_{19} there is a pallet with target M_3 and in the node N_{16} there is a pallet with target M_2. In fact, in order to remove the deadlock, the pallet in the node N_{16} should move to the node N_7 or N_{20} or the pallet in the node N_{19} should move to the node N_{22}; in any case these decisions require N_{RH} to be large enough to obtain a performance index less then the one obtained with a smaller N_{RH} and the two pallet locked in the nodes N_{16} and N_{19}. On the other hand, the introduction of the integral effect in the performance index, as mentioned above, allows removing the deadlock with a smaller N_{RH} that makes faster the control algorithm computation. Moreover, N_{RH} should not exceed the minimum number of steps $\bar{n}_i$ required by the machines M_i to work the pallets (in this case $N_{RH} < \bar{n_{35}} = 9$), otherwise the optimization problem would not activate the command to load the machines due to the high penalty on their states x_3.
The performance index (2.20) must be minimized under the physical and logical constraints described by the *MLD* model (2.18a)-(2.18c). Once the sequence of optimal controls $u_{i,j}(k+h-1), h = 1,\ldots,N_{RH}$ has been computed, according to the receding horizon approach, only the first value $u_{i,j}(k)$ is applied and the overall procedure is repeated at the next step.

2.6 MPC validation

The properties of the algorithm have been analyzed in many simulation and experimental tests. In the following, a long simulation experiment is first described in detail to thoroughly study the performance of the controlled

system. Then, an experiment performed with the real system is reported and analyzed.

2.6.1 Simulation results

The *MLD* model and the control algorithm have been implemented in MATLAB by means of the YALMIP modeling language [74]. Preliminary experiments, here not reported for brevity, have been performed to validate the model. Then, many simulation experiments have been run for the fine tuning of the weights of the performance index, which have been finally chosen as follows: $q_{ui,j} = 0.02$, $q_{\eta i} = 1$, $q_{xi} = 10000$, and $\lambda_{32,1} = \lambda_{32,2} = \lambda_{33,2} = \lambda_{33,3} = \lambda_{34,1} = \lambda_{35,1} = 10$, while it has been set $\lambda_{33,1} = 4$ to facilitate the violation of the zone near M_2 and to allow for a free passage to M_3. In the experiment reported below, the prediction horizon has been chosen as $N_{RH} = 4$ and five pallets have been assumed to be loaded on the transport line at $k = 0$. Specifically, their initial states and targets are:

pallet 1 (green) in N_{16} with Target 2;
pallet 2 (yellow) in N_{19} with Target 3;
pallet 3 (blue) in N_5 with Target 3;
pallet 4 (red) in N_6 with Target 3;
pallet 5 (grey) in N_7 with Target 2.

The simulation results are presented in terms of pallet paths, see Figure 2.10. For each pallet, the following comments are in order.

- *pallet 1*: it follows the shortest path to reach its targets M_2, M_1, and M_2 again (still not reached at the end of the reported simulation;
- *pallet 2*: it moves from its initial position to the target M_3 along a path much longer than the shortest one. In fact, it is initially moved to node N_{22}, instead of to node N_{16}, in order to leave the free passage to the pallet 1. In the interval $k \in \{12,\dots,39\}$ it stays in N_9, and in the interval $k \in \{40,\dots,71\}$ it stays in N_{10} to allow the pallets 3 and 4 to reach their target M_3, complete their processing, exit and move to their new targets. This behavior is due to the weights on the permanence of the pallets in the neighbors of the machines (node N_{12} in this case);
- *pallet 3*: it goes straight to its target, save for a stop in N_{10} instead of N_{12} in order to wait that the pallet 4 exits N_{34};

- *pallet 4*: once it has released from machine M_3 (node N_{34}), it moves to its new target M_2 (node N_{33}). Then it stops in N_{18} and, just before pallet 5 exits N_{33}, it moves to N_{16} so as to be ready to enter N_{33};
- *pallet 5*: also this pallet follows a longer path. In fact, instead of staying in N_7 so as to be ready to reach N_{33} through N_{16}, it moves to N_{15} so to allow the pallets 3 and 4 to reach their target N_{34}.

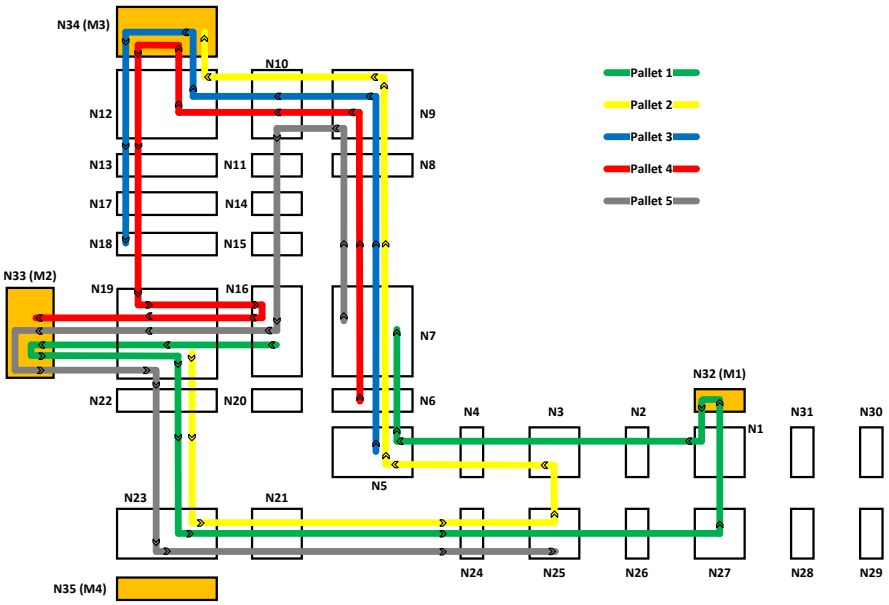


Figure 2.10: Simulation pallet paths.

In order to highlight the ability of the control algorithm to manage the pallet movements avoiding deadlocks, in Figures 2.11 and 2.12 the number of the pallet visiting each node N_i, $i = 32, \ldots, 35$ over a long simulation interval (1600 sampling) are plotted. It is apparent the absence of deadlocks and the relative load of each machine.

To evaluate the computational burden of the *MPC* algorithm, simulations have been carried on with different values of N_{RH} and with a different number of pallets on the transport line. The mean values of the times required for the optimization are listed in Table 2.1[1]. In order to interpret these re-

[1]Simulations carried out on a computer Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.10GHz 2.10 GHz, 16.0 GB Installed Memory (RAM), System type 64-bit Operating System, x64-based processor, Windows 8.1 Pro., MATLAB R13a, YALMIP R20141127, CPLEX R12.4, CPLEX settings: Parallelmode = 0, Threads = 0.

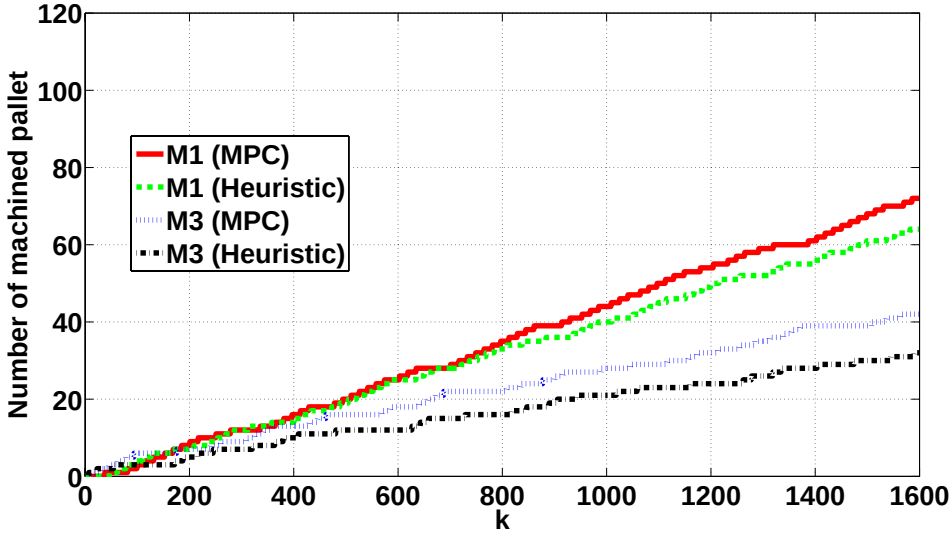


Figure 2.11: Machined pallets by M_1 and M_3.

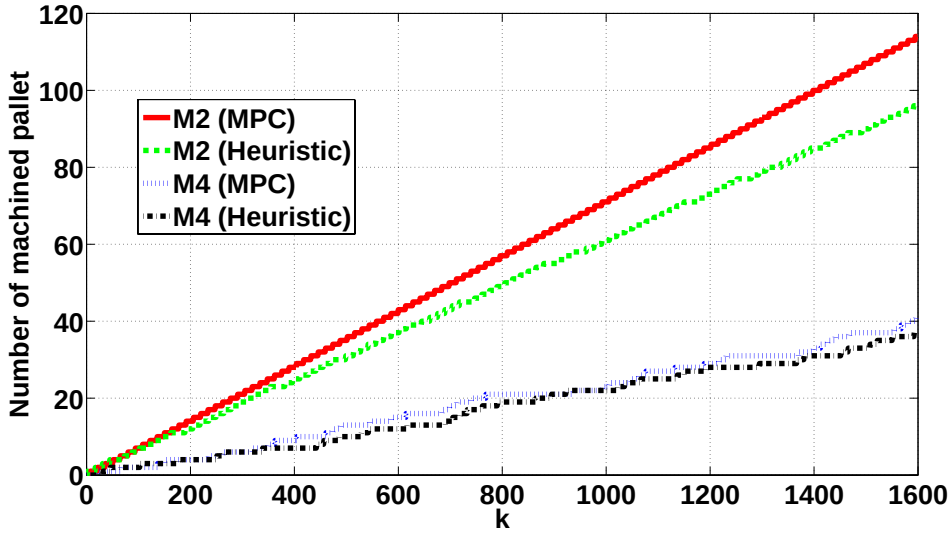


Figure 2.12: Machined pallets by M_2 and M_4.

sults, note that on average the actuation of the pallets requires about $5[s]$ to move them from a *BZ* to an adjacent one. Then, if the on-line optimization does not exceed $10[s]$, the overall time required to compute and actuate the control action is of about $15[s]$. Since the fastest machine is M_4, which completes the discharge of the electronic board in about $45[s]$, 3 pallets movements (or more) are allowed for any cycle of the machines, which is

judged to be a satisfactory value in view of the configuration of the transport line.

N.Pallet	$\mathbf{N_{RH}=4}$	$\mathbf{N_{RH}=5}$	$\mathbf{N_{RH}=6}$	$\mathbf{N_{RH}=7}$
3	0.17	0.33	0.58	1.13
4	0.27	0.50	1.04	2.45
5	0.37	0.87	2.39	7.23
6	0.42	1.26	5.09	19.42
7	0.53	1.64	7.72	37.12
8	0.69	3.09	24.91	>100.00
9	0.85	6.01	>100.00	>100.00
10	1.22	15.13	>100.00	>100.00

Table 2.1: On-line optimization computation time $[s]$.

These considerations, together with the results of Table 2.1, show that the computational burden of the algorithm is largely acceptable for the considered case (five pallets with $N_{RH}=4$), with also the possibility to use more pallets and a larger prediction horizon N_{RH}. Obviously, as the number of pallets and/or the length of the prediction horizon increase, the computational issue can become relevant. However, this problem could be partially solved by resorting to the distributed optimization methods already described in e.g. [10, 55].
Finally, the obtained results have been compared to the performances provided by a strategy based on a set of heuristic rules, as usually done in the process industry. Specifically for the de-manufacturing pilot plant:

- in order to create pallet loop-paths and reduce pallet traffic jam, the commands $u_{1,27}$, $u_{27,28}$, $u_{3,25}$, $u_{7,16}$, $u_{16,7}$, $u_{14,17}$ and $u_{17,14}$ have been disabled;

- in order to avoid deadlocks and to allow the pallets to exit the machines, the commands $u_{10,12}$, $u_{18,19}$, $u_{22,23}$, $u_{27,1}$ are not fired if the machines M_3, M_2, M_4 and M_1, respectively, contain a pallet;

- at node N_3 priority is given to the pallet coming from N_2, and at N_{24} priority is given to the one coming from N_{23};

- if in the node N_{10} there is a pallet with target M_2 then the command $u_{10,12}$ can not be not fired so to prefer the pallet path through the node N_{11} and then leave free the path from M_4 to M_2 through the nodes N_{12}-N_{18};

- if in the node N_{19} there is a pallet with target M_1 and in the node N_{15} there is a pallet with target M_2 then the command $u_{15,16}$ can not be fired so to stop the pallet in N_{15} and give priority to the pallet in node N_{19} so to leave free the path to the machine M_2.

According to these rules, the following paths have been defined for the pallets with different targets. *Path (i)*: from M_1 the pallet to be sent to the testing machine M_2 is moved along the nodes $N_1 - N_{11}$, $N_{14} - N_{16}$; *Path (ii)*: the pallet from M_3 to M_2 is moved along the nodes N_{12}, N_{13}, $N_{17} - N_{19}$; *Path (iii)* the pallet exiting M_2 and to be sent to the reworking machine M_3 is moved along the path N_{19}, N_{16}, N_{20}, N_{21}, N_{24}, N_{25}, $N_3 - N_{10}$, N_{12} or N_{19}, $N_{22} - N_{25}$, $N_3 - N_{10}$, N_{12}; *Path (iv)*: from the testing machine M_2 to the discharge machine M_4 the path is N_{19}, N_{22}, N_{23}; *Path (v)*: from M_2 to the unload station M_1 the pallet with repaired board follows the path N_{19}, N_{16}, N_{20}, N_{21}, $N_{24} - N_{27}$, N_1 or N_{19}, $N_{22} - N_{27}$, N_1. Starting from the same initial conditions of the simulation experiment previously described, the pallets have been moved according to this heuristic approach. The results achieved, in terms of pallets visiting the machines, are reported in Figures 2.11 and 2.12, in which it is clearly shown the comparison between the two approaches: the *MPC* approach is able to use the degrees of freedom allowed by the structure of the transport line in a more efficient way than the heuristic rules, and the throughput of the plant is increased. The total number of pallet processed by each machine with the two strategies is also reported in Table 2.2, together with the percentage increment provided by *MPC*.

Algorithm	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{M_4}$
MPC	72	114	42	41
heuristic	64	96	32	37
% increment	12.5	18.8	31.3	10.8

Table 2.2: Total number of pallets processed by machines M_1, M_2, M_3, M_4: Comparison between the *MPC* algorithm and heuristic rules.

2.6.2 Experimental results

The *MPC* algorithm has been implemented in the *DCPIP*, see Section 5.2.2 for the detailed implementation of the software platform, which has allowed to implement the interface of the *MPC* controller developed in MATLAB. First, a discrete event simulation model of the de-manufacturing plant developed in SIMIO [58] has been used to test the software implementation,

see [23]. Then, the algorithm has been used to control the real system. In both cases, the algorithm works on the current state, either of the SIMIO simulator or of real plant, which can be different from the one predicted with the *MLD* model due to the unpredictable value of $\Gamma_{33}(k)$. In fact, in the real plant, the value of $\Gamma_{33}(k)$ is defined as an outcome of the testing machine M_2, while in the MATLAB implementation described in the previous section and in the SIMIO simulator it has been set randomly equal to M_1 or M_3 or M_4.
Since the real de-manufacturing plant is still on set-up phase, only four pallets have been available for the experimental tests, and the real machining operations of the machines $M_1 - M_4$ have been implemented in terms of plain delays. All the tuning parameters of the *MPC* algorithm have been set equal to those used in the simulation experiment previously described.
The tests performed on the real system have shown very satisfactory control performances, see Figure 2.13 where the paths followed by the pallets in one experiment are reported. Since the behavior of the system is very similar to the one of the simulation experiments, a detailed description of the pallets movements is not reported here, while a video of the controlled real system is available[2] to witness the performances of the proposed approach.

[2]Supplementary downloadable material is available at http://ieeexplore.ieee.org/search. This includes the video of the experiment and the readme file. This material is 30.519 MB in size.

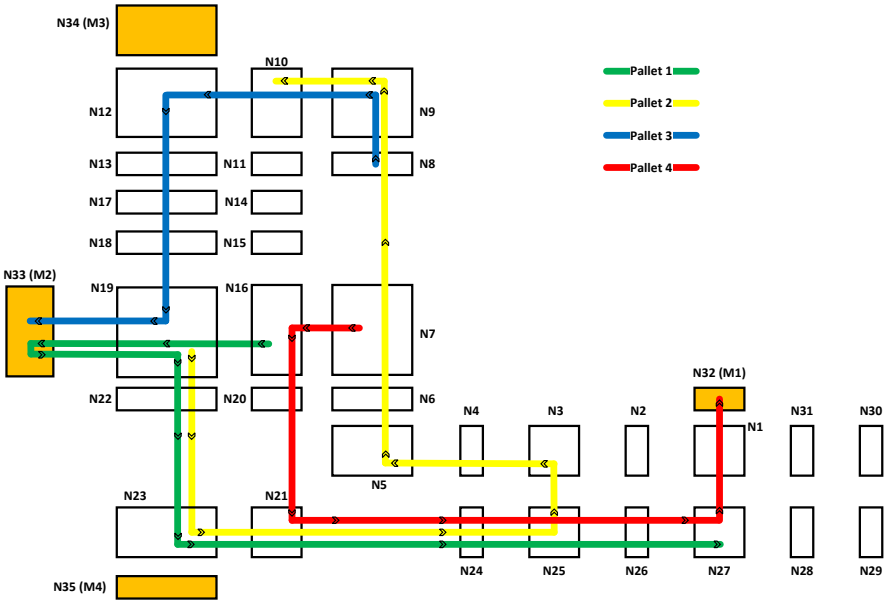


Figure 2.13: Real plant pallet paths.

Chapter 3

Production scheduling of parallel machines

The energy efficiency of manufacturing production systems is becoming a crucial topic for many reasons, such as the need to minimize the energy consumption of industrial plants, to resize the factory energy supply infrastructures, and to limit the CO_2 emissions, see e.g. [38, 54, 85, 102]. Different levels of manufacturing efficiency have been considered in the literature [43]: (i) the process level, which concerns the energy interaction related to the physical machining operations; (ii) the machine level, which considers both processing and auxiliary operations; (iii) the production line level, which refers to a group of different machines and, finally, (iv) the factory level, which concerns the high-level managing of different production lines, possibly interacting and sharing common appliances. In general, improving the efficiency at the lower levels (machine and process) is a complex task because it may result in worsening quality and costs or it may require the deployment of new and more advanced processing techniques. By contrast, energy efficiency at the production or factory level can be improved by designing suitable production scheduling and planning algorithms. This level of optimization is usually preferred because it is less invasive and does not effect quality and costs. For this reason, the development of optimization algorithms for the solution of scheduling problems, such as job shop, flow shop, and flexible flow shop, has been the subject of a huge scientific effort, see e.g. [90] and the references reported there. Recent contributions explicitly dealing with the energy efficient scheduling of production systems are reported in [2, 15, 21, 32, 39, 46].

This chapter considers the problem of optimizing on-line the production scheduling and buffer management of a multiple-line production plant com-

posed by L machines M_i, $i = 1,...,L$, which can operate at different speeds corresponding to different energy demands. The path from a common source node, where the part to be processed is assumed to be always available, to each machine may differ in the number of buffer nodes, and the energy required to move the part along these transportation lines must be suitably considered in the computation of the overall energy consumption. Therefore, the control problem consists of computing, at each sampling instant, the sequence of commands to be applied to the transportation lines and the processing speed of the machines in order to optimize the throughput of the system and to limit the overall energy consumption. This problem, which shares some similarities with the classical flexible flow shop problem, has been motivated by the optimal management of the demanufacturing plant described in [29,30]. Specifically, this plant is made by a number of machines and a multi-path transportation line, part of which is made by two parallel independent transport lines which start from the same source node and feed two independent machines. The design of the optimal pallet routing has been already considered in [23, 24], where however the target machine of each pallet has been assumed to be a-priori given.

The optimal scheduling of parallel machines, which must guarantee the completion of a given number of tasks by assigning them to different machines, has been considered in many papers, see e.g. the review [103] and the references therein. This problem is known to be very complex, see [121], and therefore the proposed solutions are mainly based on the development of heuristics, see e.g. [31, 57, 92].

On the contrary, the approach here proposed relies on *MPC*, a technique not yet widely popular in the field of discrete manufacturing as already mentioned in Section 1.3, save for the notable exceptions of [1, 14, 40, 117–119, 124], where problems related to the management of supply chains have been studied. Specifically, in the problem here considered, *MPC* recursively computes the optimal sequence of buffer commands and machines' processing times over a given prediction horizon by minimizing the overall energy consumption and maximizing the future production. Optimization is performed under suitable constraints on the production, on the electric power involved, and under the physical constraints imposed by the system. These constraints are described by logical statements, which in turn are transformed into algebraic relations among boolean variables, see [12]. Moreover, the machines are represented by finite state machine models, so that the overall system to be optimized is described by a *MLD* model, see Section 3.1, according to which the resulting optimization problem belongs to the class of *MILP* problems, for which fast solvers are available as de-

scribed in Section 3.2.
The simulation results reported in Section 3.3 clearly show that the algorithm can be easily adapted to obtain different behaviours by means of the tuning of simple and easy-to-understand parameters of the cost function. Moreover, the proposed method allows to cope with dynamic changes of the minimum production and maximum absorbed power and to choose the constraints to be violated in case of infeasibility. All these features are very difficult to be achieved with standard scheduling techniques based on the solution of *MILP* problems or on heuristics.
Many extensions of the results reported in this chapter can be considered. The first, quite simple, could deal with the inclusion of constraints on the early production of parts. Other improvements could be related to the use of lagrangian relaxation methods to simplify the optimization phase in case of large scale systems, or to the inclusion of non deterministic behaviours of the machines. For all these reasons, it is believed that model-based solutions like the one here proposed will open the way to the optimal management of high performance manufacturing plants.

Nomenclature

(B) machine *Busy* state of the Finite State Machine (FSM) of a machine
(E) machine *End* state of the FSM of a machine
(F) machine *Free* state of the FSM of a machine
M_i $i-th$ machine
$N_{i,j}$ generic node of the production system
p_i number of buffer nodes of the $i-th$ production line
q_i machine M_i absorbed power in the busy state
$f_{e_i}(\eta_i)$ function representing the absorbed power when the machine M_i is busy
$u_{i,j}$ binay variable trigger which moves the part from node $N_{i,j-1}$ to $N_{i,j}$
$x_{i,j}$ logical state related to the node $N_{i,j}$
u input vector of MLD model
x state vector of MLD model
y output vector of MLD model
δ binary auxiliary variables vector of MLD model
β_i logic output representing the busy state of the $i-th$ machine
γ_i trigger starting signal of the $i-th$ machine
δ_i^h auxiliary logical variables
η_i number of time instants in which the machine will be in the busy state
$\overline{\eta}_i$ η_i upper bound
$\underline{\eta}_i$ η_i lower bound
D_i number of possible values η_i can take
Ψ_i logic output representing the end of the processing of the machine M_i
δt sampling time
t simulation time

About the mathematical model of M_i:
z_i^l logic state denoting whether the machine is free
z_i^c integer state counting the number of time instants in the busy state
z_i^η integer state to "hold" the value of η_i during the transition $F \rightarrow B$
μ_i logical variable used to trigger the transition $F \rightarrow B$
z_F logic state which is true when the machine is free
z_B logic state which is true when the machine is busy
z_E logic state which is true when the machine is in the end state
z_c integer state counting the time instants spent on the busy state

z_η	integer state holding the value of η set during the transition $F \to B$ and $F \to E$
ξ_j	$j = 1, \cdots, 4$, auxiliary logical variables

About *MPC*:

N_{RH}	prediction horizon
RH	prediction horizon steps
Q_{prod}	weight related to the production
Q_{energy}	weight related to the energy consumpion
Q_{move}	weight related to the useless movements of the parts
Q_{part}	weight penalizing the presence of parts in the nodes if not needed
Q_{store}	weight penalizing the storage costs of the parts
R^m_{dead}	$m = 1, \cdots, N-1$, cost function exponential weights
q_{max}	maximum allowable power
P_{min}	minimum allowable production
ε_q	slack variable related to the power
ε_p	slack variable related to the production
S_q	weight related to the slack variable ε_q
S_p	weight related to the slack variable ε_p
J_l	$l = 1, \cdots, 4$, cost functions
$\bar{t}$	interval time in which the production P_{min} is guaranteeded

3.1 Problem formulation

The generic structure of the production system considered in this thesis is sketched in Figure 3.1: it consists of L parallel production lines, each one with p_i buffer nodes, $p_i \in \mathbb{N}^+$, $i = 1,\dots,L$, ended by a machine M_i.
The machines $M_1,\dots,M_L$ are assumed to have a controllable and variable duration processing time related to the required energy to perform the machining operations. This means that it is possible to choose whether a machine must process the next part at full or slow speed with consequent high or low energy demand. The binary variable $u_{i,j}$, $i = 1,\dots,L$, $j = 1,\dots,p_i$ represents the trigger which moves the part from node $N_{i,j-1}$ to $N_{i,j}$. Specifically, $u_{i,j} = 0$ if the part is not moved from $N_{i,j-1}$ to $N_{i,j}$ or if node $N_{i,j-1}$ is empty, while $u_{i,j} = 1$ if the part is moved from $N_{i,j-1}$ to $N_{i,j}$.
The control problem consists in moving the parts from the root node N_0 to the machines $M_1,\dots,M_L$ and in deciding the processing time of each machine, while ensuring that constraints on maximum power and minimum production are fulfilled.

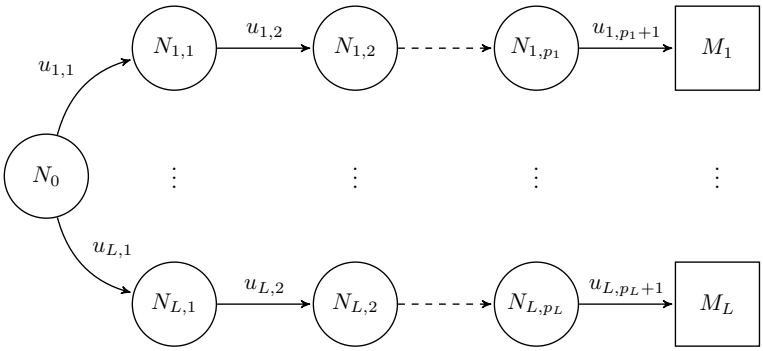


Figure 3.1: Production system.

We will consider the following assumption:

Assumption 1 *The root node N_0 always contains a part, i.e. there is always a part ready to be processed by the system.*

3.1.1 Node model

Let $x_{i,j}$ be a logical state related to node $N_{i,j}$, see Figure 3.2, and let $x_{i,j} = 1$ when $N_{i,j}$ contains a part and $x_{i,j} = 0$ otherwise. The variable $u_{i,j}$ is logical as well, and $u_{i,j} = 1$ means that the part will be moved from $N_{i,j-1}$ to $N_{i,j}$.

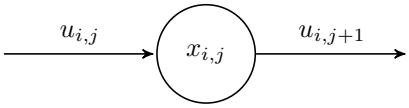


Figure 3.2: Node model $N_{i,j}$.

Letting k be the discrete time index, the dynamics of the logic state is given by

$$x_{i,j}(k+1) = x_{i,j}(k) + u_{i,j}(k) - u_{i,j+1}(k) \tag{3.1}$$

In order to simplify the notation, from now on we will drop the time index k when not required for clarity of presentation. Moreover, the superscript $^+$ will denote the variable at the next time instance, so that, given a generic variable $\varphi(k)$, the symbol φ will correspond to $\varphi(k)$ and φ^+ to $\varphi(k+1)$. According to this notation, (3.1) can be written as

$$x^+_{i,j} = x_{i,j} + u_{i,j} - u_{i,j+1} \tag{3.2}$$

The inputs $u_{i,j}$ must be suitably constrained in order to prevent the states taking values different from zero and one, and to avoid unrealistic configurations, such as moving a part out of an empty node. In particular, it is possible to move a part into $N_{i,j}$ if and only if all the following conditions are fulfilled:

1. the node $N_{i,j-1}$ contains a part;
2. the node $N_{i,j}$ is empty or it contains a part which is moved to $N_{i,j+1}$ at the same time instant.

These conditions can be rewritten using logical operators as

$$x_{i,j-1} \wedge (\neg x_{i,j} \vee (x_{i,j} \wedge u_{i,j+1})) \tag{3.3}$$

which, according to the propositional calculus rules [12, 75, 79, 93, 122], is equivalent to

$$\begin{aligned} u_{i,j} &\leq x_{i,j-1} \\ u_{i,j} &\leq 1 - x_{i,j} + u_{i,j+1} \end{aligned} \tag{3.4}$$

As for the root node N_0, Assumption 1 implies that $x_0 = 1$ at any time instant, allowing its dynamics to be neglected. However, only one part at a time can be moved out of N_0 and thus the following constraint must be ful-

filled

$$\sum_{i=1}^{L} u_{i,1} \leq 1 \tag{3.5}$$

The interface between the final node N_{i,p_i} of each line and the corresponding machine M_i will be discussed in the following together with the models of the machines.

3.1.2 Simplified Machine Model (SMM)

Let the i-th machine M_i, $i = 1, \ldots, L$, be represented by a finite state machine which describes the following behaviour:

- the machine can either be *busy* (B) or *free* (F);
- the transition $F \rightarrow B$ occurs on the rising edge of a logic (binary) input, denoted by γ_i, which may be seen as a "starting" signal;
- together with the starting signal, a second integer input, denoted with η_i, must be set. The machine will then stay in the busy state for η_i time instants and, afterwards, it will go back to be free, i.e. the transition $B \rightarrow F$ will occur;
- when the machine is busy, the output representing the absorbed power, denoted by q_i, must be set according to a properly designed function $q_i = f_{e_i}(\eta_i)$, related to the processing time;
- during the last time instant in which the machine is busy, the logic output representing the end of the processing, denoted by ψ_i, must be set;
- when busy, the logic output β_i, representing the busy state of the machine, must be set to one.

Therefore, each machine M_i can be represented by the block shown in Figure 3.3

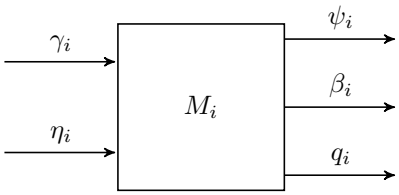


Figure 3.3: Machine block.

Mathematical model

In view of the previous definitions, the mathematical model of M_i can be derived by defining the following internal states:

- z_i^l: logic state denoting whether the machine is free ($z_i^l = 0$) or busy ($z_i^l = 1$);
- z_i^c: integer state counting the number of time instants in the busy state;
- z_i^η: integer state to "hold" the value of η_i during the transition $F \to B$.

In addition, consider the following auxiliary logical variable

$$\mu_i = 1 \leftrightarrow z_i^c \geq z_i^\eta - 1 \tag{3.6a}$$

which will be used to trigger the transition $B \to F$, as illustrated in Figure 3.4.

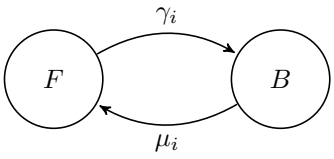


Figure 3.4: *FSM* of a machine.

The dynamics of the states is given by

$$z_i^{l+} = (z_i^l \wedge \neg\mu_i) \vee (\neg z_i^l \wedge \gamma_i) \tag{3.6b}$$

$$z_i^{c+} = \begin{cases} z_i^c + 1 & \text{if } z_i^l \\ 0 & \text{if } \neg z_i^l \end{cases} \tag{3.6c}$$

$$z_i^{\eta+} = \begin{cases} z_i^{\eta} & \text{if } z_i^l \\ \eta_i & \text{if } \neg z_i^l \end{cases} \tag{3.6d}$$

These relations describe the following facts:

- (3.6b) states that the machine will be busy at the next time instant either if it is currently busy and the processing is not yet completed, i.e. $\mu_i = 0$, or if it is currently free and the starting trigger γ_i is set;
- (3.6c) defines z_i^c as an increasing counter only when busy, which is reset when free;
- (3.6d) defines z_i^{η} as a holder of the last value that η_i has taken when in the free state. This value is kept as long as the machine is busy.

The absorbed power is defined as a function of the value of z_i^{η}, since it is different from zero only when the machine is in the busy state and is related to the production duration. No assumptions on the class of this function have been made[1], but since $z_i^{\eta} \in \mathbb{N}$, it is required to be defined only for integer positive values. In particular, assume that $\eta_i \in [\underline{\eta}_i, \bar{\eta}_i] \subseteq \mathbb{N}$, $\bar{\eta}_i \geq \underline{\eta}_i \geq 1$ and let $D_i = \bar{\eta}_i - \underline{\eta}_i + 1$ be the number of possible values η_i can take. The function $q_i = f_e(z_i^{\eta})$ can then be obtained by defining D_i auxiliary logical variables, denoted by δ_i^h, $h = 1, \ldots, D_i$ constrained as follows

$$\sum_{h=1}^{D_i} \delta_i^h = 1 \tag{3.6e}$$

$$\sum_{h=1}^{D_i} h\delta_i^h = z_i^{\eta} \tag{3.6f}$$

The absorbed power is then computed as

$$q_i = \begin{cases} \sum_{h=1}^{D_i} f_{e_i}(h)\delta_i^h & \text{if } z_i^l \\ 0 & \text{if } \neg z_i^l \end{cases} \tag{3.6g}$$

The other two logic outputs can be easily obtained from the state variables as

[1] It should be noted that the function $q_i = f_{e_i}(\eta_i)$ should be monotonically decreasing, because, in general, the faster processing (i.e. small η_i), the higher absorbed power (i.e. big q_i). However, at least from a theoretical point of view, nothing prevents it to behave differently.

$$\beta_i = z_i^l \tag{3.6h}$$

and, for the end of cycle

$$\psi_i = z_i^l \wedge \mu_i \tag{3.6i}$$

which can be recast as a set of inequalities as described in [12].

Machine-buffer interface

The machines must be properly connected to the buffer lines as in Figure 3.1. It can be noted that each machine behaves similarly to a node. The main difference is in the transition from being busy to free. In fact, each buffer node $N_{i,j}$ can be freed by setting the controllable variable $u_{i,j+1}$, while the machines are automatically freed once the processing is completed. Therefore, with respect to Figure 3.1 and the notation presented in Section 3.1.1, each machine can be seen as a node by setting

$$\begin{aligned} x_{i,p_i+1} &= \beta_i \\ u_{i,p_i+1} &= \gamma_i \end{aligned} \tag{3.7}$$

Concerning constraints (3.4), due to the inability to pull a part out of the machine, a part in N_{i,p_i+1} can be moved into M_i only when M_i is free, i.e. $x_{i,p_i+1} = 0$. Therefore, the set of constraints for the last node of each line is

$$\begin{aligned} u_{i,p_i+1} &\leq x_{i,p_i} \\ u_{i,p_i+1} &\leq 1 - x_{i,p_i+1} \end{aligned} \tag{3.8}$$

The underlying assumption is that each machine must be free for at least one time instant before a new part can be processed. This assumption will be removed in the next paragraph in which an enhanced machine model is introduced.

3.1.3 Enhanced Machine Model (EMM)

The model presented in Section 3.1.2 is valid only when the machine cannot achieve continuous processing operations, meaning that it is not possible to unload the machined part and load a new one at the same time. In fact, the previous model forces the machine to be free for at least one time instant (unload operation) before going back to busy. However, some industrial applications may achieve continuous processing operations and therefore an

enhanced machine model is developed to take this behaviour into account. As for the simpler model, consider the state machine in Figure 3.5:

- *Free* (F): the machine is not operating and waiting for a part to be processed
- *Busy* (B): the machine is operating and it is not the last operative time instant
- *End* (E): the machine is operating in its last operative time instant

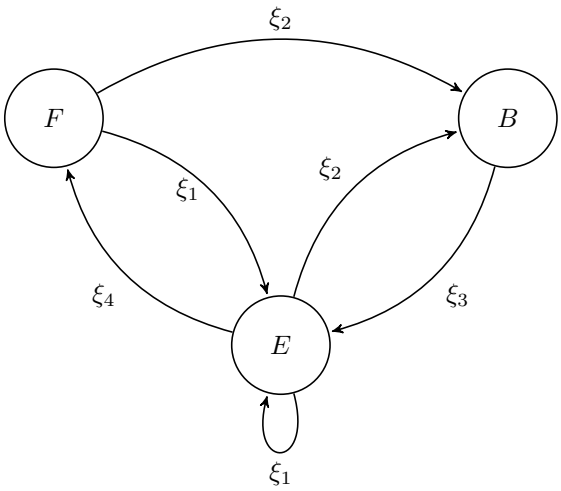


Figure 3.5: *FSM* of a enhanced machine.

The main difference with the previous model is the ability to directly switch from the end state to the busy one, i.e. to begin the machining of a new part immediately after the current one has finished. There is no need to go through the free state. Inputs and outputs are defined in the same way as in the previous model. However, the output power q_i must be different from zero when the active state is either busy (B) or end (E). In fact, these states only differ from a timing point of view, i.e. E is active during the last time instant of the processing, while B during the previous ones.

Mathematical model

In the following, to simplify the notation the index i denoting the specific machine will be dropped. Consider the following state variables:

- z^F: logic state which is true when the machine is free;

- z^B: logic state which is true when the machine is busy;
- z^E: logic state which is true when the machine is in the end state;
- z^c: integer state counting the time instants spent on the busy state;
- z_η: integer state "holding" the value of η set during the transition $F \to B$ and $F \to E$.

The transitions are defined by the following auxiliary logical variables

$$\xi_1 = 1 \leftrightarrow \gamma \wedge \eta = 1 \tag{3.9a}$$
$$\xi_2 = 1 \leftrightarrow \gamma \wedge \eta > 1 \tag{3.9b}$$
$$\xi_3 = 1 \leftrightarrow z^c \geq z^\eta - 2 \tag{3.9c}$$
$$\xi_4 = 1 \leftrightarrow \neg\gamma \tag{3.9d}$$

In other words, the following scenarios are possible:

1. From the free state, it is possible to go to the busy or end states depending on the value of the processing time η. In particular, if $\eta > 1$ the active state will be B and a long processing will start, otherwise it will be E leading to a new one-step machining;
2. When the active state is B, the machine will remain busy until the remaining processing time is only one step, that is $\xi_3 = 1$. At this point, the machine will switch to E;
3. From the E state it is possible to:
   (a) begin a new long processing ($\eta > 1$) by switching to B;
   (b) begin a new one-step processing ($\eta = 1$) by staying in E;
   (c) end the current processing by going back to F.

The dynamics of the state is given by

$$z^{F+} = (z^F \wedge \neg\xi_2) \vee (z^E \wedge \xi_4) \tag{3.9e}$$
$$z^{B+} = (z^F \wedge \xi_2) \vee (z^B \wedge \neg\xi_3) \vee (z^E \wedge \xi_2) \tag{3.9f}$$
$$z^{E+} = (z^F \wedge \xi_1) \vee (z^B \wedge \xi_3) \vee (z^E \wedge \xi_1) \tag{3.9g}$$
$$z^{c+} = \begin{cases} z^c + 1 & \text{if } z^B \\ 0 & \text{if } \neg z^B \end{cases} \tag{3.9h}$$
$$z^{\eta+} = \begin{cases} z_i^\eta & \text{if } z^B \\ \eta_i & \textit{otherwise} \end{cases} \tag{3.9i}$$

Concerning the outputs, (3.6e), (3.6f) still apply in this context. However, (3.6g) must be changed to

$$q = \begin{cases} \sum_{h=1}^{D} f_e(h)\delta^h & \text{if } z^B \vee z^E \\ 0 & \textit{otherwise} \end{cases} \tag{3.9j}$$

The busy output β is now defined as

$$\beta = z^B \vee z^E \tag{3.9k}$$

and the end of cycle ψ

$$\psi = z^E \tag{3.9l}$$

The logical states z^B, z^F and z^E must also be constrained as follows

$$z^F + z^B + z^E = 1 \tag{3.9m}$$

In fact, although the combination of the graph topology in Figure 3.5, together with the definitions of ξ_1, ξ_2, ξ_3 and ξ_4 and the dynamics of z^B, z^F and z^E prevent the activation of more that one state at a time, constraint (3.9m) is still required to avoid the potential feasibility of the initial state $z^F = z^B = z^E = 0$, which would lead to a wrong evolution of the system.

As in the case of the *SMM*, model (3.9) belongs to the class of *MLD* and thus it can be reformulated as a suitably constrained linear system [12].

Machine-buffer interface

The machine-buffer interface is very similar to what discussed in Section 3.1.2. In particular, (3.7) still applies for the enhanced model with β defined in (3.9k). By contrast, it is now possible to start the machining of a new part when the part being currently processed is about to end. More specifically, if the active state of M_i is E, which corresponds to $\psi_i = 1$, the machine can process a new part. Such behaviour is achieved by setting the constraints in place of (3.8)

$$\begin{aligned} u_{i,p_i+1} &\leq x_{i,p_i} \\ u_{i,p_i+1} &\leq 1 - x_{i,p_i+1} + \psi_i \end{aligned} \tag{3.10}$$

From a conceptual point of view, it can be noted that the role of ψ_i in (3.10) is equal to that of $u_{i,j+1}$ in (3.4). Indeed, ψ_i can be seen as a signal which moves the part out of the machine M_i in the same way as $u_{i,j+1}$ moves it out of $N_{i,j}$.

3.1.4 Mixed logical dynamical model of the system

The system described in the previous Sections is characterized by the mutual coexistence of discrete time dynamics and logical variables. Such systems are referred as *hybrid*. Different classes of hybrid dynamical models have been developed in the literature: Piecewise Affine (*PWA*), *MLD*, Linear Complementary (*LC*), Extended Linear Complementary (*ELC*) and Max-Min-Plus-Scaling (*MMPS*) Systems, see [49] and the references therein. From now on, *MLD* models [12] will be considered since they are particularly suitable for control purposes.
As already discussed in Section 2.5, a *MLD* system can be described by the following linear relations

$$x(k+1) = Ax(k) + B_u u(k) + B_\delta \delta(k) + B_z z(k) \tag{3.11a}$$

$$y(k) = Cx(k) + D_u u(k) + D_\delta \delta(k) + D_z z(k) \tag{3.11b}$$

$$E_\delta \delta(k) + E_z z(k) \leq E_u u(k) E_x x(k) + E \tag{3.11c}$$

where:

- $x = [x_c, x_l]$ is the state vector and $x_c \in \mathbb{R}^{n_c}$, $x_l \in \{0,1\}^{n_l}$;

- $u = [u_c, u_l]$ is the input vector and $u_c \in \mathbb{R}^{m_c}$, $u_l \in \{0,1\}^{m_l}$;
- $y = [y_c, y_l]$ is the output vector and $y_c \in \mathbb{R}^{p_c}$, $y_l \in \{0,1\}^{p_l}$;
- $\delta \in \{0,1\}^{r_l}$ is a vector of binary auxiliary variables;
- $z \in \mathbb{R}^{r_c}$ is a vector of continuous variables.

In [12] it is shown how logical relationships, such as (3.6b), as well as implications between continuous and binary variables, e.g. (3.6c), can be recast as a set of linear inequalities by introducing a certain number of auxiliary variables, resulting in a model in the form of (3.11a)- (3.11c) [2].

3.1.5 Overall plant model

In summary, the overall system to be controlled is defined by the connection of all the nodes and the machines accordingly to the plant topology. Therefore, the plant model can be seen as the block shown in Figure 3.6 where:

- $u_{i,j}$, $i = 1,\ldots,L$, $j = 1,\ldots,p_i$ are controllable inputs which cause the parts to move and the machines to start;
- η_i, $i = 1,\ldots,L$, are the controllable inputs which define the processing time of each part;
- q_i, ψ_i and β_i, $i = 1,\ldots,L$, are the measured outputs as described in Section 3.1.2.

Note that the use of either the simple of the enhanced machine model is transparent with respect to the definition of inputs and outputs, since they differ only in the internal dynamics. Therefore, it is possible to define a unique control problem which will adapt itself to the actual model of the machines being used. Moreover, it is also possible to include configurations in which both models coexist to describe the behavior of different machines.

[2] For the Simplified Machine Model: $n_c = 7$, $n_l = 2$, $m_c = 2$, $m_l = 5$, $p_c = 2$, $p_l = 4$, $r_c = 6$, $r_l = 10$. For the Enhanced Machine Model: $n_c = 7$, $n_l = 6$, $m_c = 2$, $m_l = 5$, $p_c = 2$, $p_l = 4$, $r_c = 6$, $r_l = 24$.

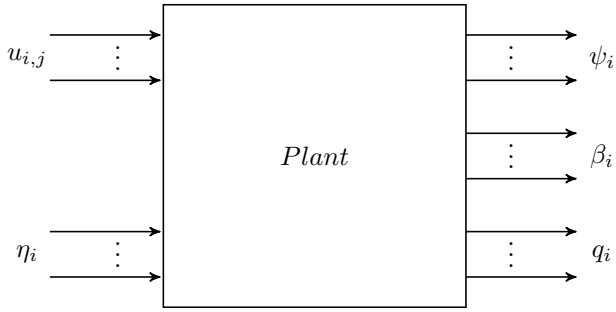


Figure 3.6: Schematic block of the plant.

3.2 Model predictive control

According to the *MPC* strategy described in Section 1.2, the cost function to be minimized includes the following terms:

1. *Production*: a negative cost in order to maximize the production. It can be measured in terms of end-of-cycles of the machines, indicated by the outputs ψ_i, $i = 1, \ldots, L$;
2. *Energy consumption*: a positive cost to minimize the overall energy consumption;
3. *Movements*: a positive term weighting useless movements of the parts;
4. *Part*: a positive weight penalizing the presence of parts in the nodes if not needed to fulfil the requirements.

Based on the previous considerations, the cost function has been selected as follows

$$\begin{aligned} J_1 = &- Q_{prod} \sum_{k=t}^{t+N_{RH}-1} \sum_{i=1}^{L} \psi_i(k) + Q_{energy} \Delta t \sum_{k=t}^{t+N_{RH}-1} \sum_{i=1}^{L} q_i(k) \\ &R_{move} \sum_{k=t}^{t+N_{RH}-1} \sum_{i=1}^{L} \sum_{j=1}^{p_i} u_{i,j}(k) + Q_{part} \sum_{k=t}^{t+N_{RH}-1} \sum_{i=1}^{L} \sum_{j=1}^{p_i} x_{i,j}(k) \end{aligned} \tag{3.12}$$

where RH is the prediction horizon, t is the current time index, Δt is the adopted sampling time and the weights Q_{prod}, Q_{energy}, R_{move} and Q_{part} are design parameters. The maximum absorbed power and minimum production requirements along the prediction horizon can be included in the optimization problem by defining q_{max} as the overall maximum allowable

power and P_{min} as the minimum allowable production. Then, the following constraints must be considered

$$\sum_{i=1}^{L} q_i(k) \leq q_{max} \tag{3.13a}$$

$$\sum_{k=t}^{t+N_{RH}-1} \sum_{i=1}^{L} \psi_i(k) \geq P_{min} \tag{3.13b}$$

In order to avoid infeasibility problems, due to the potential impossibility to contemporarily fulfill the above relations, constraints (3.13) can be modified to allow their violation when necessary by adding slack variables ε_q and ε_p as follows

$$\sum_{i=1}^{L} q_i(k) \leq q_{max} + \varepsilon_q \tag{3.14a}$$

$$\sum_{k=t}^{t+N_{RH}-1} \sum_{i=1}^{L} \psi_i(k) \geq P_{min} - \varepsilon_p \tag{3.14b}$$

$$\varepsilon_q, \varepsilon_p \geq 0 \tag{3.14c}$$

These slack variables must be heavily weighted in the cost function in order to prevent them to be different from zero when the optimization problem is feasible, so that the cost function becomes

$$J_2 = J_1 + S_p \varepsilon_p + S_q \varepsilon_q \tag{3.15}$$

where the coefficients S_p and S_q take sufficiently high values. Therefore the optimization problem can be stated as

$$\min_{u_{i,j},\, \eta_i,\, i=1,\ldots,L, j=1,\ldots,p_i, \varepsilon_p, \varepsilon_q} J_2 \tag{3.16}$$

subject to (3.14).

Avoiding deadlocks and guaranteeing due date

The previous formulation of the *MPC* optimization problem does not guarantee a given due date, a property often required in industrial applications. In fact, the constraints (3.14) are implemented according to the receding horizon approach, and the optimal solution can even cause deadlocks of the

system when the minimum production requirement is small. This is shown in the very simple scenario depicted Figure 3.7, where it is assumed that the prediction horizon is $N_{RH} = 5$ steps, the minimum production requirement is $P_{min} = 1$ and the maximum processing time is $\eta = 2$.

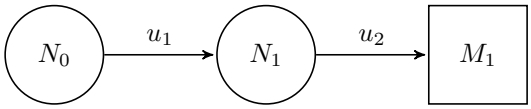


Figure 3.7: Example of a simple configuration.

The production requirement can be satisfied by operating at the slower speed (i.e. $\eta = 2$), and the controller will choose this scenario in order to minimize the absorbed power, given that $f_e(1) > f_e(2)$. However, two solutions still solve the problem; in fact, since only four time steps are required to process the part and the horizon is one step longer, the remaining step can be put on the leading (Figure 3.8a) or trailing (Figure 3.8b) edge of the horizon. These solutions have the same optimal cost and thus the controller will randomly choose one of them. However, the first one (Figure 3.8a) must be avoided, since it locks the system due to the *RH* implementation. In the following, two solutions to this problem are proposed. The first one does not introduce hard constraints, but does not guarantee a given level of production, while the second one is characterized by additional constraints which enforce the fulfillment of a prescribed due date.

(a) First Solution: initial delay

(b) Second Solution: no initial delay

Figure 3.8: Two feasible solutions for the same problem.

Exponential weighting

This solution consists in increasingly weighting the control variables along the horizon, so that the optimal solution tends to activate the control variables at the beginning of the prediction horizon. With reference to the previous example, this means that the former solution is "cheaper" than the latter. The new cost function is

$$J_3 = J_2 + \sum_{k=t}^{t+N_{RH}-1} R_{dead}^{k-t} \sum_{i=1}^{L} \sum_{j=1}^{p_i} u_{i,j}(k) \tag{3.17}$$

where $R_{dead}^0 < R_{dead}^1 < \ldots < R_{dead}^{N_{RH}-1}$. In addition, $R_{dead}^{N_{RH}-1}$ must be much smaller than any other weight appearing in (3.15) in order for the optimal solution to be unaffected, besides the removal of any initial delay.

Guaranteed due date

In the second solution, the production P_{min} is guaranteed at every RH time steps. This can be achieved by letting $\bar{t} = \lfloor t/N_{RH} \rfloor N_{RH}$ and by introducing in the optimization problem the additional constraint

$$\sum_{k=\bar{t}}^{\bar{t}+N_{RH}-1} \sum_{i=1}^{L} \psi_i(k) \geq P_{min} \tag{3.18}$$

which is not forced in a RH form, but makes reference to successive time intervals of length RH. Also in the case of these additional constraints, the use of suitable slack variables can be necessary to avoid feasibility properties, as discussed in (3.14), (3.15) for constraints (3.13). In any case, the resulting optimization problem is more stringent than the one with the exponential weighting, so that the feasibility issue can become more critical.

Storage costs

The parts produced in the N_{RH} time steps considered the due date constraint have usually to be stored before their delivery, with related storage costs. In order to minimize these costs, the production can be weighted in the performance index so as to postpone it as long as possible. This can be easily considered in the problem formulation with the new performance index

$$J_4 = J_3 + \sum_{k=\bar{t}}^{\bar{t}+N_{RH}-1} [Q_{store} \sum_{i=1}^{L} \psi_i(k)] \tag{3.19}$$

where Q_{store} is a proper weight.

3.3 Simulation experiments

The algorithm here proposed has been implemented in MATLAB using the MPT Toolbox [51] and the HYSDEL [114] modelling language. In the following, two experiments are described to highlight its main features. The first one focuses on the analysis of the system behaviour in response to variations of the minimum production regardless of the energy consumption, while the second one on the production maximization constrained by a limited amount of available power. Both experiments are based on the plant depicted in Figure 3.9, where the two machines can process the parts at slow speed (two time instants, $\eta = 2$) or at full speed (one time instant, $\eta = 1$), but they differ in the absorbed power, as listed in Table 3.1.

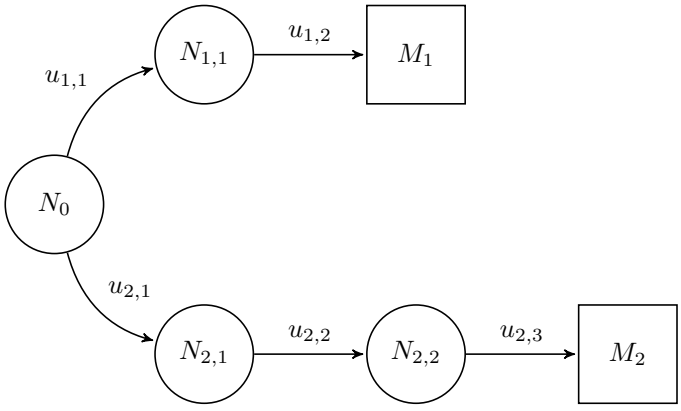


Figure 3.9: Experimental plant.

Machine	Processing Time Instants	Absorbed Power
M_1	1	2.40 $[kW]$
M_1	2	1.05 $[kW]$
M_2	1	2.20 $[kW]$
M_2	2	1.00 $[kW]$

Table 3.1: Machines absorbed power

Note that the absorbed power refers to the whole processing which is assumed to be uniformly split during the machining. For instance, M_2 requires an overall energy of $2.00\Delta t\,[kJ]$, where $\Delta t[s]$ is the adopted sampling time measured in seconds, to process a part at slow speed, that is in two time instants. Some remarks about the selected values are in order:

- as expected, slow processing times lead to lower absorbed power;
- M_1 is always more powerful than M_2, given the same processing time. However, from Figure 3.9, it can be noted that M_2 is farther than M_1 from the source node. Therefore, M_2 may not always be the obvious optimal choice.

The following tuning parameters are used in both experiments:

- the prediction horizon is $N_{RH} = 6$;
- the sampling time is $\Delta t = 60[s]$;
- the movement cost is $R_{move} = 0$;
- the storage cost is $Q_{store} = 0$;
- the weights of the slack variables are $S_q = 10^6$ and $S_p = 10^4$.

Unless otherwise specified, the exponential weighting method described in Section 3.2 to avoid deadlocks has been used with $R^h_{dead} = 0.01(h+1)$, $h = 0, \ldots, N_{RH} - 1$.

3.3.1 Minimum production

The aim of this experiment is to demonstrate the scheduling capabilities of the algorithm in case of dynamic variations of the minimum production requirement. The cost function parameters have been chosen as $Q_{part} = 10$, $Q_{prod} = 1.2 \times 10^5$, $Q_{energy} = 1$. Note that, even if $Q_{prod} \gg Q_{energy}$, the energy consumption has a more relevant impact on the performance index since the power is measured in terms of $[W]$ and the energy magnitude is 10^5, then it is expected to be minimized with the production as low as required to fulfil the minimum specification. Moreover, if the minimum production is set to zero, the trivial solution of turning all the plant off will be the optimal one to minimize the energy consumption.
In the simulation experiment, the maximum power constraint (3.13a) has been neglected and the minimum roduction has been changed as follows:

- at $t = 0[min]$ only one part must be processed during the prediction horizon, i.e. it has been set $P_{min} = 1$;
- at $t = 20[min]$, $P_{min} = 2$;
- at $t = 40[min]$, $P_{min} = 4$;
- at $t = 60[min]$, $P_{min} = 6$;
- at $t = 80[min]$, $P_{min} = 8$.

Simplified Machine Model

Figure 3.10 shows the resulting scheduling of the machines as well as the buffer management when the model described in Section 3.1.2 is used. The dashed vertical lines in Figure 3.10a denote the changes in the minimum production requirement. The following remarks can be stated:

- from $t = 0[min]$ to $t = 20[min]$, machine M_2 is preferred to M_1. This is not surprising since the low production allows the controller to choose the cheapest machine at slow speed, that is M_2 for two time steps;
- from $t = 20[min]$ to $t = 40[min]$, machine M_2 is still preferred to M_1. In order to fulfil the higher production requirement, the idle time of M_2 is reduced;
- from $t = 40[min]$ to $t = 60[min]$, M_1 is periodically scheduled at slow speed;
- from $t = 60[min]$ to $t = 80[min]$, the production requirement further increases and both machines switch at full speed, as expected;
- from $t = 80[min]$ to $t = 100[min]$, both machines already operate at full speed but the production is further increased. This results in a missing production, as shown in Figure 3.10b.

The previous experiment has been repeated by substituting the exponential weighting with the due date constraint described in Section 3.2. The results achieved are reported in Figure 3.11. A direct comparison between Figures 3.10 and 3.11 shows that the exponential weighting method produces a slightly more regular solution in terms of machines use and energy consumption. Moreover, if the minimum production requirement changes during the fixed time window where the constraint (3.18) is active, the due date constraint can lead to feasibility problems, see time $t = 40[min]$ for example. For this reason, the exponential weighting can be preferred when there are not hard production constraints.
The same experiment has been performed by including in the performance index to be minimized a term weighting the storage time, as shown in (3.19). Specifically, the weight $Q_{store} = 3 \cdot 10^4$ has been used and the results obtained are shown in Figure 12. By comparing the results of Figures 11 and 12, it is easy to see the different behavior of the machines in the interval from $t = 20[min]$ to $t = 40[min]$, in which the machines use a higher power and then work faster in order to reduce the storage time of the produced parts.

Enhanced Machine Model

The plots related to the use of the enhanced machine model are illustrated in Figure 3.13. The following observations can be made:

- from $t = 0[min]$ to $t = 20[min]$, the optimal scheduling is equal to the previous case. In fact, the minimum production is low enough to avoid any continuous machining;
- from $t = 20[min]$ to $t = 40[min]$, M_1 is never turned on. On the other hand, M_2 takes advantage of the continuous machining to fulfil the requirements. Note that this solution is comparable to the previous case. Indeed, the only difference is in the different scheduling of the idle time of M_2;
- from $t = 40[min]$ to $t = 60[min]$, M_2 is continuously processing the parts at slow speed, while M_1 periodically operates at slow speed. Comparing these results to those of Figure 3.10, the idle time of the more expensive M_1 is now bigger, leading to a lower energy consumption;
- from $t = 60[min]$ to $t = 80[min]$, both machines are continuously scheduled at slow speed. Note that in Figure 3.10 they were periodically scheduled at full speed, but the same production is achieved;
- from $t = 80[min]$ to $t = 100[min]$, a missing production occurs, as depicted in Figure 3.13b. It may be argued that the machines should both operate at full speed, which is not possible due to the plant topology.

3.3.2 Production maximization

In the second experiment, the cost function has been tuned in order to provide the maximum production and minimize the overall energy consumption at the same time. The cost function parameters have been chosen as follows: $Q_{part} = 1$, $Q_{prod} = 2 \times 10^5$, $Q_{energy} = 1$. In contrast with the previous experiment, the production is maximized with respect to the maximum available power. The absorbed power has been dynamically constrained to a maximum value. In particular, the following bounds have been applied:

- from $t = 0[min]$ to $t = 20[min]$, the problem is unconstrained;
- from $t = 20[min]$ to $t = 40[min]$, the maximum absorbed power is $q_{max} = 4.5[kW]$;

- from $t = 40[min]$ to $t = 60[min]$, it is decreased to $q_{max} = 2.2[kW]$;
- from $t = 60[min]$ to $t = 80[min]$, it is further decreased to $q_{max} = 2.0[kW]$;
- finally, from $t = 80[min]$ to $t = 100[min]$, the maximum power is set to $q_{max} = 1.0[kW]$ and the minimum production is increased from zero to $P_{min} = 4$.

Simple Machine Model

Figure 3.14 presents the results obtained with the simple machine model. The following observations are in order:

- during the first interval $t \leq 20[min]$, the machines are scheduled to run at full speed. In fact, the production is maximized with higher priority than the absorbed power, which is also unbounded;
- from $t = 20[min]$ to $t = 40[min]$ an interesting behaviour is observed: at about $t = 25[min]$ the plant reaches steady state conditions where the number of produced parts is equal to the one in the previous time interval. The difference is in the management of the buffer nodes. Indeed, node $N_{2,2}$ is now always filled by a part and the two machines work alternately. One may argue that this configuration is intuitively better than the previous one because it achieves the same production with lower energy consumption. However, this is not true due to the cost associated to the parts in the nodes, which is now higher. Different tunings of the cost function parameters may change this result;
- from $t = 40[min]$ to $t = 60[min]$ the upper bound of the power is further decreased and the machines must switch to slow speed production mode;
- from $t = 60[min]$ to $t = 80[min]$ the machines operate out of phase in order to fulfil the further reduced maximum absorbed power;
- finally, during the last interval, the maximum power is so small that the controller must operate only M_2 at slow speed. Moreover, the minimum production bound cannot be fulfilled and, since $S_p < S_q$, its violation is preferred to the power one.

Enhanced Machine Model

The results are illustrated in Figure 3.15. The following remarks are in order:

- during the first three intervals (up to $t = 60[min]$), the machines are continuously scheduled at slow speed. In fact, as previously discussed, due to the plant topology it is not possible to continuously schedule both the machines at full speed. However, the production is equal to the previous case and the overall absorbed power is much smaller, especially in the first interval ($t < 20[min]$);
- from $t = 60[min]$ to $t = 80[min]$, M_1, which requires an higher power, is turned off, while M_2 is continuously scheduled at slow speed;
- during the last interval, the same behaviour as in the previous case is observed. However, M_2 can now continuously operate reducing the missing production.

3.3.3 Computational issues

The proposed algorithm requires to solve a *MILP* problem at any time instant; the mean values of the times required by the optimization at any time step in the simulation experiments previously described are listed in Table 3.2 [3]. It is apparent that, in the considered example, the required computations are fully compatible with a real-time implementation. However, it is also clear that solving the optimization problem at any time step can be computationally demanding, and time required depends on the the number of machines, the model adopted to describe their behavior, the number of nodes, and the length of the prediction horizon. To this regard, it must be noted that also more standard *MILP* formulations can lead to practically unsolvable problems, see the following Section 3.3.5, while heuristic approaches can be very conservative. In the case of large-size problems, a possible solution consists in resorting to lagrangian relaxation techniques, which can be referred to spatial and/or temporal distributions, as already suggested in [10, 76, 93].

[3] The simulations have been carried out on a computer Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.10GHz 2.10 GHz, 16.0 GB Installed Memory (RAM), System type 64-bit Operating System, x64-based processor, Windows 8.1 Pro., MATLAB R14b, YALMIP R20150204, CPLEX R12.4, CPLEX settings: Parallelmode = 0, Threads = 0.

	Machine Model	
Experiment	**EMM**	**SMM**
Minimum production	0,400	0,303
Production maximization	0,456	0,367

Table 3.2: Mean computational time $[s]$ per optimization step

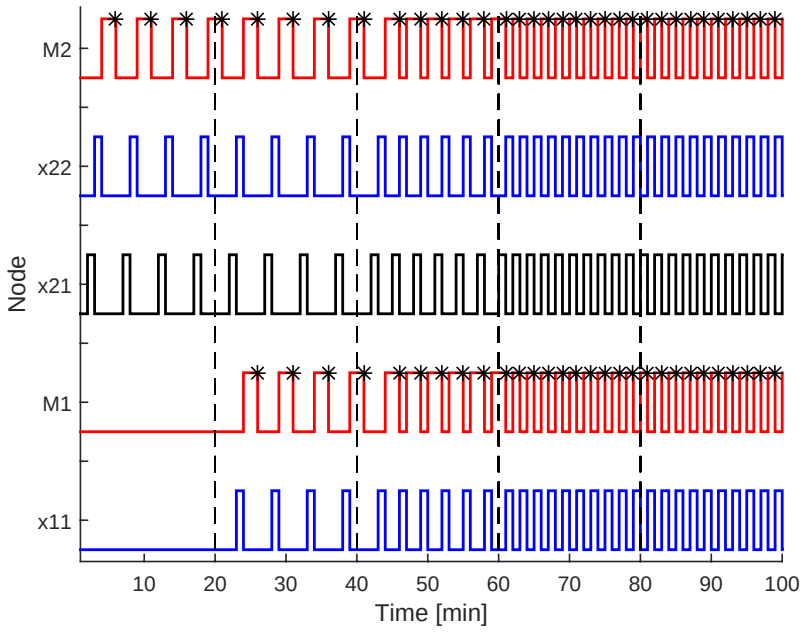
3.3.4 Sensitivity to the cost function parameters

Some experiments have been performed with the Simplified Machine Model to analyze the sensitivity of the solution with respect to the parameter Q_{prod} weighting the production in the cost function (3.12). To this end, it has been set $N_{RH} = 6$, $Q_{energy} = 1$, $P_{min} = 1$, $Q_{part} = 0$, while Q_{prod} has been varied. The number of produced parts in the time interval $[1, 100]$ is reported in Figure 3.16 as a function of Q_{prod}. As expected, the curve is non decreasing and, due to the discrete nature of the production process, it is characterized by a stepwise form. Notably, when $Q_{prod} \leq 1.2 \times 10^5$ the contribution of the energy consumption dominates and the system works at the minimum production, while, if $Q_{prod} \geq 1.85 \times 10^5$, the contribution of the production sets the system to the maximum possible production value.

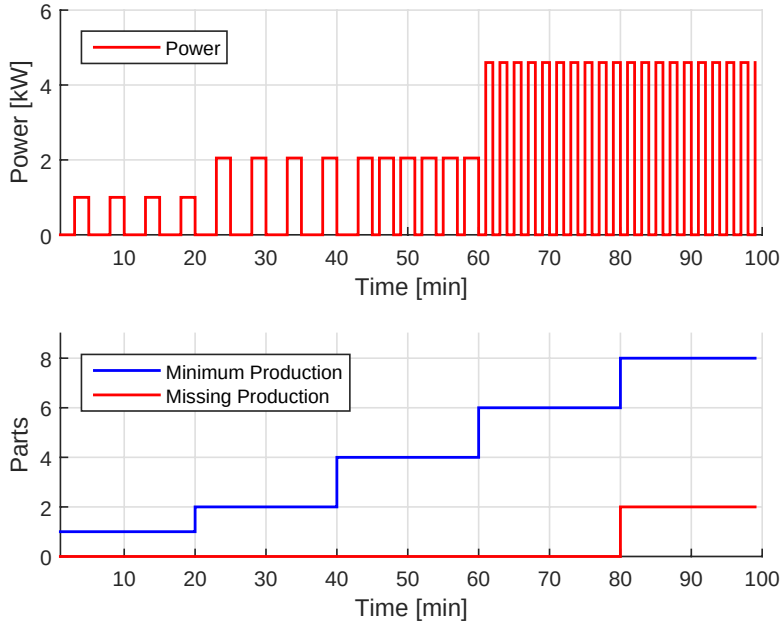
3.3.5 Comparisons with a non *RH* solution

The performances of the algorithm based on the simplified machine model have been evaluated by comparing the *RH* implementation and the non *RH* solution where the goal has been to compute the control variables along a long prediction horizon of thirty time instants. The same cost function has been adopted in the two cases, with $Q_{prod} = 1.2 \times 10^5$ and $Q_{energy} = 1$ and $Q_{part} = 10$. It has been set $P_{min} = 20$ for non *RH* solution while the *RH* implementation has been computed with $N_{RH} = 6$, $P_{min} = 4$, and including the due date constraint (3.18). In the non *RH* solution formulation the final state has been forced to be null, i.e. with empty buffers and machines, so that the computed recipe can be repeatedly applied for successive periods of the same length. The results obtained with this approach are reported in Figure 3.17, while those provided by the *RH* implementation are shown in Figure 3.18. Although the number of produced parts is the same in the two cases, the solution provided by non *RH* implementation is more homogeneous in terms of required power, with lower peaks. However, two comments are in order. First, the solution of the non *RH* problem requires $413.891[s]$, while the *RH* approach calls for a total time $9.910[s]$ (average

time of $0.330[s]$ per step). In addition, with non *RH* implementation the computational time exponentially increases, so that it is almost impossible to consider time windows larger than 35-40 sampling times. On the contrary, the computational time associated with *RH* is constant. Second, the *RH* implementation allows for much more promptness in front of possible faults of the system's components or external disturbances in view of the repeated sequence of optimizations performed at any time instant, while the non *RH* solution should be recomputed to cope with these perturbations. As a matter of fact, with *RH* a feedback control law is implemented at any sampling time, while the non *RH* approach leads to an open-loop control law which is prone to disturbances and/or modeling errors.

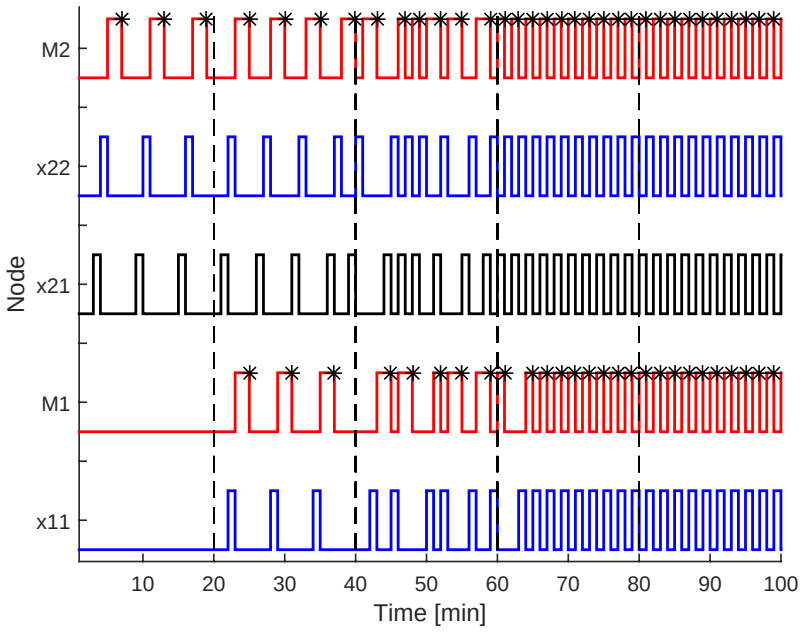


(a) Machine scheduling and buffer management. The black asterisks denote the end of the machining.

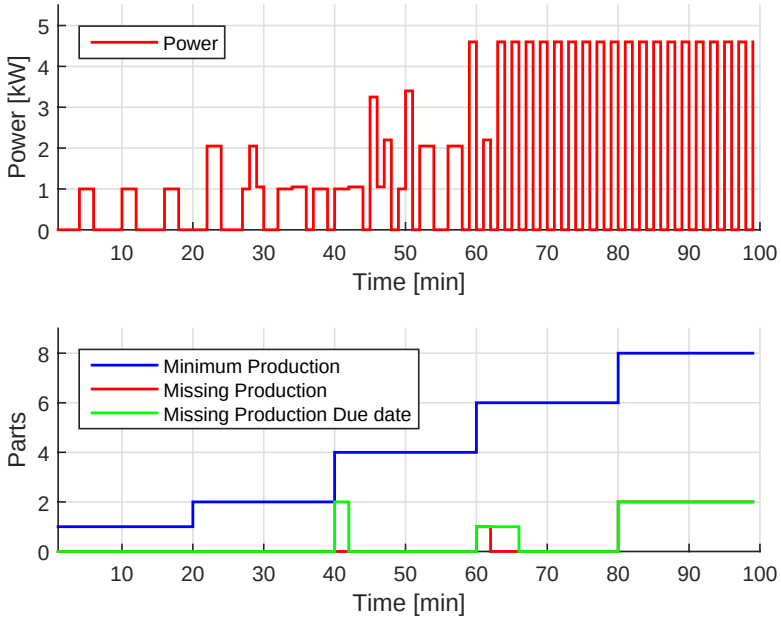


(b) Production requirement, missing production (bottom) and absorbed power (top).

Figure 3.10: Power Minimization using the simple machine model.

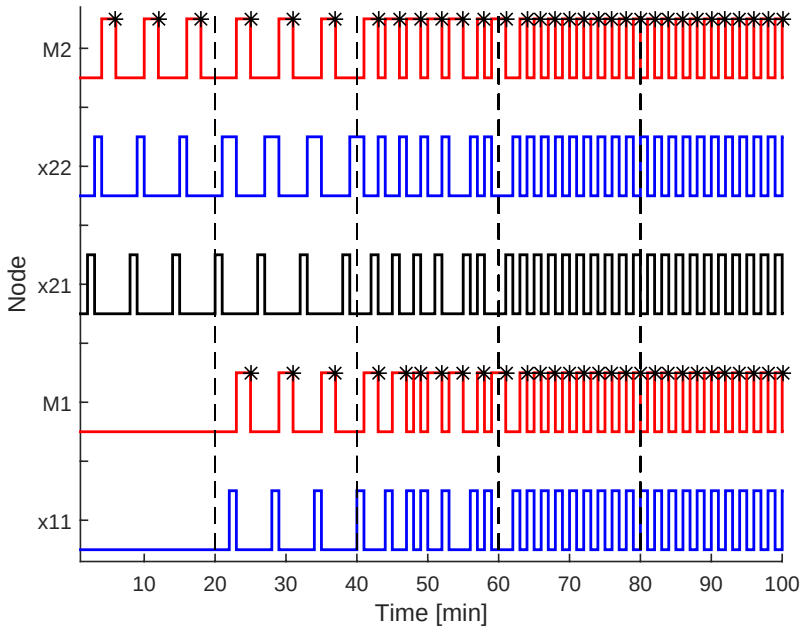


(a) Machine scheduling and buffer management. The black asterisks denote the end of the machining.

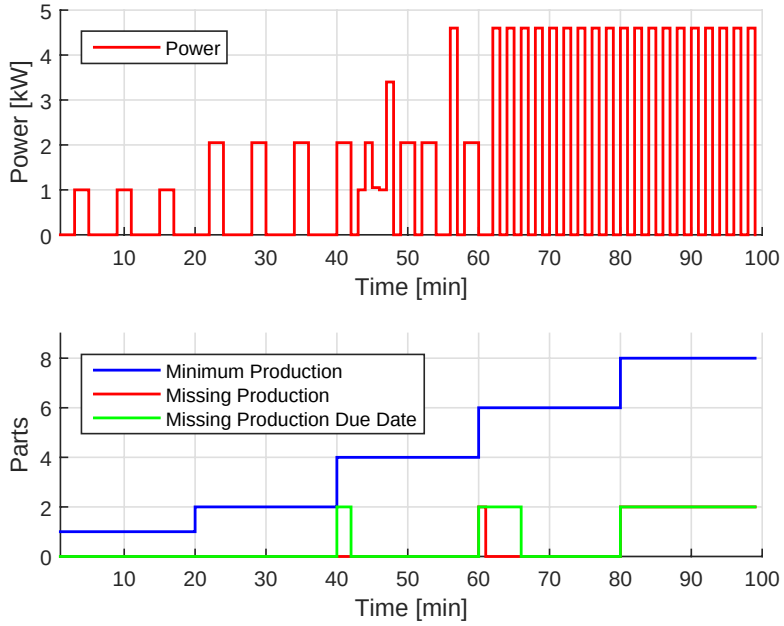


(b) Production and Absorbed Power.

Figure 3.11: Minimum production constrained problem with Due date constraint.

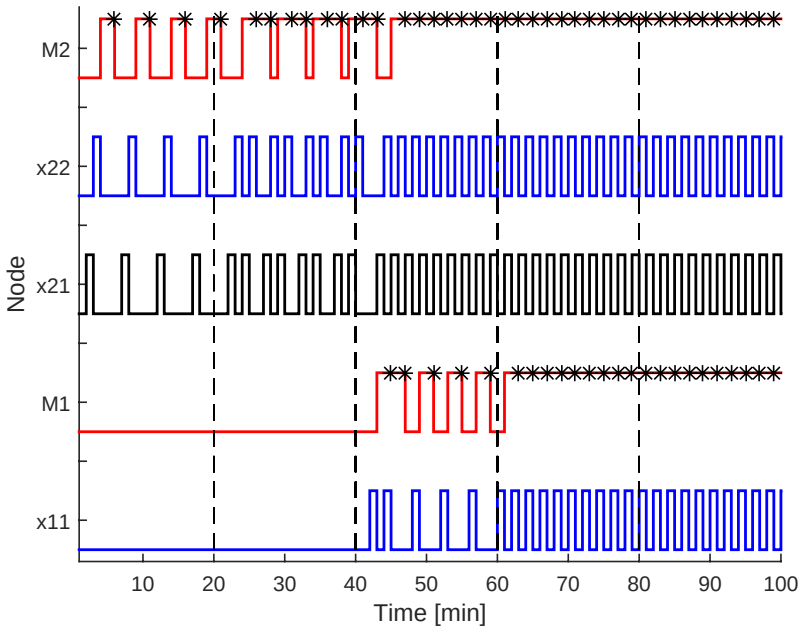


(a) Machine scheduling and buffer management. The black asterisks denote the end of the machining.

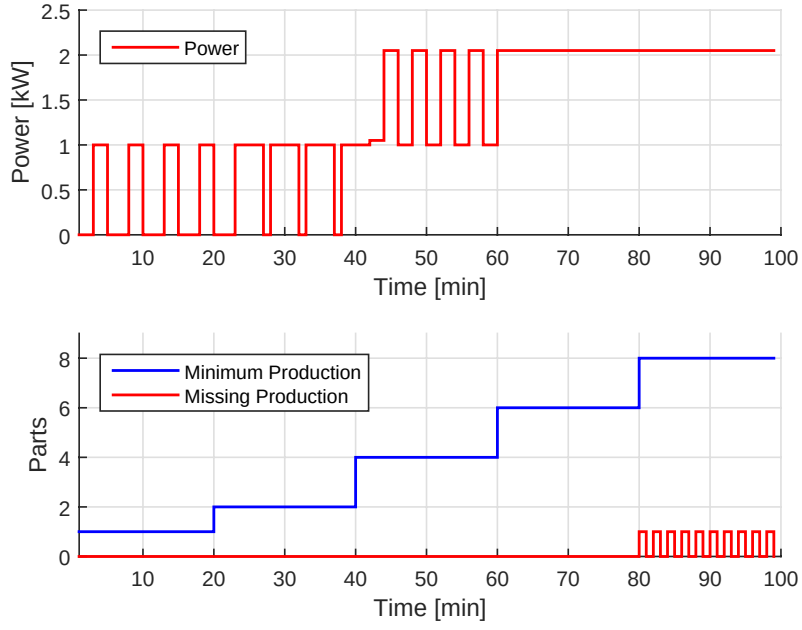


(b) Production and Absorbed Power.

Figure 3.12: Minimum production constrained problem with storage costs.

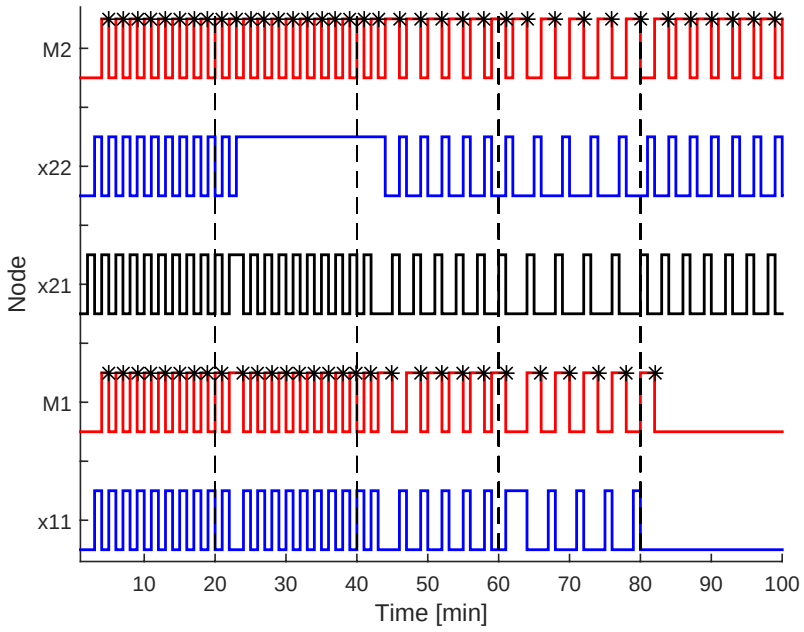


(a) Machine scheduling and buffer management. The black asterisks denote the end of the machining.

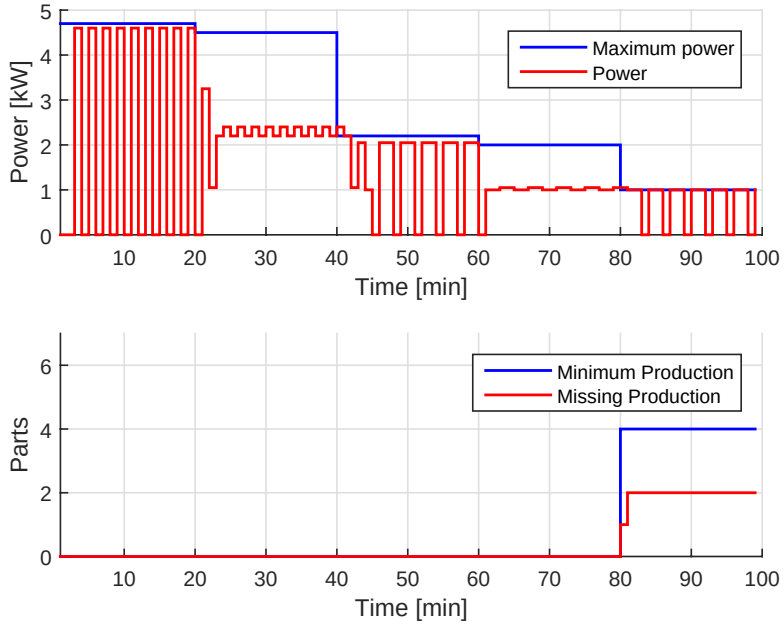


(b) Production requirement, missing production (bottom) and absorbed power (top).

Figure 3.13: Power Minimization using the enhanced machine model.

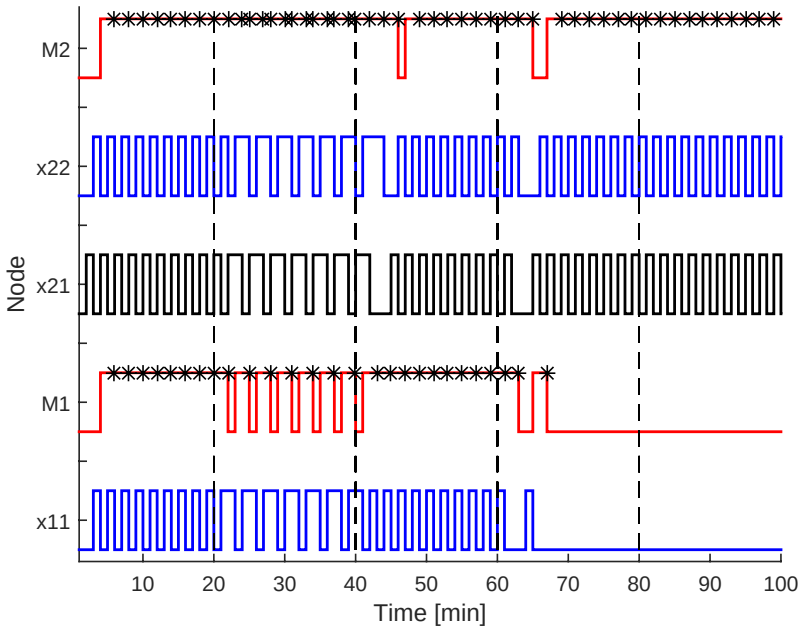


(a) Machine scheduling and buffer management. The black asterisks denote the end of the machining.

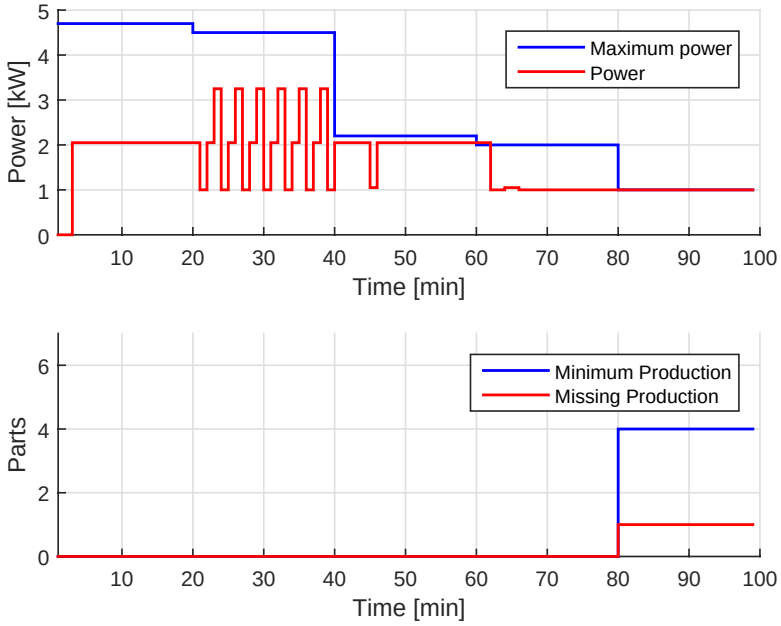


(b) Production and Absorbed Power.

Figure 3.14: Production maximization using the Simple Machine Model.



(a) Machine scheduling and buffer management. The black asterisks denote the end of the machining.



(b) Production and Absorbed Power.

Figure 3.15: Production maximization using the Enhanced Machine Model.

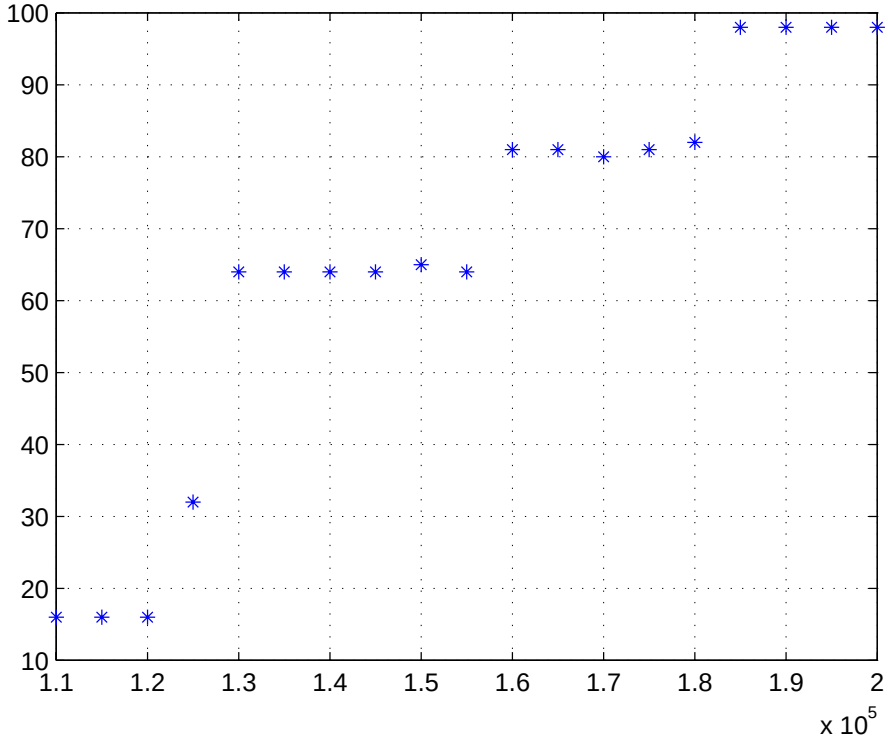
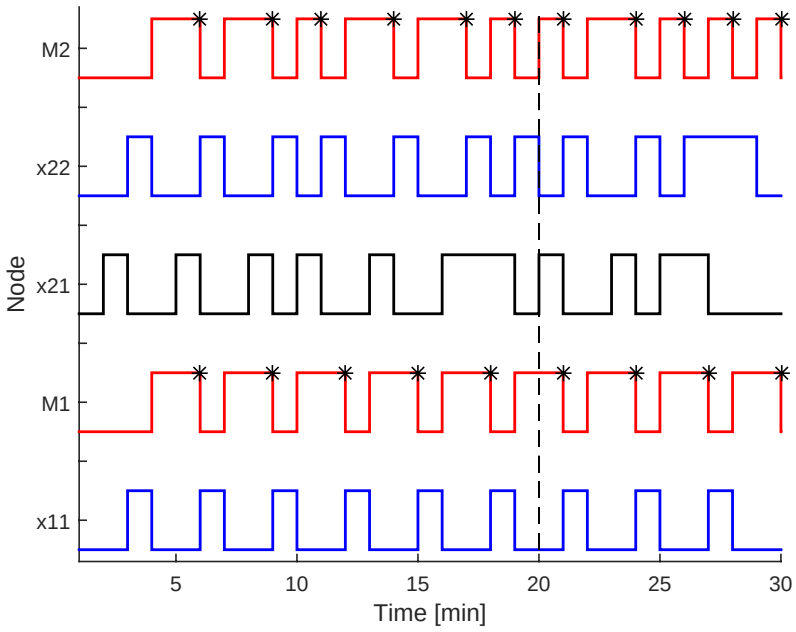
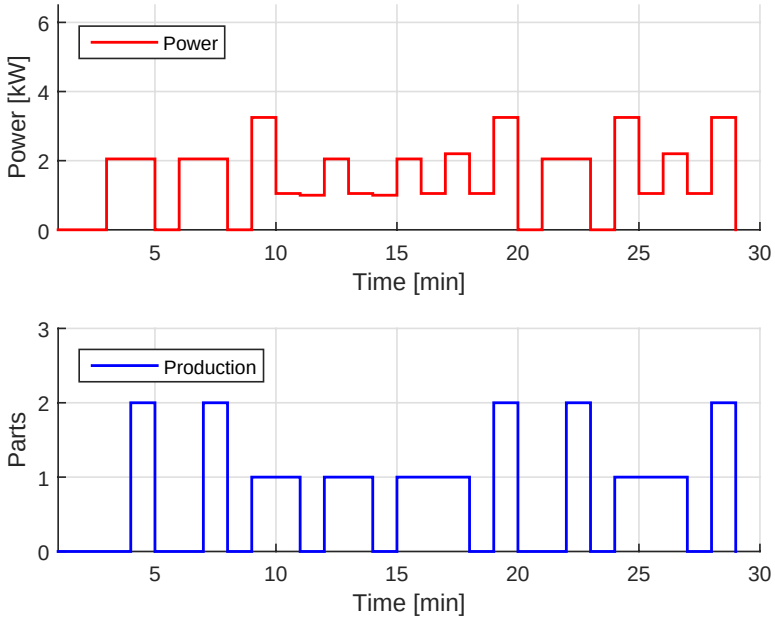


Figure 3.16: Produced parts.

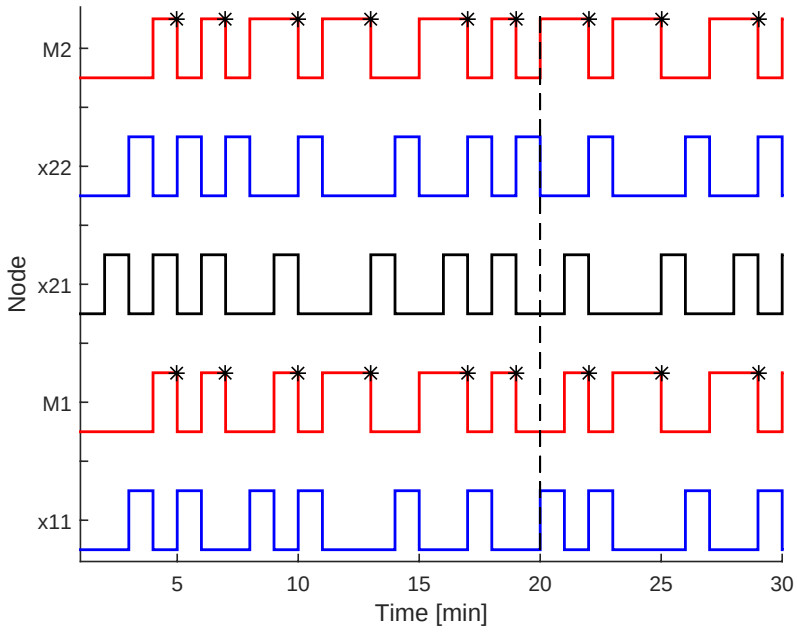


(a) Machine scheduling and buffer management. The black asterisks denote the end of the machining.

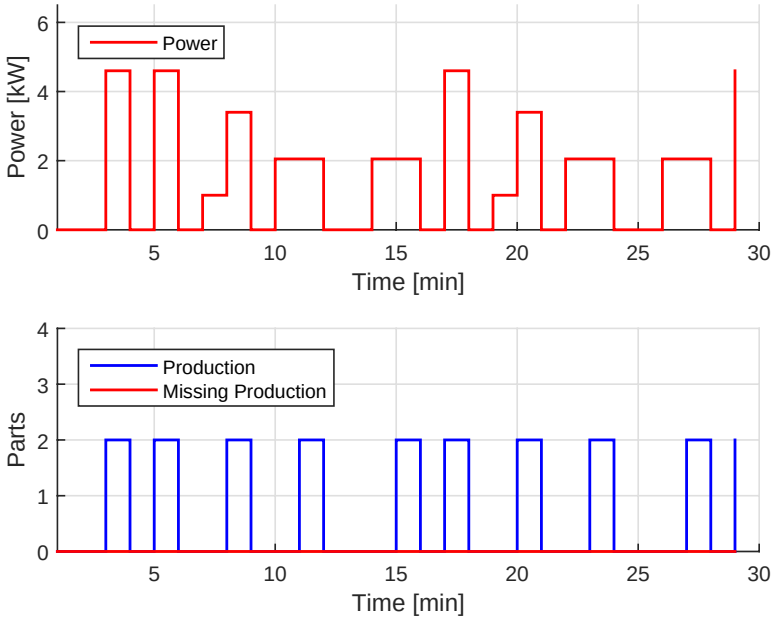


(b) Production and Absorbed Power.

Figure 3.17: Production maximization using MILP optimization.



(a) Machine scheduling and buffer management. The black asterisks denote the end of the machining.



(b) Production and Absorbed Power.

Figure 3.18: Production maximization using RH optimization.

Chapter 4

Automated storage / retrieval system

In this chapter, *MPC* is applied to a specific type of systems used in the advanced manufacturing field that is AS/RS [65]. The introduction of AS/RS improves inventory management and control, increases storage capacity and reliability, and reduces unnecessary labour costs. One of the main components of an AS/RS is the storage/retrieval crane, which is used to pick up and drop off items. Research shows that there are many ways to address issues related to the control of AS/RS, e.g. [98] mentions several methods for storage assignment and [13] focuses on the estimation of the travel time of the crane.

The complexity of control of AS/RS is related to the number of depots and the storage policy of the system. Many papers use a random storage policy, which allows a pallet to be stored randomly on any available storage location. Due to the flexibility of such a policy, the number of possible solutions enlarges, which causes a higher complexity to optimally solve the storage assignment problem. Many researchers solve this problem with heuristics [3, 34, 47, 78]. Recent research trend considers a new Petri nets approach to obtain modular and compact models for automated warehouse systems, based on the merging between hybrid Petri nets and colored Petri nets [5, 6]. A solution to problems with a dedicated storage policy is described by [44], where in contrary to a random storage policy, a dedicated storage policy predefines a unique storage location for each storage request. The paper [44] shows that this problem can be solved in polynomial time by using the fact that the crane always returns to a depot.

In this thesis a different approach is suggested to control AS/RS with a random storage policy. Since *MPC* has many advantages, see Section 1.1, it is

desirable to use it to control the *AS/RS* which is described in Section 4.1 together with its dynamical model and constraints detailed in Section 4.2. The complex application of *MPC* to *AS/RS* is overcame by describing the system in terms of a *MLD* system, detailed in Section 4.3. The transformation from logical propositions to an *MLD* system by means of the propositional calculus, allows the control problem can be expressed as a *MILP* problem. The combination *MLD-MPC* has been applied [11, 35, 83], but to our best knowledge, only few applications for discrete-event manufacturing systems have been reported [20, 22, 24], and never for *AS/RS*.
The case study used to evaluate the *MPC* performances concerns a laboratory stacker crane and the simulations results, presented in Section 4.1 and discussed in Section 4.4, illustrate the good performance of the control algorithm.
A topic for future work involves improving the way of modelling *AS/RS* by reducing the complexity of the model and decreasing the required computation time. The focus hereby should be on the reduction of the number of integer variables, since these determine the complexity of an integer linear programming problem. The current model can easily be extended to a form with multiple final storage nodes. Next to these improvements and extensions, it would be interesting to compare the *MLD-MPC* method to other control methods using *MPC*, such as time instant optimisation *MPC* [116] and heuristics.

Nomenclature

n	number of pallets
N	set of nodes
N_i	$i-th$ node
N_f	set of final storage nodes
N_s	set of source nodes
N_T	set of temporary storage nodes
S_i	set of adjacent nodes to N_i
p	type of pallet
P	set of pallet types
k	discrete time instant
Υ_i	$i \in \mathbb{N}$, crane discrete position when it is at the specific $i-th$ location
$P_i(k)$	state and position of the crane
$G_i(k)$	state of the node
$u_{ij}(k)$	command to move the crane from N_i to N_j
$v_i(k)$	command to load a pallet from the crane onto N_i
$w_i(k)$	command to load a pallet from N_i onto the crane
$D(k)$	customer demand
$\delta_i(k)$	Boolean auxiliary variable
x	$x \in \mathbb{R}$ and $x \in X$
X	a given bounded set
$U_i(k)$	sum of the command variable moving a pallet from a node N_i to an adjacent one
$f(\cdot)$	$f : \mathbb{R} \mapsto \mathbb{R}$
m	min of $f(\cdot)$
ε	very small positive number
δ	logical variable
X_i	literal

About *MLD-MPC*:

$x(k)$	state variables vector
$u(k)$	control actions vector
$\delta(k)$	Boolean auxiliary variables vector
$z(k)$	continuous auxiliary variables vector
$J(k)$	performance index
J_{node}	performance index used to give certain nodes priority
J_{crane}	performance index used to reduce the total time the crane is carrying a pallet
$J_{penalty}$	performance index used to penalize unnecessary crane movements

J_{pallet}	performance index used to satisfy as much as possible the customer demand
α_i	$i = 1, \cdots, 4$, performance index weights
N_{RH}	prediction horizon
$\gamma_i(k)$	Boolean auxiliary variable used to implement J_{node}
q_i	weight factor in J_{node}
$\eta(k)$	Boolean auxiliary variable used to implement J_{crane}
q_c	penalty factor for the movement of the crane used to implement $J_{penalty}$
q_p	penalty factor for the movement of the piston used to implement $J_{penalty}$
$\varsigma(k)$	Boolean auxiliary variable used to implement J_{pallet}

4.1 The automated storage / retrieval system

Description

The crane is located in the laboratory of the Electronics, Automation, and Bioengineering department of the Politecnico di Milano, Italy. It is a scale model of a real stacker crane, see Figure 4.1.

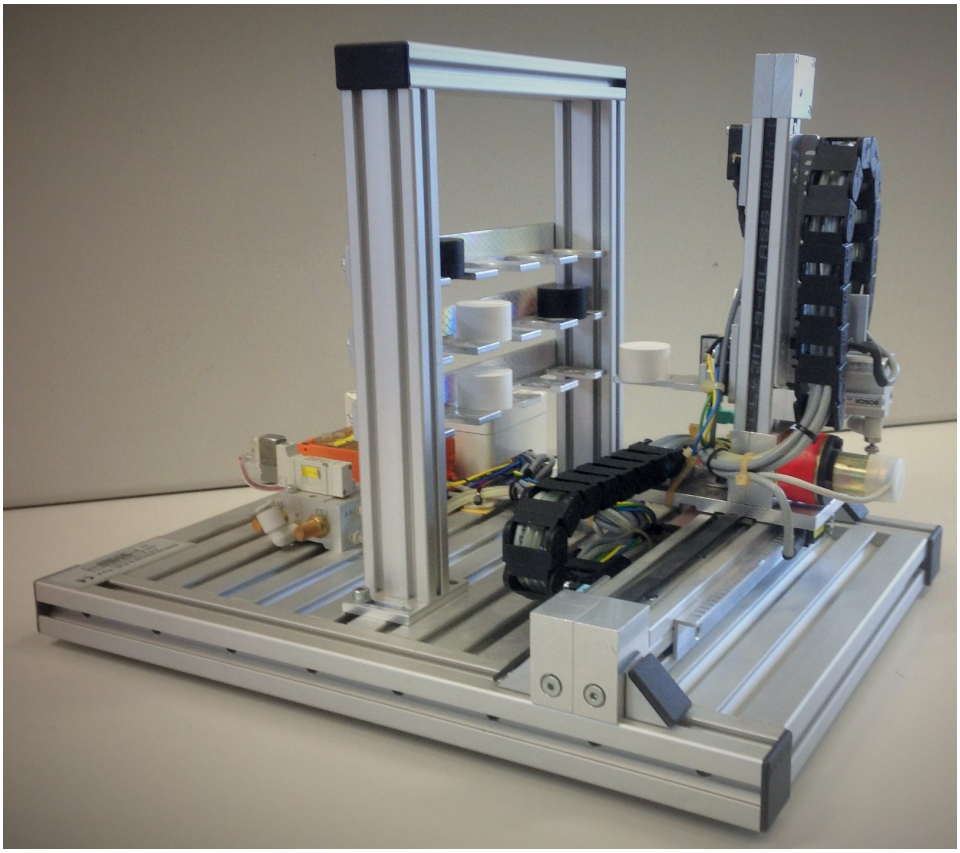


Figure 4.1: Stacker crane plant.

The system consists of two main components: the storage facility, a three-storey warehouse with four slot for each store level, and the crane.
The storage facility can store different type of cylindrical pallets at different locations (the types of pallet are identified by means of different colors).
The crane is based on two tracks, actuated by electrical motors, used to move the crane on x- and y-axes (respectively horizontal and vertical axes) of a Cartesian plane parallel to the warehouse structure and a pneumatic

piston which move an hand effector designed to manage the pallets.
The electrical motors are actuated by $24[V]$ electrical power so, by the switch the power supply it is possible to start the motor in one direction or in the opposite one. According to that, a specific software function has been implemented in MATLAB platform so to manage each motor power supply; in this way, by means of two different Boolean variables, it is possible to move each crane axes forward or backward.
The position of the crane with respect to each x- and y-axes is measured by means of a photo-sensors coupled to an optical linear bar so the measured position is obtained by acquiring the Boolean signals coming from the photo-sensor. This means that a specific software function, implemented in MATLAB platform, has been used to elaborate such Boolean signals and then calculate the crane position in terms of continuous variable.
The position of the end effector, pick position or back position, is measured by means of two proximity switches which give Boolean signals.
The presence of a pallet in a specific location of the warehouse is detected by means of a photo-sensor which gives Boolean signal.
In order to be able to design an advanced control system for the storage system independently from the scaled plant technology and physical implementation, a low level control system has been implemented in MATLAB platform. In particular specific low level control functionalities have been developed so that:

- Control actions
  - u_{ij} is the Boolean control variable designed to move the crane from one specific $i-th$ location to another specific $j-th$ one. This function has been implemented based on static tables;
  - v_i is the Boolean control variable designed to put a pallet into a specific $i-th$ location, that means it represents the control variable which actuate the pneumatic piston into the pick position;
  - w_i is the Boolean control variable designed to pick a pallet from a specific $i-th$ location, that means it represents the control variable which actuate the pneumatic piston into the back position;
- Measured variables
  - $\Upsilon_i, i \in \mathbb{N}$, is the variable designed to give the discrete position of the crane when it is at the specific $i-th$ location. As a matter of fact, the low level control function implemented to compute such variable

consists on the integration and combination of the Boolean signals coming from the crane axis photo-sensors;

Representation

The system so implemented can be modeled by a set of nodes N, see Figure 4.2. One of these nodes can be defined as the source node, $N_s \in N$, where new pallets arrive for storage. Another node can be defined as final storage node, $N_f \in N$, where the customers can pick up their order. When this node is filled, temporary storage places can be used to store the extra delivered pallets. The temporary storage places are represented by all nodes that are neither the source node nor the final storage node, resulting in the subset $N_T \subset N$ with temporary storage nodes.
In this analysis, a single-level-deep stationary storage facility is assumed, which means that the crane can directly reach the stored pallets. Solutions to multiple-levels-deep storage problems are discussed by [125] while this research will focus on single unit-load aisle-captive AS/RS.

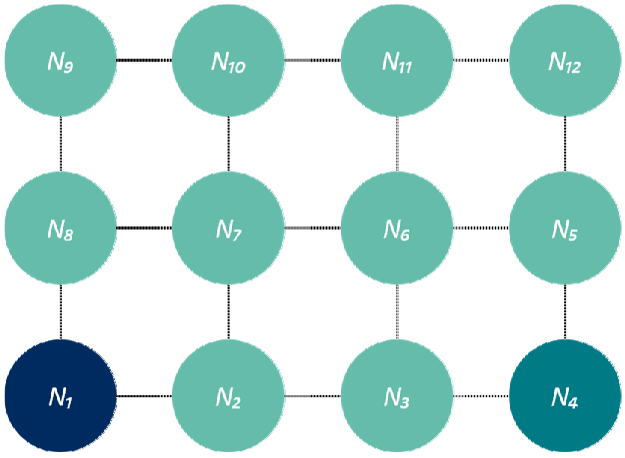


Figure 4.2: Schematic representation of the storage facility.

This means that there is one crane per storage facility that cannot leave its designated aisle, and that cannot carry more than one pallet at a time [98]. In this study, it is assumed that at each event step the crane can only travel to an adjacent node. Therefore for every node N_i, a set $S_i \subset N$ of adjacent nodes is defined, e.g. $S_3 = \{N_2, N_4, N_6\}$ according the model depicted in Figure 4.2.

Moreover the different types of pallets can be defined as $p \in P$. It is defined P as follows: $P = \{2, \cdots, |P|+1\}$, where $|P|$ is the number of different pallet types handled by the system, since the values of 0 and 1 will be used in the model to define the absence of the crane in front of a node and the absence of a pallet on the crane, respectively. It is assumed that every pallet has the same final storage node $N_{\mathrm{f}} \in N$.

4.2 Dynamic model

In this Section the dynamic model of the AS/RS will be described. First the state and decision variables are defined and then the dynamic equations and constraints are derived.

4.2.1 State variables

For each node $N_i \in N$, with $i \in \{1, \ldots, n\}$ (i.e. $n = |N|$ number of pallets) two state variables can be defined to describe the system.
The first variable is $P_i(k)$, which represents the state and the position of the crane (i.e. does the crane carry a pallet or not and is the crane in front of node N_i or not). Since the type of pallet is not detectable because none suitable sensor is available in the scaled plant, then the crane position variable Υ_i is not directly usable in terms state variable so, combined with the internal dynamic model of the system, it is just used to compute $P_i(k)$.
The second state variable, $G_i(k)$ represents the state of the node (i.e. does node N_i store a pallet or not). For the same reason such that the type of pallet is not detectable, the Boolean signal coming from the photo-sensor which detects the pallet presence into a specific location is combined with the internal dynamic model of the system so to compute $G_i(k)$. In particular the Booleand signal is used to verify that in the real plant the pallet is effectively stored in a location so to avoid discrepancies between the real plant and its dynamic internal model.

These variables are defined in the following way

$$P_i(k) = \begin{cases} p & \text{, if the crane is at } N_i \text{ with a pallet type } p \\ 1 & \text{, if the crane is at } N_i \text{ without a pallet} \\ 0 & \text{, otherwise} \end{cases}$$

$$G_i(k) = \begin{cases} p & \text{, if } N_i \text{ stores a pallet of type } p \\ 0 & \text{, otherwise} \end{cases}$$

4.2.2 Decision variables

The following decision variables can be defined to represent, together with the state variables, the dynamical model of the AS/RS

$$u_{ij}(k) = \begin{cases} 1 & \text{, the crane moves from } N_i \text{ to } N_j \\ 0 & \text{, otherwise} \end{cases}$$

$$v_i(k) = \begin{cases} 1 & \text{, a pallet is loaded from the crane onto } N_i \\ 0 & \text{, otherwise} \end{cases}$$

$$w_i(k) = \begin{cases} 1 & \text{, a pallet is loaded from } N_i \text{ onto the crane} \\ 0 & \text{, otherwise} \end{cases}$$

Such variables match with the control actions available for the scaled plant as mentioned in Section 4.1.

4.2.3 Dynamic equations

As already mentioned in Section 4.1, assuming one movement per event step, the following explicit discrete state space model can be defined, which represents the states of the system at event step $k+1$

$$P_i(k+1) = P_i(k) - \sum_{j \in S_i} u_{ij}(k)P_i(k) + \sum_{j \in S_i} u_{ji}(k)P_j(k) - v_i(k)[P_i(k)-1] + w_i(k)[G_i(k)-1] \quad (4.1)$$

$$G_i(k+1) = G_i(k) + v_i(k)P_i(k) - w_i(k)G_i(k) \quad (4.2)$$

Equation (4.1) shows that there are four different events that can change the state $P_i(k)$ of the system to a different state in the next time step, $P_i(k+1)$:

- The crane moves from N_i to N_j,
  thus $P_i(k) \geq 1$, $P_j(k) = 0$, and $u_{ij(k)} = 1$;
- The crane moves from N_j to N_i,
  thus $P_i(k) = 0$, $P_j(k) \geq 1$, and $u_{ji}(k) = 1$;
- The piston loads a pallet from the crane onto N_i,
  thus $P_i(k) \geq 2$, and $v_i(k) = 1$;
- The piston loads a pallet from N_i onto the crane,
  thus $P_i(k) = 1$, and $w_i(k) = 1$.

The correctness of (4.1) can be illustrated as follows. If at event step k the crane stays where it is and a pallet is loaded from the crane onto node N_i, (4.1) will reduce to $P_i(k+1) = P_i(k) - v_i(k)[P_i(k)-1]$ since all the other terms of the dynamic equation are equal to zero. As the crane is in front of N_i and holds a pallet of type p, $P_i(k) = p$. Moving the pallet onto the node requires $v_i(k) = 1$, which, according to the dynamic equation, results in the next state $P_i(k+1) = 1$. This means that at event step $k+1$ the crane is still in front of node N_i but does not hold a pallet, which is in line with the definition of $P_i(k)$ given at the start of Section 4.2. Equation (4.2) is influenced in a similar way. If the crane moves and the piston does not, the state of the nodes do not change: $G_i(k+1) = G_i(k)$. However, if the piston moves, the state of the node changes.

4.2.4 Deliveries and demands

For the source node and final storage node, dynamic equation (4.2) applies, but some extra rules need to be regarded.
The arrivals of new pallets at the source node are predefined e.g. with a uniform distribution. Per event step, maximal one pallet arrives. When node N_s is empty, i.e. $G_1(k) = 0$, a new pallet arrives at N_s and $G_1(k+1) =$

p, depending on the type of pallet. When $G_1(k) \neq 0$, dynamic equation (4.2) applies to determine $G_1(k+1)$.
The customer demand D, which is chosen uniformly distributed over P, influences the final storage node N_f. When the demand is satisfied, the state of final storage node N_f, at even step k, is $G_\mathrm{f}(k) = D(k)$; next, the pallet will be removed from the final storage node by the customer and thus $G_\mathrm{f}(k+1) = 0$. When the customer demand is not satisfied, G_f will be calculated according to dynamic equation (4.2).

4.2.5 Constraints

The following constraints describe the restrictions of the system and apply to every $N_i \in N$:

- Only one event per time step k may occur

$$\sum_{j \in S_i} u_{ij}(k) + \sum_{j \in S_i} u_{ji}(k) + v_i(k) + w_i(k) \leq 1 \tag{4.3}$$

- If the crane is not at node N_i, the crane cannot move from a node N_i to node N_j

$$P_i(k) = 0 \rightarrow \sum_{j \in S_i} u_{ij}(k) = 0 \tag{4.4}$$

- A pallet cannot be loaded from the crane onto node N_i if the crane is not positioned in front of N_i, carries no pallet, or the node carries already a pallet

$$(P_i(k) \leq 1) \vee (G_i(k) \geq 1) \rightarrow v_i(k) = 0 \tag{4.5}$$

  where $\vee$ denotes the disjunction;

- A pallet cannot be loaded from node N_i onto the crane if the crane is not in front of N_i, already carries a pallet or N_i carries no pallet

$$(P_i(k) \neq 1) \vee (G_i(k) = 0) \rightarrow w_i(k) = 0 \tag{4.6}$$

Note that (4.4), (4.5) and (4.6) result in non-linear constraints. However, these constraints can be reformulated in a linear way by the use of propositional calculus [26, 94]. In this way auxiliary variables are created, as will be illustrated next.

Example

By adding Boolean auxiliary variables to the model, $\delta_i(k)$, (4.4) is reformulated, resulting in the following expressions

$$P_i(k) = 0 \rightarrow \delta_i(k) = 1$$
$$\delta_i(k) = 1 \rightarrow \sum_{j \in S_i} u_{ij}(k) = 0$$

For the ease of reading, from now on, in this example, $\sum_{j \in S_i} u_{ij}(k) = U_i(k)$.
The following two proposition rules are used to translate the constraints [12].

Proposition 1

Assuming that $x \in X$, where X is a given bounded set, $m = \min_{x \in X} f(x)$ and ε a very small positive number, leads to the following statement
$[f(x) \leq 0] \rightarrow [\delta = 1]$ if and only if $f(x) \geq \varepsilon + (m - \varepsilon)\delta$.
Let a literal X_i represent a statement which is either true or false, e.g. $x \geq 1$.
One can associate to a literal X_i a Boolean (auxiliary) variable $\delta_i \in \{0, 1\}$.
If X_i is true, $\delta_i = 1$, and $\delta_i = 0$ otherwise.

Proposition 2

The following expressions and linear constraint can be seen to be equivalent
$X_1 \rightarrow X_2$ is equivalent to $\delta_1 - \delta_2 \leq 0$.
Since $\min_k P_i(k) = 0$, $P_i(k) = 0 \rightarrow \delta_i(k) = 1$ is equivalent to $P_i(k) \leq 0 \rightarrow \delta_i(k) = 1$. Note that the latter formulation is now in the right form to use with the propositional rule of Proposition 1. This results in the first set of constraints that need to be added to the model

$$P_i(k) \geq \varepsilon - \varepsilon\delta_i(k)$$

Note that $m = 0$.
Next, the rule of Proposition 2 is used to add $\delta_i(k) = 1 \rightarrow U_i(k) = 0$ to the model. This results in the following constraint

$$\delta_i(k) + U_i(k) \leq 1$$

It is left to the interested reader to derive the other constraints with the use of propositional calculus.

4.3 MPC formulation

The dynamic model of the AS/RS described in the previous Section, can be translated into a *MLD* formulation using propositional calculus as explained above. As already discussed in Section 2.5, the *MLD* formulation has the following form

$$x(k+1) = Ax(k) + B_u u(k) + B_\delta \delta(k) + B_z z(k) \tag{4.7a}$$
$$y(k) = Cx(k) + D_u u(k) + D_\delta \delta(k) + D_z z(k) \tag{4.7b}$$
$$E_\delta \delta(k) + E_z z(k) \leq E_u u(k) E_x x(k) + E \tag{4.7c}$$

where $x(k)$ is the vector of state variables, $u(k)$ the vector of control actions, with elements $u_{ij}(k)$, $\delta(k)$ is the vector of Boolean auxiliary variables and $z(k)$ is a vector of continuous auxiliary variables.

The performance index J is defined at event step k as

$$J(k) = \alpha_1 J_{\text{node}}(k) + \alpha_2 J_{\text{crane}}(k) + \alpha_3 J_{\text{penalty}}(k) + \alpha_4 J_{\text{pallet}}(k) \tag{4.8}$$

where $\alpha_1, \alpha_2, \alpha_3$, and α_4 are weight factors. The objective function consists of the following parts:

- J_{node} can be used to give certain nodes priority over others to load or unload them faster, e.g. it is desired to keep the source node empty for new pallets that arrive;
- J_{crane} is used to reduce the total time the crane is carrying a pallet. This is to avoid the system storing a pallet on the crane instead of loading it into the node;
- J_{penalty} penalises unnecessary crane movements;
- J_{pallet} tries to satisfy the customer demand as soon as possible.

The performance index (4.8) must be minimized under the constraints (4.3)-(4.6). Once the sequence of optimal control actions has been computed over a prediction horizon N_{RH}, according to the *RH* approach, only the values of the first event step are applied. The overall procedure is repeated at the next event step.

Objective J_{node} can be used to define preferences for certain nodes. To do so, the following Boolean auxiliary variable is used

$$\gamma_i(k) = \begin{cases} 1 & \text{, if a pallet is present in node } N_i \\ 0 & \text{, otherwise} \end{cases}$$

By multiplying this variable by a weight factor, the system will try to store pallets in nodes with a lower weight factor rather than a higher weight factor, when minimising the performance index. This formulation is represented by

$$J_{\text{node}}(k) = \sum_{t=1}^{N_{RH}} \sum_{i=1}^{n} q_i \gamma_i(k+t-1),$$

where q_i is the weight factor corresponding to node N_i. As described before, J_{crane} is used to unload a pallet from a node instead of keeping it stored on the crane. By defining $\eta(k)$, a Boolean auxiliary variable, considering the presence of a pallet on the crane

$$\eta(k) = \begin{cases} 1 & \text{, if the crane holds a pallet} \\ 0 & \text{, otherwise} \end{cases}$$

this results in

$$J_{\text{crane}}(k) = \sum_{t=1}^{N_{RH}} \eta(k+t-1)$$

To save energy, it is desired to minimise the total amount of crane movements. This is represented by J_{penalty}. Defining a penalty of q_{c} and q_{p} for movements of the crane and piston respectively, results in

$$J_{\text{penalty}}(k) = \sum_{t=1}^{N_{RH}} \Big[q_{\text{c}} \sum_{(i,j)\in N\times N} u_{ij}(k+t-1) + q_{\text{p}} \sum_{i=1}^{n} \big(v_i(k+t-1) + w_i(k+t-1) \big) \Big]$$

The definition of the customer satisfaction in J_{pallet} will be elaborated next. It is desired to satisfy the customer demand as soon as possible. If the pallet

type present in final storage node N_f is equal to the requested pallet type by the customer $D(k)$, it is set the Boolean auxiliary variable $\varsigma(k) = 1$

$$\varsigma(k) = \begin{cases} 1 & \text{, if } D(k) = G_\mathrm{f}(k) \\ 0 & \text{, otherwise} \end{cases}$$

This can be transformed into a linear form using propositional calculus. The following objective function for the customer satisfaction is proposed

$$J_\mathrm{pallet}(k) = -\sum_{t=1}^{N_{RH}} \varsigma(k+t-1)$$

It is desired to satisfy the customer as soon as possible, which means that the term $\sum_{t=1}^{N_{RH}} \varsigma(k+t-1)$ must be maximised. Since the total objective function $J_\mathrm{pallet}(k)$ will be minimised, the objective function is described by the negative sum of $\varsigma(k)$.
The objective function and the *MLD* model of the system have now been transformed into a integer linear programming problem, which can be optimised using adequate solvers.

4.4 Case study

A case study is performed on a laboratory stacker crane to test the *MPC* application on an *AS/RS*. This Section first summarizes the system used and then discusses the simulation results.
Since the system does not include elements with a partially uncertain behaviour, like the one of M_2 in the de-manufacturing process described in Chapter 2, the simulation results obtained with a reliable model of the *AS/RS* are highly significative.

4.4.1 Laboratory stacker crane

As mentioned in Section 4.1, the storage facility can store pallets on three different levels, where each level has four places available for storage. This means that the whole storage facility can take up to twelve pallets (i.e. $n = 12$), see Figure 4.1 and its representation in Figure 4.2. Here N_1 is the source node where the pallets, that need to be stored, are delivered. Node N_4 is defined as the final storage node, where customers can come

to pick up their order. There are three different types of pallets used, so $P = \{2,3,4\}$.
The temporary storage places are represented by all nodes that are not the source node nor the final storage node. This results in: $N_\mathrm{s} = N_1$ is the source node, $N_\mathrm{f} = N_4$ is the final storage node and $N_\mathrm{T} = \{N_2, N_3, N_5, N_6, N_7, N_8, N_9, N_{10}, N_{11}, N_{12}\}$ is the set with temporary storage nodes.

4.4.2 Weight factors

It is desired to remove a pallet from the source node N_1 as soon as possible and therefore q_1 is set higher than the other q_i weight factors of the temporary storage nodes. Next to that, it is preferable to keep the final destination node, N_4, empty as long as an order cannot be satisfied. Therefore it is decided to set $q_1 = 1$ and $q_4 = 1$ and to set $q_i = 0, i \in N_\mathrm{T}$, since there is no preference in loading or unloading certain temporary storage.
Since the crane movements consume significantly more energy than the piston movements, the weight factor of the crane movement should be set higher than the other and therefore $q_\mathrm{c} = 2$ and $q_\mathrm{p} = 1$ are chosen.
The weight factors $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ are set according to importance of the different objectives. Since customer satisfaction has priority number one, α_4 has a higher value than the other factors. Second most important is the satisfaction of J_node and, on a shared third place, come J_crane and J_penalty. The chosen values for the weight factors are $\alpha_1 = 1$, $\alpha_2 = 0.12$, $\alpha_3 = 0.1$ and $\alpha_4 = 10$. The prediction horizon N_{RH} has been set to 10.

4.4.3 Simulation results

A basic example of the evolution of the system over time is given in Figure 4.3.

For the ease of understanding, here only four nodes of the whole system are depicted, including source node N_1 and final storage node N_4. At event step $k = 1$ the customer at node N_4 demands a pallet of type 4, i.e. $D(1) = 4$. At this time, the variable $\varsigma(1) = 0$ since node N_4 is empty and thus the performance index can be optimised by making $\varsigma(k) = 1$ with k as small as possible. At the same time, i.e. $k = 1$, there is a pallet of the demanded type available at the source node N_1, so the optimised control input sequence allows managing the pallet movement step by step over time to final storage node N_4, so that the customer can pick up his order. Note that the crane first moves one node per time step and then the piston moves to place the pallet

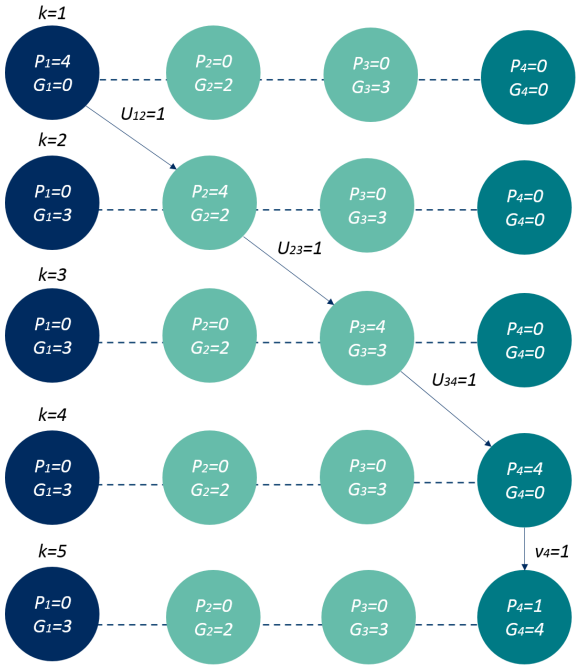


Figure 4.3: Example of state transitions.

in the storage facility (i.e. $v_4(4) = 1$).
Differently from the example of Figure 4.3, if a pallet type p requested from the customer is already stored in a temporary storage node, the *MPC* algorithm will control its movement towards the final storage node N_4. Moreover, in case a pallet p type is requested by the customer and two pallets of the same type are stored in a temporary storage node and in the source node N_1 respectively, then the *MPC* algorithm will menage to pick up the pallet from node N_1 since the weight $q_1 = 1$ and the weight related to temporary storage node N_i is $q_i = 0$. By picking up the pallet from the source node, the final performance index value will be optimised.
Finally, if no pallet is available to satisfy the customer demand, neither in the source node nor in any temporary storage node, then the *MPC* algorithm will not implement any pallet movement even if the performance index value increases over time due to the unsatisfied customer demand at final storage node N_4. In such a case, as soon as a pallet of the requested type p is placed into the source node N_1, the control will manage its movement as previously discussed.

In order to implement the simulation experiments, the *MLD* model and the *MPC* formulation together with the objective function, are translated into an integer linear programming problem. The control problem is implemented in the YALMIP [74] modelling language so simulations experiments have been performed.
Figure 4.4 shows the pallet movements conducted from simulation. The horizontal axis shows the events steps while the vertical axis represents all the node numbers. The graph shows, for every event step, in which node a pallet is present. By connecting the positions of a pallet over time, the lines in the graph are created. Note that, when a pallet is moved from one node to another, a linear line is depicted in the graph even though the crane does not necessarily travel in a linear line. Each pallet type is represented by a different colour.

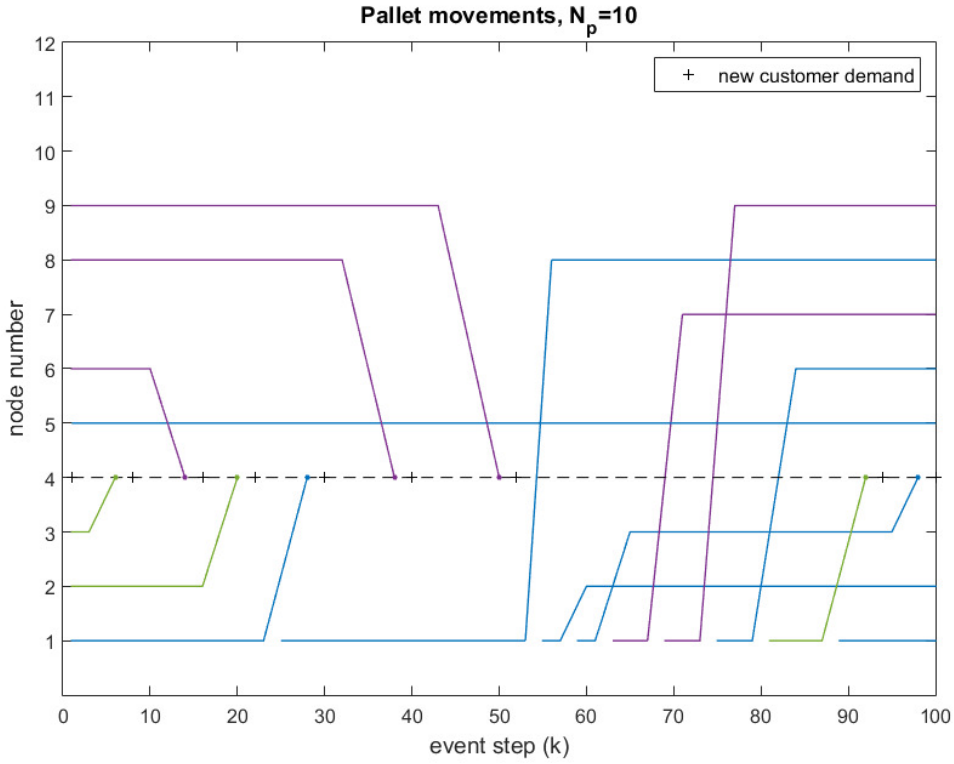


Figure 4.4: Simulation results: movement of the pallets.

The graphs shows that at event step $k = 0$ there are pallets present in nodes $N_1, N_2, N_3, N_5, N_6, N_8$ and N_9. The first pallet movement takes place from

node N_3 to N_4, to satisfy the customer demand. It can also be noted that after a pallet is removed from source node N_1, this node is soon refilled with a new pallet that arrives from outside the system. Based on the type of pallet demanded by the customer at node N_4, the control algorithm finds the optimal movements of the pallets through the system according to the pallets present in the system and the objective function formulation.

Chapter 5

An energy consumption evaluation methodology for manufacturing plants

The need to limit the CO_2 emissions [37,85], resize the factory energy supply infrastructure and minimize the energy consumption, represents a factor that leads to equalize global living standards at the level of the industrialized regions [102] and to create new perspectives on energy efficiency in business decisions [33,54], besides saving plant installation and production costs.
In order to design and manage energy efficient factories [16,82], manufacturing companies require tools [27,36,50,52] for the prediction and computation of the energy consumption of process equipments. Interesting reviews of the main approaches proposed so far can be found in [56, 104]. Many of these methods are based on Discrete Event System (*DES*) simulation, see e.g. [108, 120]. In particular, the widely popular approach proposed in [120] introduces the concept of "Energy Block", i.e. the specific energy consumption behavior that a machine can assume in its operating states, like "turned-off", "start-up", "warm-up", "stand-by", "processing" or "stopping". By associating to each operating state an energy consumption pattern, identified by a power profile, it is possible to compute the overall energy consumption of the machines in different operating conditions. Further extensions of this approach have been reported in [107, 108], where the plant auxiliary systems have also been considered. In [41, 48, 87] the operating states of the process equipments and the associated energy consumption have been modeled in terms of *FSM*, see [19,53,69], a formalism suitable for dynamic simulation. A refined approach has been implemented

in [25], where three different aspects of process equipments are taken into account, namely the mechanical, the logic control algorithms and the energy characteristics. While the mechanical behavior is modeled by means of *DES*, the logic control and energy aspects are described in terms of *FSM*. In any case, it must be recalled that the accurate knowledge of many plant parameters, which can be directly measured, computed, or derived from technical nominal data, is fundamental for the proper computation of the energy consumption of the machines, see [18, 60, 61].
A drawback of the approaches based on the use of energy blocks can be due to their limited flexibility and scalability properties. In fact, the number of operating states and the associated required power profiles, are in general variable and depend on the specific product to be processed in terms of the machining operations and production process technology, the material to be machined, the topology of the system and the adopted control logic. Therefore, when these conditions change, it can be difficult to compute, or reliably estimate, the energy consumption and the power peak loads. Notably, this information would be very useful both to the plant designers and to the management engineers: for the first ones to optimize the factory layout still during the plant design workflow phase while for the second ones to optimize the on-line control of the production system according to specific performance indexes.
Motivated by the above reasons, in Section 5.1 a *DES*-based approach for the computation of the energy consumption of discrete systems is proposed, i.e. systems where the energy consumption is mainly due to the on / off switching of the actuators governed by the control logic. In the proposed *CSM* method, the modeling phase includes both the mechanical behavior of the system and the analysis of the actuators' characteristics, typically electric motors or pneumatic actuators, in terms of their absorbed power. In this way, the resulting *DES* model is suitable to dynamically describe the energy consumption due to the logic state (on / off) of each actuator managed by the emulated plant control system, as described in Section 5.2. Then, based on the actuators' absorbed power parameters specified in terms of either field measurements or nominal data, the instantaneous power required by each machine [4, 81] and by the whole production plant is computed. It follows that the evaluation of the energy behavior is largely independent of the factors defining the operating states of the machines and of the adopted control strategy, since it is based on the effective power instantaneously absorbed by the active actuators [99]. On the other hand, *CSM* does not consider many basic functions of a process equipment, like lubrication, chip removal, tool changing and so on, which can often dominate the energy re-

quirements, see [45].
The potentialities of *CSM* have been tested on the pallet transport line of the de-manufacturing pilot plant [29, 30, 84] showed in Figure 2.1, where the main energy consumption is due to the activation of the actuators moving the pallet from a transport module to an adjacent one. In order to implement and validate *CSM* in the considered test case, *DCPIP* has been used. Then, the pallet transport line has been modeled into the SIMIO *DES* platform [58] interfaced to the *DCPIP*. Finally, the *DCPIP* has been connected to the pilot plant itself and the measured power effectively absorbed by the system has been compared to the value computed in simulation. The results achieved show that *CSM* is able to provide an accurate prediction of the power absorbed by the transport line and of the overall energy consumption of the system, as shown in Section 5.3.
With *CSM* it is possible to compute the plant power and energy consumption during the production system design phase. This allows to design the control strategies focusing on the minimization of the energy consumption and the absorbed power peaks. In this way, the value of the peaks can be maintained under given thresholds with limited plant energy costs [106] and consequently the plant power supply system can be appropriately sized.
Future developments could concern the testing of *CSM* in different and meaningful test cases. In addition, the characteristics of *CSM* will be generalized to allow it application to wider classes of manufacturing systems. To this end, a standardized simulation library of process equipments could be developed to support the engineers involved in the plant design activities.

Nomenclature

M_1	load/unload robot cell
M_2	testing machine
M_3	reworking machine
M_4	discharge machine
T_n	$n-th$ transport module
$\Upsilon_{i,j}$	$j-th$ position of the $i-th$ transport module
$\mathscr{A}_i$	nominal absorbed power of the $i-th$ actuator
$\mathscr{B}_i$	energy consumption of the $i-th$ actuator belonging to a stacker crane
$\mathscr{C}_i$	Boolean control variable of the $i-th$ actuator
$\mathscr{D}_i$	equivalent electric energy consumption of the $i-th$ actuator belonging to a stacker crane
$\mathscr{E}$	total energy consumption for a device or plant
$\mathscr{E}_{T_n}$	energy consumption of the $n-th$ transport module
$\mathscr{F}_i$	energy consumption of the $i-th$ actuator belonging to a pneumatic piston block
$\mathscr{G}_i$	equivalent electric energy consumption of the $i-th$ actuator belonging to a pneumatic piston block
$\mathscr{P}$	total absorbed power for a device or plant
$\mathscr{P}_i$	instantaneous absorbed power of the $i-th$ actuator
$\mathscr{P}_{T_n}$	instantaneous absorbed power of the $n-th$ transport module
$f_j(\cdot)$	pulse function in the $j-th$ period of time

5.1 The CSM energy consumption methodology

The basic idea of *CSM* consists of associating a specific power profile to each actuator driven by its logic control action and considered in the *DES* model of the process equipment. Specifically, *CSM* can be summarized as follows:

- Assume to have n actuators whose state (switched on / off) can be modified at fixed and synchronous time intervals Δt. Usually Δt also corresponds to the adopted simulation step time;
- Letting t be the continuous-time index, for each actuator define by $\mathscr{C}_i(t)$ the Boolean control variable corresponding to the activation/deactivation command, i.e. $\mathscr{C}_i(t) = 1$ if the $i-th$ actuator is working at time t while $\mathscr{C}_i(t) = 0$ if the actuator is in idle;
- Denote by $\mathscr{A}_i$ the (known) nominal absorbed power of the $i-th$ actuator in working conditions. For simplicity we consider here $\mathscr{A}_i$ as constant, although this assumption could be easily relaxed to consider time varying power profiles or modulating control actions;
- Compute the instantaneous absorbed power of the $i-th$ actuator at time t as

$$\mathscr{P}_i(t) = \mathscr{A}_i \cdot \mathscr{C}_i(t) \tag{5.1}$$

and the total absorbed power as

$$\mathscr{P}(t) = \sum_{i=1}^{n} \mathscr{P}_i(t) \tag{5.2}$$

- Compute the total energy consumption at any time $t = k\Delta t + \tau$, $k = 0,1,2,\ldots$, with τ belonging to the interval $[0,t)$, as

$$\mathscr{E}(k\Delta t + \tau) = \mathscr{E}(k\Delta t) + \mathscr{P}(k\Delta t) \cdot \tau \tag{5.3}$$

Concerning this procedure, some remarks are in order. First, from (5.2) it is apparent that small size actuators can be neglected, with significant advantages in terms of modeling effort. However, some care must be placed in removing actuators which, although characterized by a small instantaneous absorbed power, remain active for long periods of time. In fact, their contribution to the total energy consumption could be significant, as apparent from the integral nature of (5.3).

A second consideration concerns the power profile $\mathscr{A}_i$. As already noted, and extensively discussed in [27], the machine absorbed power profile depends on many factors, such as the specific operating machine technology, the material to be machined and the product to be produced in terms of machining process. Therefore, it would be useful to design a software data structure able to set the most suitable instantaneous power value to be assigned to $\mathscr{A}_i$ at each simulation step. In this regard, nominal data taken from the technical data-sheets, or measured from the field, can be used, see [60]. In general, compared to the approaches requiring the definition of the energy states of the process equipment, *CSM* is simpler, since the description of the control behavior is often already available from the control engineers who design the control system. In addition, as clearly expressed by (5.1) and (5.2), the estimate of the maximum power peak is immediate. This implies that, in case of modified control sequences, there is no need to reanalyze the system from the point of view of new energy states, which in turn would require additional field measurements and the power profile reformulation. This is very important in the industrial production process design to guarantee flexibility to the plant designer and to the control engineers. Indeed, with *CSM* it is easy to design and simulate the process plant, and then to evaluate the total absorbed power maximum peak according to the defined plant control policy. In addition, it is possible to modify the plant layout and the control system functionalities in order to limit the plant power requirements, so obtaining significant savings in the production system costs both in terms of electrical power supply infrastructure and of energy purchasing [106].

5.2 The application of CSM to a de-manufacturing plant

According to the de-manufacturing pilot plant description, see Section 2.1, the nominal absorbed power of the actuators used onto the transport modules is listed in Table 5.1.

Actuator	*Nominal absorbed power* ($\mathscr{A}_i$)	*Equivalent electric energy* ($\mathscr{D}_i$)
M_Tr_F/B	$64[W]$	–
$M\_Sc1/2\_Tr\_L/R$	$64[W]$	–
$Ev\_Sc1/2\_U/D$	$1.3[W]$	$1.9[J]$
Ev_P	$1.3[W]$	$0.8[J]$

Table 5.1: Actuators nominal absorbed power.

The data showed in Table 5.1 concerning the pneumatic actuators $Ev\_Sc1/2\_U/D$ and Ev_P deserve some comments. First, it must be noted that for these elements, the energy required by the auxiliaries, i.e. the compressor, must be considered, see [108], [107], [115]. Specifically, the $Ev\_Sc1/2\_U/D$ actuator remains in the Switched-On state for about $1s$, that represents the interval of time needed to complete the stacker crane up or down movement. This means that for each movement the $Ev\_Sc1/2\_U/D$ energy consumption $\mathscr{B}_i$ is

$$\mathscr{B}_i = 1.3 \cdot 1 = 1.3[J] \tag{5.4}$$

Moreover, also the compressed air consumed by $Ev\_Sc1/2\_U/D$ must be considered. According to [87, 107] the required energy can be translated into equivalent electrical power as follows. The pneumatic cylinder that moves up and down the stacker crane has a diameter equal to $30[mm]$ and a length equal to $10[mm]$, which corresponds to a volume of air equal to $7[ml]$ at the atmospheric pressure. From [99], the equivalent electric energy required by the compressor, used in the de-manufacturing plant, to compress $1[Nl]$ of air to a pressure of $7[bar]$ is $275[\frac{J}{l}]$. This means that the equivalent electric energy $\mathscr{D}_i$ required to move up the stacker crane is equal to

$$\mathscr{D}_i = 275 \cdot 7 \cdot 10^{-3} \simeq 1.9[J] \tag{5.5}$$

as reported in Table 5.1.
In (5.4) the energy consumption is calculated as the product between electric power and time, while (5.5) represents an energy contribution independent of time. As it will be discussed below, this implies that the amount of energy calculated by (5.4) depends on the interval of time in which the actuator is switched on, while the second term depends only on the number of activation cycles of the actuator. Finally, two additional considerations are in order. The first one concerns the amount of equivalent electric energy in the period of time during which the actuator is switched-on. Since the adopted pneumatic actuator consists of a single effect piston, so that the return movement is carried out by a spring, the equivalent energy contribution must be computed only once for each of the *mi* actuation cycle. The second consideration concerns the step that affects the total energy consumption function due to the equivalent electric energy consumption, which is independent of the simulation interval of time Δt. These two considerations allow to extend (5.3) as follows:

$$\begin{aligned}\mathscr{E}(k\Delta t+\tau) = \mathscr{E}(k\Delta t) + \mathscr{P}(k\Delta t)\cdot\tau + \\ + \sum_{i=1}^{n}\left\{\mathscr{D}_i\cdot\left[\sum_{j=1}^{mi} f_j(\mathscr{C}_i(k\Delta t))\right]\right\} + \\ + \sum_{i=1}^{n}\left\{\mathscr{G}_i\cdot\left[\sum_{j=1}^{mi} f_j(\mathscr{C}_i(k\Delta t))\right]\right\}\end{aligned} \tag{5.6}$$

where the function $f_j(\mathscr{C}_i(k\Delta t))$ is 1 in the $j-th$ period of time during which the $i-th$ actuator is switched-on, 0 otherwise.
Similar considerations hold true for the piston Ev_P used to block and unblock the pallet. The nominal electric power absorbed by the electro-valve that drives the piston requires $1.3[W]$ for the same interval time of $1[s]$ and then, as in (5.4), the consumed energy is equal to

$$\mathscr{F}_i = 1.3\cdot 1 = 1.3[J] \tag{5.7}$$

The pneumatic piston has a diameter equal to $20[mm]$ which leads to

$$\mathscr{G}_i = 275\cdot 3.14\cdot 10^{-3} \simeq 0.8[J] \tag{5.8}$$

5.2.1 Control system structure

More than one pallet can be placed on the transport line at the same time, and the goal of the control system is to determine, at any time instant, the movement of the pallets along the line to optimize the throughput of the system, to avoid deadlocks, to minimize the overall energy consumption and the absorbed power. In turn, this is equivalent to compute the commands to the actuators of the transport modules which allow for the pallets movements. With this objective in mind, the control system has been designed according to the hierarchical structure shown in Figure 5.1. At the higher level, an optimization algorithm recursively computes the optimal movement of the pallets along the transportation line, i.e. the control sequences to be actuated, based on the current status of the system given by the number of pallets on the conveyor and the status of the machines. A detailed description of the implemented high level control algorithm is reported in [24]. At the lower level, a set of *PLC*, one for each transport module, acquires the sensors' signals and drives the actuators to implement the required control actions. The two control loops run at different time scales, the high level works with $1[s]$ cycle time, while the sampling time adopted at the low level is equal to $100[ms]$.

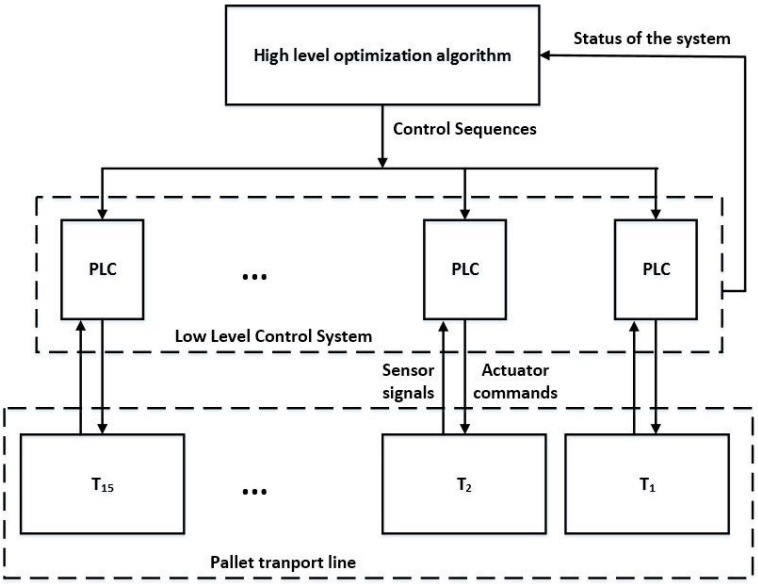


Figure 5.1: De-manufacturing pilot plant control architecture.

5.2.2 Dynamic control platform for industrial plants

DCPIP, a software platform based on the $C++$ object oriented programming language, has been implemented and used to implement the control scheme and to apply the *CSM* methodology. *DCPIP* is structured in terms of kernels, dynamically generated by the *main cycle* during its start-up. The kernels represent the software modules which implement specific control and communication functionalities and, since the object oriented paradigms have been used for the *DCPIP* implementation, then the generic kernel can be described in terms of a class object. In particular the *main cycle* reads a text file containing the IDentification (*ID*) numbers of the machines defining the plant to be controlled and the constructors of the identified machine classes start building the corresponding dynamic data structure and setting the control and energy consumption algorithms. In Figure 5.2 the *DCPIP* Class diagram is showed.

The kernels implemented in the *DCPIP* are:

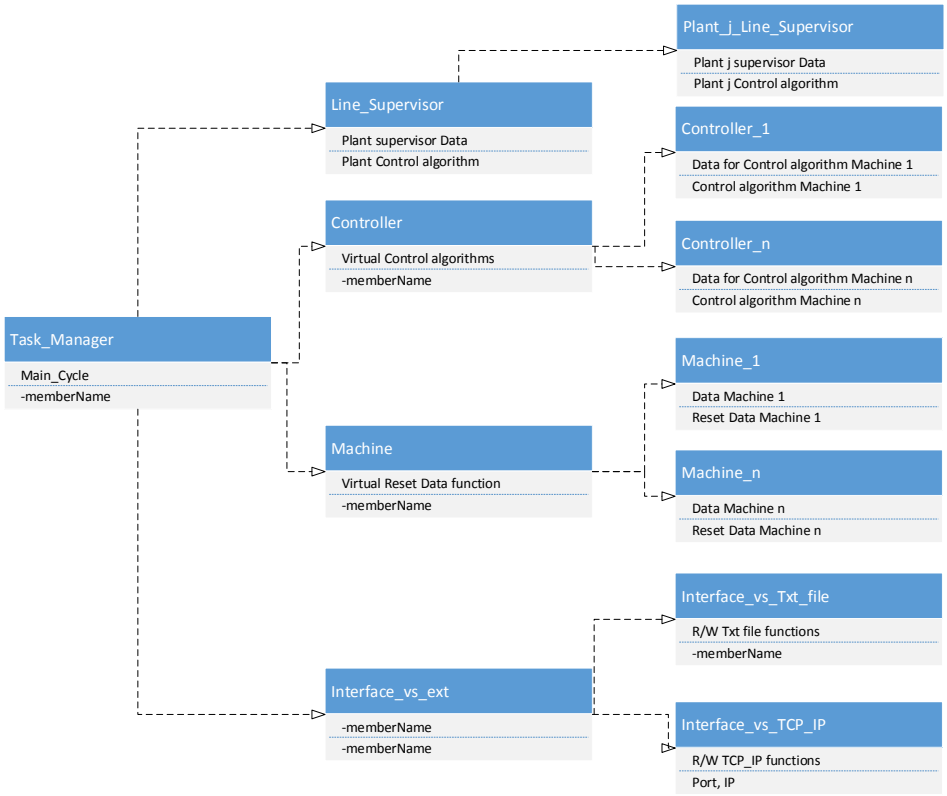


Figure 5.2: *DCPIP* class diagram.

- *Task Manager*: it has to contain all the variable and methods for the execution of each other kernel. In particular it starts the dynamic generation of each kernel, runs the *DCPIP main cycle*, which scans the Input variables coming from the plant *PLC* or plant simulator, runs the control algorithms contained into the controller kernels and update the Output variables to be sent to the plant *PLC* or plant simulator;

- *Line Supervisor*: it implements the control algorithm of the plant supervisor controller. In particular it communicates with each machine controller kernel besides with the plant *PLC* or plant simulator, by exchanging Input / Output data via text files or *TCP* protocol. The most important object oriented aspect of this class is the inheritance because a derived class instantiated from the *Line Supervisor* one could be implemented as control algorithm for specific plant. This means that the *Line Supervisor* kernel is based on virtual functions which will be specialized into the derived classes. Finally it can stop running specific controller

kernels or set them into the stand-by mode so to emulate the off-line state of specific plant operating machines;

- *Plant j Line Supervisor*: it implements the specific plant supervisor control algorithm;
- *Machine*: it implements the data structure required by the referring machine control algorithm implementation;
- *Machine n*: it implements the specific machine data structure;
- *Controller*: it contains the control algorithm of the corresponding machine to be controlled. Since each machine is characterized by its own control functionalities, this implies that the *Controller* class has some virtual functions which are specialized into the derived controller class. In particular the first two functionalities are specialized for the specific machine to be controlled, then they are defined as virtual functions, while the last two ones are specific for each machine:
  - *Execute control*: it implements the control algorithm of the machine controller;
  - *Execute set-up*: it implements the machine set-up sequence;
  - *Calculate working time*: it implements the machine working time computation;
  - *Reset internal variables*: it reset the machine controller internal variables;
- *Controller n*: it implements the specific machine control algorithm;
- *Interface vs ext*: it implements the communication among the *Machine* controllers, the *Line Supervisor* controller and the plant *PLC* or plant simulator. This class has got methods to execute the *Read / Write* Input / Output data, according to the different communication methods specialized in specific derived classes. For this reason, it is a virtual class;
- *Interface vs Txt file / TCP*: they implement the specific communication methods.

Switching the Input / Output data communication method is a very easy way because it just needs to set the specific software class (*Interface vs Txt file* or *TCP*). This characteristic has been used for the *CSM* validation by firstly interfacing the *DCPIP* with the discrete event simulator of the transport line and then by connecting it to the real plant. Notably, no modification of any *DCPIP* software data has been required.

In order to connect *DCPIP* with the real plant or the plant simulator so to implement the control algorithm, a suitable software architecture has been designed, see Figure 5.3.

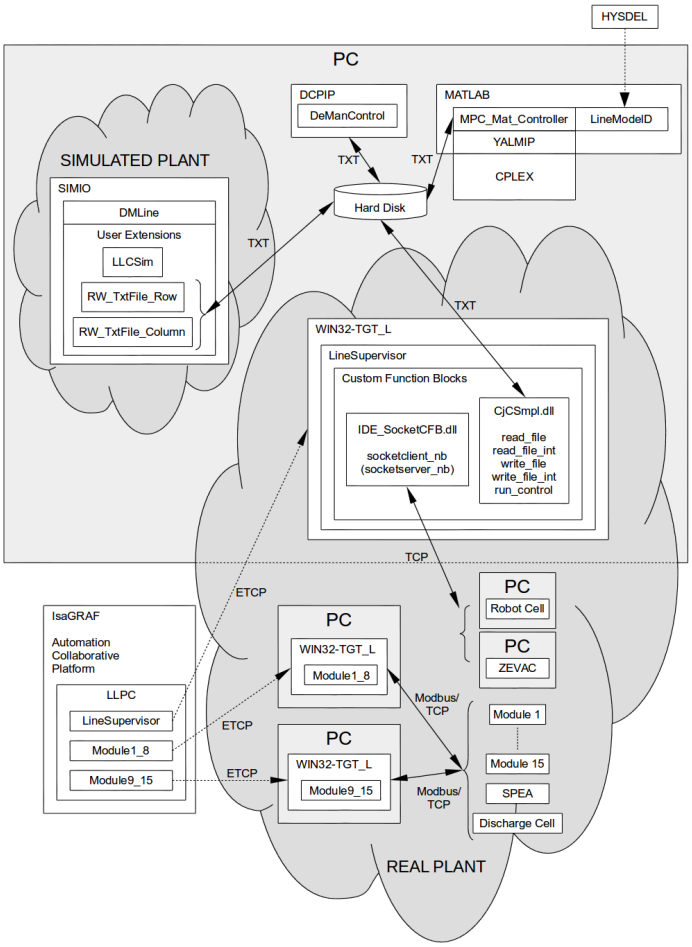


Figure 5.3: Software architecture for *DCPIP* implementation.

On a *PC* they are hosted:

- *DCPIP*, which runs the control algorithm of the plant to be controlled. It acquires via text file the Input data written on the Hard Disk coming from the real plant (WIN32-TGT_L ISaGRAF platform) or the plant simulator (SIMIO platform), elaborates such Input and writes the corresponding Output data via text file on the same Hard Disk. In order to implement advanced control algorithm, *DCPIP* is linked via text file with the MAT-

LAB platform (together with YALMIP and HYSDEL tools) and CPLEX optimizer;

- MATLAB platform, together with YALMIP and HYSDEL software tools is dedicated to run advanced control algorithm as *MPC*. The optimization process is performed by CPLEX software. In particular HYSDEL is used to generate the *MLD* model of the plants while YALMIP allows an easy interface for the control algorithm building and the CPLEX optimization functionalities management;

- SIMIO platform runs the discrete event plant model by exchanging via text file the Input / Output plant variables, in particular by reading the control actions sent by the *DCPIP* calculates the regarding process variables that are sent back to *DCPIP*;

- WIN32-TGT_L ISaGRAF platform represents the software interface of the real plant controllers towards *DCPIP*. In particular it exchanges via text file the Input / Output plant variables as the plant simulator does. In order to get the data coming from the real plant controller, a *TCP* communication protocol is implemented. On the real plant there are two kinds of controllers: soft-controllers, running on PCs, and plant control devices (i.e. *PLC*), according to the specific device to be controlled.

On the real plant, ISaGRAF soft-controllers, running on PCs, and plant control devices (i.e. *PLC*) are used to manage the different machines (for the case study here considered the machines consist on the fifteen transport modules and the operating machines M_1-M_4 described in Section 2.1.

5.2.3 Discrete event system simulator of the plant

CSM has been first tested in simulation. To this end, a Discrete Event System Simulator (*DESS*) of the plant has been implemented, in particular the transport line has been modeled in the SIMIO *DES* platform according to the modeling and simulation techniques described in [25], that is by describing the control behavior of a machine by means of a FSM. The complete model has been obtained by aggregating fifteen transport module simulation models [23], each one based on the following main items: (*i*) a network of nodes and paths describing the flow of the pallet; (*ii*) a set of SIMIO processes used to manage the pallet movement thorough the network of nodes; (*iii*) the FSM control behavior, translated into *C#* code and

implemented into SIMIO custom step in order to be integrated with the mechanical behavior, see Figure 5.4; (iv) the computation of the total absorbed power and of the total energy according to (5.2) and (5.3). The choice to implement (5.2) and (5.3) into each transport module *DES* model depends on the specific control system architecture of the de-manufacturing pilot plant. In fact each transport module is managed by an own *PLC* which is deputed to drive the transport module working function by acquiring the sensor signals and by setting the actuator commands.

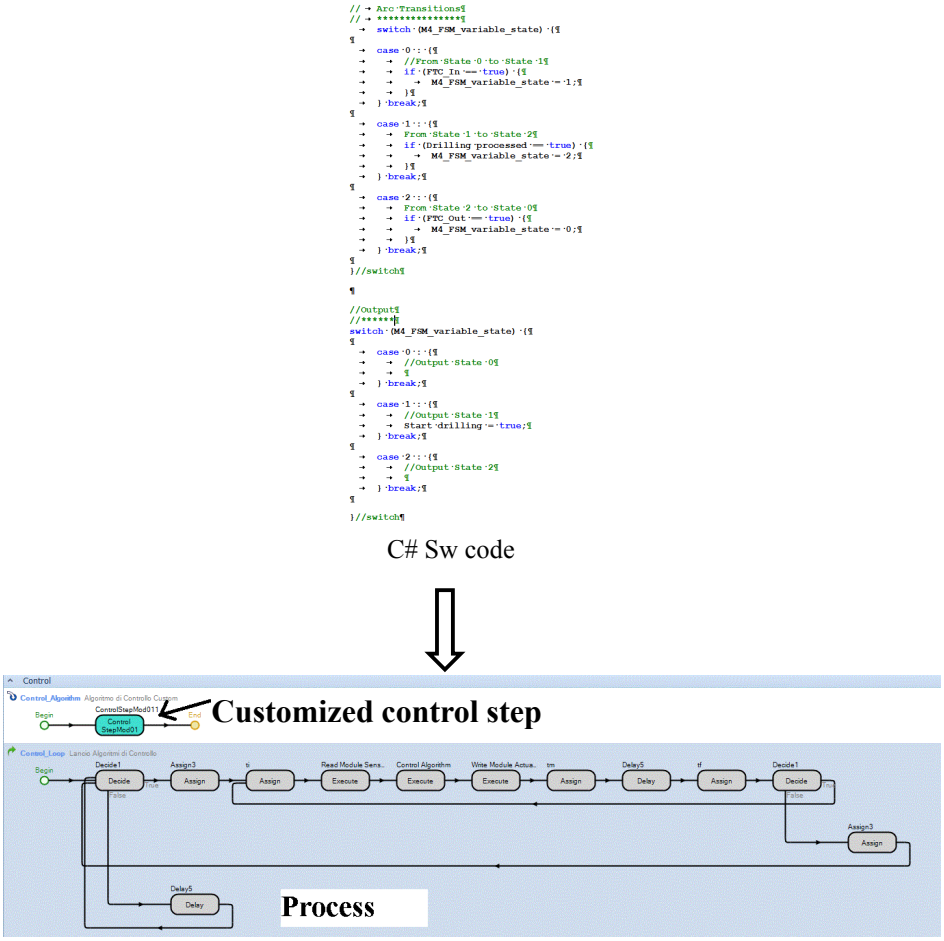


Figure 5.4: SIMIO customized control step.

5.3 Simulation and experiments results

5.3.1 CSM applied to a simple introductory example

Consider the control sequence used to move a pallet from the buffer zone $\Upsilon_{1,1}$ to $\Upsilon_{1,3}$. Denote by $\mathscr{A}_{\hat{i}}(t)$ and $\mathscr{A}_{\check{i}}(t)$ the instantaneous power absorbed by the stacker crane down movement and the main track motor forward movement while $\mathscr{C}_{\hat{i}}(t)$ and $\mathscr{C}_{\check{i}}(t)$ the related Boolean control variables. For simplicity consider a constant power profile independent of time t, so we can assume $\mathscr{A}_{\hat{i}}(t) = \mathscr{A}_{\hat{i}}$ and $\mathscr{A}_{\check{i}}(t) = \mathscr{A}_{\check{i}}$. The transport module total absorbed power is

$$\mathscr{P}_{T_1}(t) = \mathscr{A}_{\hat{i}} \cdot \mathscr{C}_{\hat{i}}(t) + \mathscr{A}_{\check{i}} \cdot \mathscr{C}_{\check{i}}(t) \tag{5.9}$$

Assume that the control sequence starts at time zero, see Figure 5.5. The Boolean control variables $\mathscr{C}_{\hat{i}}(0)$ and $\mathscr{C}_{\check{i}}(0)$ are set to 1, and the total absorbed power is given by (5.9) in the whole interval time between the initial time and τ_2, when $\mathscr{C}_{\hat{i}}(t)$ is set to 0, even if the power computation runs at each interval time Δt. This means that, if Δt is $100[ms]$ and τ_2 is equal to $2s$, the absorbed power update (5.9) is run 20 times. At time τ_2 the power $\mathscr{A}_{\hat{i}} \cdot \mathscr{C}_{\hat{i}}(t)$ becomes null and the T_1 power computation value is only given by the second term in (5.9). At the instant time τ_5 the control action $\mathscr{C}_{\check{i}}(t)$ is also reset to 0 and also the second term of (5.9) becomes null.

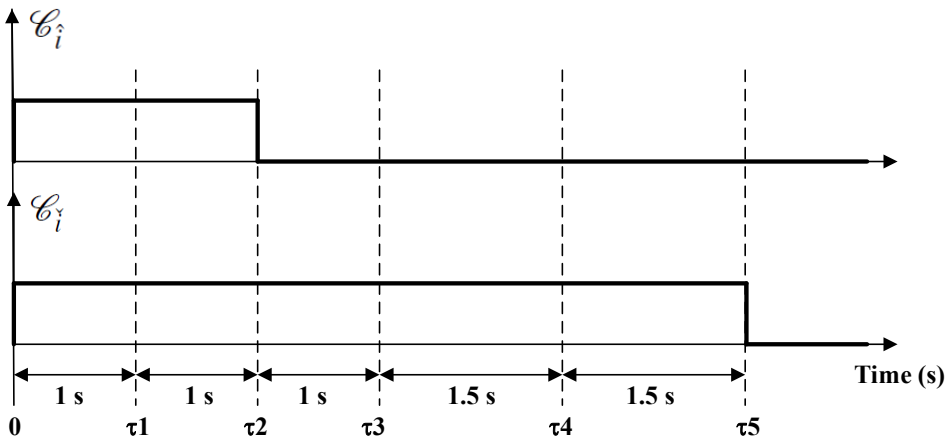


Figure 5.5: Control variables plot.

The total absorbed power is plotted in Figure 5.6, where also the total energy consumption is depicted.

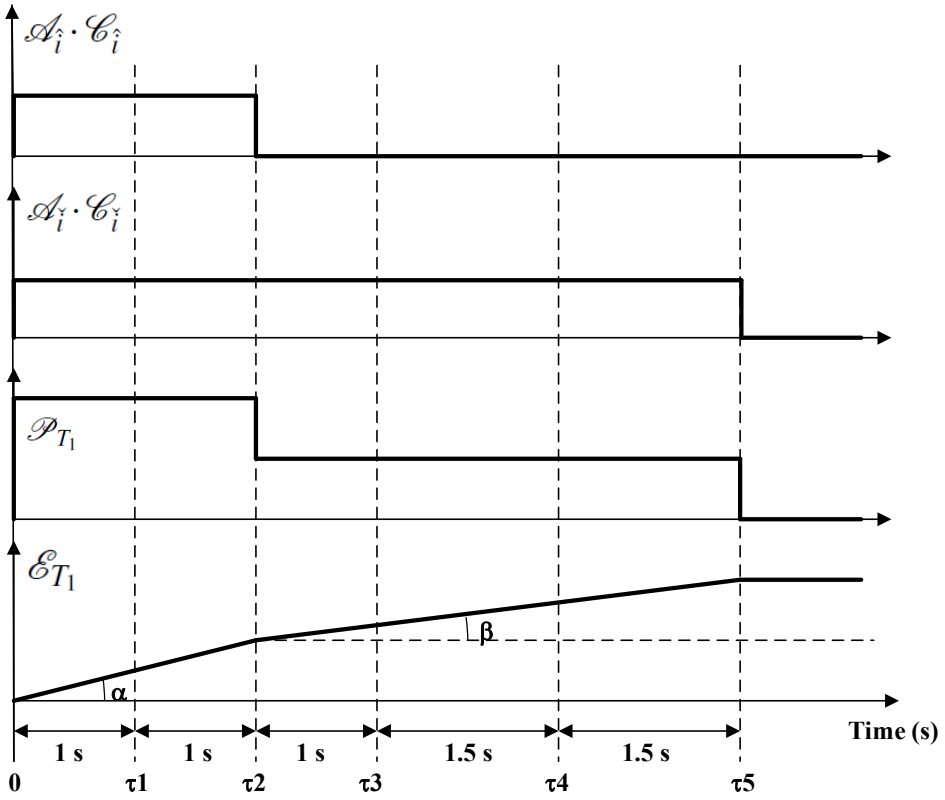


Figure 5.6: Power and energy plot.

In particular, in the first interval of time from 0 to τ_2 the energy consumption is calculated as a straight line with gradient $tg(\alpha)$, equal to the total absorbed power value

$$\frac{d\mathscr{E}_{T_1}}{dt} = \mathscr{A}_{\hat{i}} + \mathscr{A}_{\check{i}} \tag{5.10}$$

Then at τ_2 the total energy consumption is equal to

$$\mathscr{E}_{T_1}(\tau_2) = 0 + \mathscr{A}_{\hat{i}} + \mathscr{A}_{\check{i}} \tag{5.11}$$

At this instant, the control action related to the stacker crane becomes null and only the main track power profile is integrated. In this way the gradient $tg(\beta)$ of the energy consumption curve decreases to the value

$$\frac{d\mathscr{E}_{T_1}}{dt} = \mathscr{P}_{T_1}(t) = \mathscr{A}_{\hat{i}} \tag{5.12}$$

At τ_5 the total energy consumption will be equal to

$$\mathscr{E}_{T_1}(\tau_5) = \mathscr{E}_{T_1}(\tau_2) + \mathscr{A}_{\hat{i}} \cdot (\tau_5 - \tau_2) \tag{5.13}$$

which rearranged becomes

$$\mathscr{E}_{T_1}(\tau_5) = (\mathscr{A}_{\hat{i}} \cdot \tau_2) + (\mathscr{A}_{\hat{i}} \cdot \tau_5) \tag{5.14}$$

(5.14) shows that the total energy consumption is computed as the summation of the energy consumed by each actuator in all the intervals time in which they have been switched-on. After the time τ_5 the total energy consumption curve remains constant because the two actuators are switched-off and then no power is absorbed.

5.3.2 CSM applied to the simulated model of the pallet transport line

The constant power absorbed by the electronic devices of each transport module is equal to $30[W]$, for a total amount of $450[W]$. Based on the data (and on the notation of the actuators label) reported in Table 5.1 and in view of the previous considerations, see (5.4)- (5.8), the contributions of the electro-valves to the overall absorbed power and energy consumption is negligible. Therefore, only the main and stacker crane tracks power contributions have been considered in (5.1), while the contribution of the actuators $Ev\_Sc1/2\_U/D$ and Ev_P has been neglected.
The simulation experiment consists of moving two pallets on the transport line according to the control sequence described in Table 5.2.

Note that at most two actuators are activated at the same time to allow for a simple representation and analysis of the system's behavior. This control sequences have been implemented and run inside the *DCPIP*. In the simulation experiment, the *DCPIP* has been connected to the transport line *DESS* model and the contribution of each transport module has been computed.
The total power profile and the consumed energy are shown in Figure 5.7-5.8, respectively. It is apparent from Figure 5.7 that the constant power ($450[W]$) absorbed by the whole pallet transport line due to the electronic

Control step	*Step time* $[s]$	*Switched-on actuator*	*Transport module*
1	0.0	M_Tr_F	T_1
		M_Tr_F	T_2
2	11.7	M_Tr_F	T_1
		M_Tr_F	T_2
		M_Tr_F	T_2
		$Sc1\_Tr\_L$	T_3
3	20.9	M_Tr_F	T_2
		M_Tr_F	T_3
4	33.6	M_Tr_F	T_3
		M_Tr_F	T_4
5	40.0	M_Tr_F	T_4
		M_Tr_F	T_5
6	57.4	M_Tr_F	T_2
		$Sc1\_Tr\_L$	T_3
		M_Tr_F	T_5
7	66.7	M_Tr_F	T_3
		$Sc1\_Tr\_L$	T_5
		$Sc1\_Tr\_L$	T_6
8	77.5	M_Tr_F	T_3
		M_Tr_F	T_4
		M_Tr_F	T_6
9	90.6	$Sc1\_Tr\_L$	T_4
		$Sc1\_Tr\_R$	T_8
10	98.9	M_Tr_F	T_6
		M_Tr_F	T_8

Table 5.2: Control sequence and actuators activation.

control devices, represents roughly the 70% of the maximum peak of absorbed power, which leads to the linear trend of the consumed energy shown in Figure 5.8. This is due to the small number of actuators switched on at the same time. The total energy consumption at the end of the simulation ($102.9[s]$) is equal to $54558[J]$.

In order to analyze more in detail the simulation results, consider for example the control step 1. The high level controller requires to move the pallet from $\Upsilon_{1,1}$ to $\Upsilon_{1,3}$ with the actuator M_Tr_F of the transport module T_1, while the second control action moves the pallet from $\Upsilon_{2,1}$ to $\Upsilon_{2,3}$ with the actuator M_Tr_F of T_2. From the power profiles of these transport modules, reported in Figure 5.9 and Figure 5.11, it is possible to see that the total absorbed power rises from $30[W]$, i.e. their base power, to $94[W]$,

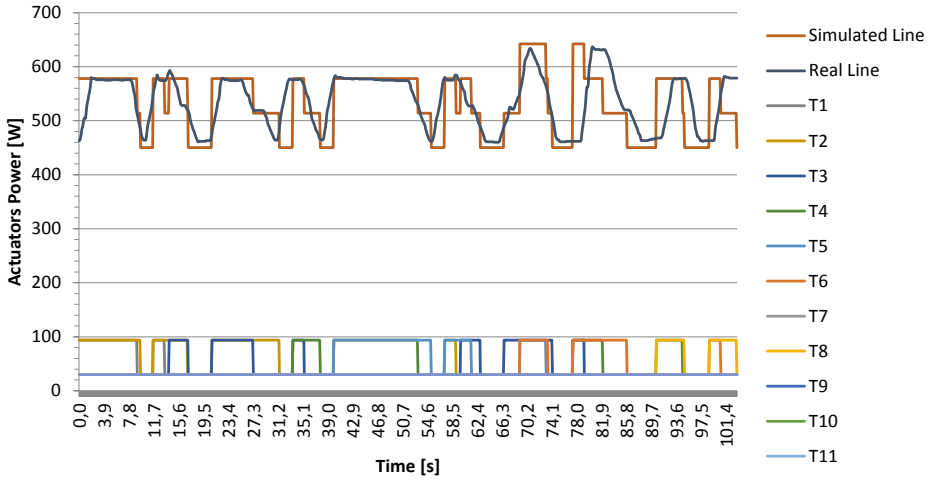


Figure 5.7: Pallet transport line power simulation and measurement results.

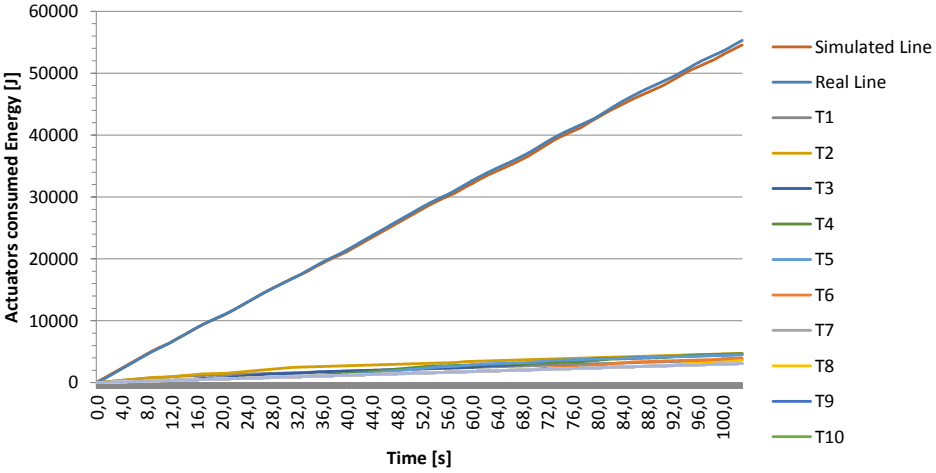


Figure 5.8: Pallet transport line energy consumption simulation and measurement results.

and this value is maintained in the interval of time required to complete the pallet movement. Then, the power absorbed by each transport module falls down to $30[W]$. Correspondingly, as shown in Figure 5.7, close to the time instant $9[s]$ the total power has a negative step equal to $64[W]$. This depends on the fact that, once the pallet has left the transport module T_1, the corresponding actuator M_Tr_F is switched-off, while the actuator M_Tr_F of T_2 is still switched-on. In fact, the analysis of the simulation results shows that M_Tr_F of T_1 switches-off at time $9.1[s]$ while M_Tr_F of T_2 switches-off at time $9.6[s]$, when the total power absorbed by the transport

line is brought back to the minimum value $450[W]$. In order to complete the analysis, the energy consumption of the transport modules T_1 and T_2 is shown in Figure 5.10 and Figure 5.12, respectively. From these figures it is easy to see that the energy consumed by the transport modules electronic devices, labelled "Base", is greater than the total energy consumed by each actuator used to perform the pallet movement. This is due to the specific simulation experiment in which only two pallets have been moved, so that the contribution of the active actuators is quite small compared to the one of the Base energy consumption.

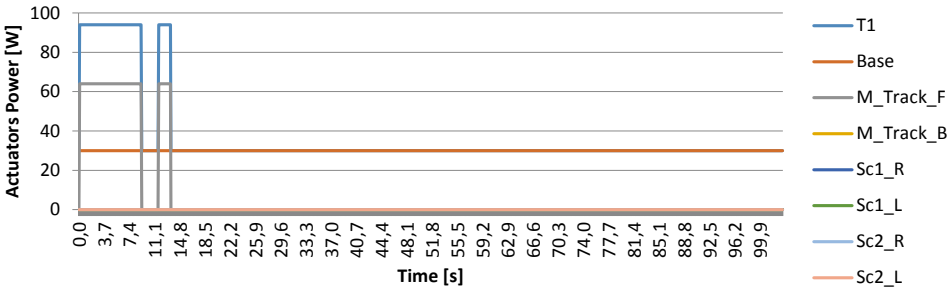


Figure 5.9: Transport module T_1 power simulation results.

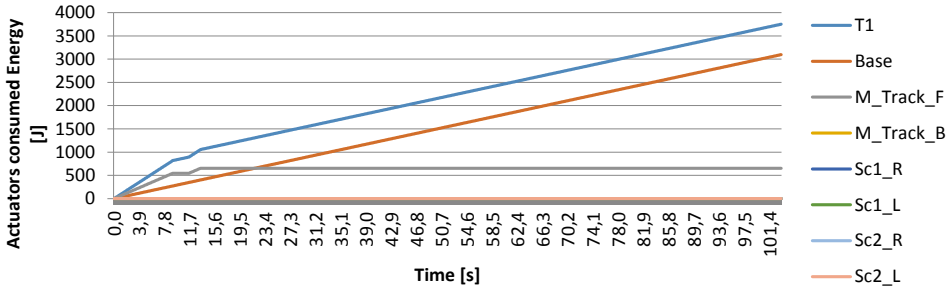


Figure 5.10: Transport module T_1 energy consumption simulation results.

5.3.3 CSM applied to the de-manufacturing pilot plant

DCPIP has also been used to control the real plant. In this case, the high level optimizing controller runs in *DCPIP*, while the low level controllers are implemented in the ICS Triplex ISaGRAF platform by resorting to the Sequential Functional Chart (*SFC*) programming language of *IEC*61131 –

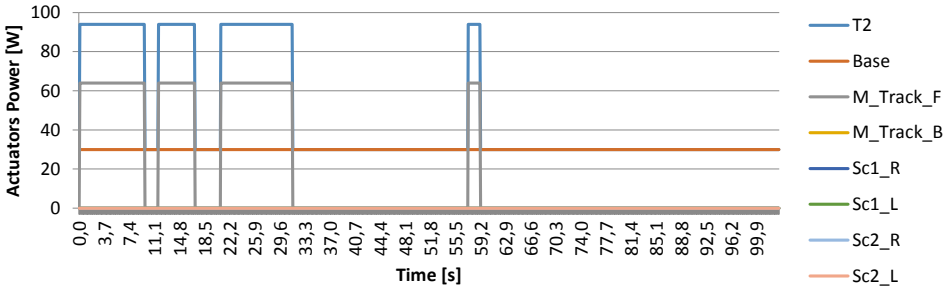


Figure 5.11: Transport module T_2 power simulation results.

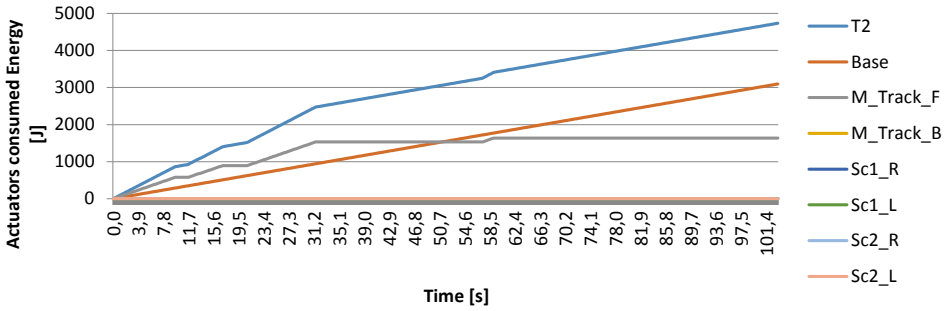


Figure 5.12: Transport module T_2 energy consumption simulation results.

3 standard, see for example an implemented *LLCS* sequence in Figure 5.13.

The measurement instrumentation has been used to collect and store the power and energy consumption of the whole pallet transport line, see Figure 5.14.

The same control sequence considered in the simulation experiment and listed in Table 5.2 has been used. Since the available instrumentation is able to acquire only one measure of power and energy consumption, only their total value, referred to the whole transportation line, has been acquired. The measured transients are compared to those computed in simulation with *CSM* in Figure 5.7- 5.8, respectively.
From Figure 5.7 it is apparent that the transients of the absorbed power, due to the activation/deactivation of the actuators, are not pure steps, but exhibit a sort of "ramp-type" behavior. This is due to a filtering action performed by the measurement system. However, the effect of this filtering is negligible on the average, since the overestimated energy due to the actuators' switch on phase is compensated by the underestimated energy due to their

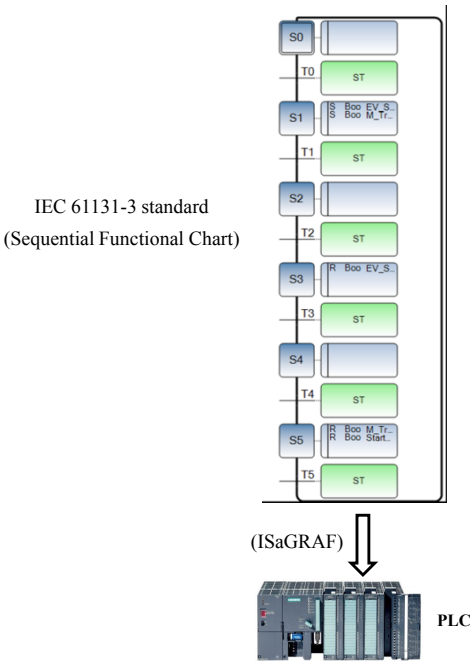


Figure 5.13: ISaGRAF plant low level control system implementation.

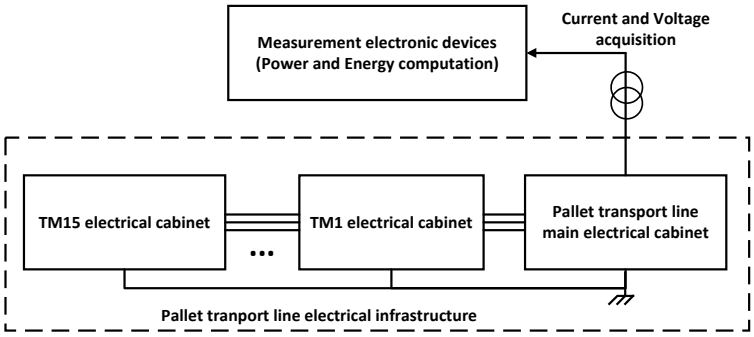


Figure 5.14: Measurement system architecture.

switch-off. As for the power peaks and their duration, Figure 5.7 shows a very good fitting between the simulated results and the field measurements. Finally, Figure 5.8 shows that also the energy consumption computed with *CSM* and the simulation model fits well the real plant data. In fact, the difference between the measured and computed energy consumption at the end of the experiment is equal to $1589[J]$ over real total consumption of $57522[J]$. In terms of power, the difference between the mean simulated and measured values is equal to $15[W]$, i.e. the 0.5% of the measured total

absorbed power. This small difference is mainly due to the neglected effect of the actuators $Ev\_Sc1/2\_U/D$ and Ev_P.

Chapter 6

Conclusions

In this thesis, in order to cope with discrete-event control problems typical of the manufacturing field, different industrial applications have been taken into account so to assess the adequacy of advanced control systems in terms of "easy to design and use" and performances.

Firstly, efficient routing of the pallets in networks made by machines and transportation lines has been studied with the aim to avoid bottlenecks, starvation, congestion, and to maximize throughputs. In this scenario, *MPC* has been applied to a de-manufacturing transport line in which a multi-pallet, dynamic multi-target problem has to be solved. The dynamic system of the transport line has been formulated as a *MLD* system. Then, a performance index has been defined and the optimal control sequence has been recursively computed and applied according to a receding horizon approach.
Secondly, the problem of optimizing on-line the production scheduling and buffer management of a multiple-line production plant composed by L machines M_i, $i = 1, ..., L$, which can operate at different speeds corresponding to different energy demands, has been taken into account. The path from a common source node, where the part to be processed is assumed to be always available, to each machine may differ in the number of buffer nodes and the energy required to move the part along these transportation lines must be suitably considered in the computation of the overall energy consumption. Therefore, the considered control problem consists of computing, at each sampling instant, the sequence of commands to be applied to the transportation lines and the processing speed of the machines in order to optimize the throughput of the system and to limit the overall energy consumption.
As third case study, a specific type of AS/RS has been considered. The

complexity of the control algorithm for a AS/RS is related to the number of depots and the storage policy of the system. The application of *MPC* to AS/RS has been developed by describing the system in terms of a *MLD* system. The transformation fron logical propositions to an *MLD* system allows the control problem to be expressed as a *MILP* problem.
Finally, in order to be able to compute the energy consumption for a manufacturing plant, a specific new energy consumption computation method has been defined and validated, based on the *DES* approach for the computation of the energy consumption of discrete systems, i.e. systems where the energy consumption is mainly due to the on/off switching of the actuators governed by the control logic.

The different simulations and experiments have shown a good adaptability of the *MPC* algorithm to the different applications, in particular the *MPC* can be easily adapted to obtain different behaviours of the controlled system by means of the tuning of simple and easy-to-understand parameters of the cost function.
In any case, the increasing of both the prediction horizon and the *MLD* system dimension, besides the *MILP* nature of the optimization problem, make relevant the computational burden of the *MPC* algorithm.

For these reasons, future research activity involving the different systems considered in this thesis, could be respectively aimed at:

- reducing the complexity of the de-manufacturing transport line *MLD* model, see Section 2.4, and then the optimization problem computation time, by using the Lagrangean decomposition methods [10, 55, 76, 113]. In particular the application of the temporal Lagrangean decomposition would allow to split the prediction time into sub-intervals and, at the same time, to keep unchanged the whole transport line topology and the regarding mechanical and control constraints;
- customizing the algorithm to the de-manufacturing transport line [89] by considering the possibility to dynamically change the different operating machines' working function settings in order to further optimize the production line efficiency, with constraints on the early machining of parts or with non deterministic behaviours of the machines themselves;
- reducing the complexity of the laboratory stacker crane model by generalizing the AS/RS description in order to decrease the required computation time. The aim is hereby focused on the reduction of the number of in-

teger variables, since these determine the complexity of a *MILP* problem. For example the model developed in this thesis can easily be extended to a form with multiple final storage nodes. Next to these improvements and extensions, it would be interesting to compare the *MPC* method to other control methods such as time instant optimisation *MPC* [116] and heuristics.

Appendices

Appendix A

MLD model of the transport line

The *MLD* formulation of the system has been obtained by linearizing the system dynamic equations and by translating the logic propositions into linear inequalities by means of the propositional calculus [12, 75, 79, 93, 122].

Preliminaries

First recall that the product between two boolean variables $\nu_3 = \nu_1\nu_2$ is equivalent to

$$-\nu_1 + \nu_3 \leq 0 \tag{A.1a}$$
$$-\nu_2 + \nu_3 \leq 0 \tag{A.1b}$$
$$+\nu_1 + \nu_2 - \nu_3 \leq 1 \tag{A.1c}$$

The product between a real variable $f(x)$ and a boolean one ν such that $g(x) \triangleq \nu f(x)$, is equivalent to

$$g(x) \leq M\nu \tag{A.2a}$$
$$g(x) \geq m\nu \tag{A.2b}$$
$$g(x) \leq f(x) - m(1-\nu) \tag{A.2c}$$
$$g(x) \geq f(x) - M(1-\nu) \tag{A.2d}$$

with $M \triangleq max\, f(x)$ and $m \triangleq min\, f(x)$.

For what concern the proposition $[f(x) \leq 0] \leftrightarrow [\nu = 1]$, it can be linearized as

$$f(x) \leq M(1-\nu) \tag{A.3a}$$
$$f(x) \geq \varepsilon + (m-\varepsilon)\nu \tag{A.3b}$$

and the proposition $[f(x) \leq 0] \rightarrow [\nu = 1]$ can be linearized by means of the formulation

$$f(x) \geq \varepsilon + (m-\varepsilon)\nu \tag{A.4}$$

in which ε represents a small tolerance beyond which the constraint is violated.

In case of boolean variables only,

$$P_1 \rightarrow P_2 \wedge P_3 \quad \Longrightarrow \quad \delta_1 \leq \delta_2,\, \delta_1 \leq \delta_3 \tag{A.5}$$
$$P_1 \wedge P_2 \rightarrow P_3 \quad \Longrightarrow \quad \delta_1 + \delta_2 - \delta_3 \leq 1 \tag{A.6}$$
$$P_1 \rightarrow P_2 \quad \Longrightarrow \quad \delta_1 - \delta_2 \leq 0 \tag{A.7}$$
$$P_1 \leftrightarrow P_2 \quad \Longrightarrow \quad \delta_1 - \delta_2 = 0 \tag{A.8}$$
$$P_n \leftrightarrow P_1 \vee P_2 \vee \ldots P_k \quad \Longrightarrow \begin{cases} \delta_1 + \delta_2 + \cdots + \delta_k \geq \delta_n, \\ -\delta_j + \delta_n \geq 0,\ j = 1,\ldots,k \end{cases} \tag{A.9}$$

Dynamic equation linearization

The dynamic equation (2.4) concerning the target propagation can be rewritten in the following form

$$\Gamma_i(k+1) = \Gamma_i(k) + \sum_{j \in I_{i,in}} \tilde{\Gamma}_{j,i}(k) - \sum_{j \in I_{i,out}} \tilde{\Gamma}_{i,j}(k), \qquad i = 1,\ldots,31 \tag{A.10}$$

by applying (A.2a) - (A.2d) to

$$\tilde{\Gamma}_{j,i}(k) = \Gamma_i(k) u_{j,i}(k), \qquad i = 1,\dots,31, \qquad j \in I_{i,in} \tag{A.11a}$$
$$\tilde{\Gamma}_{i,j}(k) = \Gamma_i(k) u_{i,j}(k), \qquad i = 1,\dots,31, \qquad j \in I_{i,out} \tag{A.11b}$$

For what concern the linearization of (2.9), the first step regards the product between two boolean variables, which is implemented according to (A.1a) - (A.1c)

$$\tilde{\rho}_i(k) = \delta_i(k)\vartheta_i(k), \qquad i = 1,\dots,31 \tag{A.12a}$$
$$\hat{\rho}_i(k) = \vartheta_i(k) u_{j,i}(k), \qquad i = 1,\dots,31, \qquad j \in I_{i,in} \tag{A.12b}$$
$$\check{\rho}_i(k) = \vartheta_i(k) u_{i,j}(k), \qquad i = 1,\dots,31, \qquad j \in I_{i,out} \tag{A.12c}$$

leading to

$$\eta_i(k+1) = \eta_i(k) + \tilde{\rho}_i(k) + \sum_{j \in I_{i,in}} [\eta_j(k) + 1]\hat{\rho}_i(k) + \\ - \sum_{j \in I_{i,out}} \eta_i(k)\check{\rho}_i(k), \qquad i = 1,\dots,31 \tag{A.13}$$

The product between a real variable and a boolean one is solved according to (A.2a) - (A.2d)

$$\hat{\eta}_i(k) = \eta_i(k)\hat{\rho}_i(k), \qquad i = 1,\dots,31 \tag{A.14a}$$
$$\check{\eta}_i(k) = \eta_i(k)\check{\rho}_i(k), \qquad i = 1,\dots,31 \tag{A.14b}$$

by obtaining the linearized form

$$\eta_i(k+1) = \eta_i(k) + \tilde{\rho}_i(k) + \sum_{j \in I_{i,in}} [\hat{\eta}_j(k) + \hat{\rho}_i(k)] - \sum_{j \in I_{i,out}} \check{\eta}_i(k), \\ i = 1,\dots,31 \tag{A.15}$$

The machine target propagation dynamic equation (2.16) can be translated as

$$\Gamma_i(k+1) = \Gamma_i(k) + \sum_{j \in I_{i,in}} \tilde{\Gamma}_j(k) u_{j,i}(k) - \sum_{j \in I_{i,out}} \tilde{\Gamma}_i(k) u_{i,j}(k) + \\ + \delta_{i,23}(k) \bar{\Gamma}_i - \hat{\Gamma}_i(k), \qquad i = 32, \ldots, 35 \tag{A.16}$$

by adopting (A.11a) - (A.12b) with $i = 32, \ldots, 35$ and

$$[\delta_{i,23}(k) = 1] \leftrightarrow [x_{i2}(k) \wedge (n_i(k) \geq \bar{n}_i)], \qquad i = 32, \ldots, 35 \tag{A.17}$$

so that

$$\hat{\Gamma}_i(k) = \Gamma_i(k) \delta_{i,23}(k), \qquad i = 32, \ldots, 35 \tag{A.18}$$

Moreover it is possible to state

$$[\nu_i(k) = 1] \leftrightarrow [n_i(k) \geq \bar{n}_i], \qquad i = 32, \ldots, 35 \tag{A.19}$$

which in turn can be linearized according to the statements (A.3a) - (A.3b) leading to

$$\bar{n}_i - n_i \leq \bar{n}_i [1 - \nu_i(k)], \qquad i = 32, \ldots, 35 \tag{A.20a}$$
$$\bar{n}_i - n_i \geq \varepsilon + (m - \varepsilon) \nu_i(k), \qquad i = 32, \ldots, 35 \tag{A.20b}$$

So (A.17) can be rewritten as

$$[\delta_{i,23}(k) = 1] \leftrightarrow [x_{i2}(k) \wedge \nu_i(k))], \qquad i = 32, \ldots, 35 \tag{A.21}$$

and then the propositional calculus statements (A.5), (A.6) lead to

$$x_{i2}(k) + \nu_i(k) - \delta_{i,23}(k) \leq 1, \qquad i = 32, \ldots, 35 \tag{A.22a}$$
$$x_{i2}(k) - \delta_{i,23}(k) \geq 0, \qquad i = 32, \ldots, 35 \tag{A.22b}$$
$$\nu_i(k) - \delta_{i,23}(k) \geq 0, \qquad i = 32, \ldots, 35 \tag{A.22c}$$

For what concern the machine counter equation (2.17), it follows that

$$n_i(k+1) = n_i(k) - \tilde{n}_i(k) + x_{i2}(k) - \tilde{x}_i(k), \quad i = 32, \ldots, 35 \tag{A.23}$$

with

$$\tilde{n}_i(k) = n_i(k) x_{i1}(k), \qquad i = 32, \ldots, 35 \tag{A.24}$$

$$\tilde{x_i}(k) = x_{i1}(k)x_{i2}(k), \qquad i = 32,\ldots,35 \tag{A.25}$$

Equation (A.25) consists of the product between two boolean variables, so it can be linearized by using the transformation (A.1a) - (A.1c) while the equations (A.11a), (A.12b), (A.18), and (A.24) consist of the product between a real variable $f(x)$ and a boolean one v, then they can be linearized by means of the transformation (A.2a) - (A.2d).

Logic proposition linearization

Concerning the equation (2.3), since the second term consists of a summation of binary variables $u_{i,j}$ according to which (2.2) is stated, then it can be formulated by means of (A.4)

$$\Gamma_i(k) \geq \varepsilon u_{i,j}, \qquad i = 1,\ldots,35, \quad j \in I_{i,out} \tag{A.26}$$

since $min\Gamma_i(k) = m = 0$.

Concerning the equations (2.5), each one of the three term is decomposed as follows:

$$[\Gamma_i(k) > 0] \leftrightarrow [v'(k) = 1] \tag{A.27}$$

is linearized according to (A.3a) - (A.3b);

$$[\sum_{j \in I_{i,out}} u_{i,j} = 0] \leftrightarrow [v''(k) = 1] \tag{A.28}$$

is translated in

$$\sum_{j \in I_{i,out}} u_{i,j} + v''(k) \geq 1 \tag{A.29a}$$

$$\sum_{j \in I_{i,out}} u_{i,j} + v''(k) \leq 1 \tag{A.29b}$$

Similarly the term

$$[\sum_{j \in I_{i,in}} u_{j,i} = 0] \leftrightarrow [v'''(k) = 1] \tag{A.30}$$

can be translated in the two inequalities

$$\sum_{j\in I_{i,in}} u_{j,i} + v'''(k) \geq 1 \tag{A.31a}$$

$$\sum_{j\in I_{i,in}} u_{j,i} + v'''(k) \leq 1 \tag{A.31b}$$

In this way, (2.5) can be formulated as $[v'(k) \wedge v''(k)] \rightarrow [v'''(k) = 1]$ which is linearized according to (A.6), and leads to

$$1 + v'''(k) \geq v'(k) + v''(k) \tag{A.32}$$

Similarly to (2.5), the linearization of the equations (2.7) can be carried out by stating $[\sum_{j\in I_{i,out}} u_{i,j} = 0] \leftrightarrow [v'(k) = 1]$, which leads to inequalities similar to (A.29a) and (A.29b), and $[\sum_{j\in I_{i,in}} u_{j,i} = 0] \leftrightarrow [v''(k) = 1]$, which leads to inequalities similar to (A.31a) and (A.31b).
Then (2.7) can be formulated as $[\delta_i(k) = 1] \leftrightarrow [v'(k) \wedge v''(k)]$
which, according to (A.5), (A.6), can be written as

$$\delta_i(k) \leq v'_i(k) \tag{A.33a}$$

$$\delta_i(k) \leq v''_i(k) \tag{A.33b}$$

$$v'_i(k) + v''_i(k) - \delta_i(k) \leq 1 \tag{A.33c}$$

For what concern the machine EFA evolution described by equations (2.10) and (2.12), by means of (A.6) they can be respectively written as

$$x_{i1}(k) + u_{j,i}(k) - [1 - x_{i1}(k+1)] \leq 1, \quad i = 32,\ldots,35,\ j \in I_{i,in} \tag{A.34}$$

$$x_{i1}(k) + u_{j,i}(k) - x_{i2}(k+1) \leq 1, \quad i = 32,\ldots,35,\ j \in I_{i,in} \tag{A.35}$$

$$x_{i3}(k) + u_{i,j}(k) - [1 - x_{i3}(k+1)] \leq 1, \quad i = 32,\ldots,35,\ j \in I_{i,out} \tag{A.36}$$

$$x_{i3}(k) + u_{i,j}(k) - x_{i1}(k+1) \leq 1, \quad i = 32,\ldots,35,\ j \in I_{i,out} \tag{A.37}$$

since (2.1) and (2.2) are stated.
Similarly, the equations (2.11) can be respectively written as

$$x_{i2}(k) + \nu_i(k) - [1 - x_{i2}(k+1)] \leq 1, \qquad i = 32, \ldots, 35 \tag{A.38}$$

$$x_{i2}(k) + \nu_i(k) - x_{i3}(k+1) \leq 1, \qquad i = 32, \ldots, 35 \tag{A.39}$$

with $\nu_i(k)$ defined in (A.19).

The equations (2.13) and (2.14) can be respectively stated as

$$[1 - x_{i1}(k)] - [1 - u_{j,i}(k)] \leq 0, \; i = 32, \ldots, 35, \; j \in I_{i,in} \tag{A.40}$$

$$[1 - x_{i3}(k)] - [1 - u_{i,j}(k)] \leq 0, \; i = 32, \ldots, 35, \; j \in I_{i,out} \tag{A.41}$$

according to (A.7).

In order to translate the (2.19a) - (2.19d), the terms $\Gamma_i(k) = \bar{\Gamma}_n$ can be stated according to (A.3a) - (A.3b), so obtaining

$$[\sigma_{32}(k) = 1] \leftrightarrow [\nu_{27}(k) \vee \nu_{31}(k) \vee \vartheta_{32}(k)] \tag{A.42a}$$

$$[\sigma_{33}(k) = 1] \leftrightarrow [\nu_{19}(k) \vee \vartheta_{33}(k)] \tag{A.42b}$$

$$[\sigma_{34}(k) = 1] \leftrightarrow [\nu_{12}(k) \vee \vartheta_{34}(k)] \tag{A.42c}$$

$$[\sigma_{35}(k) = 1] \leftrightarrow [\nu_{23}(k) \vee \nu_{35}(k) \vee \vartheta_{35}(k)] \tag{A.42d}$$

which, according to (A.9), lead to

$$\nu_{27}(k)+\nu_{31}(k)+\vartheta_{32}(k)-\sigma_{32}(k)\geq 0 \tag{A.43a}$$
$$-\nu_{27}(k)+\sigma_{32}(k)\geq 0 \tag{A.43b}$$
$$-\nu_{31}(k)+\sigma_{32}(k)\geq 0 \tag{A.43c}$$
$$-\vartheta_{32}(k)+\sigma_{32}(k)\geq 0 \tag{A.43d}$$
$$\nu_{19}(k)+\vartheta_{33}(k)-\sigma_{33}(k)\geq 0 \tag{A.43e}$$
$$-\nu_{19}(k)+\sigma_{33}(k)\geq 0 \tag{A.43f}$$
$$-\vartheta_{33}(k)+\sigma_{33}(k)\geq 0 \tag{A.43g}$$
$$\nu_{12}(k)+\vartheta_{34}(k)-\sigma_{34}(k)\geq 0 \tag{A.43h}$$
$$-\nu_{12}(k)+\sigma_{34}(k)\geq 0 \tag{A.43i}$$
$$-\vartheta_{34}(k)+\sigma_{34}(k)\geq 0 \tag{A.43j}$$
$$\nu_{23}(k)+\nu_{35}(k)+\vartheta_{35}(k)-\sigma_{35}(k)\geq 0 \tag{A.43k}$$
$$-\nu_{23}(k)+\sigma_{35}(k)\geq 0 \tag{A.43l}$$
$$-\nu_{35}(k)+\sigma_{35}(k)\geq 0 \tag{A.43m}$$
$$-\vartheta_{35}(k)+\sigma_{35}(k)\geq 0 \tag{A.43n}$$

Then by defining $\mu \triangleq \sigma u$, the terms of (2.20) concerning the penalization of the presence of a pallet in the nodes adjacent to $M_1,\ldots,M_4$ can be linearized according to (A.1a) - (A.1c)

$$-\sigma_m(k)+\mu_{m,i,j}(k)\leq 0 \tag{A.44a}$$
$$-u_{i,j}(k)+\mu_{m,i,j}(k)\leq 0 \tag{A.44b}$$
$$+\sigma_m(k)+u_{i,j}(k)-\mu_{m,i,j}(k)\leq 1 \tag{A.44c}$$

with

$$(m,i,j)\in\{(32,27,1),(32,31,1),(33,7,16),(33,15,16),(33,18,19),\\ (34,10,12),(35,22,23)\}.$$

Finally, in order to translate (2.8), the terms $\Gamma_i(k)\geq 1$ and $\Gamma_i(k)\leq 4$ can be stated according to (A.3a) - (A.3b) so that

$$[\vartheta_i(k)=1]\leftrightarrow[\nu_{i1}(k)\wedge\nu_{i2}(k)],\qquad i=1,\ldots,31 \tag{A.45}$$

which, according to (A.5) and (A.6), leads to

$$\nu_{i1}(k) + \nu_{i2}(k) - \vartheta_i(k) \leq 1, \qquad i = 1, \ldots, 31 \tag{A.46a}$$
$$\nu_{i1}(k) - \vartheta_i(k) \geq 0, \qquad i = 1, \ldots, 31 \tag{A.46b}$$
$$\nu_{i2}(k) - \vartheta_i(k) \geq 0, \qquad i = 1, \ldots, 31 \tag{A.46c}$$

Appendix B

Low level control system sequences

B.1 Transport module sensors and actuator

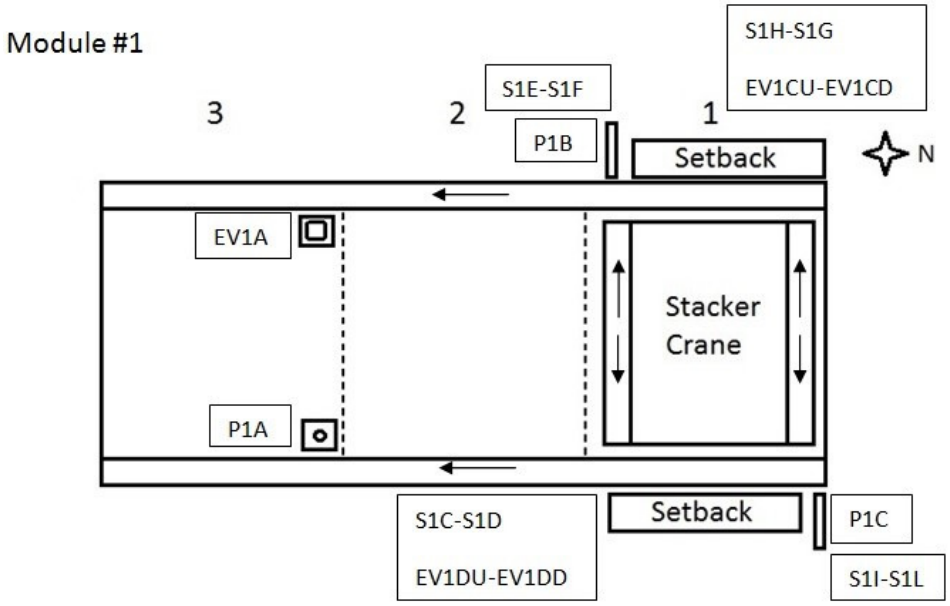


Figure B.1: Transport module N. 1.

- It is the module that receive the pallet from the robotic arms;
- Two accumulation areas $(1;3)$;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P1A	Presence sensor (piston lock)
S1A-S1B	Stacker crane Up/Down
P1B; P1C	Presence sensor (setbacks)
S1C-S1D; S1G-S1H	Setback Up/Down
S1E-S1F; S1I-S1L	Presence sensor (setbacks) Up/Down

ACTUATORS:	
M1F	Module′s tracks Forward
m1R-m1L	Stacker crane tracks Right/Left

EV1A	Piston Lock Down
EV1BU-EV1BD	Stacker crane Up/Down
EV1CU-EV1CD; EV1DU-EV1DD	Setback Up/Down

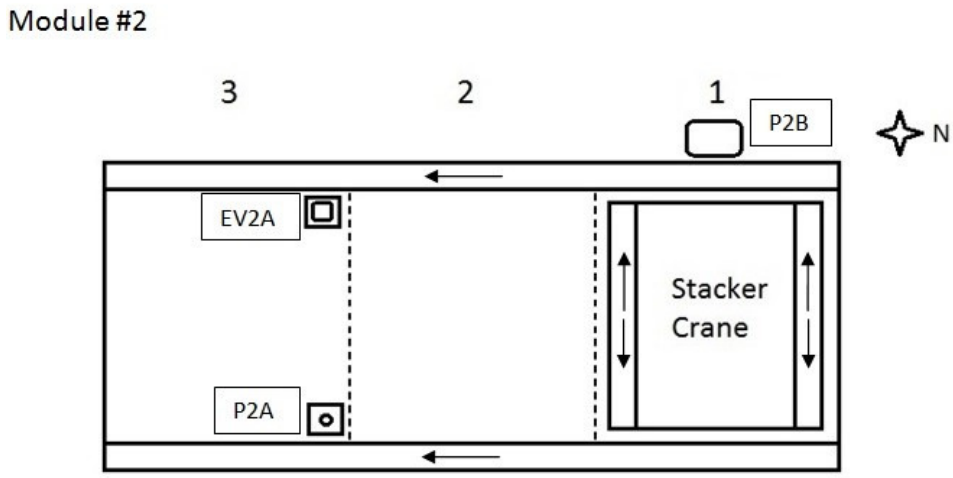


Figure B.2: Transport module N. 2.

- Two accumulation areas $(1;3)$;
- Mono-directional module$'$s tracks;
- Bi-directional stacker crane$'$s tracks;

SENSORS:	
P2A	Presence sensor (piston lock)
S2A-S2B	Stacker crane Up/Down
P1B; P1C	Presence sensor (setbacks)
S2B	Sliding presence sensor

ACTUATORS:	
M2F	Module$'$s tracks Forward
m2R-m2L	Stacker crane tracks Right/Left
EV2A	Piston Lock Down
EV2BU-EV2BD	Stacker crane Up/Down

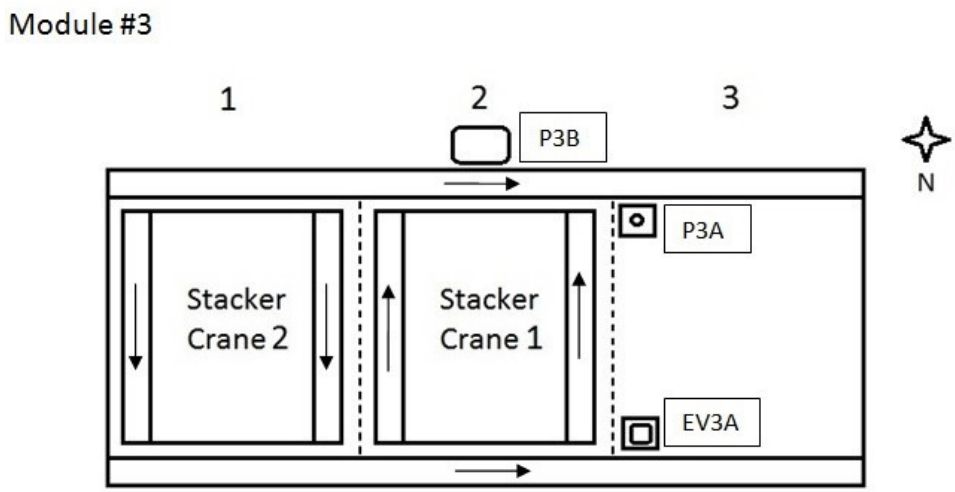


Figure B.3: Transport module N. 3.

- Two accumulation areas $(2;3)$, the pallet can not stop in the buffer zone number 1;
- Two stacker cranes;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P3A	Presence sensor (piston lock)
S3A-S3B	Stacker crane 1 Up/Down
P3B	Sliding presence sensor
S3C-S3D	Stacker crane 2 Up/Down

ACTUATORS:	
M3F	Module′s tracks Forward
m3aL	Stacker crane′s tracks 1 Left
m3bR	Stacker crane′s tracks 2 Right
EV3A	Piston Lock Down
EV3BU-EV3BD	Stacker crane 1 Up/Down
EV3CU-EV3CD	Stacker crane 2 Up/Down

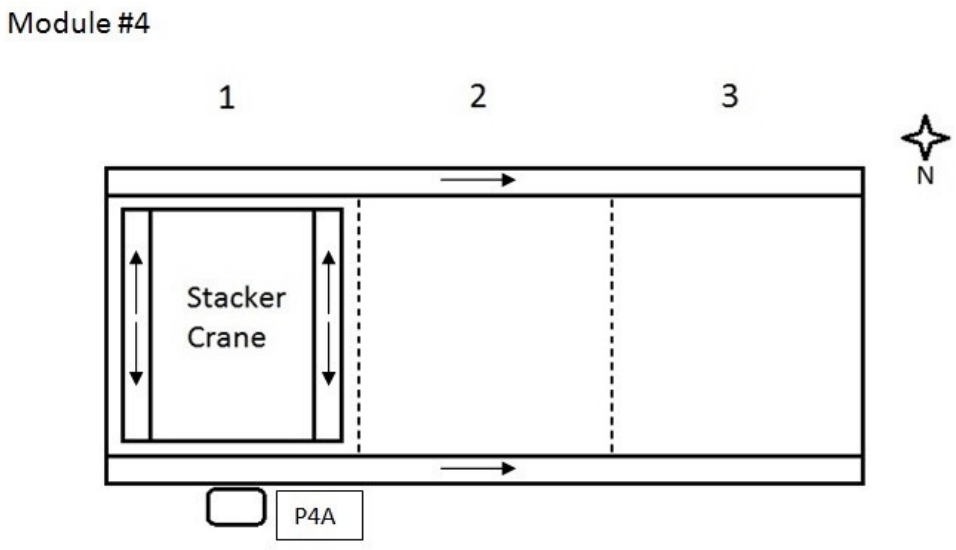


Figure B.4: Transport module N. 4.

- One accumulation area (1);
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:

P4A	Sliding presence sensor
S4A-S4B	Stacker crane Up/Down

ACTUATORS:

M4F	Module′s tracks Forward
m4aR-m4aL	Stacker crane′s tracks Right/Left
EV4AU-EV4AD	Stacker crane Up/Down

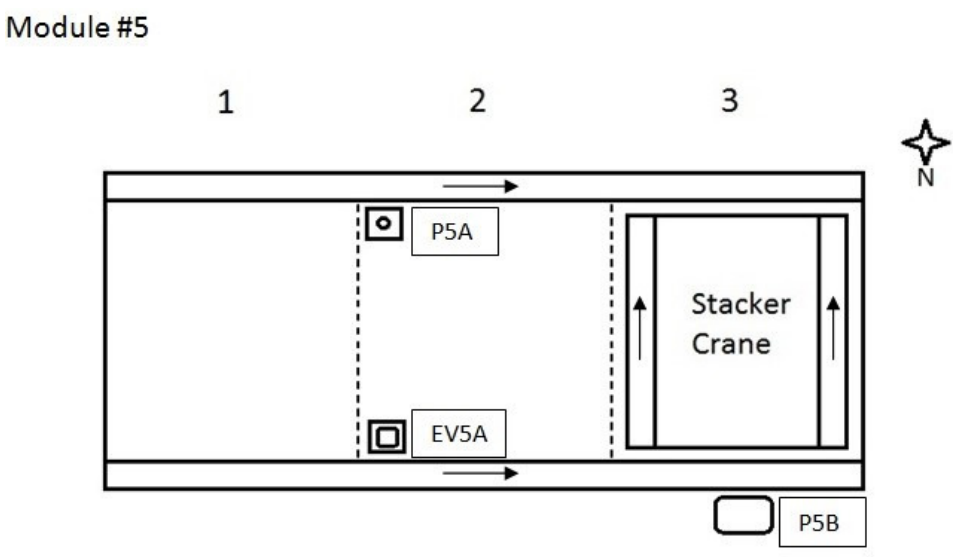


Figure B.5: Transport module N. 5.

- Two accumulation areas $(2;3)$;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P5A	Presence sensor (piston lock)
S5A-S5B	Stacker crane 1 Up/Down
P5B	Sliding presence sensor

ACTUATORS:	
M5F	Module′s tracks Forward
m5L	Stacker crane′s tracks 1 Left
EV5A	Piston Lock Down
EV5BU-EV5BD	Stacker crane Up/Down

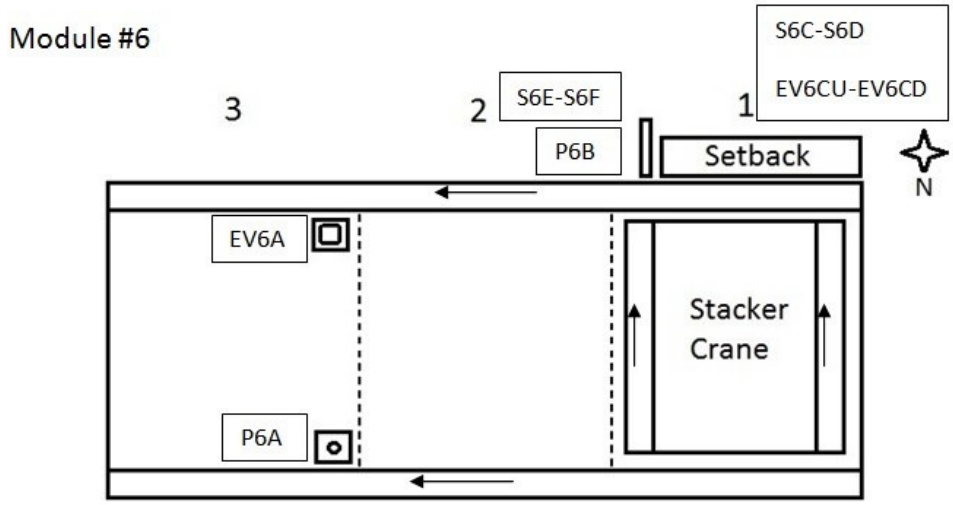


Figure B.6: Transport module N. 6.

- Two accumulation areas $(1;3)$;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P6A	Presence sensor (piston lock)
S6A-S6B	Stacker crane 1 Up/Down
P6B	Presence sensor (setback)
S6C-S6D	Setback Up/Down
S6E-S6F	Presence sensor (setback) Up/Down

ACTUATORS:	
M6F	Module′s tracks Forward
m6R	Stacker crane′s tracks Right
EV6A	Piston Lock Down
EV6BU-EV6BD	Stacker crane Up/Down
EV6CU-EV6CD	Setback Up/Down

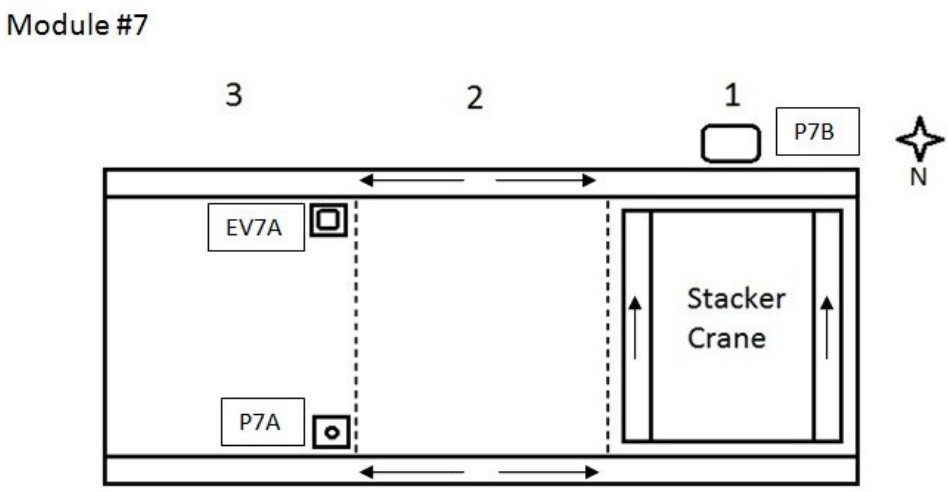


Figure B.7: Transport module N. 7.

- Two accumulation areas $(1;3)$;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P7A	Presence sensor (piston lock)
S7A-S7B	Stacker crane Up/Down
P7B	Sliding presence sensor
ACTUATORS:	
M7F-M7B	Module′s tracks Forward/Backward
m7R	Stacker crane′s tracks Right
EV7A	Piston Lock Down
EV7BU-EV7BD	Stacker crane Up/Down

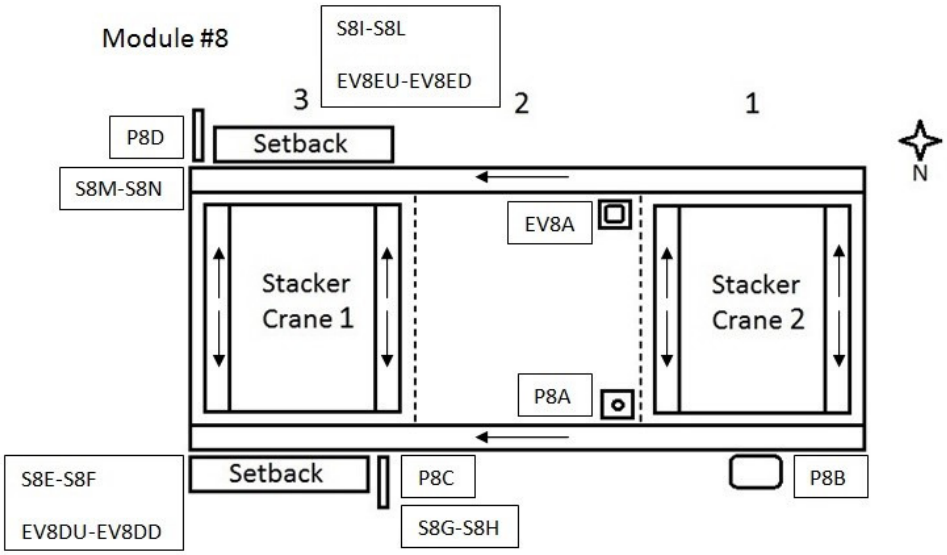


Figure B.8: Transport module N. 8.

- Three accumulation areas $(1;2;3)$;
- Two stacker cranes;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P8A	Presence sensor (piston lock)
S8A-S8B	Stacker crane 1 Up/Down
P8B	Sliding presence sensor
S8C-S8D	Stacker crane 2 Up/Down
P8C-P8D	Presence sensor (setbacks)
S8G-S8H; S8M-S8N	Presence sensor (setbacks) Up/Down
S8E-S8F; S8I-S8L	Setback Up/Down

ACTUATORS:	
M8F	Module′s tracks Forward
m8aR-m8aL	Stacker crane′s tracks 1 Right/Left
m8bR-m8bL	Stacker crane′s tracks 2 Right/Left
EV8A	Piston Lock Down

EV8BU-EV8BD	Stacker crane 1 Up/Down
EV8CU-EV8CD	Stacker crane 2 Up/Down
EV8DU-EV8DD; EV8EU-EV8ED	Setback Up/Down

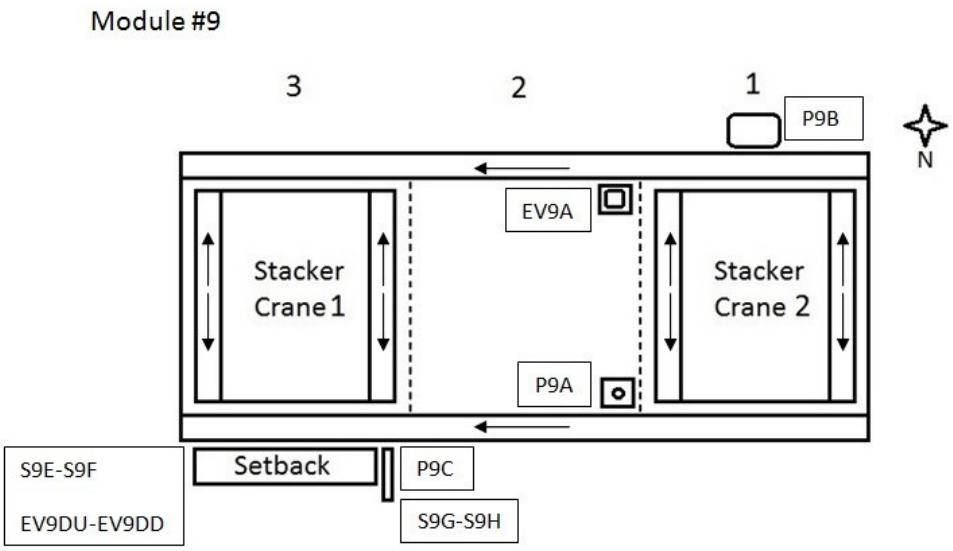


Figure B.9: Transport module N. 9.

- Three accumulation areas $(1;2;3)$;
- Two stacker cranes;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P9A	Presence sensor (piston lock)
S9A-S9B	Stacker crane 1 Up/Down
P9B	Sliding presence sensor
S9C-S9D	Stacker crane 2 Up/Down
P9C	Presence sensor (setbacks)
S9G-S9H	Presence sensor (setbacks) Up/Down
S9E-S9F	Setback Up/Down

ACTUATORS:	
M9F	Module′s tracks Forward
m9aR-m9aL	Stacker crane′s tracks 1 Right/Left
m9bR-m9bL	Stacker crane′s tracks 2 Right/Left
EV9A	Piston Lock Down

EV9BU-EV9BD	Stacker crane 1 Up/Down
EV9CU-EV9CD	Stacker crane 2 Up/Down
EV9DU-EV9DD	Setback Up/Down

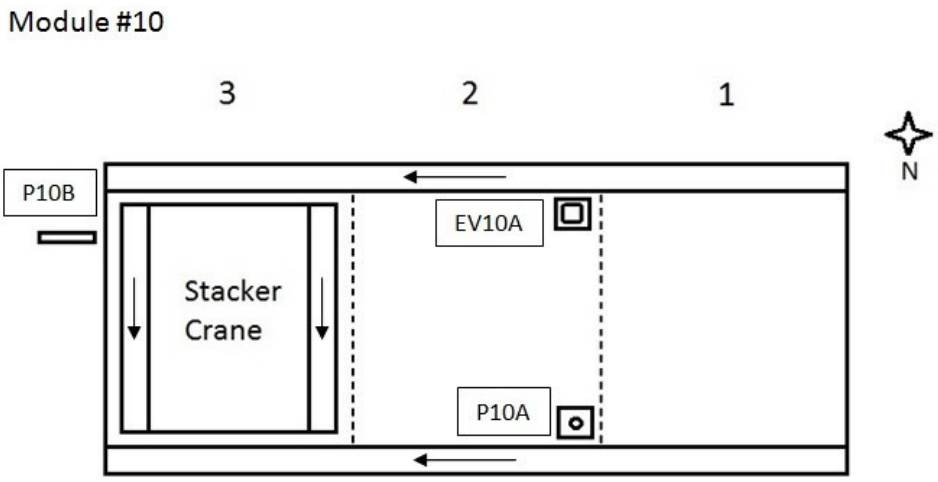


Figure B.10: Transport module N. 10.

- Two accumulation areas $(1;2)$;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P10A	Presence sensor (piston lock)
S10A-S10B	Stacker crane Up/Down
P10B	Presence sensor

ACTUATORS:	
M10F	Module′s tracks Forward
m10L	Stacker crane′s tracks Left
EV10A	Piston Lock Down
EV10BU-EV10BD	Stacker crane Up/Down

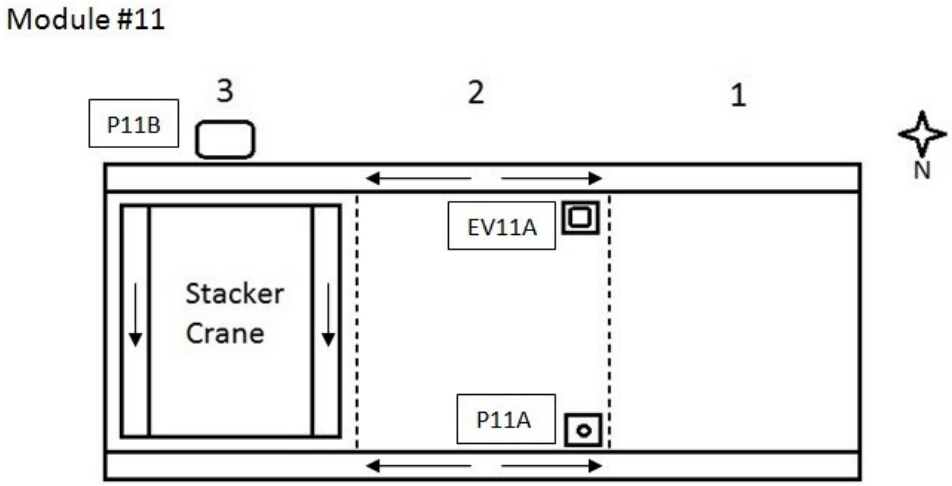


Figure B.11: Transport module N. 11.

- Two accumulation areas $(1;2)$;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P11A	Presence sensor (piston lock)
S11A-S11B	Stacker crane Up/Down
P11B	Sliding presence sensor
ACTUATORS:	
M11F/M11B	Module′s tracks Forward/Backward
m11L	Stacker crane′s tracks Left
EV11A	Piston Lock Down
EV11BU-EV11BD	Stacker crane Up/Down

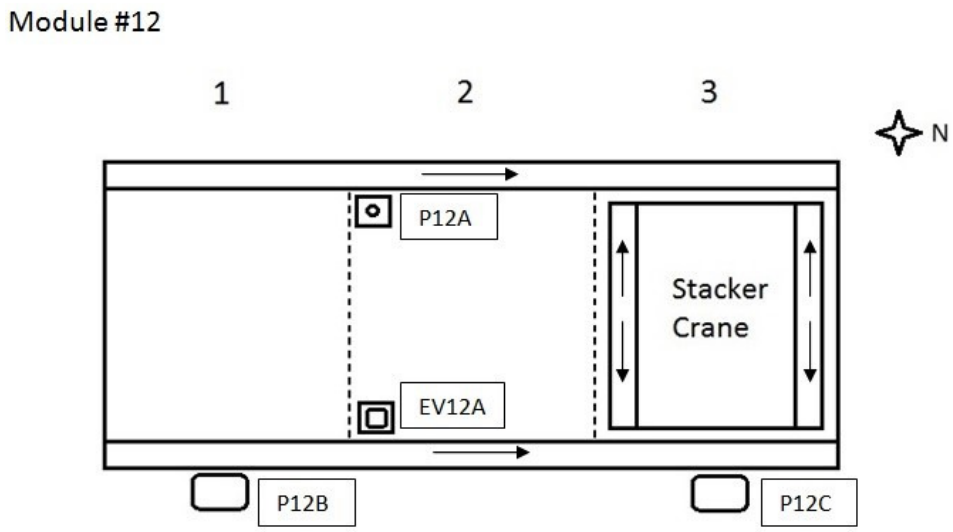


Figure B.12: Transport module N. 12.

- Two accumulation areas $(2;3)$;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;
- The sensor P12B is not used from the control system;

SENSORS:	
P12A	Presence sensor (piston lock)
S12A-S12B	Stacker crane Up/Down
P12B; P12C	Sliding presence sensor
ACTUATORS:	
M12F	Module′s tracks Forward
m12L-m12R	Stacker crane′s tracks Right/Left
EV12A	Piston Lock Down
EV12BU-EV12BD	Stacker crane Up/Down

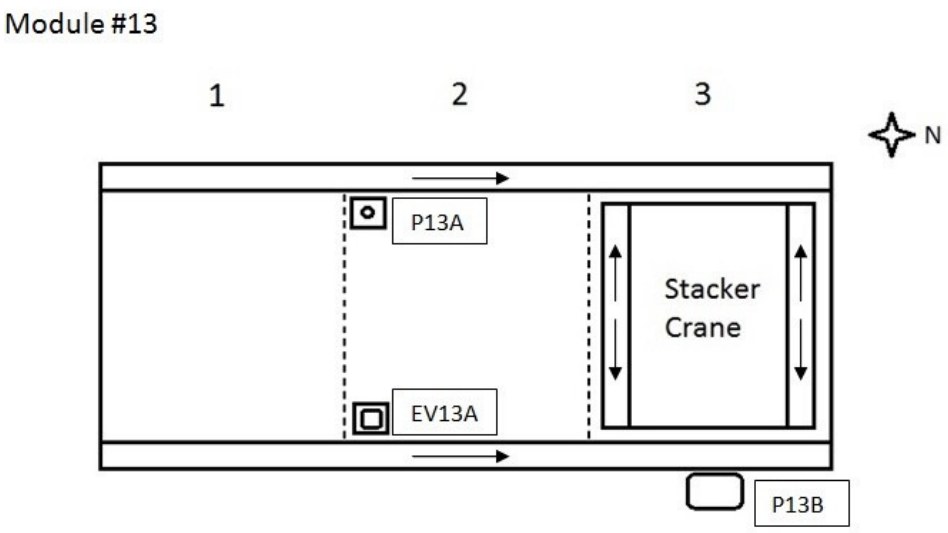


Figure B.13: Transport module N. 13.

- Two accumulation areas $(2;3)$;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P13A	Presence sensor (piston lock)
S13A-S13B	Stacker crane Up/Down
P13B	Sliding presence sensor

ACTUATORS:	
M13F	Module′s tracks Forward
m13L-m13R	Stacker crane′s tracks Right/Left
EV13A	Piston Lock Down
EV13BU-EV13BD	Stacker crane Up/Down

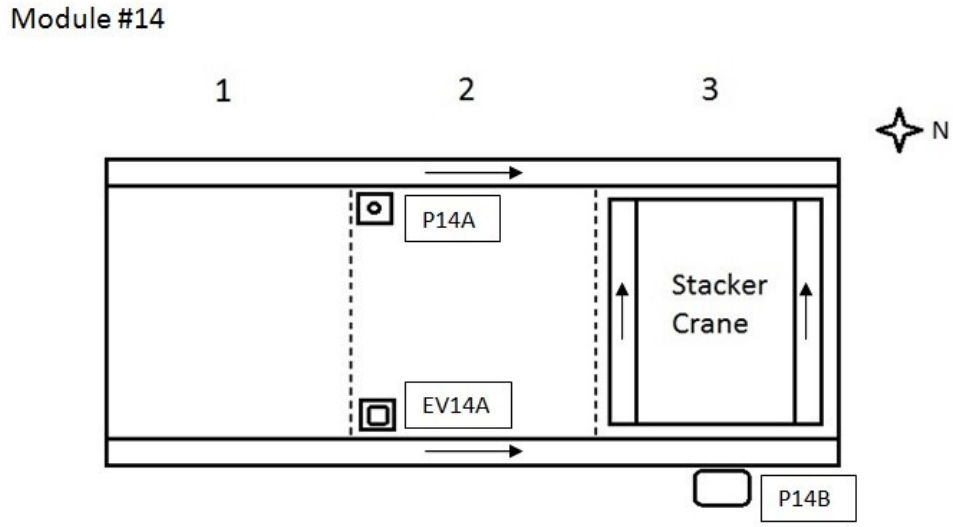


Figure B.14: Transport module N. 14.

- Two accumulation areas $(2;3)$;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P14A	Presence sensor (piston lock)
S14A-S14B	Stacker crane Up/Down
P14B	Sliding presence sensor

ACTUATORS:	
M14F	Module′s tracks Forward
m14L	Stacker crane′s tracks Left
EV14A	Piston Lock Down
EV14BU-EV14BD	Stacker crane Up/Down

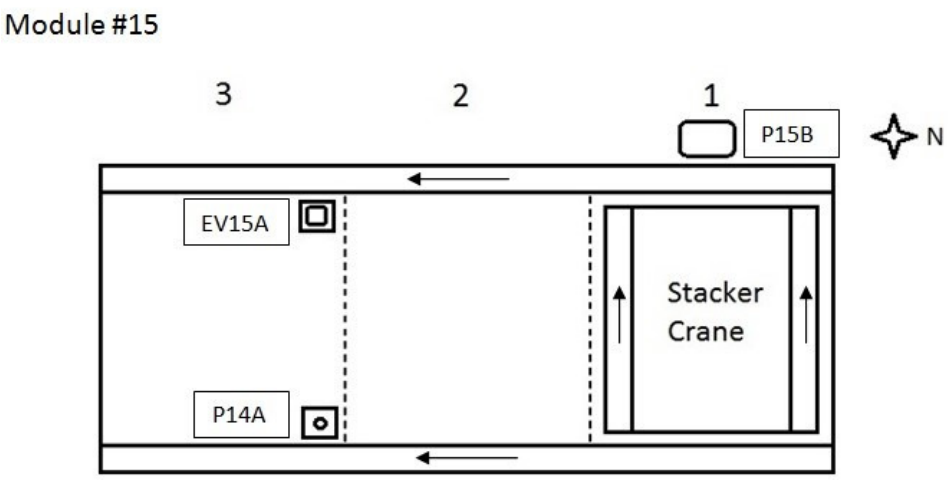


Figure B.15: Transport module N. 15.

- Two accumulation areas $(1;3)$;
- Mono-directional module′s tracks;
- Bi-directional stacker crane′s tracks;

SENSORS:	
P15A	Presence sensor (piston lock)
S15A-S15B	Stacker crane Up/Down
P15B	Sliding presence sensor

ACTUATORS:	
M15F	Module′s tracks Forward
m15R	Stacker crane′s tracks Right
EV15A	Piston Lock Down
EV15BU-EV15BD	Stacker crane Up/Down

B.2 Transport module control sequences

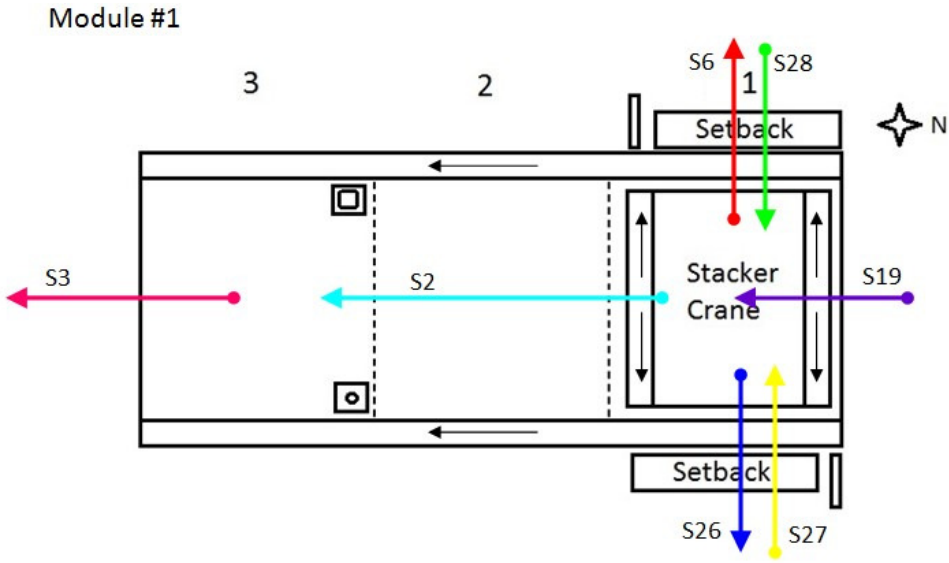


Figure B.16: Transport module N. 1 with sequences.

SEQUENCES:

S2:	from Stacker crane 1	to Piston lock
S3:	from Piston lock	to Next Module
S28:	from External Right (Setback Right) (Setback Left)	to Stacker crane 1
S27:	from External Left (Setback Left) (Setback Right)	to Stacker crane 1
S36:	from Stacker crane 1 (Robot cell)	to External Right
S26:	from Stacker crane 1 (Setback Right) (Setback Left)	to External left
S19:	from Previous Module	to Setback Right

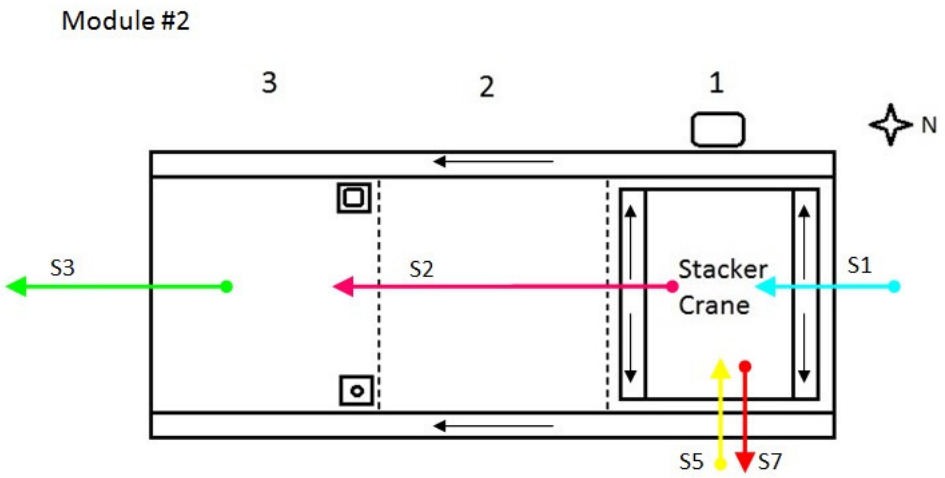


Figure B.17: Transport module N. 2 with sequences.

SEQUENCES:

S1:	from Previous module	to Stacker crane 1
S2:	from Stacker crane 1	to Piston lock
S3:	from Piston lock	to Next Module
S7:	from Stacker crane 1	to External Left
S5:	from External Left	to Stacker crane 1

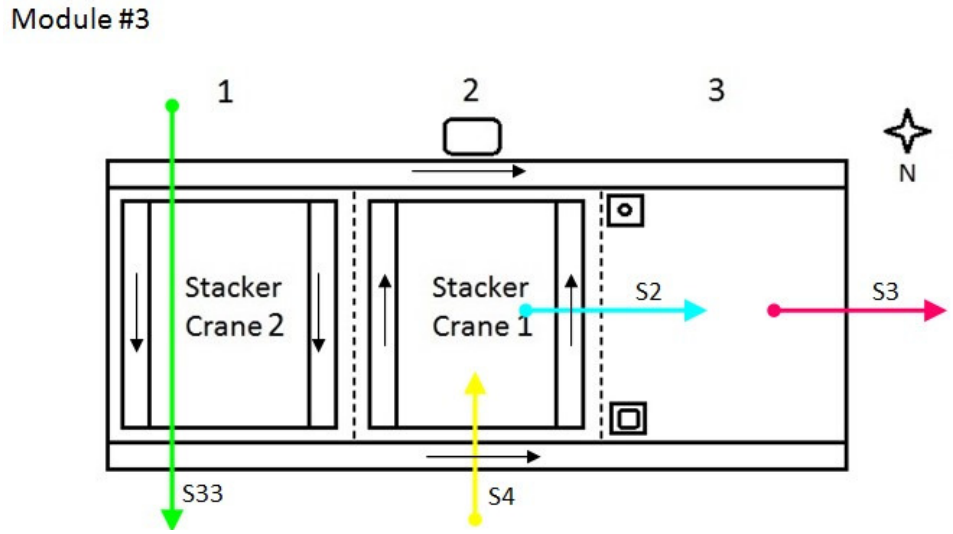


Figure B.18: Transport module N. 3 with sequences.

SEQUENCES:

S2:	from Stacker crane 1	to Piston lock
S3:	from Piston lock	to Next Module
S33:	from External Left (through SC2)	to External Right
S4:	from External Right	to Stacker crane 1

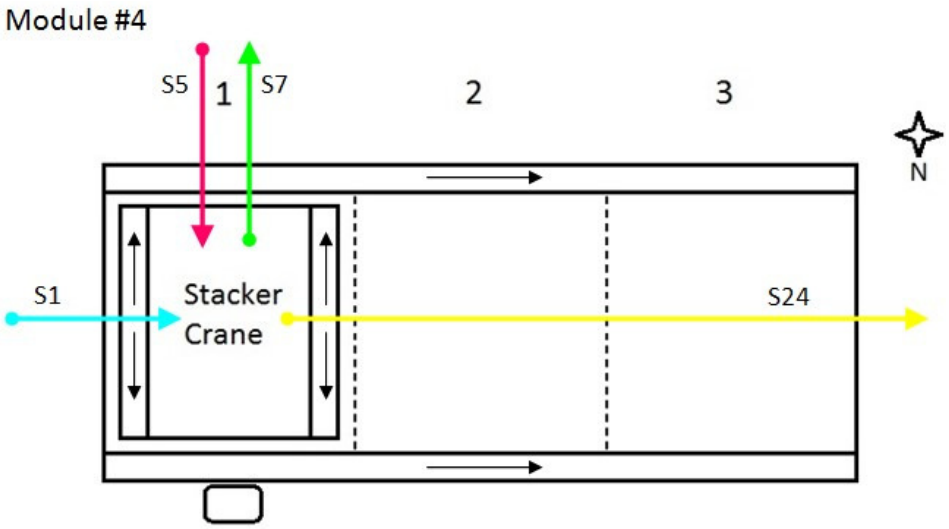


Figure B.19: Transport module N. 4 with sequences.

SEQUENCES:

S1:	from Previous module	to Stacker crane 1
S5:	from External Left	to Stacker crane 1
S7:	from Stacker crane 1	to External Left
S24:	from Stacker crane 1 (2 Buffer Zones)	to Next Module

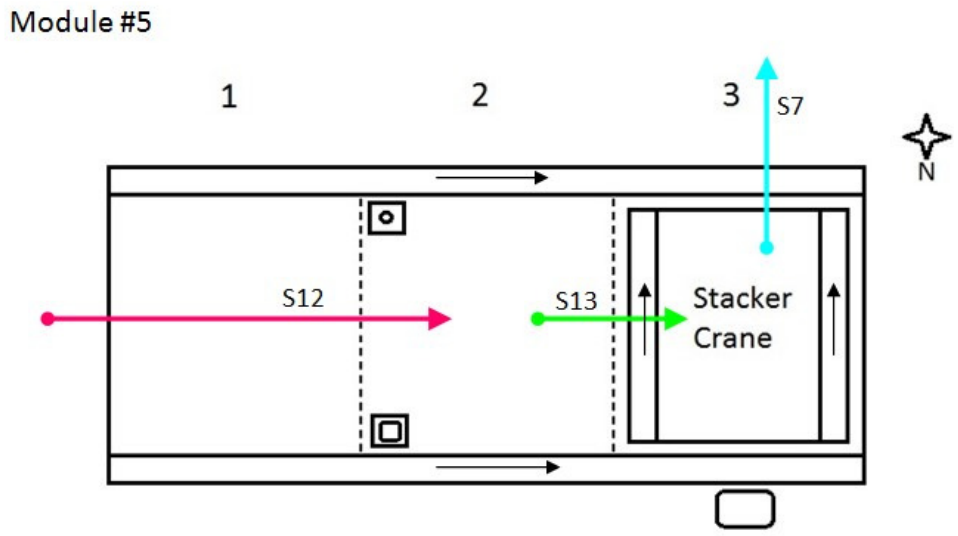


Figure B.20: Transport module N. 5 with sequences.

SEQUENCES:

S7:	from Stacker crane 1	to External Left
S12:	from Previous module	to Piston lock
S13:	from Piston lock	to Stacker crane 1

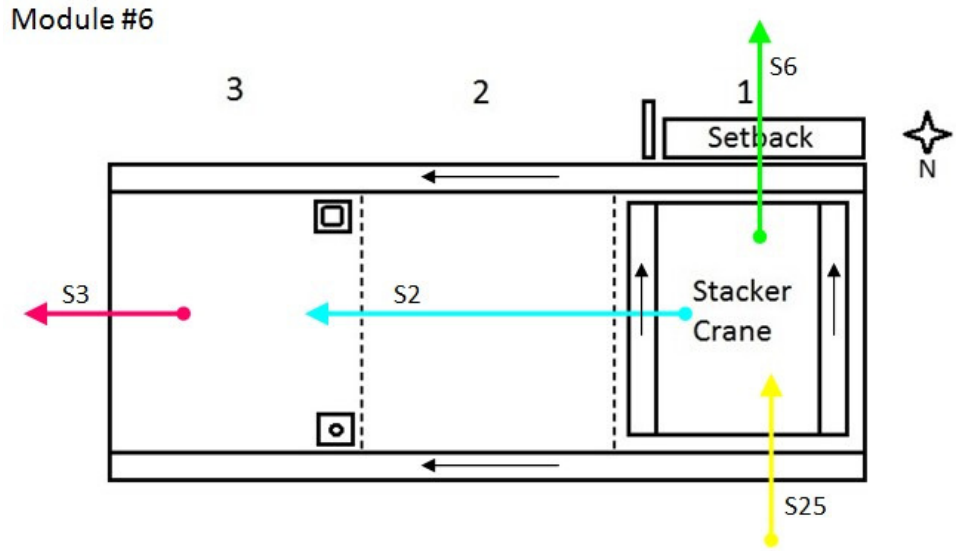


Figure B.21: Transport module N. 6 with sequences.

SEQUENCES:

S2:	from Stacker crane 1	to Piston lock
S3:	from Piston lock	to Next module
S6:	from Stacker crane 1	to External Right
S25:	from External Left (Setback Right)	to Stacker crane 1

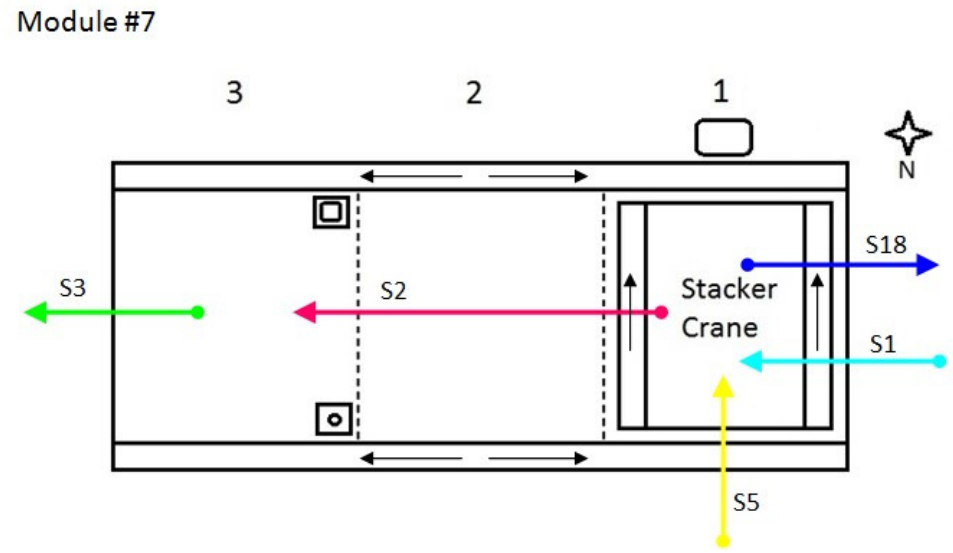


Figure B.22: Transport module N. 7 with sequences.

SEQUENCES:

S1:	from Previous module	to Stacker crane 1
S2:	from Stacker crane 1	to Piston lock
S3:	from Piston lock	to Next module
S5:	from External Left	to Stacker crane 1
S18:	from Stacker crane 1	to Previous module

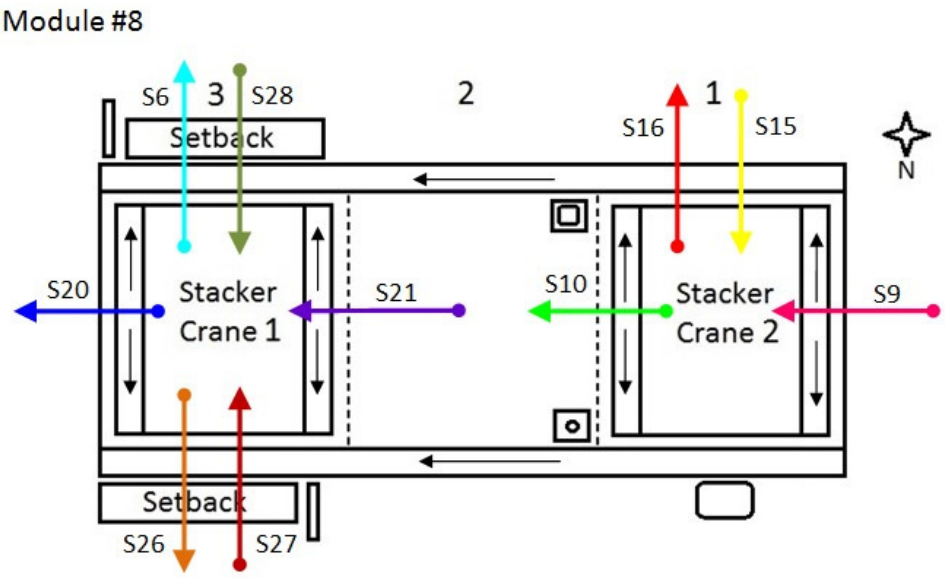


Figure B.23: Transport module N. 8 with sequences.

SEQUENCES:

S6:	from Stacker crane 1	to External Right
S9:	from Previous module	to Stacker crane 2
S10:	from Stacker crane 2	to Piston lock
S15:	from Externa Right	to Stacker crane 2
S16:	from Stacker crane 2	to External Right
S20:	from Setback Right	to Next module
S21:	from Piston lock	to Setback Right
S26:	from Stacker crane 1 (Setback Right) (Setback Left)	to External Left
S27:	from External Left (Setback Left) (Setback Right)	to Stacker crane 1
S28:	from External Right (Setback Right) (Setback Left)	to Stacker crane 1

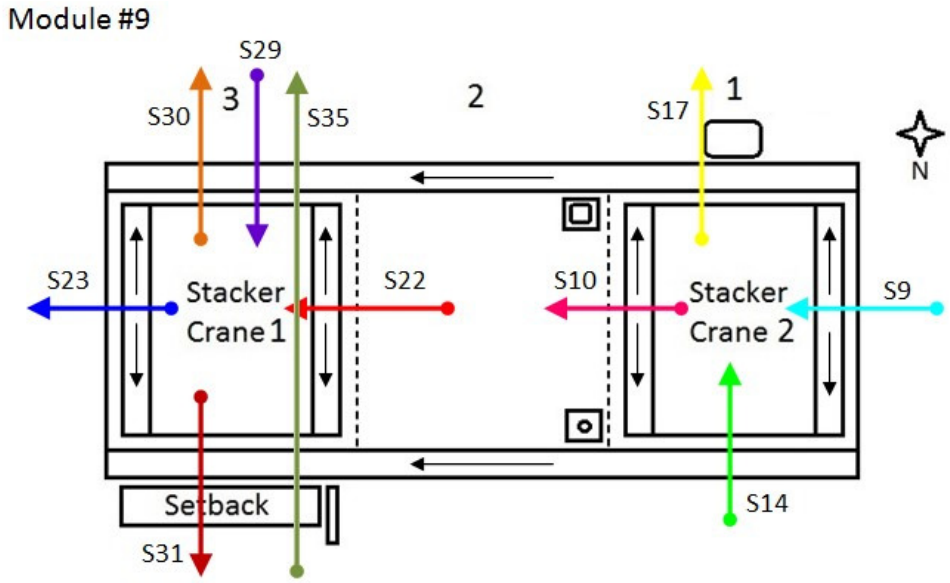


Figure B.24: Transport module N. 9 with sequences.

SEQUENCES:

S9:	from Previous module	to Stacker crane 2
S10:	from Stacker crane 2	to Piston lock
S14:	from External Left	to Stacker crane 2
S17:	from Stacker crane 2	to External Left
S22:	from Piston lock	to Setback Left
S23:	from Setback Left	to Next module
S29:	from External Right (Setback Left)	to Stacker crane 1
S30:	from Stacker crane 1 (Setback Left)	to External Right
S31:	from Stacker crane 1 (Setback Left)	to External Left
S35:	from External Left (Setback Left) (Stacker crane 1)	to External Right

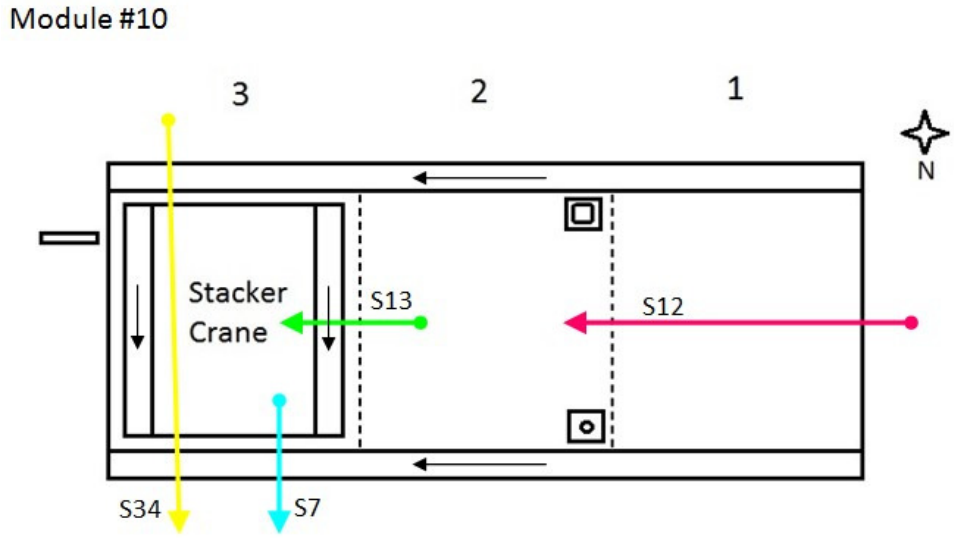


Figure B.25: Transport module N. 10 with sequences.

SEQUENCES:

S7:	from Stacker crane 1	to External Left
S12:	from Previous module	to Piston lock
S13:	from Piston lock	to Stacker crane 1
S34:	from External Right (through SC1)	to External Left

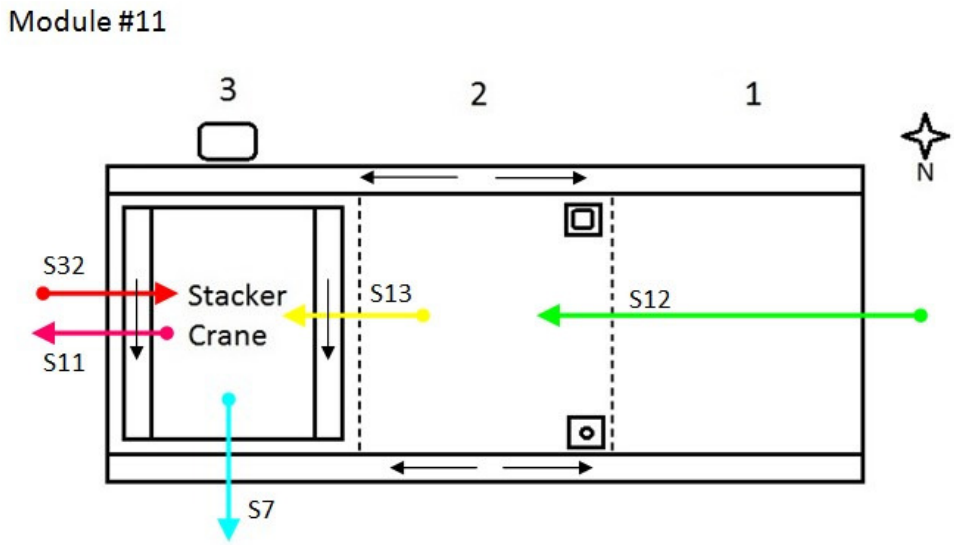


Figure B.26: Transport module N. 11 with sequences.

SEQUENCES:

S7:	from Stacker crane 1	to External Left
S8:	from Stacker crane 1 (Discharge Cell)	to Next module
S12:	from Previous module	to Piston lock
S32:	from Next module	to Stacker crane 1

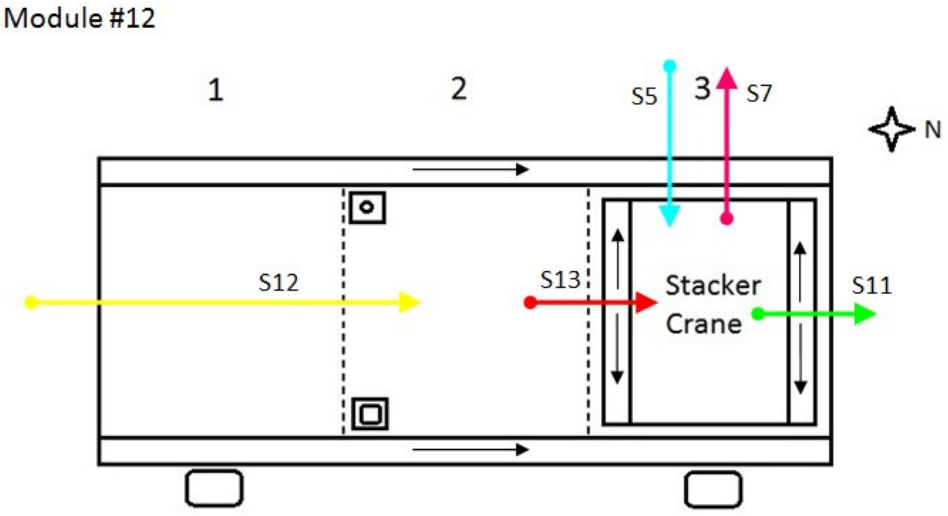


Figure B.27: Transport module N. 12 with sequences.

SEQUENCES:

S5:	from External Left	to Stacker crane 1
S7:	from Stacker crane 1	to External Left
S11:	from Stacker crane 1 (Buffer Zone 1)	to Next module
S12:	from Previous module	to Piston lock
S13:	from Piston lock	to Stacker crane 1

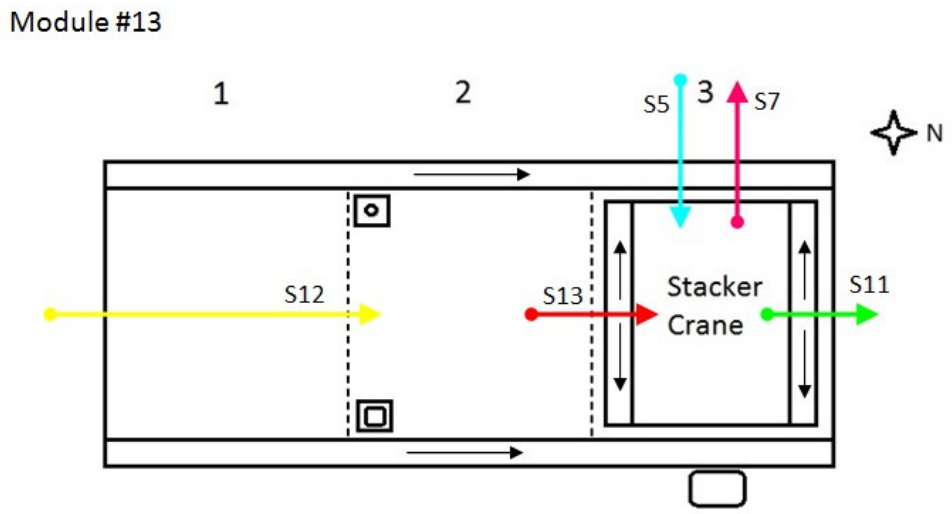


Figure B.28: Transport module N. 13 with sequences.

SEQUENCES:

S5:	from External Left	to Stacker crane 1
S7:	from Stacker crane 1	to External Left
S11:	from Stacker crane 1 (Buffer Zone 1)	to Next module
S12:	from Previous module	to Piston lock
S13:	from Piston lock	to Stacker crane 1

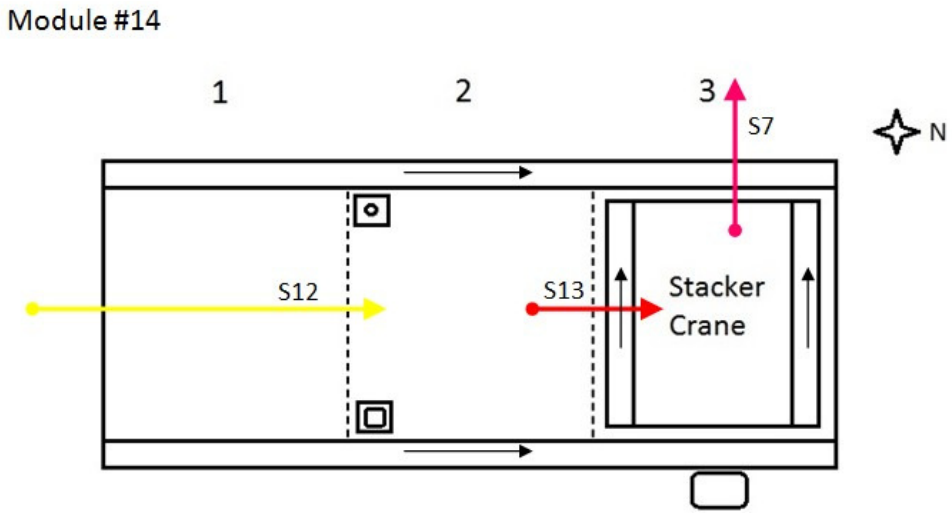


Figure B.29: Transport module N. 14 with sequences.

SEQUENCES:

S7:	from Stacker crane 1	to External Left
S12:	from Previous module	to Piston lock
S13:	from Piston lock	to Stacker crane 1

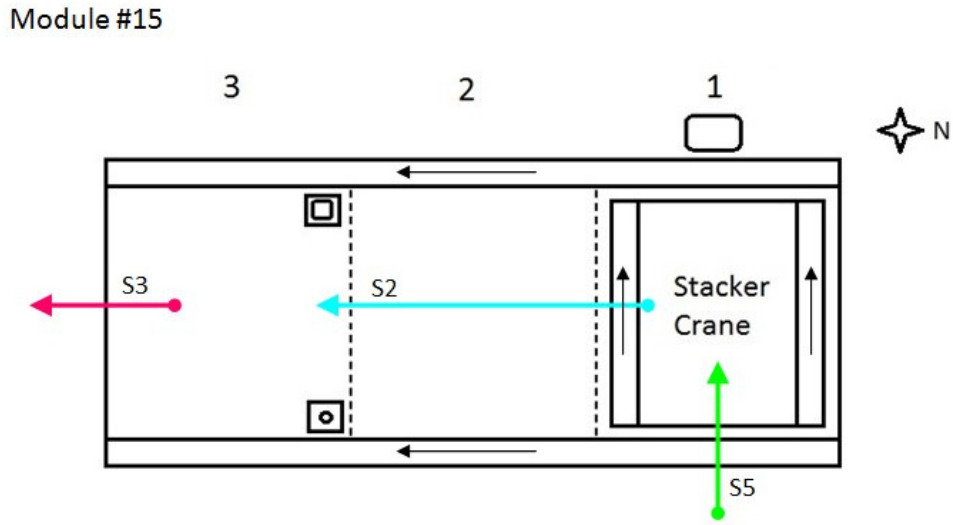


Figure B.30: Transport module N. 15 with sequences.

SEQUENCES:

S2:	from Stacker crane 1	to Piston lock
S3:	from Piston lock	to Next module
S5:	from External Left	to Stacker crane 1

B.3 Finite state machine control sequences

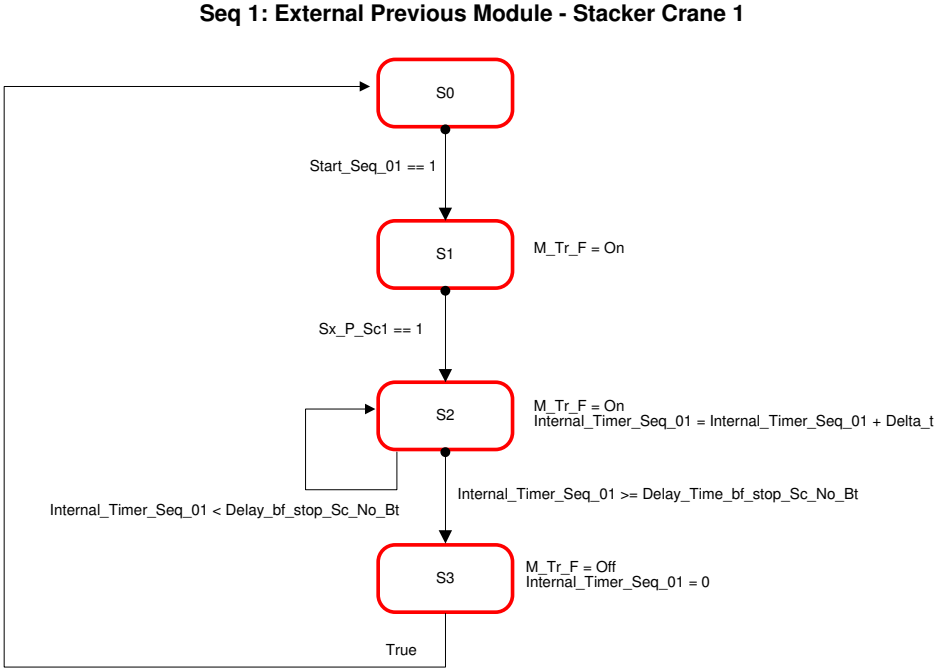


Figure B.31: LLCS sequence N. 1.

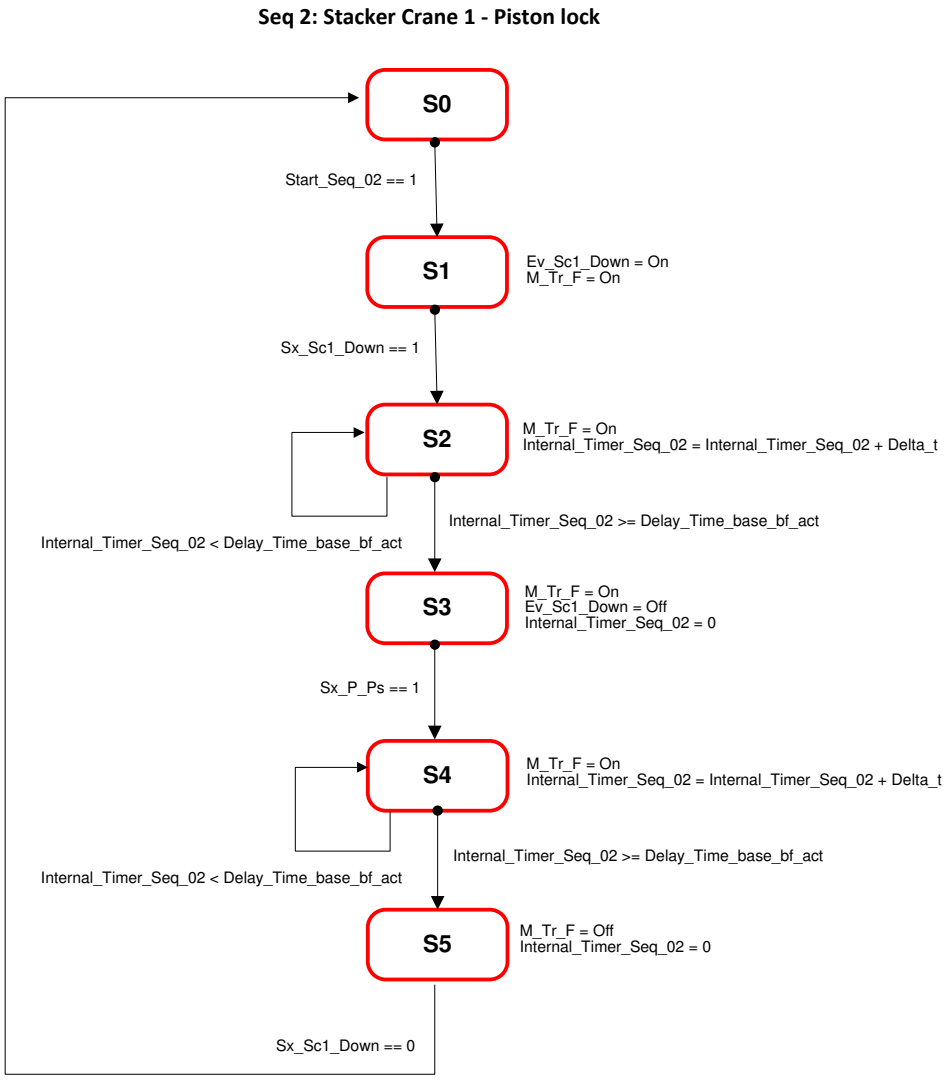


Figure B.32: LLCS sequence N. 2.

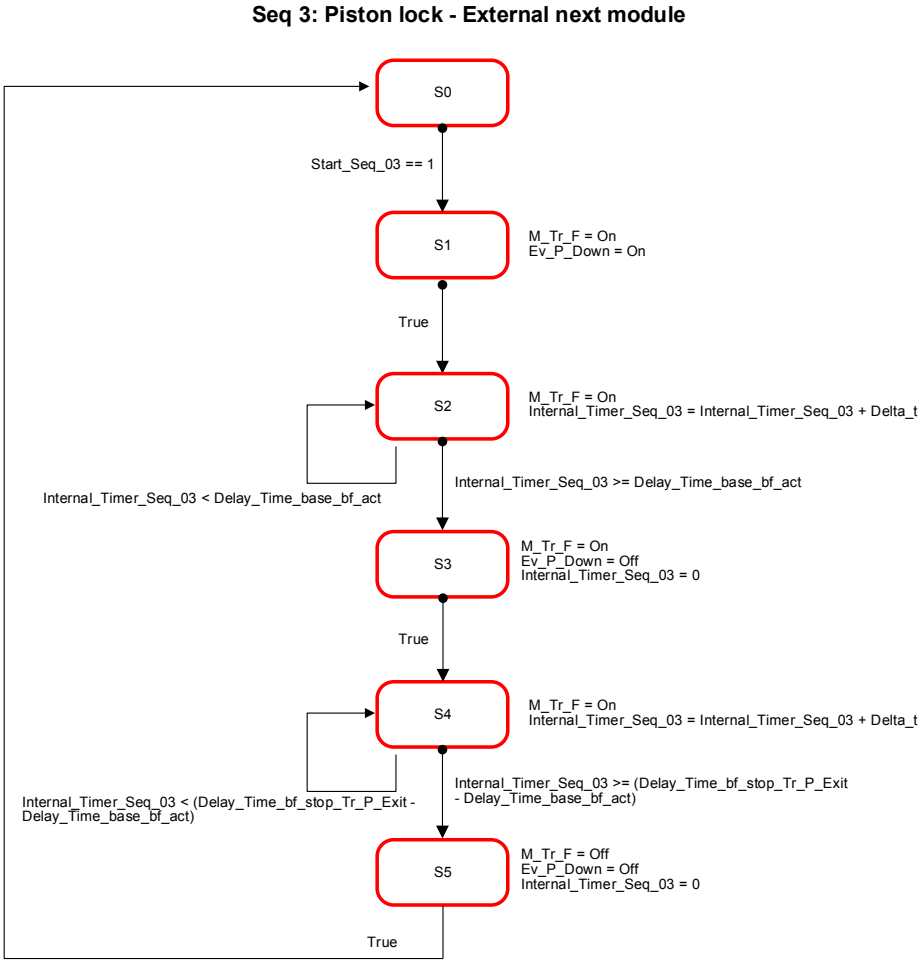


Figure B.33: LLCS sequence N. 3.

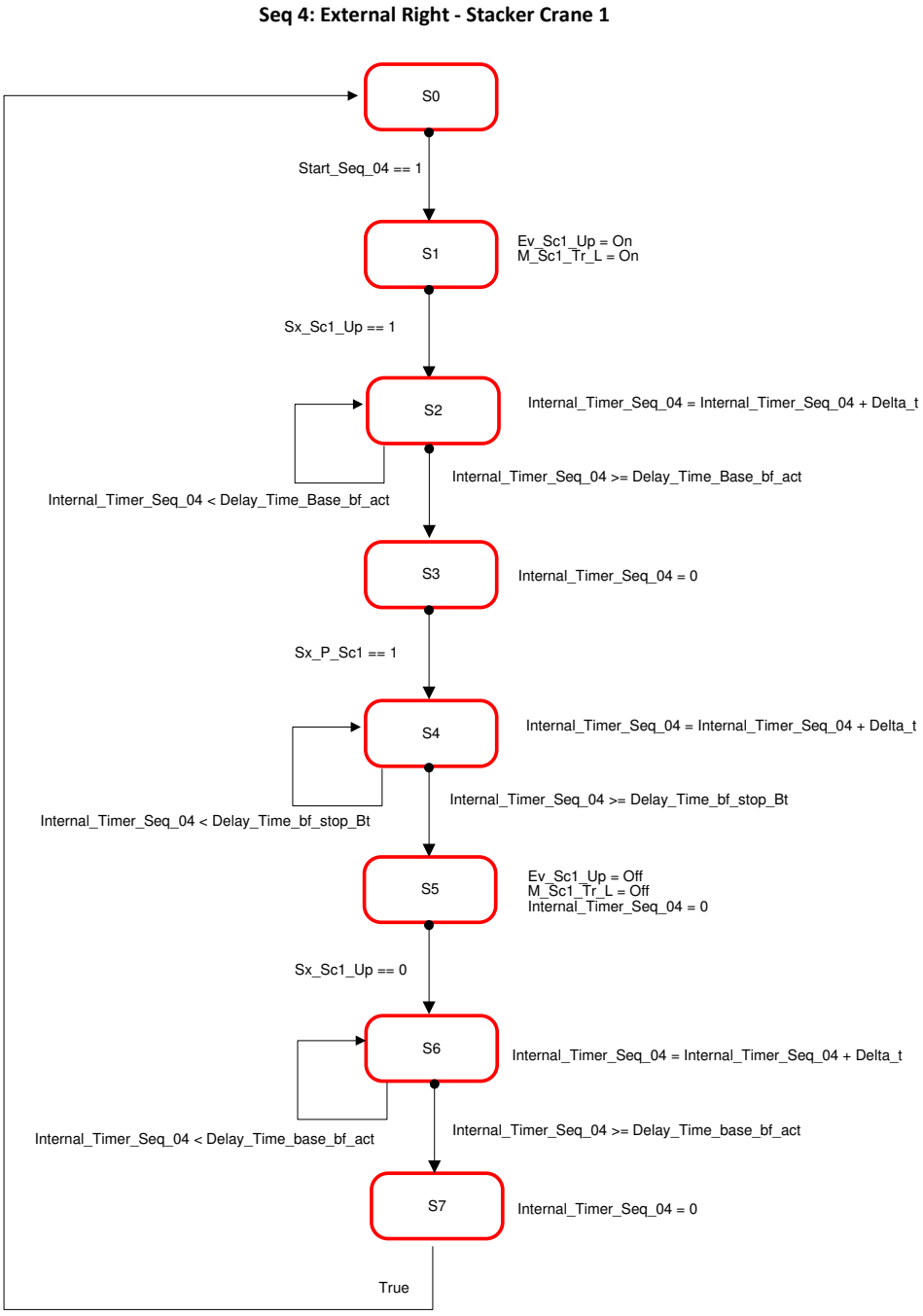


Figure B.34: LLCS sequence N. 4.

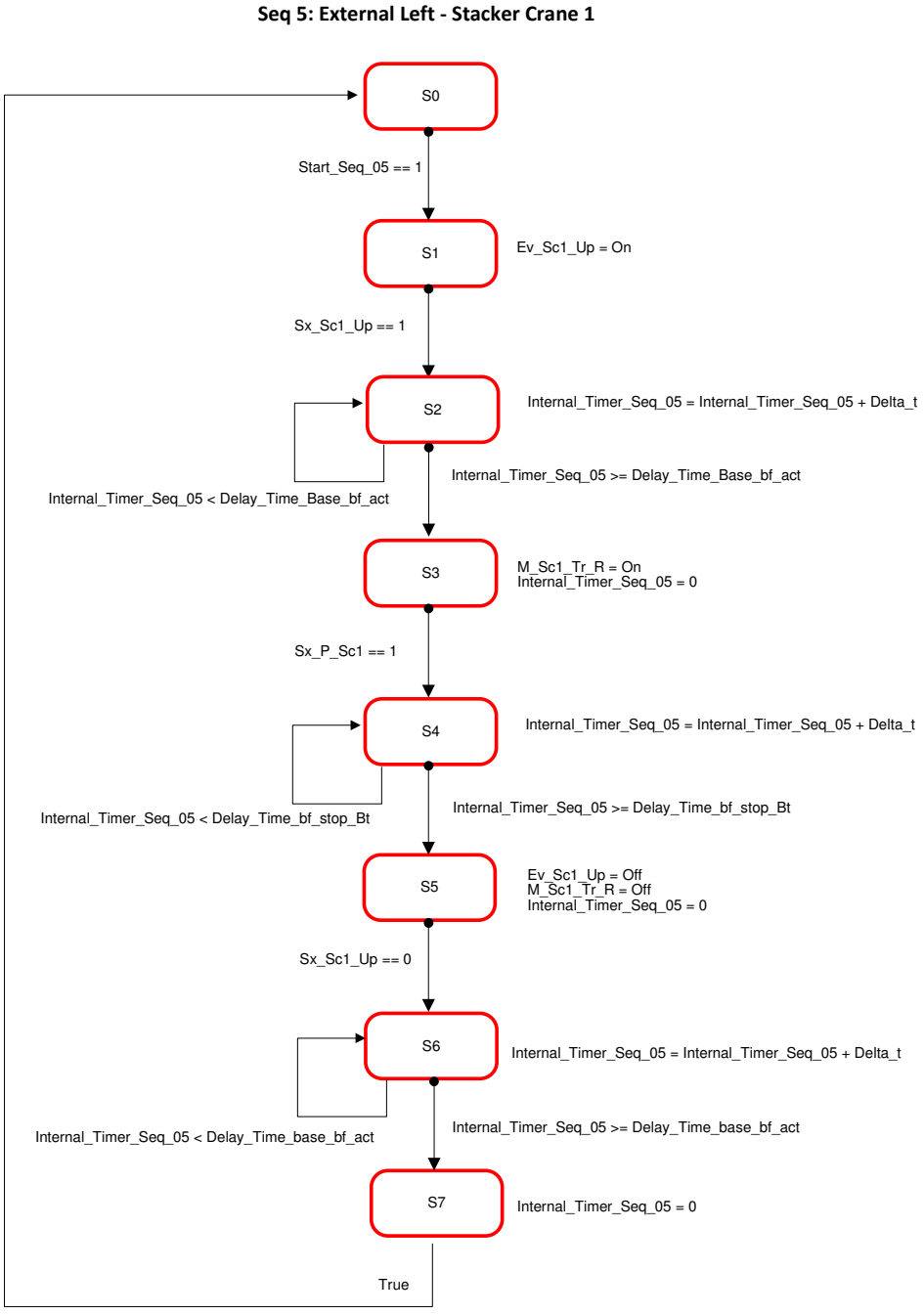


Figure B.35: LLCS sequence N. 5.

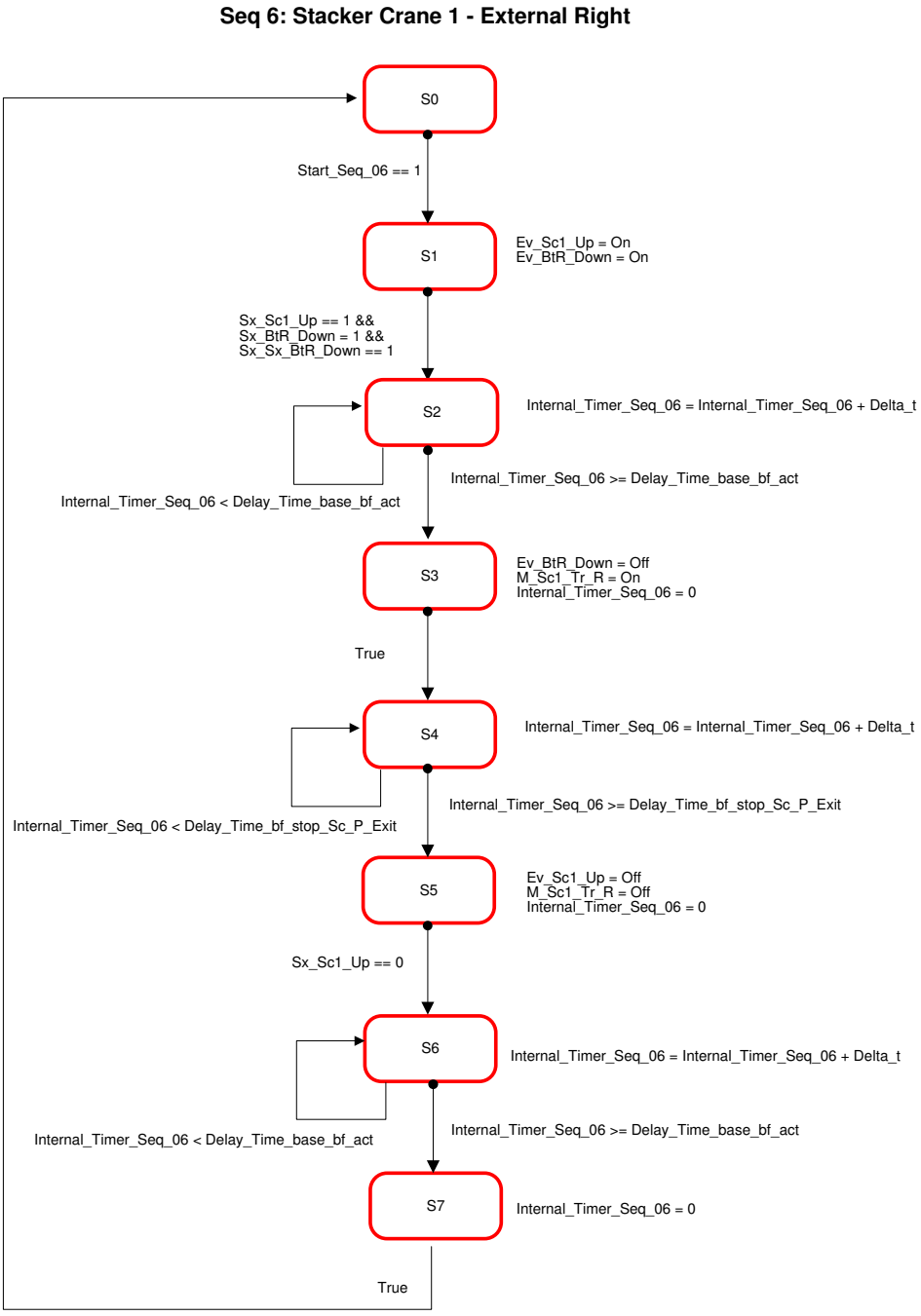


Figure B.36: LLCS sequence N. 6.

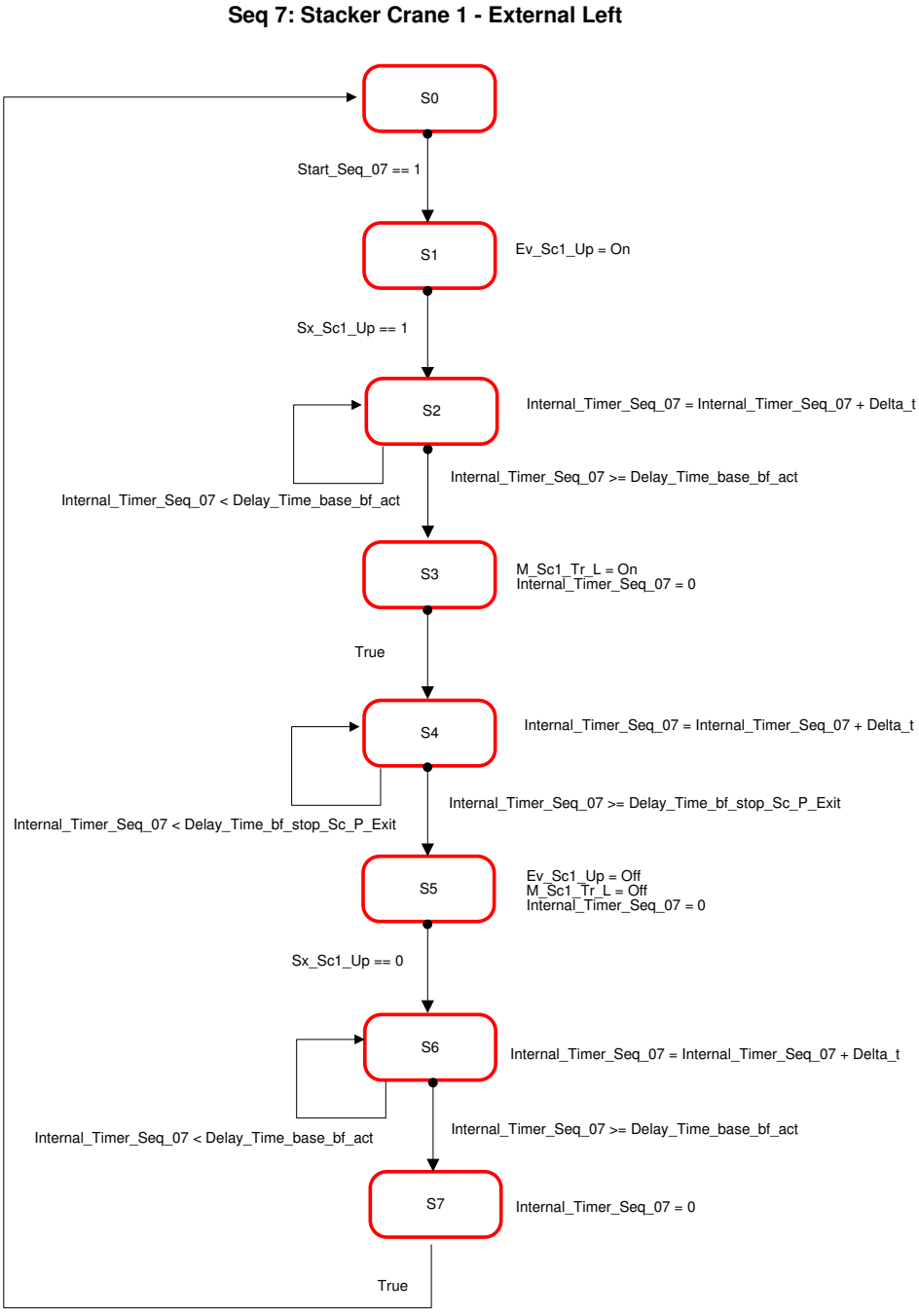


Figure B.37: LLCS sequence N. 7.

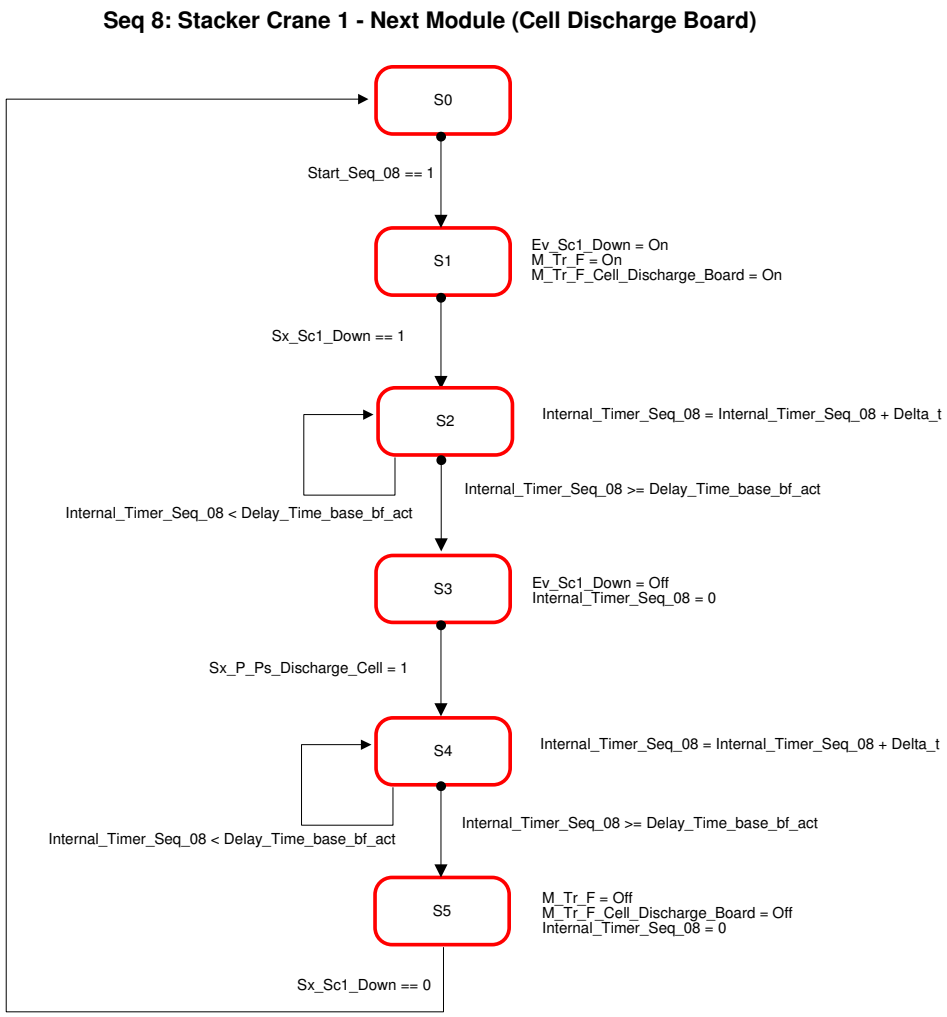


Figure B.38: LLCS sequence N. 8.

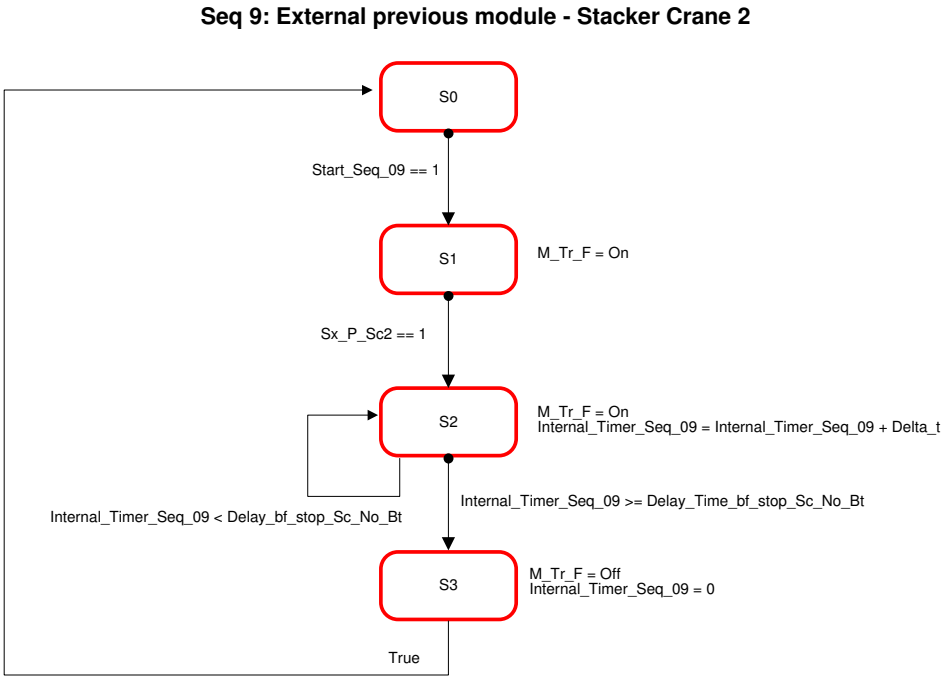


Figure B.39: LLCS sequence N. 9.

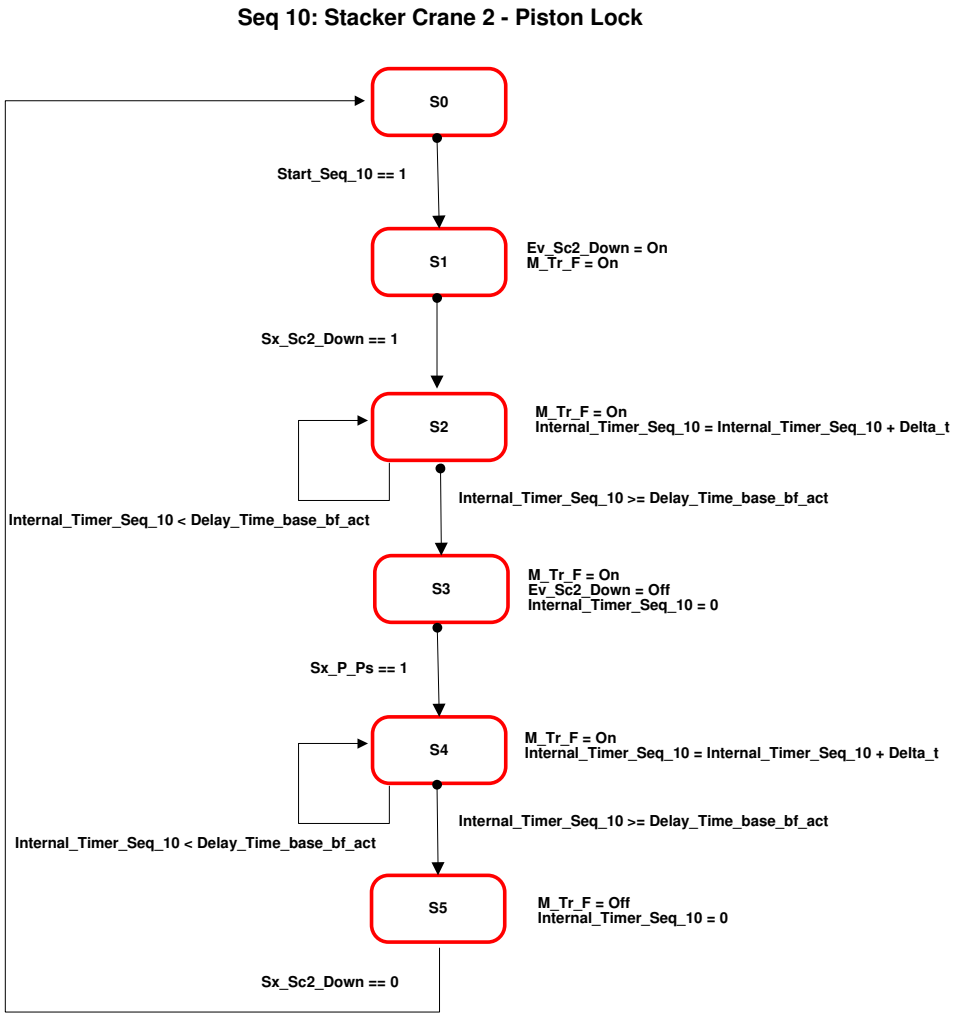


Figure B.40: LLCS sequence N. 10.

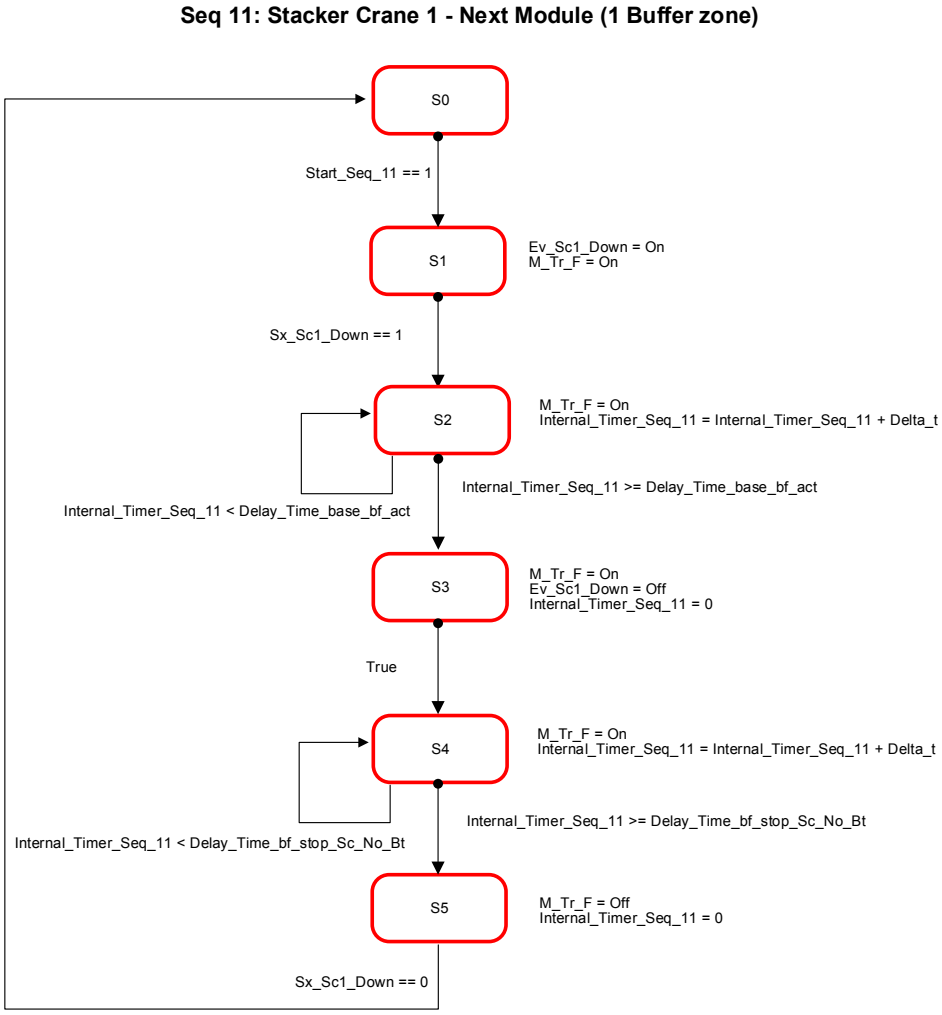


Figure B.41: LLCS sequence N. 11.

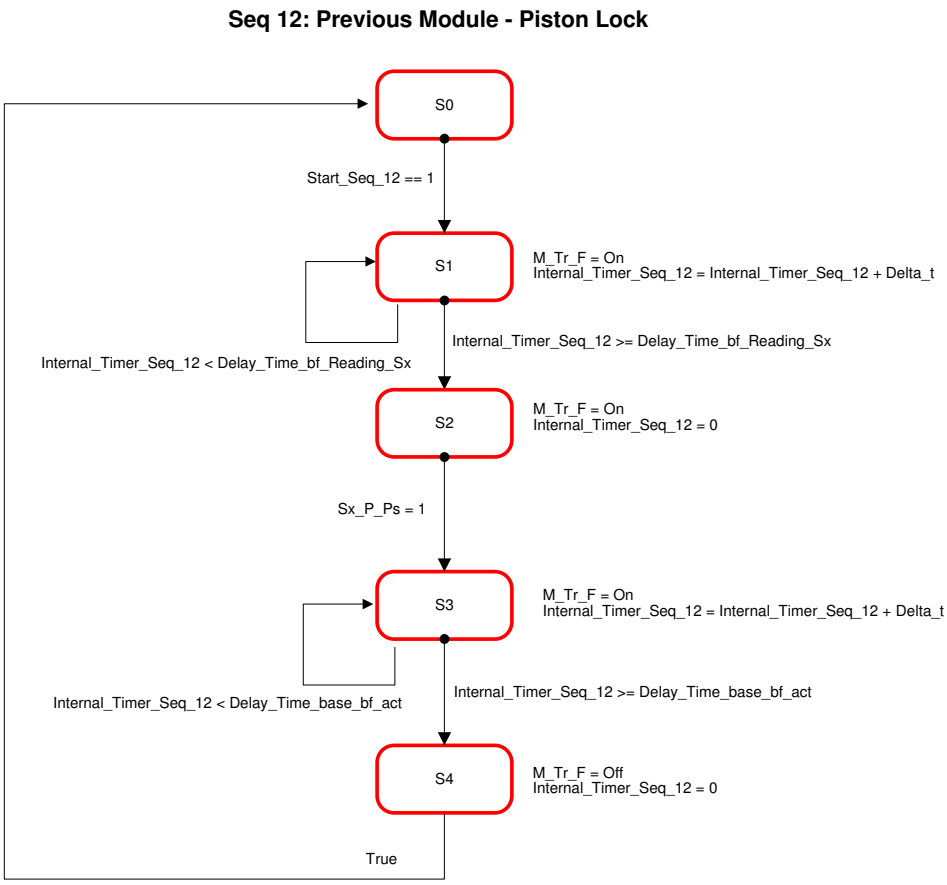


Figure B.42: LLCS sequence N. 12.

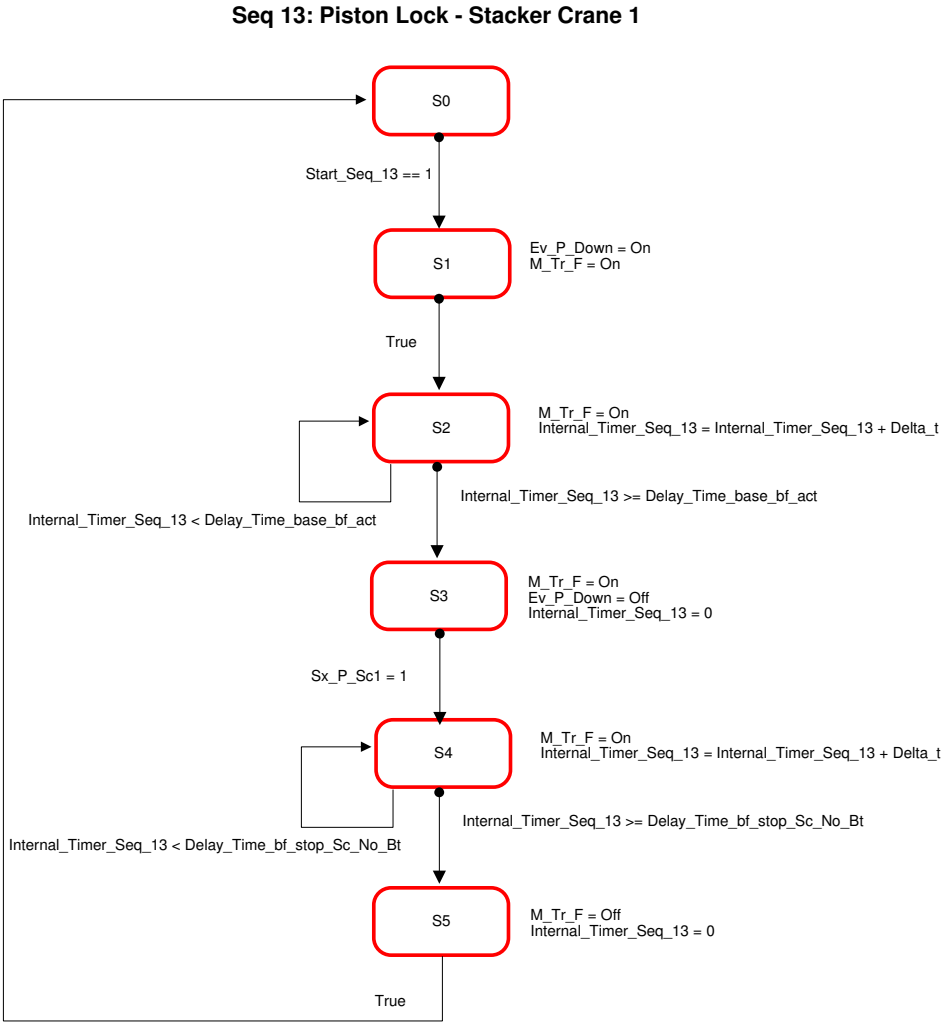


Figure B.43: LLCS sequence N. 13.

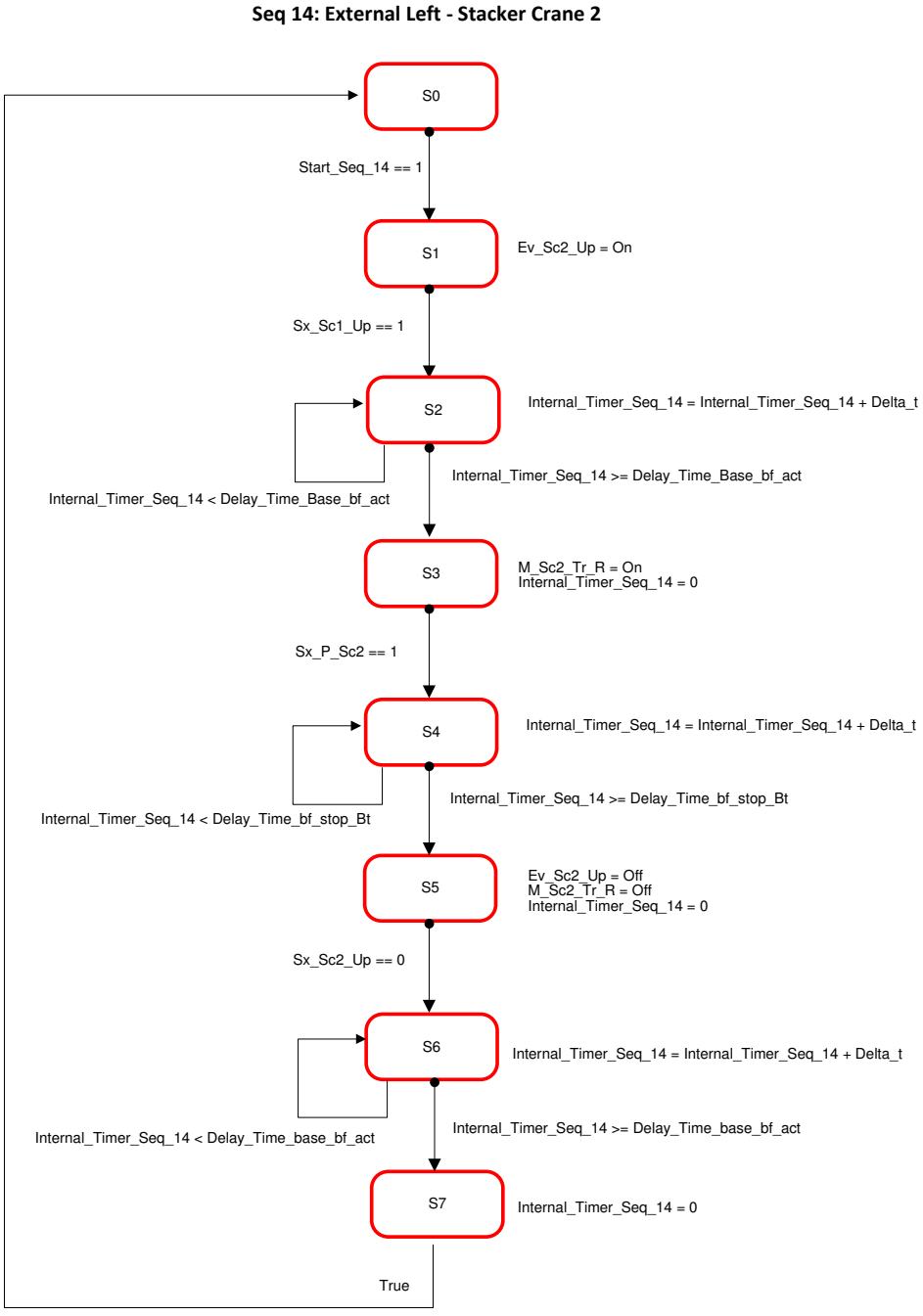


Figure B.44: LLCS sequence N. 14.

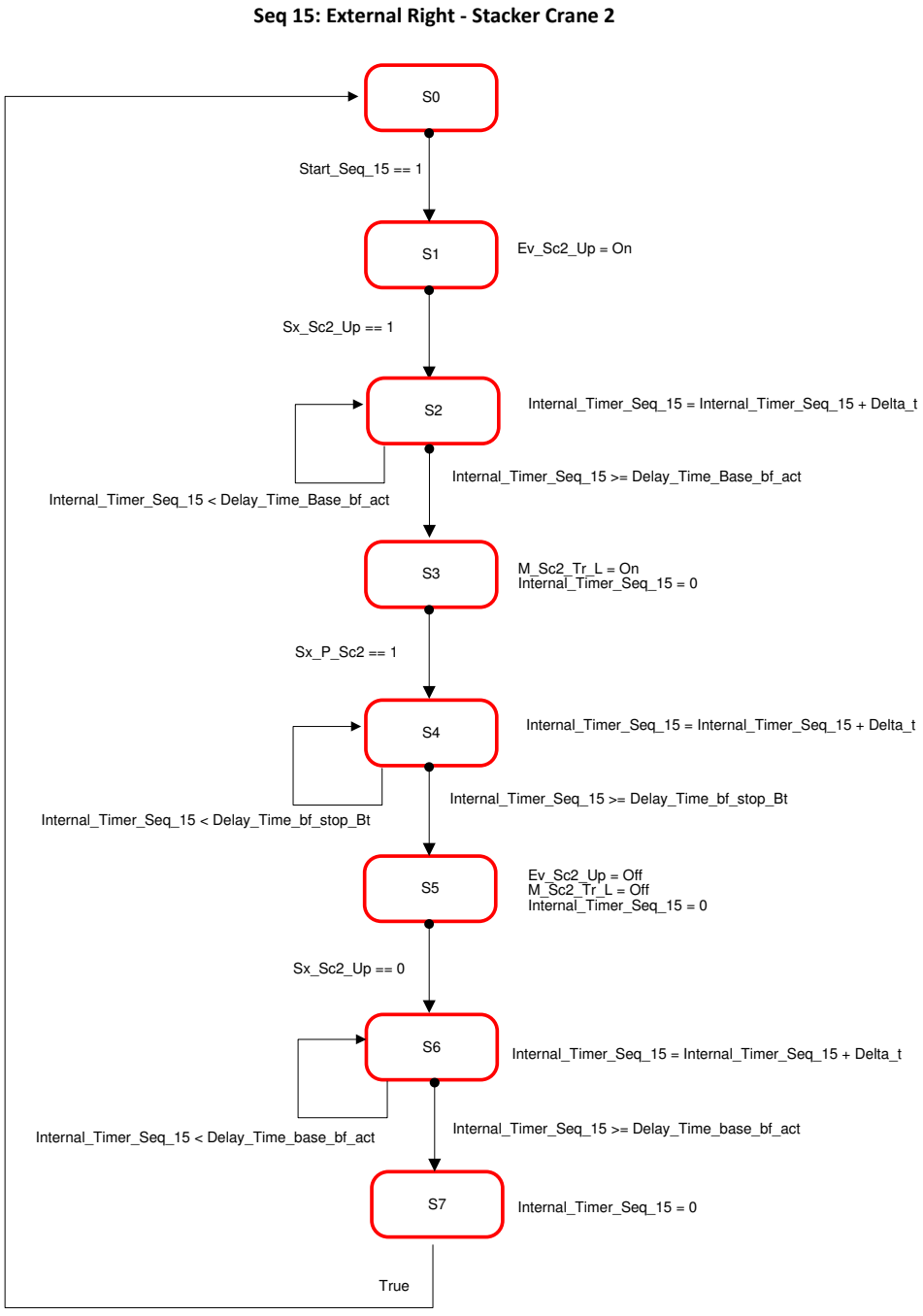


Figure B.45: LLCS sequence N. 15.

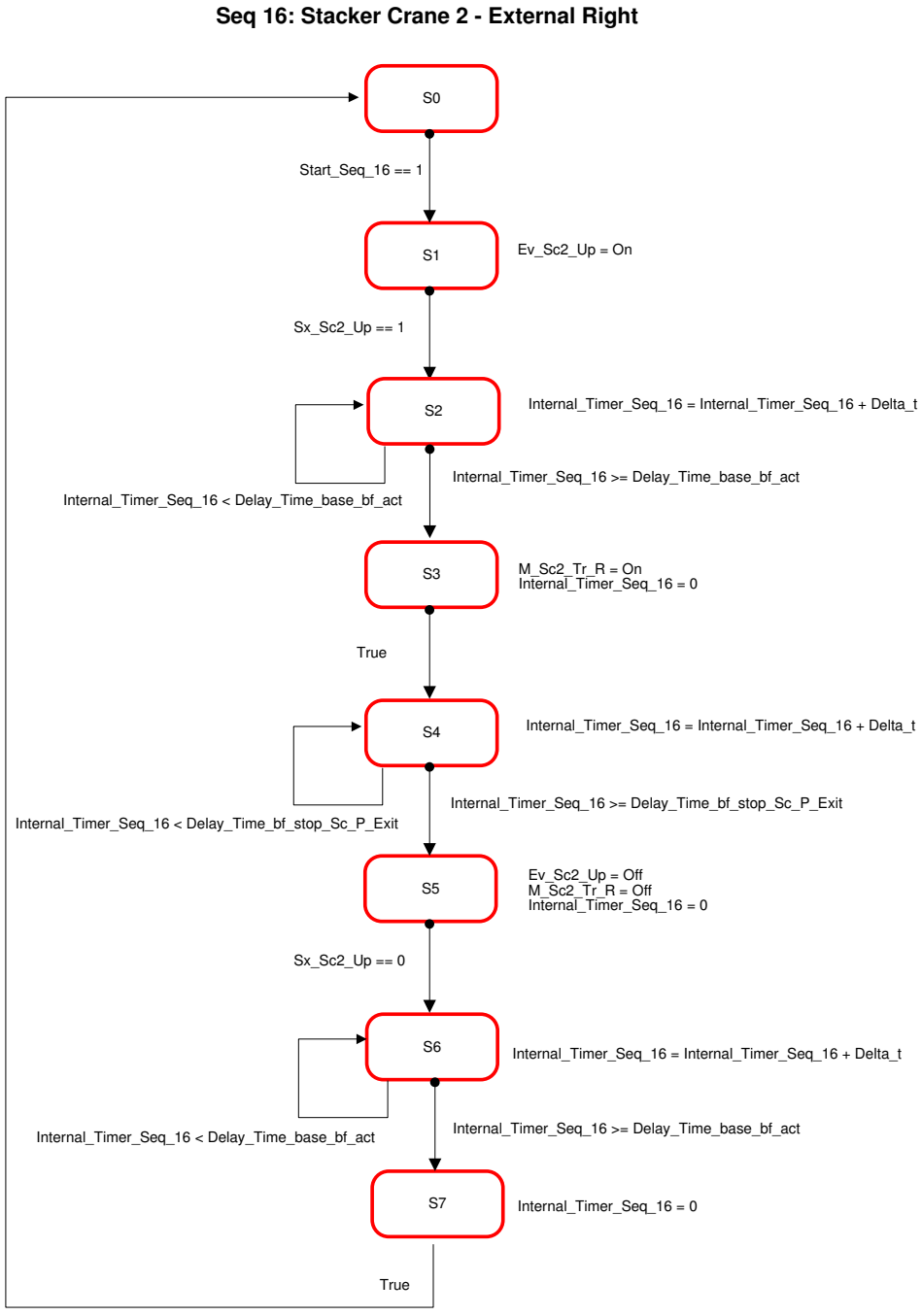


Figure B.46: LLCS sequence N. 16.

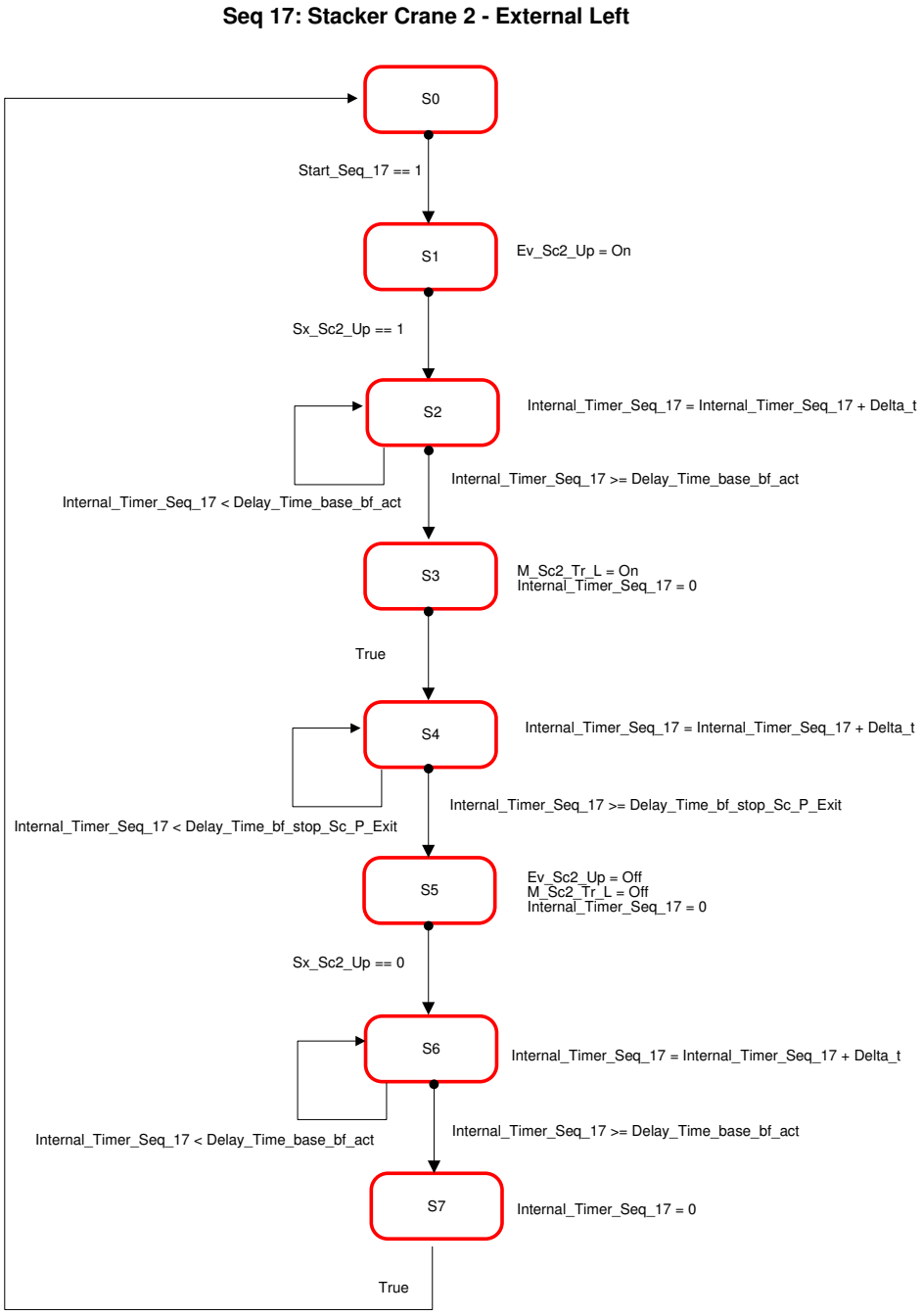


Figure B.47: LLCS sequence N. 17.

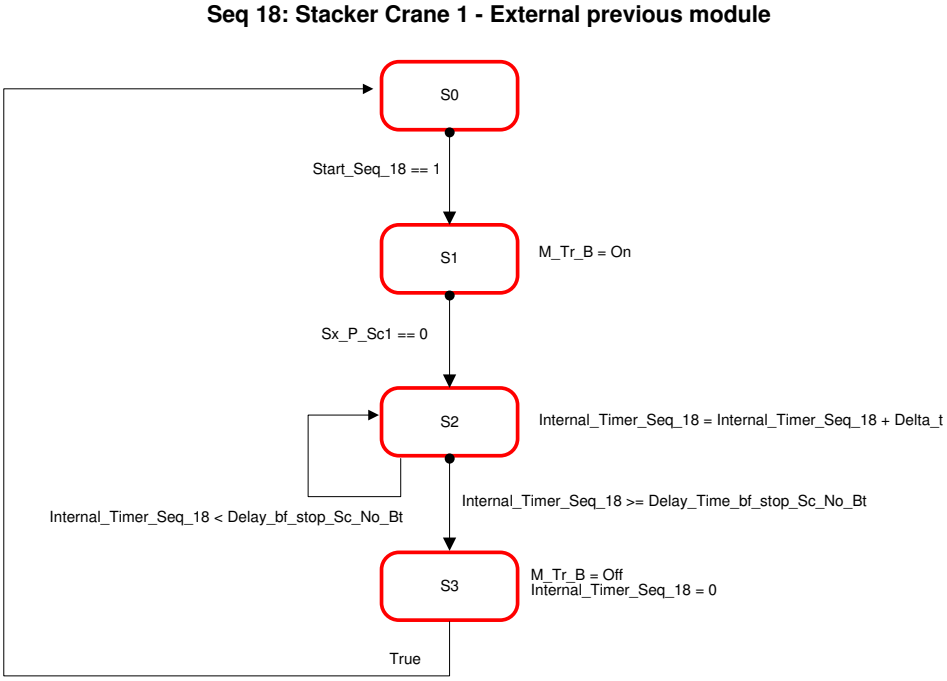


Figure B.48: LLCS sequence N. 18.

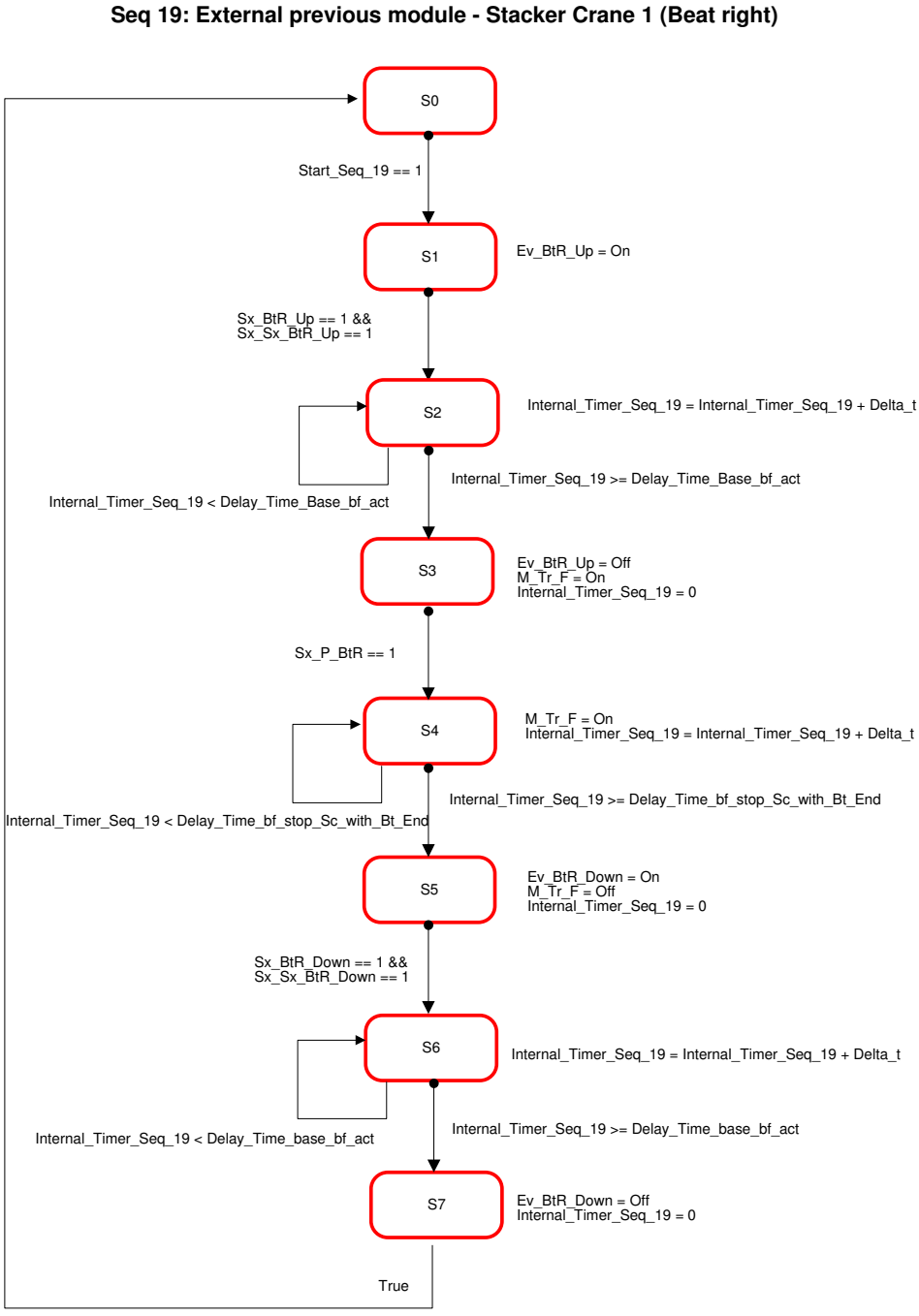


Figure B.49: LLCS sequence N. 19.

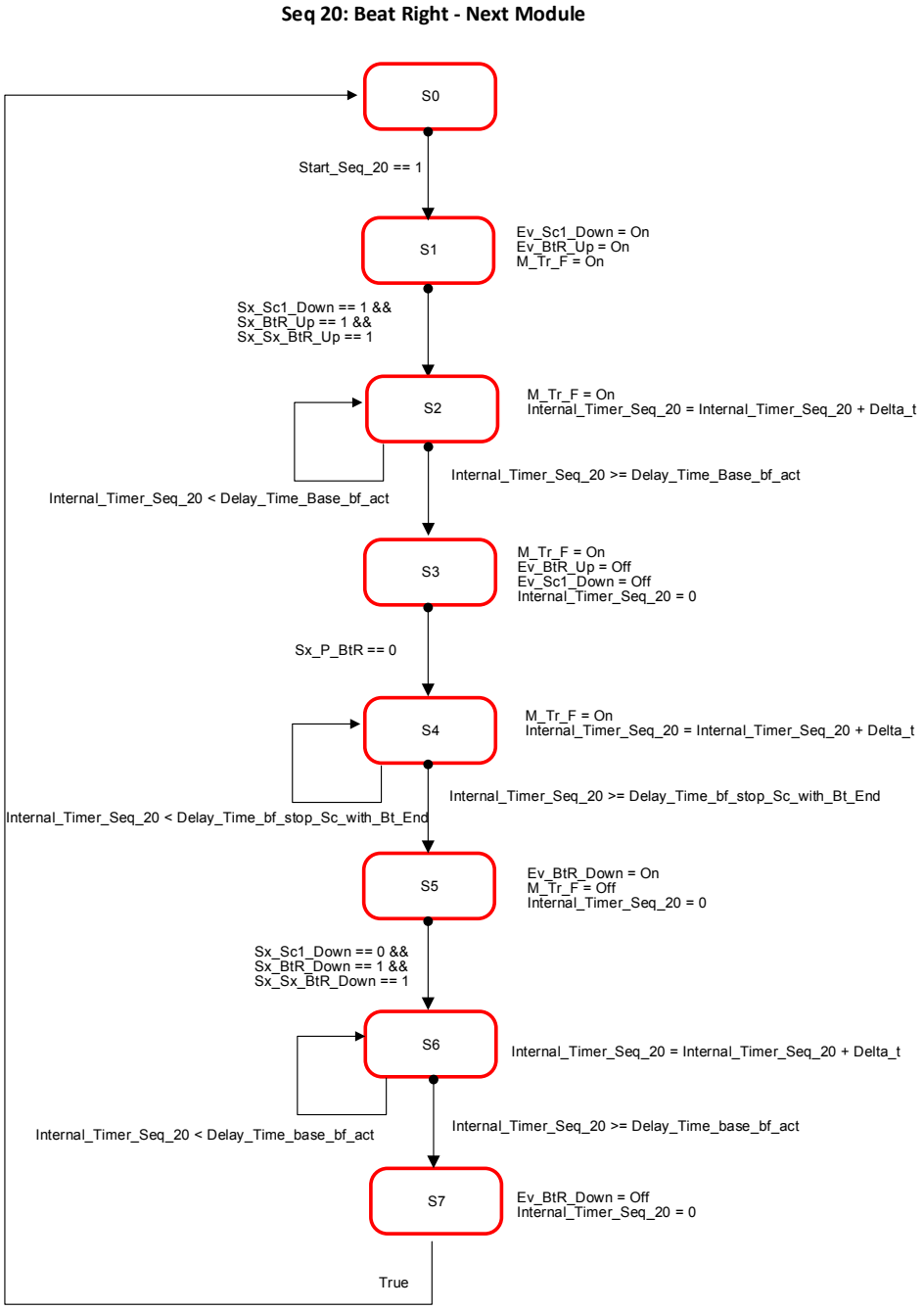


Figure B.50: LLCS sequence N. 20.

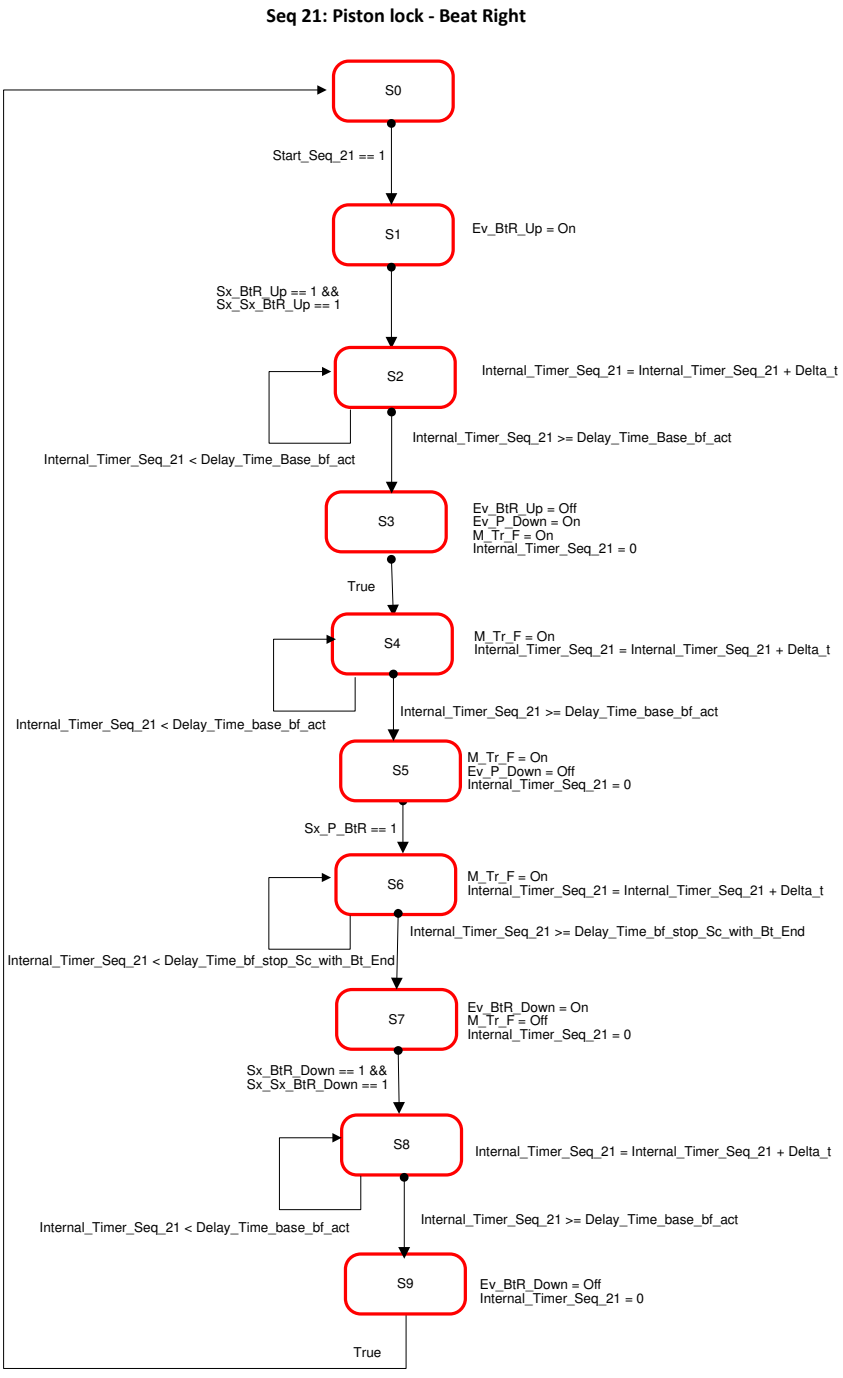


Figure B.51: LLCS sequence N. 21.

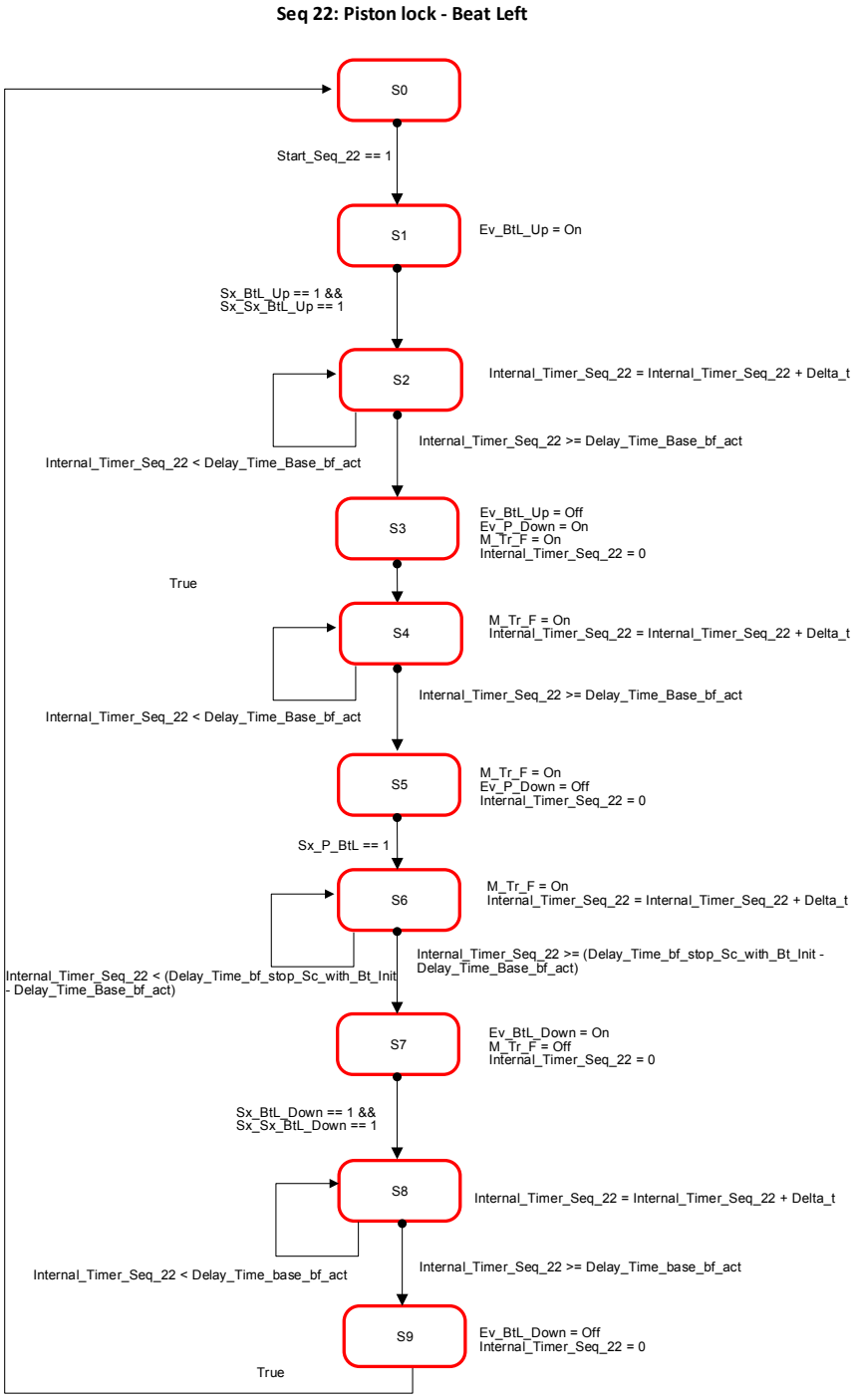


Figure B.52: LLCS sequence N. 22.

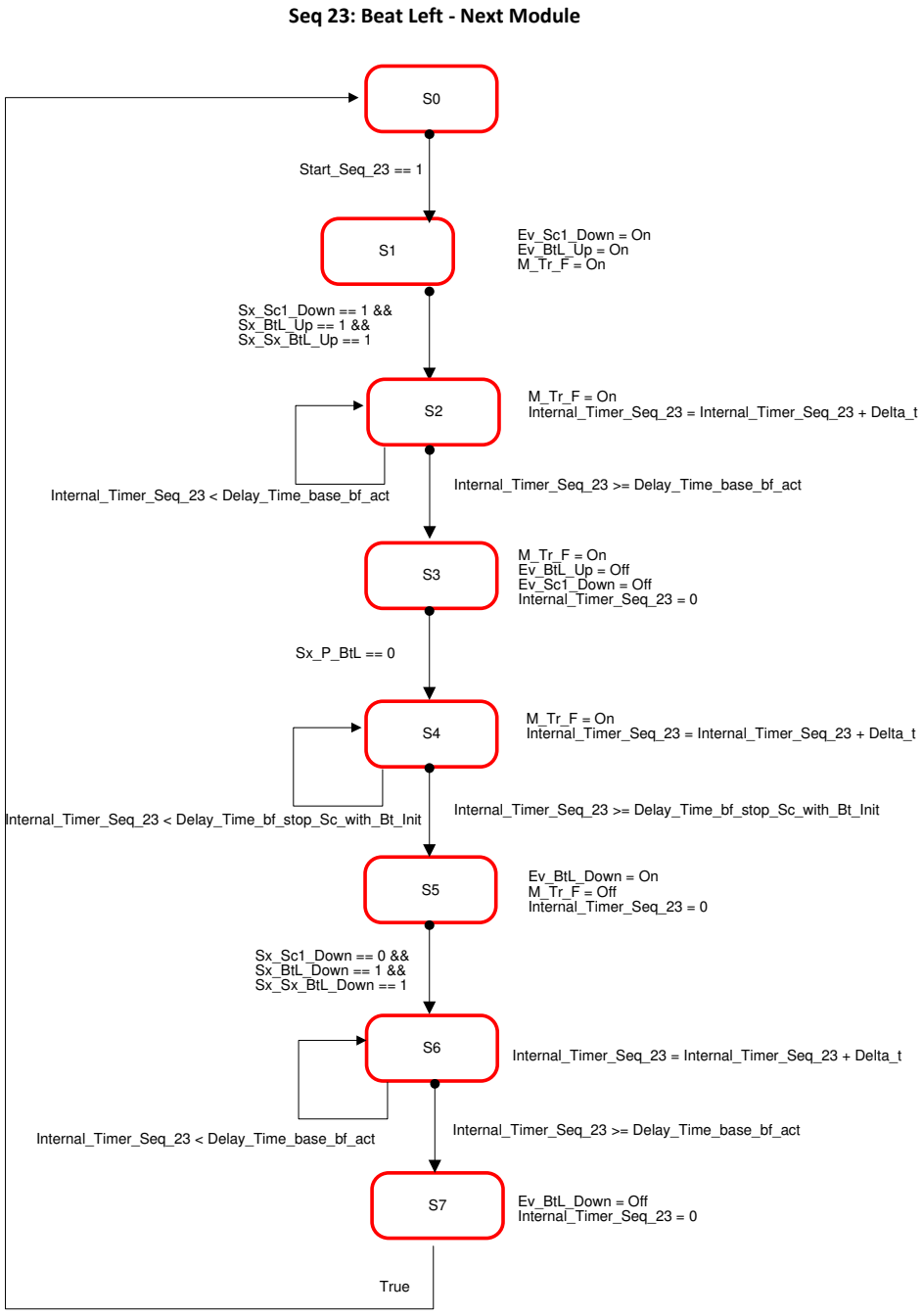


Figure B.53: LLCS sequence N. 23.

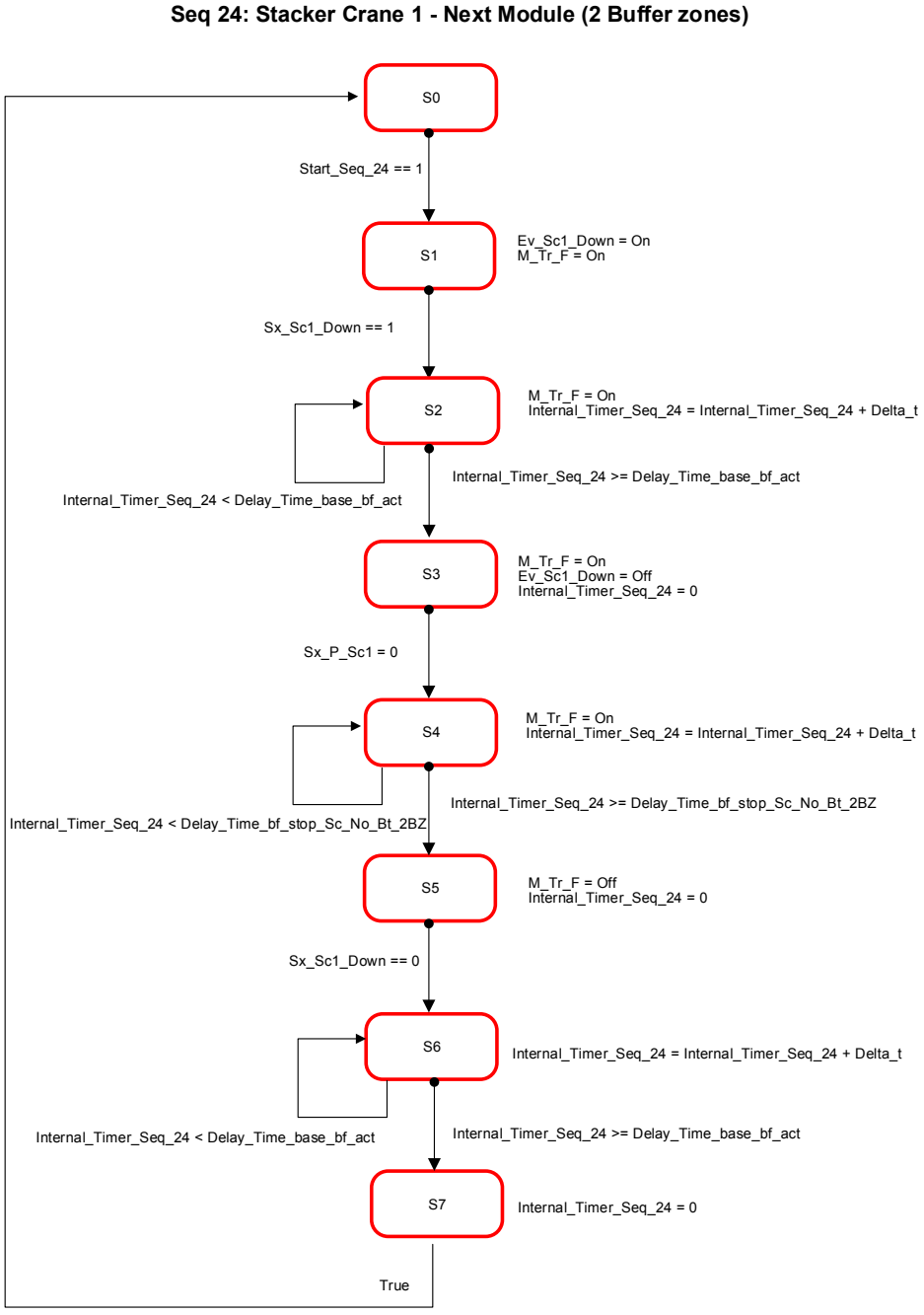


Figure B.54: LLCS sequence N. 24.

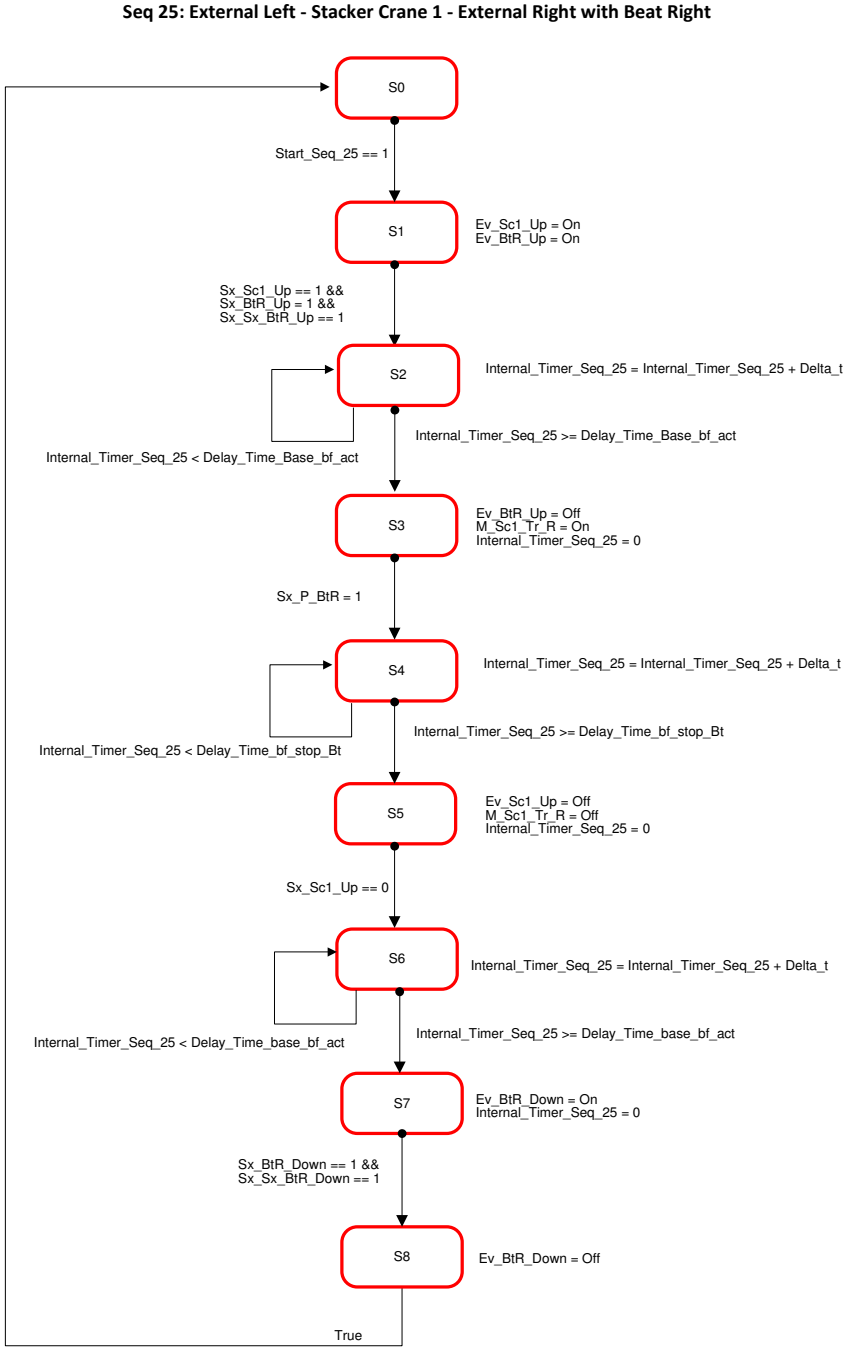


Figure B.55: LLCS sequence N. 25.

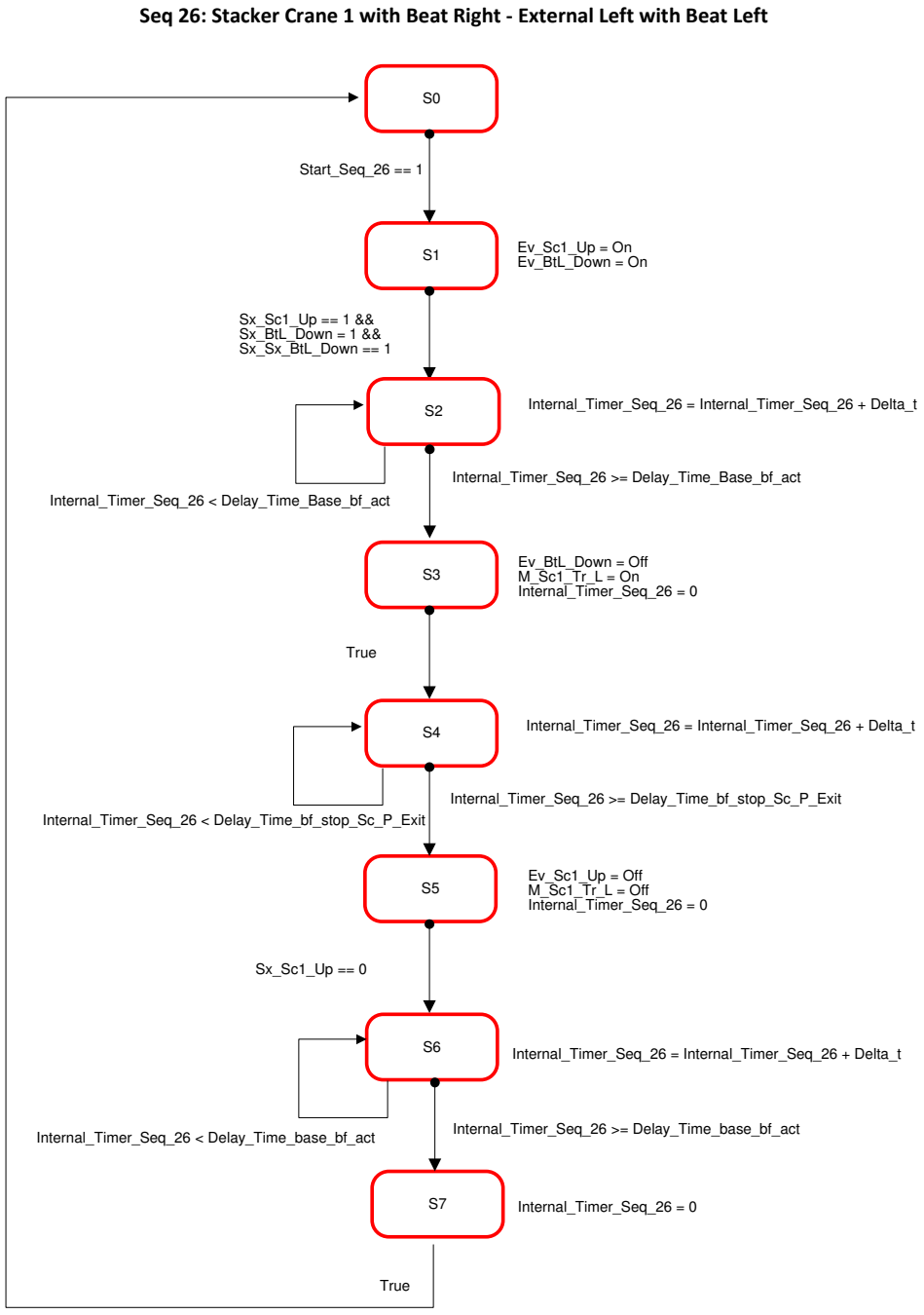


Figure B.56: LLCS sequence N. 26.

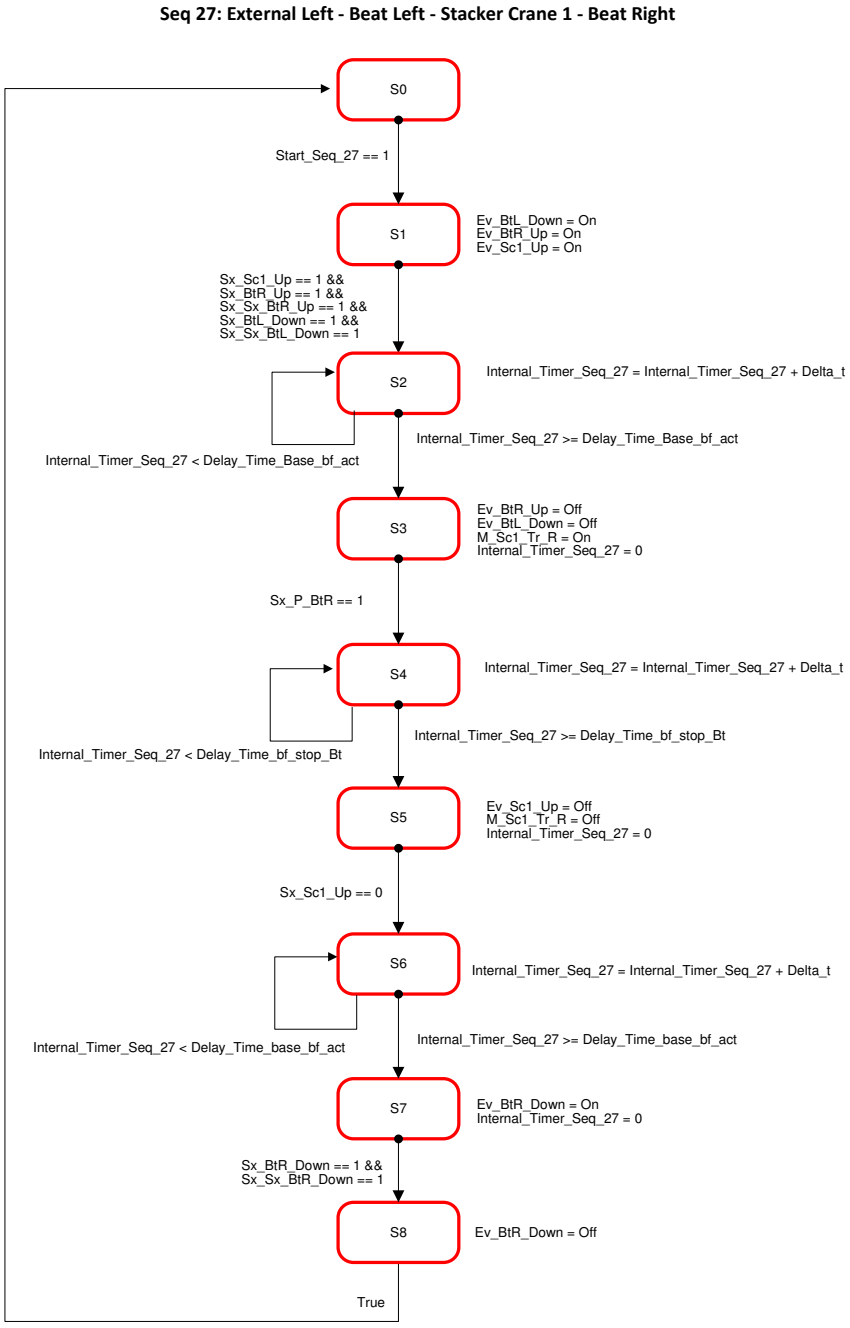


Figure B.57: LLCS sequence N. 27.

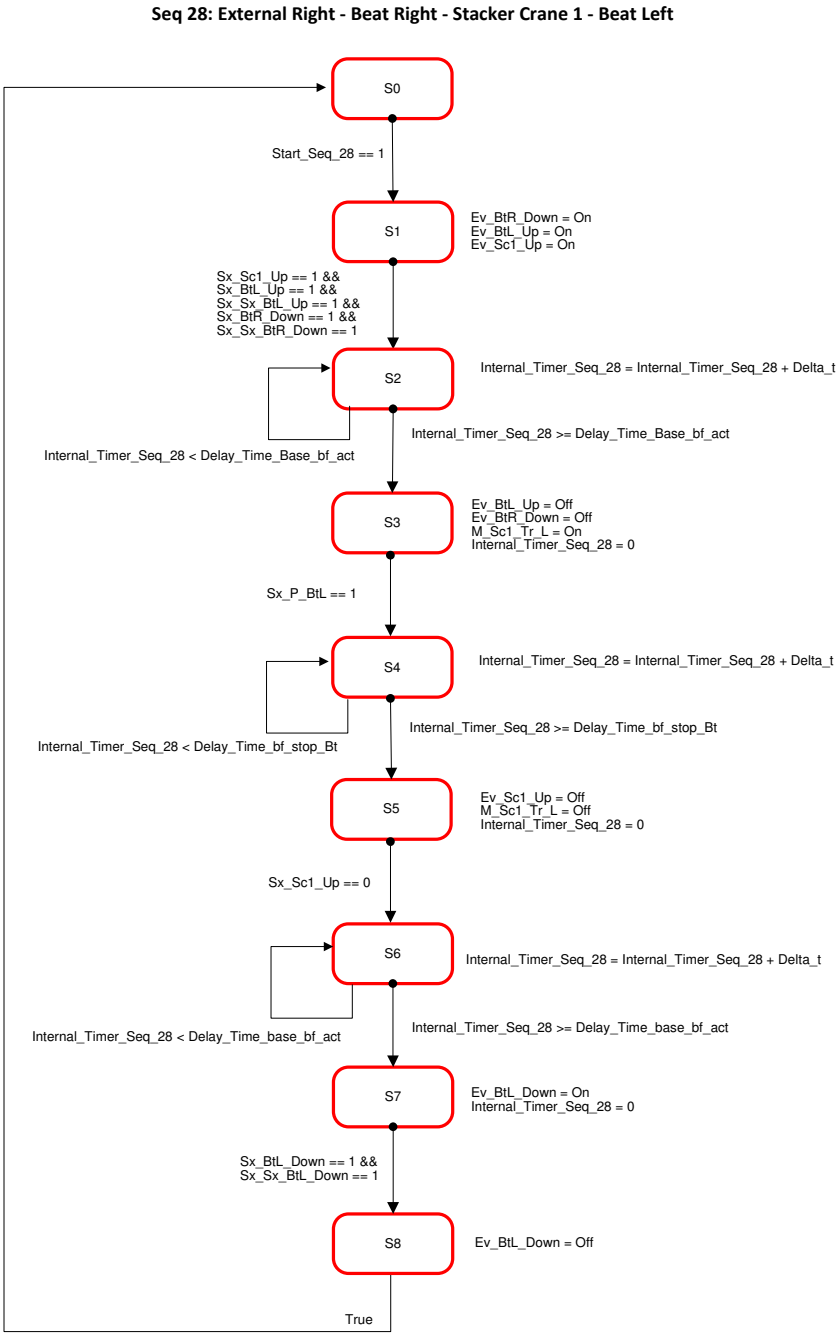


Figure B.58: LLCS sequence N. 28.

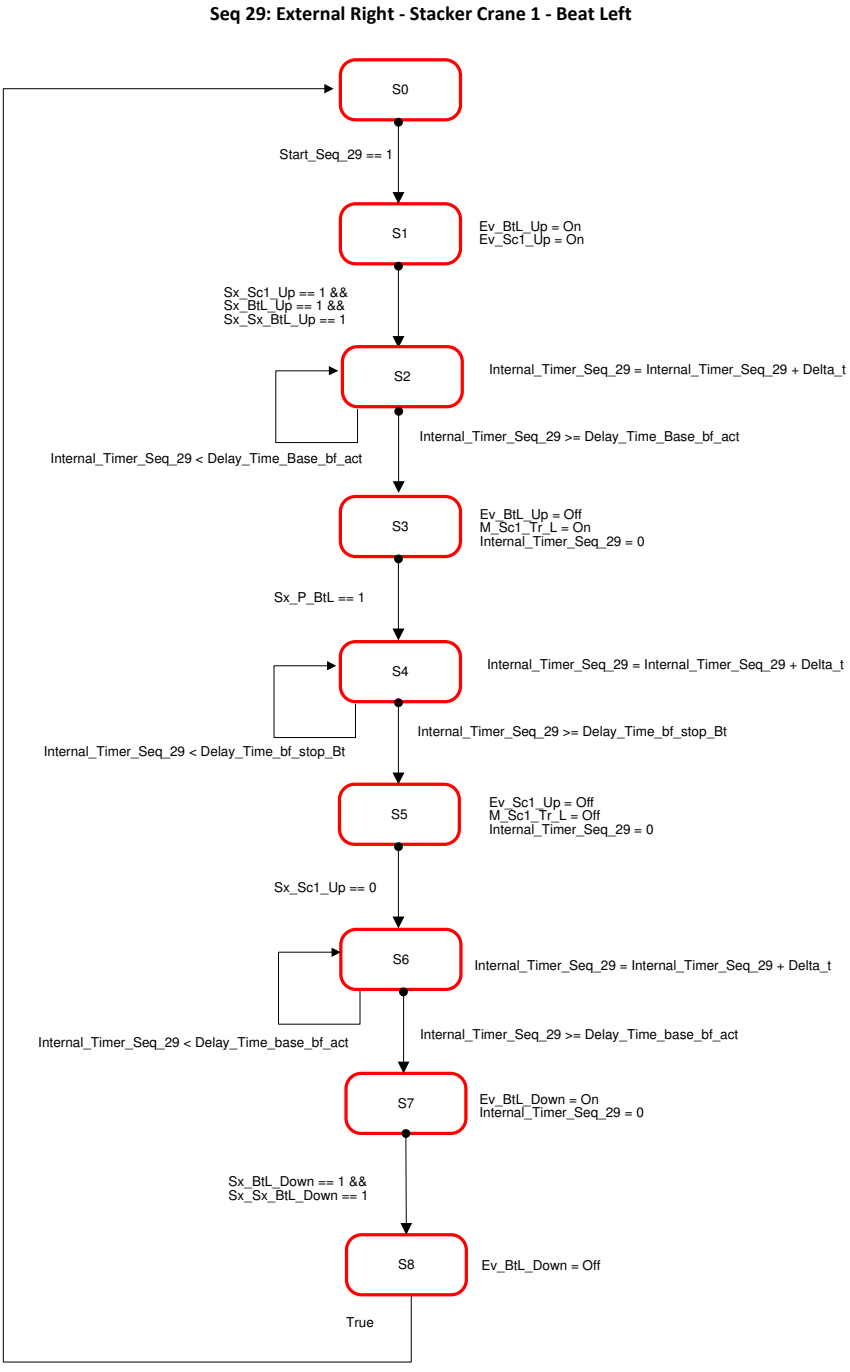


Figure B.59: LLCS sequence N. 29.

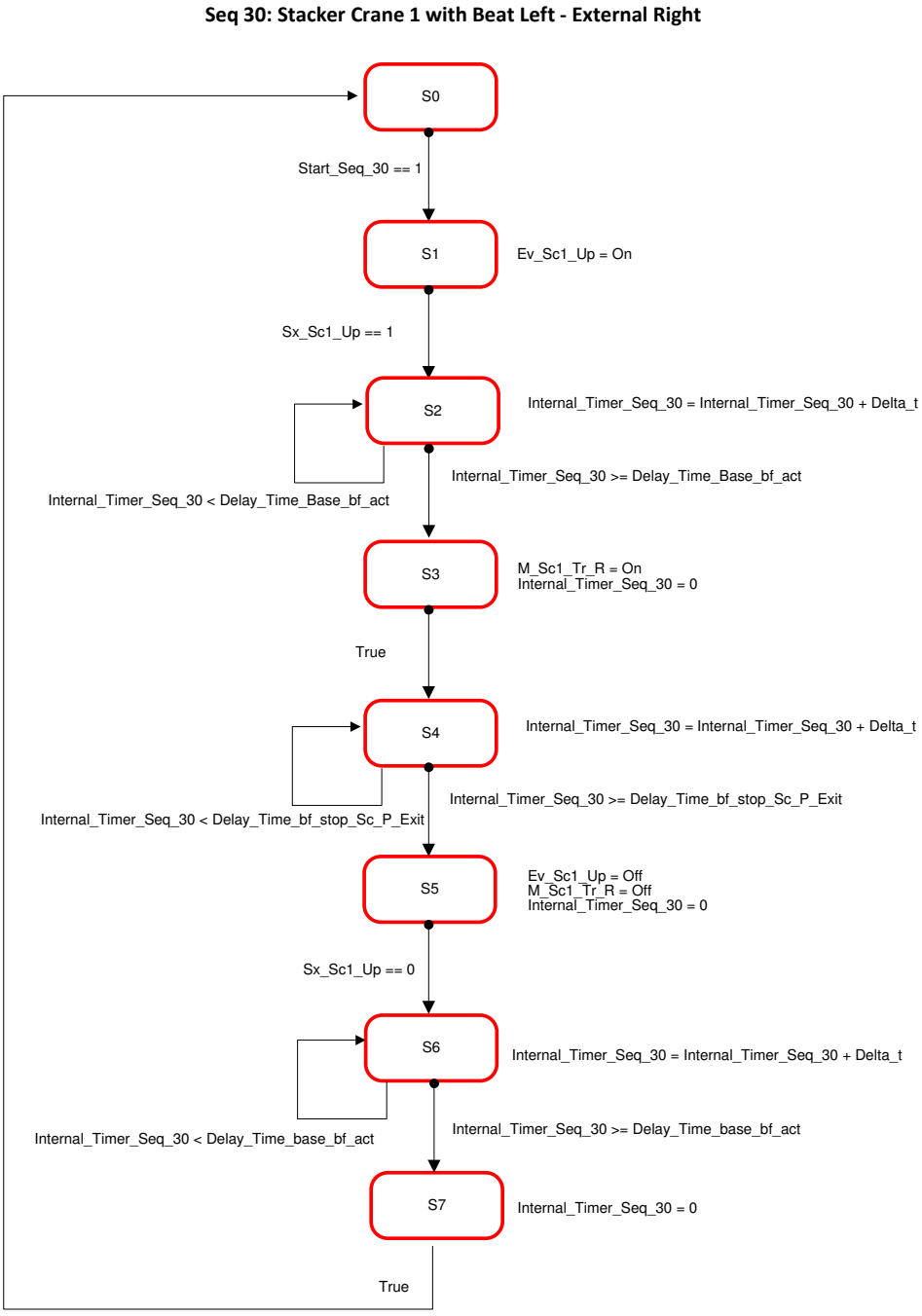


Figure B.60: LLCS sequence N. 30.

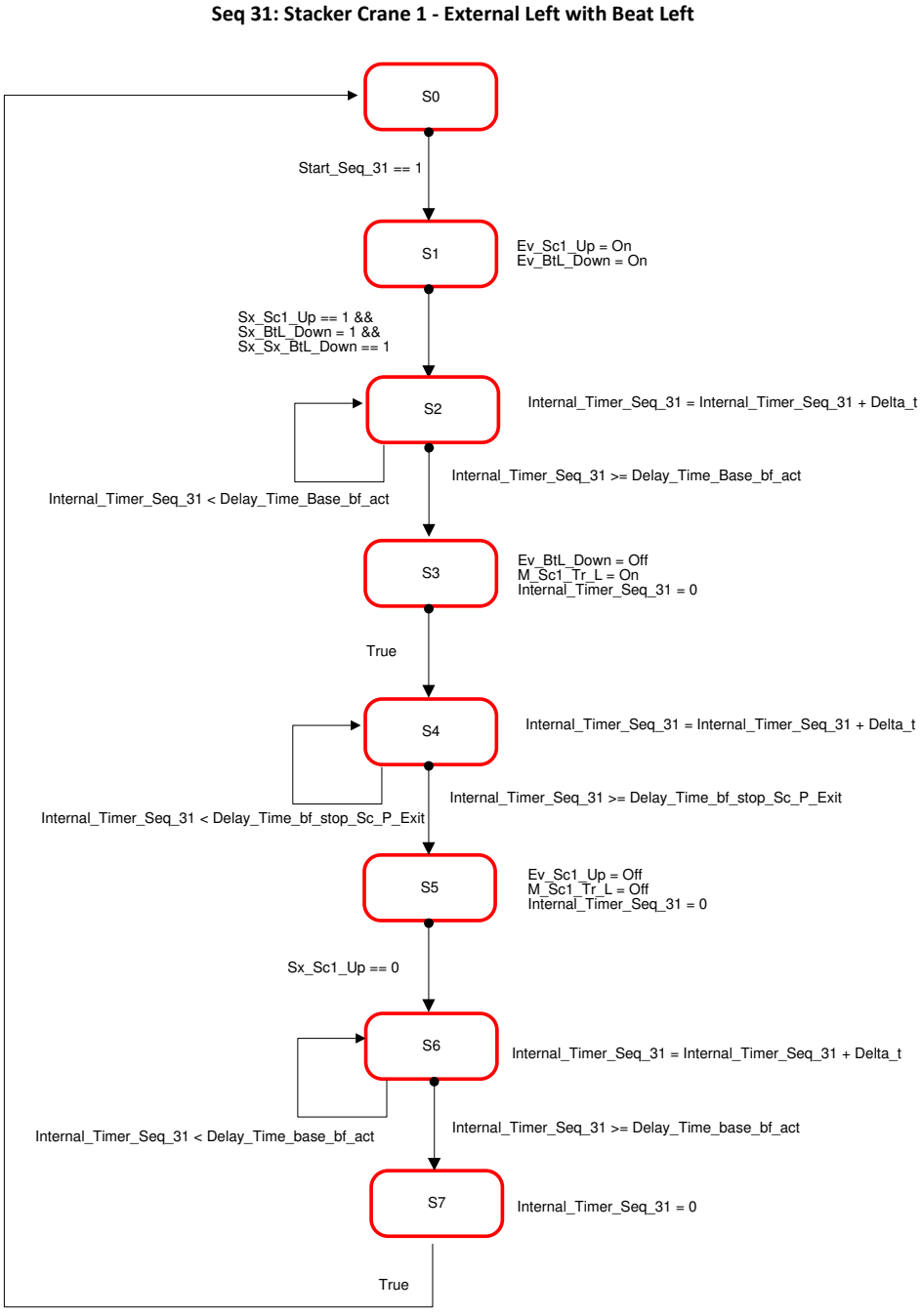


Figure B.61: LLCS sequence N. 31.

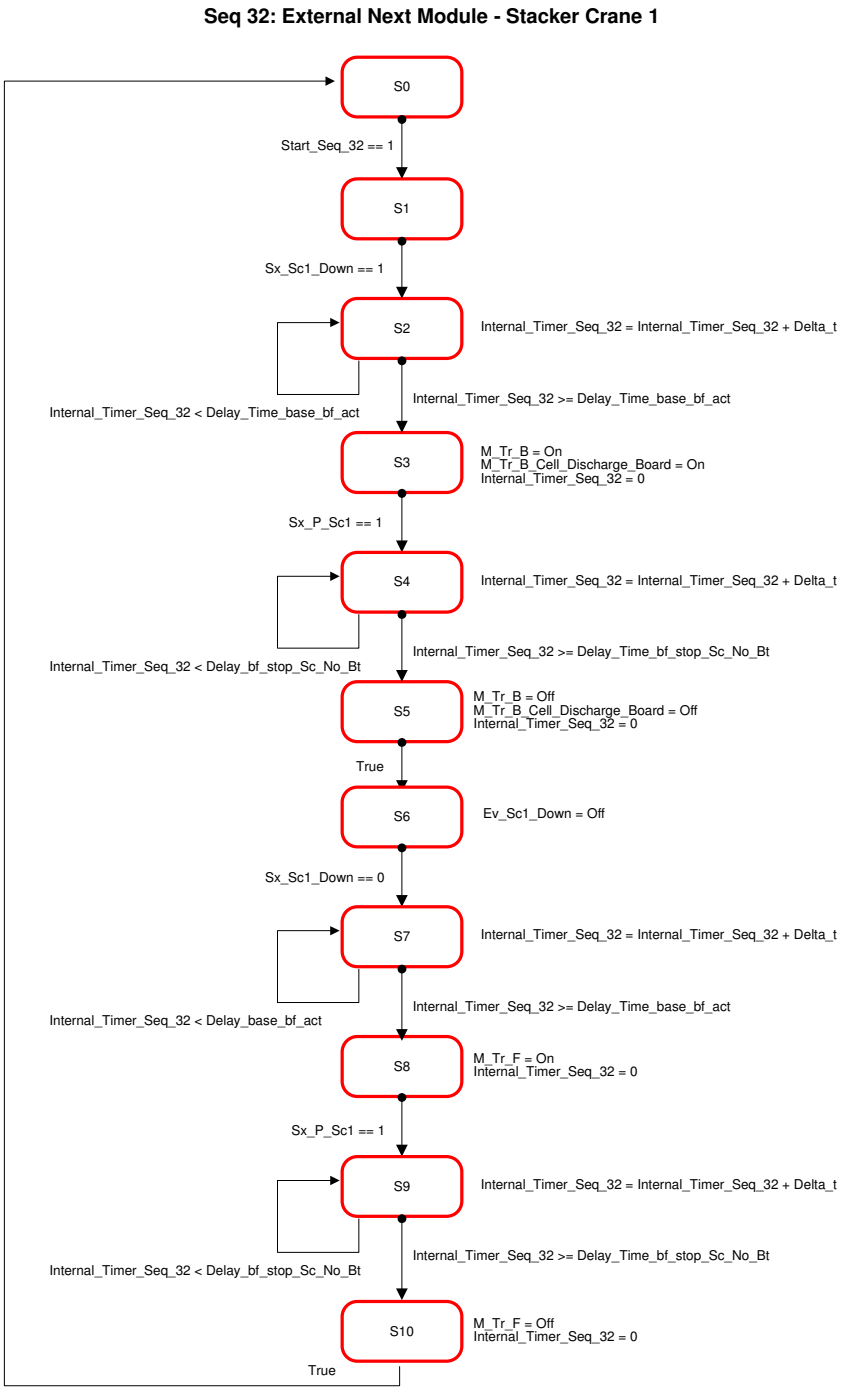


Figure B.62: LLCS sequence N. 32.

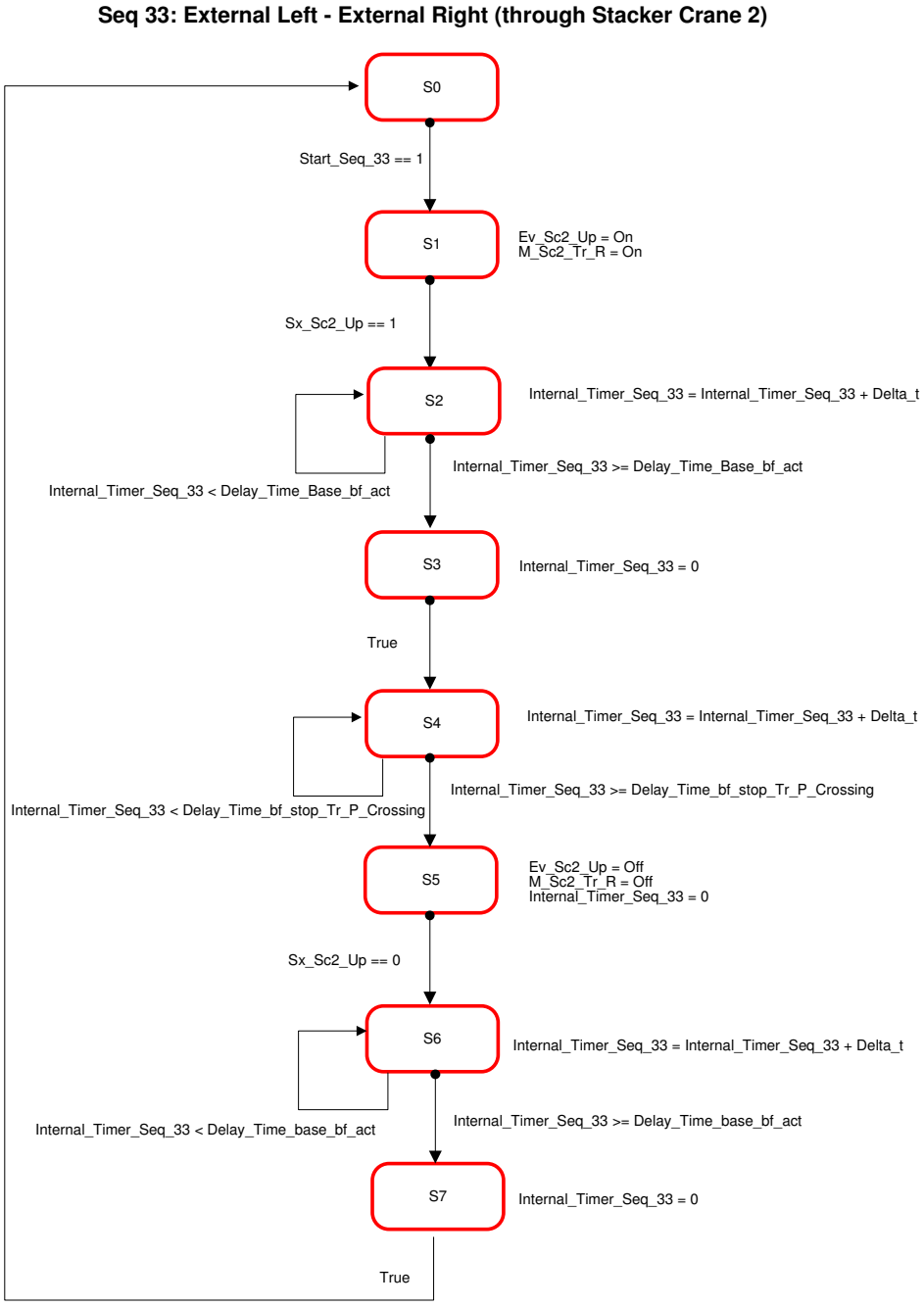


Figure B.63: LLCS sequence N. 33.

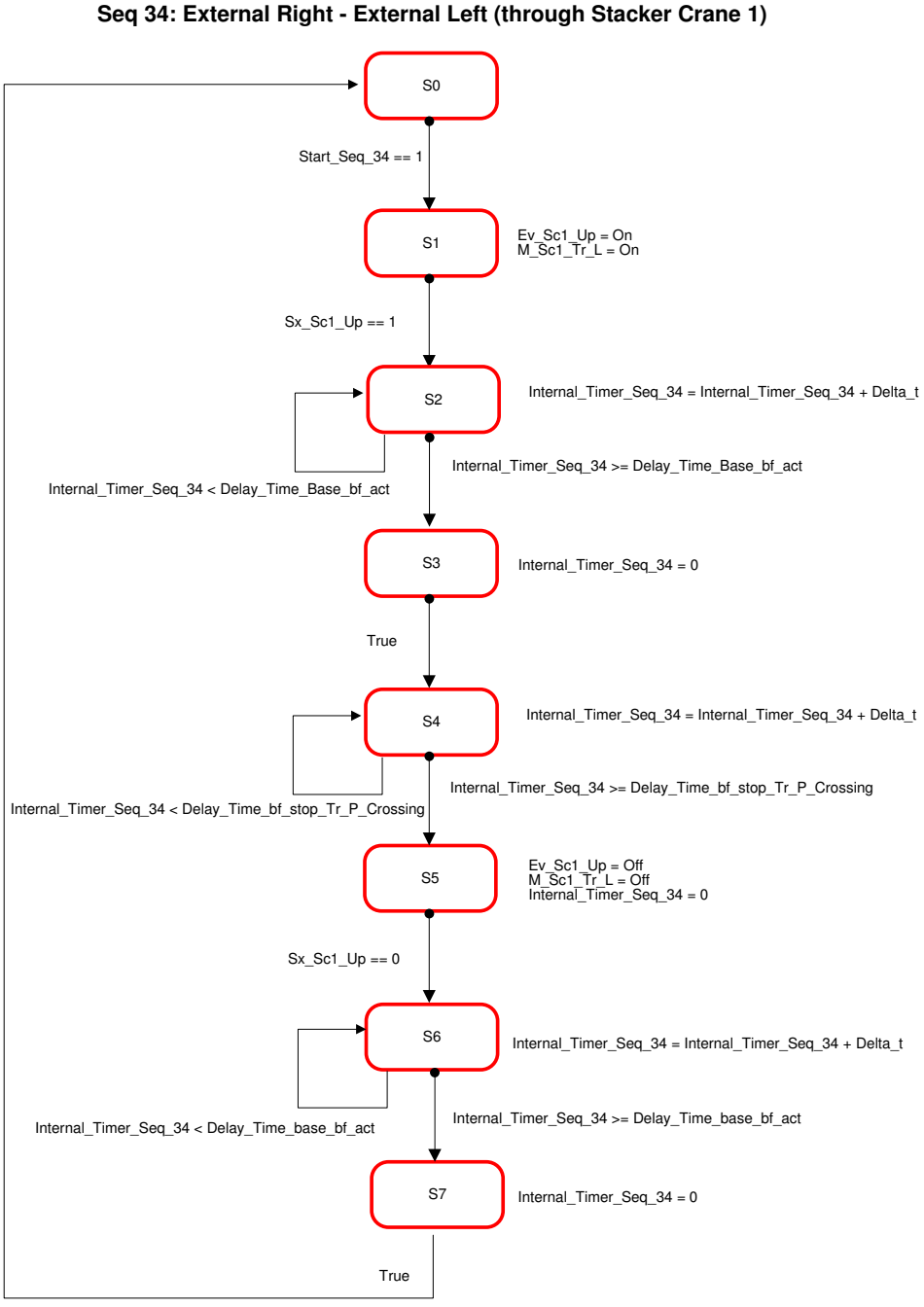


Figure B.64: LLCS sequence N. 34.

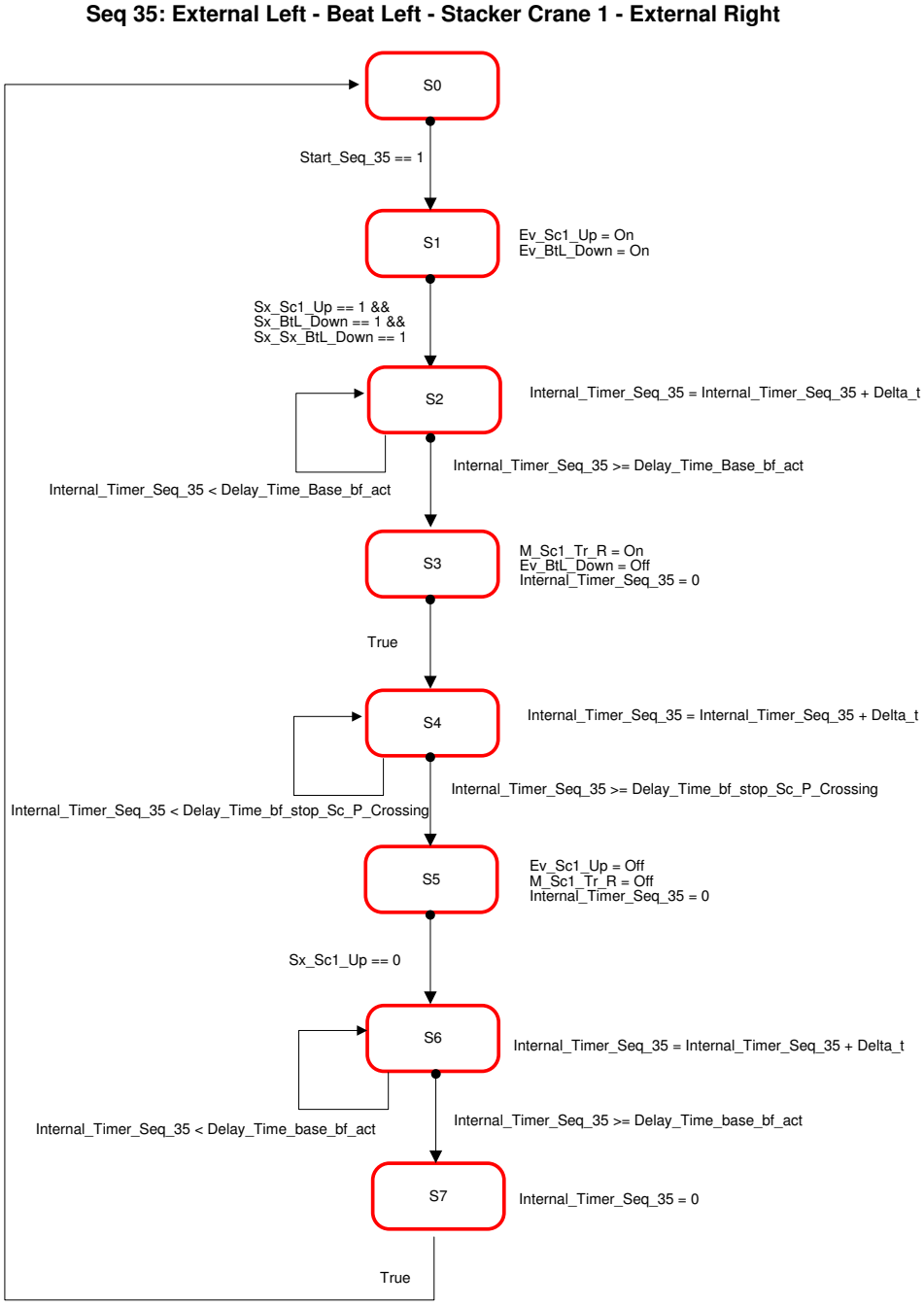


Figure B.65: LLCS sequence N. 35.

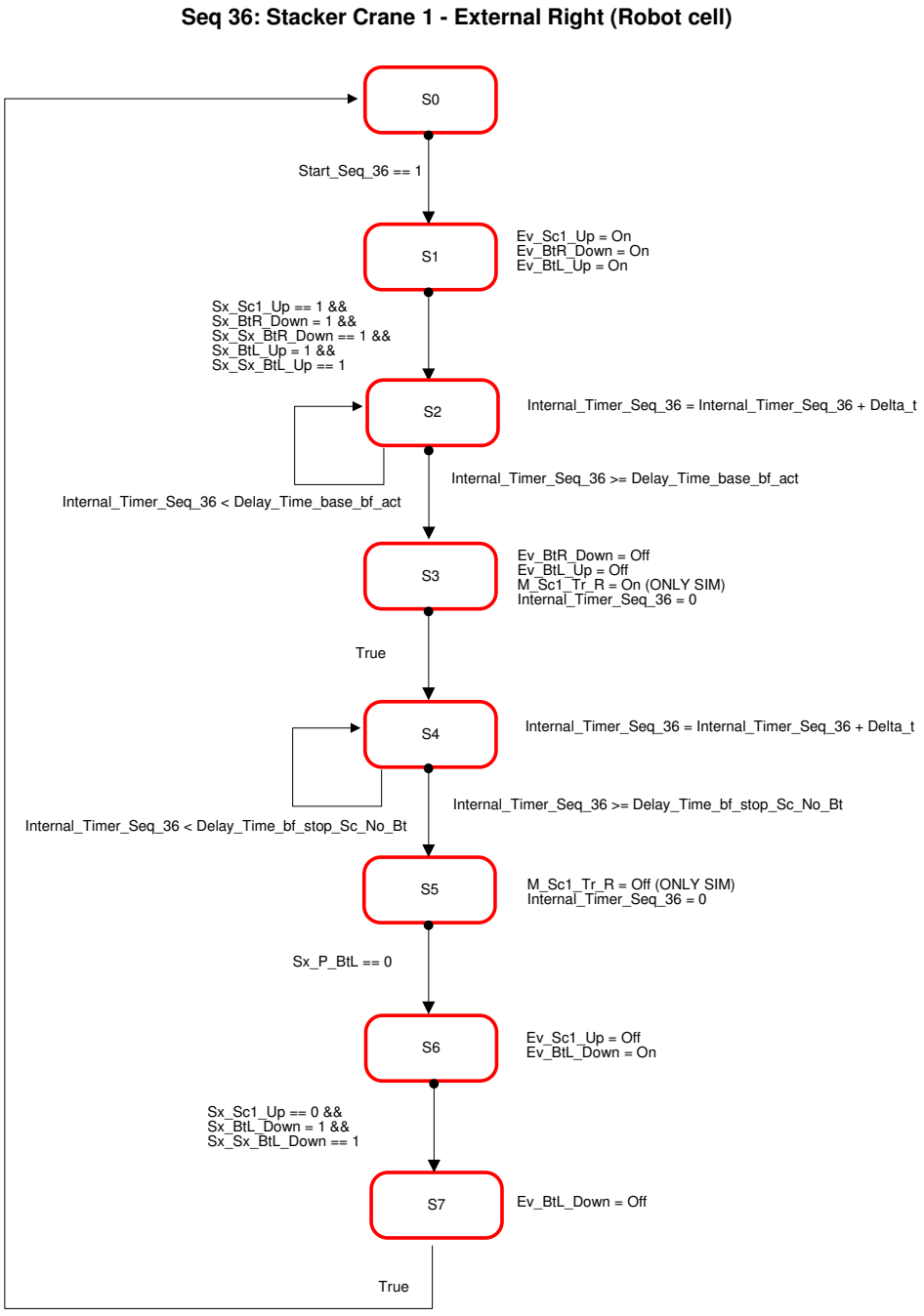


Figure B.66: LLCS sequence N. 36.

Appendix C

Low level control system sequence list

$\mathbf{S_n}$	**From**	**To**
S_1	Previous T_i	Stacker crane 1
S_2	Stacker crane 1	Block pallet piston
S_3	Block pallet piston	Next transport module
S_4	External right	Stacker crane 1
S_5	External left	Stacker crane 1
S_6	Stacker crane 1	External right
S_7	Stacker crane 1	External left
S_8	Stacker crane 1	Next transport module (M_4)
S_9	Previous transport module	Stacker crane 2
S_{10}	Stacker crane 2	Block pallet piston
S_{11}	Stacker crane 1	Next transport module
S_{12}	Previous transport module	Block pallet piston
S_{13}	Block pallet piston	Stacker crane 1
S_{14}	External left	Stacker crane 2
S_{15}	External right	Stacker crane 2
S_{16}	Stacker crane 2	External right
S_{17}	Stacker crane 2	External left
S_{18}	Stacker crane 1	Previous transport module
S_{19}	Previous transport module	Setback right
S_{20}	Setback right	Next transport module
S_{21}	Block pallet piston	Setback right
S_{22}	Block pallet piston	Setback left
S_{23}	Setback left	Next transport module
S_{24}	Stacker crane 1	Next transport module ($\Upsilon_{4,1}$)
S_{25}	External left	Stacker crane 1 (Setback right)
S_{26}	Stacker crane 1	(Setback right) External left (Setback left)
S_{27}	External left (Setback left)	Stacker crane 1 (Setback right)
S_{28}	External right (Setback right)	Stacker crane 1 (Setback left)
S_{29}	External right	Stacker crane 1 (Setback left)
S_{30}	Stacker crane 1	(Setback left) External right
S_{31}	Stacker crane 1	External left (Setback left)
S_{32}	Next transport module	Stacker crane 1
S_{33}	External left	External right (through Stacker crane 2)
S_{34}	External right	External left (through Stacker crane 1)
S_{35}	External left (Setback left)	External right (through Stacker crane 1)
S_{36}	Stacker crane 1	External right (M_1)

Table C.1: The control sequences (S_n) definition.

$\mathbf{T_n}$	$\mathbf{S_n}$
T_1	$S_2, S_3, S_{36}, S_{19}, S_{26}, S_{27}, S_{28}$
T_2	S_1, S_2, S_3, S_5, S_7
T_3	S_2, S_3, S_4, S_{33}
T_4	S_1, S_5, S_7, S_{24}
T_5	S_7, S_{12}, S_{13}
T_6	S_2, S_3, S_6, S_{25}
T_7	$S_1, S_2, S_3, S_5, S_{18}$
T_8	$S_6, S_9, S_{10}, S_{15}, S_{16}, S_{20}, S_{21}, S_{26}, S_{27}, S_{28}$
T_9	$S_9, S_{10}, S_{14}, S_{17}, S_{22}, S_{23}, S_{29}, S_{30}, S_{31}, S_{35}$
T_{10}	$S_7, S_{12}, S_{13}, S_{34}$
T_{11}	$S_7, S_8, S_{12}, S_{13}, S_{32}$
T_{12}	$S_5, S_7, S_{11}, S_{12}, S_{13}$
T_{13}	$S_5, S_7, S_{11}, S_{12}, S_{13}$
T_{14}	S_7, S_{12}, S_{13}
T_{15}	S_2, S_3, S_5

Table C.2: The control sequences (S_n) associated to the related transport modules (T_i).

$\mathbf{S_n}$	$\mathbf{T_n}$
S_1	T_2, T_4, T_7
S_2	$T_1, T_2, T_3, T_6, T_7, T_{15}$
S_3	$T_1, T_2, T_3, T_6, T_7, T_{15}$
S_4	T_3
S_5	$T_2, T_4, T_7, T_{12}, T_{13}, T_{15}$
S_6	T_6, T_8
S_7	$T_2, T_4, T_5, T_{10}, T_{11}, T_{12}, T_{13}, T_{14}$
S_8	T_{11}
S_9	T_8, T_9
S_{10}	T_8, T_9
S_{11}	T_{12}, T_{13}
S_{12}	$T_5, T_{10}, T_{11}, T_{12}, T_{13}, T_{14}$
S_{13}	$T_5, T_{10}, T_{11}, T_{12}, T_{13}, T_{14}$
S_{14}	T_9
S_{15}	T_8
S_{16}	T_8
S_{17}	T_9
S_{18}	T_7
S_{19}	T_1
S_{20}	T_8
S_{21}	T_8
S_{22}	T_9
S_{23}	T_9
S_{24}	T_4
S_{25}	T_6
S_{26}	T_1, T_8
S_{27}	T_1, T_8
S_{28}	T_1, T_8
S_{29}	T_9
S_{30}	T_9
S_{31}	T_9
S_{32}	T_{11}
S_{33}	T_3
S_{34}	T_{10}
S_{35}	T_9
S_{36}	T_1

Table C.3: The transport module (T_i) associated to the related control sequences (S_n) .

Acronyms

AS/RS	Automated Storage / Retrieval System
BZ	Buffer Zone
CSM	aCtuatorS Methodology
DCPIP	Dynamic Control Platform for Industrial Plants
DCV	Destination Coded Vehicle
DES	Discrete Event System
DESS	Discrete Event System Simulator
EFA	Extended Finite Automata
ELC	Extended Linear Complementary
EMM	Enhanced Machine Model
FSM	Finite State Machine
HLCS	High Level Control System
ID	IDentification
ITIA−CNR	Institute of Industrial Technology and Automation - National Rresearch Council
LC	Linear Complementary
LLCS	Low Level Control System
MILP	Mixed Integer Linear Programming
MLD	Mixed Logical Dynamical
MMPS	Max Min Plus Scaling
MPC	Model predictive Control
PID	Proportional Integral Derivative
PLC	Programmable Logic Controller
PWA	PieceWise Affine
RH	Receding Horizon
SCM	Supply Chain Management
SFC	Sequential Functional Chart
SMM	Simplified Machine Model

List of Figures

1.1 MPC strategy. 3
1.2 MPC schema. 4
1.3 The de-manufacturing plant. 9

2.1 The de-manufacturing plant. 18
2.2 The de-manufacturing plant structure. 19
2.3 The transport module T_8 configuration. 20
2.4 Schematic representation of the transport module actuators. 21
2.5 Schematic representation of the transport module. 22
2.6 Buffer Zone and Control Sequence transport line representation. 24
2.7 Direct graph representation of the plant. 25
2.8 EFA model of the machines. 28
2.9 Distance of a pallet placed in the node N_1 from the different target machines. 32
2.10 Simulation pallet paths. 35
2.11 Machined pallets by M_1 and M_3. 36
2.12 Machined pallets by M_2 and M_4. 36
2.13 Real plant pallet paths. 40

3.1 Production system. 46
3.2 Node model $N_{i,j}$. 47
3.3 Machine block. 49
3.4 FSM of a machine. 49
3.5 FSM of a enhanced machine. 52
3.6 Schematic block of the plant. 57
3.7 Example of a simple configuration. 59
3.8 Two feasible solutions for the same problem. 59
3.9 Experimental plant. 61
3.10 Power Minimization using the simple machine model. . . . 69

3.11 Minimum production constrained problem with Due date constraint. 70
3.12 Minimum production constrained problem with storage costs. 71
3.13 Power Minimization using the enhanced machine model. . 72
3.14 Production maximization using the Simple Machine Model. 73
3.15 Production maximization using the Enhanced Machine Model. 74
3.16 Produced parts. 75
3.17 Production maximization using MILP optimization. 76
3.18 Production maximization using RH optimization. 77

4.1 Stacker crane plant. 83
4.2 Schematic representation of the storage facility. 85
4.3 Example of state transitions. 95
4.4 Simulation results: movement of the pallets. 96

5.1 De-manufacturing pilot plant control architecture. 107
5.2 *DCPIP* class diagram. 108
5.3 Software architecture for *DCPIP* implementation. 110
5.4 SIMIO customized control step. 112
5.5 Control variables plot. 113
5.6 Power and energy plot. 114
5.7 Pallet transport line power simulation and measurement results. 117
5.8 Pallet transport line energy consumption simulation and measurement results. 117
5.9 Transport module T_1 power simulation results. 118
5.10 Transport module T_1 energy consumption simulation results. 118
5.11 Transport module T_2 power simulation results. 119
5.12 Transport module T_2 energy consumption simulation results. 119
5.13 ISaGRAF plant low level control system implementation. . 120
5.14 Measurement system architecture. 120

B.1 Transport module N. 1. 140
B.2 Transport module N. 2. 142
B.3 Transport module N. 3. 143
B.4 Transport module N. 4. 144
B.5 Transport module N. 5. 145
B.6 Transport module N. 6. 146
B.7 Transport module N. 7. 147
B.8 Transport module N. 8. 148
B.9 Transport module N. 9. 150

B.10 Transport module N. 10. 152
B.11 Transport module N. 11. 153
B.12 Transport module N. 12. 154
B.13 Transport module N. 13. 155
B.14 Transport module N. 14. 156
B.15 Transport module N. 15. 157
B.16 Transport module N. 1 with sequences. 158
B.17 Transport module N. 2 with sequences. 159
B.18 Transport module N. 3 with sequences. 160
B.19 Transport module N. 4 with sequences. 161
B.20 Transport module N. 5 with sequences. 162
B.21 Transport module N. 6 with sequences. 163
B.22 Transport module N. 7 with sequences. 164
B.23 Transport module N. 8 with sequences. 165
B.24 Transport module N. 9 with sequences. 166
B.25 Transport module N. 10 with sequences. 167
B.26 Transport module N. 11 with sequences. 168
B.27 Transport module N. 12 with sequences. 169
B.28 Transport module N. 13 with sequences. 170
B.29 Transport module N. 14 with sequences. 171
B.30 Transport module N. 15 with sequences. 172
B.31 LLCS sequence N. 1. 173
B.32 LLCS sequence N. 2. 174
B.33 LLCS sequence N. 3. 175
B.34 LLCS sequence N. 4. 176
B.35 LLCS sequence N. 5. 177
B.36 LLCS sequence N. 6. 178
B.37 LLCS sequence N. 7. 179
B.38 LLCS sequence N. 8. 180
B.39 LLCS sequence N. 9. 181
B.40 LLCS sequence N. 10. 182
B.41 LLCS sequence N. 11. 183
B.42 LLCS sequence N. 12. 184
B.43 LLCS sequence N. 13. 185
B.44 LLCS sequence N. 14. 186
B.45 LLCS sequence N. 15. 187
B.46 LLCS sequence N. 16. 188
B.47 LLCS sequence N. 17. 189
B.48 LLCS sequence N. 18. 190
B.49 LLCS sequence N. 19. 191

B.50 LLCS sequence N. 20. 192
B.51 LLCS sequence N. 21. 193
B.52 LLCS sequence N. 22. 194
B.53 LLCS sequence N. 23. 195
B.54 LLCS sequence N. 24. 196
B.55 LLCS sequence N. 25. 197
B.56 LLCS sequence N. 26. 198
B.57 LLCS sequence N. 27. 199
B.58 LLCS sequence N. 28. 200
B.59 LLCS sequence N. 29. 201
B.60 LLCS sequence N. 30. 202
B.61 LLCS sequence N. 31. 203
B.62 LLCS sequence N. 32. 204
B.63 LLCS sequence N. 33. 205
B.64 LLCS sequence N. 34. 206
B.65 LLCS sequence N. 35. 207
B.66 LLCS sequence N. 36. 208

List of Tables

2.1 On-line optimization computation time $[s]$. 37
2.2 Total number of pallets processed by machines M_1, M_2, M_3, M_4: Comparison between the MPC algorithm and heuristic rules. 38

3.1 Machines absorbed power 61
3.2 Mean computational time $[s]$ per optimization step 67

5.1 Actuators nominal absorbed power. 104
5.2 Control sequence and actuators activation. 116

C.1 The control sequences (S_n) definition. 210
C.2 The control sequences (S_n) associated to the related transport modules (T_i). 211
C.3 The transport module (T_i) associated to the related control sequences (S_n) 212

Bibliography

[1] A. Alessandri, M. Gaggero, and F. Tonelli. Min-max and predictive control for the management of distribution in supply chains. *IEEE Trans. on Control Systems Technology*, 19(5):1075 – 1089, 2011.

[2] E. Angel, E. Bampis, and F. Kacem. Energy Aware Scheduling for Unrelated Parallel Machines. In *Green Computing and Communications (GreenCom), 2012 IEEE International Conference on*, pages 533–540, Nov 2012.

[3] T. Ávila, Á Corberán, I. Plana, and J.M. Sanchis. The stacker crane problem and the directed general routing problem. *Networks*, 65(1):43–55, 2015.

[4] O.I. Avram and P. Xirouchakis. Evaluating the use phase energy requirements of a machine tool system. *Journal of Cleaner Production*, (19):699–711, 2001.

[5] F. Basile, P. Chiacchio, and J. Coppola. A hybrid model of complex automated warehouse systems - Part I: modeling and simulation. *IEEE Transactions on Automation Science and Engineering*, 9(4):640–653, 2012.

[6] F. Basile, P. Chiacchio, and J. Coppola. A hybrid model of complex automated warehouse systems - Part II: analysis and experimental results. *IEEE Transactions on Automation Science and Engineering*, 9(4):654–658, 2012.

[7] L. D. Baskar, B. De Schutter, and H. Hellendoorn. Hierarchical traffic control and management with intelligent vehicles. In *2007 IEEE Intelligent Vehicle Symposium*, pages 834–839, Istambul, Turkey, Jun 13-15 2007.

[8] L. D. Baskar, B. De Schutter, and H. Hellendoorn. Model-based predictive traffic control for intelligent vehicles: dynamic speed limits and dynamic lane allocation. In *2008 IEEE Intelligent Vehicle Symposium*, pages 174–179, Eindhoven university of technology, Jun 4-6 2008.

[9] L. D. Baskar, B. De Schutter, and H. Hellendoorn. Traffic management for automated highway systems using model-based predictive control. *IEEE Transactions on Intelligent Transportation Systems*, 13(2), Jun 2012.

[10] A. G. Beccuti, T. Geyer, and M. Morari. Temporal lagrangian decomposition of model predictive control for hybrid systems. In *43th IEEE Conf. on Decision and Control*, Atlantis, Paradise Island, Bahamas, 2004.

[11] A. G. Beccuti, T. Geyer, and M. Morari. A Hybrid System Approach to Power Systems Voltage Control. In *44th IEEE Conference on Decision and Control, and European Control Conference*, pages 6774–6779, Seville, Spain, dec 2005.

[12] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.

[13] H.N. Bessenouci, Z. Sari, and L. Ghomri. Metaheuristic based control of a flow rack automated storage retrieval system. *Journal of Intelligent Manufacturing*, 23(4):1157–1166, aug 2012.

[14] M.W. Braun, D.E. Rivera, M.E. Flores, W.M. Carlyle, and K.G. Kempf. A model predictive control framework for robust management of multi-product, multi-echelon demand networks. *Annual Reviews in Control*, 27:229–245, 2003.

[15] A.A.G. Bruzzone, D. Anghinolfi, M. Paolucci, and F. Tonelli. Energy-aware scheduling for improving manufacturing process sustainability: a mathematical model for flexible flow shops on a generalized approach to manufacturing energy efficiency. *CIRP Annals − Manufacturing Technology*, pages 459–462, 2012.

[16] K. Bunse, M. Vodicka, P. Schönsleben, M. Brülhart, and F.O. Ernst. Integrating Energy Efficiency Performance in Production Management - Gap Analysis between Industrial Needs and Scientific Literature. *Journal fo Cleaner Production*, 19:667–679, 2011.

[17] E. F. Camacho and C. Bordons. *Model Predictive Control*. Springer, 2 edition, 2007.

[18] A. Cannata, S. Karnouskos, and M. Taisch. Energy efficiency driven process analysis and optimization in discrete manufacturing. In *35th Annual Conference of IEEE on Industrial Electronics Society - IECON 2009*, Porto - Portugal, 3-5 Nov 2009.

[19] C.G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Springer, 2 edition, 2008.

[20] A. Cataldo, , and R. Scattolini. Dynamic Pallet Routing in a Manufacturing Transport Line With Model Predictive Control. *IEEE Transactions on control systems technology*, 24:1812–1819, Sep 2016.

[21] A. Cataldo, A. Perizzato, and R. Scattolini. Management of a production cell lubrication system with model predictive control. In *APMS international conference on Advances in production systems, APMS 2014*, pages 131–138, Ajaccio, France, 20-24 Sep 2014.

[22] A. Cataldo, A. Perizzato, and R. Scattolini. Production scheduling of parallel machines with model predictive control. *Control Engineering Practice*, 42:28–40, 2015.

[23] A. Cataldo and R. Scattolini. Logic control design and discrete event simulation model implementation for a de-manufacturing plant. *Automazione-plus*, 2014.

[24] A. Cataldo and R. Scattolini. Modeling and model predictive control of a de-manufacturing plant. In *2014 IEEE Int. Conf. on Control Applications (CCA) - Part of 2014 IEEE Multi-conference on Systems and Control*, pages 1855–1860, Antibes Congress Center, Nice/Antibes, France, 8-10 Oct 2014.

[25] A. Cataldo, M. Taisch, and B. Stahl. Modeling, simulation and evaluation of energy consumption for a manufacturing production line. In *IECON 2013 - 39th Annual Conference of IEEE on Industrial Electronics Society*, Vienna, Austria, 10-13 Nov 2013.

[26] T. Cavalier, P. Pardalos, and A. Soyster. Modeling and integer programming techniques applied to propositional calculus. *Computer & Operation Research*, 17:561–570, 1999.

[27] S. Chiotellis, N. Weinert, and G. Seliger. Simulation-based, energy aware production planning. In *43th CIRP International Conference on Manufacturing Systems*, pages 964–972, Vienna, 26-28 May 2010.

[28] D.W. Clarke. Application of generalized predictive control to industrial processes. *IEEE Control Systems Magazine*, 122:49–55, 1988.

[29] M. Colledani, G. Copani, and T. Tolio. De-manufacturing systems. In *Variety Management in Manufacturing. Proceedings of the 47th CIRP Conference on Manufacturing Systems*, pages 14–19, Windsor, Ontario, Canada, 28-30 Apr 2014.

[30] G. Copani, A. Brusaferri, M. Colledani, N. Pedrocchi, M. Sacco, and T. Tolio. Integrated de-manufacturing systems as new approach to end-of-Life management of mechatronic devices. In *10th Global Conf. on Sustainable Manufacturing Towards Implementing Sustainable Manufacturing*, Istanbul, Turkey, 2012.

[31] D. W. Kim and D. G. Na and F. Chen. Unrelated parallel machine scheduling with setup times and a total weighted tardiness objective. *Robotics and Computer-Integrated Manufacturing*, 19(1-2):173–181, 2003.

[32] M. Dai, D. Tang, A. Giret, M.A. Salido, and W.D. Li. Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. *Robotics and Computer–Integrated Manufacturing*, pages 418–429, 2013.

[33] A.M. Deif. A system model for green manufacturing. *Advances Production Engineering Management*, 6(1), 2011.

[34] D.R. Dooly and H.F. Lee. A shift-based sequencing method for twin-shuttle automated storage and retrieval systems. *IIE Transactions*, 40(6):586–594, 2008.

[35] J. Du, C. Song, and P. Li. Multilinear Model Control of Hammerstein-like Systems Based on an Included Angle Dividing Method and the MLD-MPC Strategy. *Industrial & Engineering Chemistry Research*, 48(8):3934–3943, 2009.

[36] J.R. Duflou, J.W. Sutherland, D. Dornfeld, C. Herrmann, J. Jeswiet, S. Kara, M. Hauschild, and K. Kellens. Towards Energy and Resource Efficient Manufacturing: A Process and System Approach. *CIRP Annals - Manufacturing Technology*, 61:587–609, 2012.

[37] European Commission. ICT and energy efficiency - the case for manufacturing. *Reccomendations of the Consultation Group*, 2009.

[38] European Commission. Factory of the future. *Multi-annual roadmap for the contractual PPP under Horizon 2020*, 2013.

[39] K. Fang, N. Uhan, F. Zhao, and J.W. Sutherland. A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction. *Journal of Manufacturing Systems*, pages 234–240, 2011.

[40] J. Ferrio and J. Wassick. Chemical supply chain network optimization. *Computers and Chemical Engineering*, 32:2481–2504, 2008.

[41] N. Frigerio, A. Matta, L. Ferrero, and F. Rusiná. Modeling Energy Sattes in Machine Tools: An Automata Based Approach. In *20th CIRP International Conference on Life Cycle Engineering*, 17-19 Apr 2013.

[42] J.Y.H Fuh, Y.S. Wong, C.Y. Lee, L. Zhuang, and K.S. Neo. Modelling, analysis and simulation for the design of a robotic assembly system. *Computer Integrated Manufacturing Systems*, 9:19–31, 1996.

[43] A. Fysikopoulos, G. Pastras, T. Alexopoulos, and G. Chryssolouris. On a generalized approach to manufacturing energy efficiency. *The International Journal of Advanced Manufacturing Technology*, pages 1–16, 2014.

[44] A.H. Gharehgozli, Y. Yu, X. Zhang, and R. De Koster. Polynomial Time Agorithms to Minimize Total Travel Time in a Two-Depot Automated Storage/Retrieval System. *Transportation Science*, 2014.

[45] T. Gutowski, J. Dahmus, and A. Thiriez. Electrical Energy Requirements for Manufacturing Processes. In *13rd CIRP International Conference on Life Cycle Engineering*, Lueven, 31 May - 2 Jun 2006.

[46] A. Haït and C. Artigues. Scheduling parallel production lines with energy costs. In *13th IFAC Symposium on Information Control Problems in Manufacturing (INCOM09)*, June 2009.

[47] M.H. Han, L.F. McGinnis, J.S. Shieh, and J.A. White. On Sequencing Retrievals In An Automated Storage/Retrieval System, volume = 19, year = 1987,. *IIE Transactions*, (1):55–66.

[48] Y. He, B. Liu, X. Zhang, H. Gao, and X. Liu. A modelling method of task-oriented energy consumption for machining manufacturing system. *Journal of Cleaner Production*, 23:167–174, 2012.

[49] W. P. M. H. Heemels, B. de Shutter, and A. Bemporad. On the equivalence of classes of hybrid dynamical models. In *Proc. of the 40th IEEE Conf. on Decision and Control*, volume 1, pages 364–369, Orlando, FLorida, USA, 4-7 Dec 2001.

[50] B. Heinzl, M. Robler, N. Popper, W. Kastner, I. Leobner, K. Ponweiser, F. Dur, and F. Bleicher. Interdisciplinary Strategies for Simulation-Based Optimization of Energy Efficiency in Production Facilities. In *Proc. of the 15th International Conference on Computer Modeling and Simulation*, pages 304–309, apr 2013.

[51] M. Herceg, M. Kvasnica, C.N. Jones, and M. Morari. Multi-Parametric Toolbox 3.0. In *Proc. of the European Control Conference*, pages 502–510, Zürich, Switzerland, July 17-19 2013.

[52] C. Herrmann, S. Thiede, S. Kara, and J. Hesselbach. Energy oriented simulation of manufacturing system - Concept and application. *CIRP Annals − Manufacturing Technology*, 60:45–48, 2001.

[53] J.E. Hopcroft, R. Motwani, and J.D. Ullman. *Introduction to automata theory, languages and computation*. Addison-Wesley, 3 edition, 2006.

[54] ICF International. Energy trends in selected manufacturing sectors: Opportunities and challenges for environmentally preferable energy outcomes. *Final Report*, 2007.

[55] J. R. Jackson and I. E. Grossman. Temporal Decomposition Scheme for Nonlinear Multisite Production Planning and Distribution Models. *Industrial and Engineering Chemistry Research*, pages 3045–3055, May 2003.

[56] M. Jahangirian, T. Eldabi, A. Naseer, L.K. Stergioulas, and T. Young. Simulation in manufacturing and business: A review. *European Journal of Operational Research*, (203):1–13, 2010.

[57] K. Jansen and L. Porkolab. Improved Approximation Schemes for Scheduling Unrelated Parallel Machines. *Mathematics of Operations Research*, 26(2):324–338, 2001.

[58] A. J. Joines and S. D. Roberts. Simulation Modeling with SIMIO: A Workbook. In *Simulation Modeling with SIMIO: A Workbook*, 2013.

[59] D.E. Kaufman, J. Nonis, and R.L. Smith. A mixed integer linear programming formulation of the dynamic traffic asignment problem. In *IEEE International conference on Systems, Man and Cybernetics*, volume 1, pages 232–235, Chicago, IL, USA, 18-21 Oct 1992.

[60] B. Krellner, R. Kunis, and G. Rünger. Modeling of Energy-Sensitive Manufacturing Process. In *9th IEEE International Conference on Industrial Informatics (INDIN)*, 2011.

[61] S.S. Krishnan, N. Balasubramanian, E. Subrahmanian, V. Arun Kumar, G. Ramakrishna, A. Murali Ramakrishnan, and A. Krishnamurthy. Machine Level Energy Efficiency Analysis in Discrete Manufacturing for a Sustainable Energy Infrastructure. In *2th International Conference on Infrastructure Systems and Services: Developing 21st Century Infrastructure Networks*, 9-11 Dec 2009.

[62] L. Kerbache and J. MacGregor Smith. Multi-objective routing within large scale facilities using open finite queueing networks. *European Journal of Operational Research*, 121:105–123, 2000.

[63] L. Lao, M. Ellis, and P. D. Christofides. Smart manufacturing: handling preventive actuator maintenance and economics using model predictive control. *AIChE Jurnal*, 60:2179–2196, Jun 2014.

[64] A. Resano Lazaro and C.J. Luis Pérez. Dynamic analysis of an automobile assembly line considering starving and blocking. *Robotics and Computer–Integrated Manufacturing*, 25:271–279, 2009.

[65] H.F. Lee. Performance analysis for automated storage and retrieval systems. *IIE Transactions*, 29(1):15–28, 1997.

[66] D. Lefebvre. Deadlock-free scheduling for manufacturing systems based on timed Petri nets and model predictive control. In *54th IEEE Conference on Decision and Control (CDC)*, pages 3013–3018, Osaka, Japan, 15-18 Dec 2015.

[67] D. Lefebvre. Approaching minimal time control sequences for timed Petri nets. *IEEE Transactions on Automatic Science and engineering*, 13(2):1215–1221, Apr 2016.

[68] D. Lefebvre and E. Leclercq. Control design for trajectory tracking with untimed Petri nets. *IEEE Transactions on Automatic Control*, 60(7):1921–1926, Jul 2015.

[69] H.R. Lewis and C. Papadimitriou. *Elements of the Theory of Computation*. Prentice Hall, 2 edition, 1997.

[70] S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn. An efficient model-based method for coordinated control of urban traffic networks. In *Int. Conference on Netwotking, Sensing and Control*, pages 8–13, Chicago, IL, USA, 10-12 Apr 2010.

[71] S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn. Model predictive control for urban traffic networks via MILP. In *IEEE American Control Conference ACC '10*, pages 2272–2277, Marriott Waterfront, Baltimore, MD, USA, Jun 30- Jul 02 2010.

[72] S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn. Fast Model Predictive Control for Urban Road Networks via MILP. *IEEE Trans. on Intelligent Transport Systems*, 12:846–856, Sept 2011.

[73] D.A. Linkers and M. Mahfonf. *Advances in model-based predictive control*. Oxford University Press, 1994.

[74] Y. Löfberg. YALMIP : A Toolbox for Modeling and Optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[75] C. Lucas, G. Mitra, and S. Moody. Tools for reformulating logical forms into zero-one mixed integer programs (MIPS). *Maths Technical Papers (Brunel University)*, (TR/03/92):1–27, Apr 1992.

[76] P.B. Luh and D.J. Hoitomt. Scheduling of manufacturing systems using the lagrangian relaxation technique. *IEEE Trans. on Automatic Control*, pages 1066–1079, 1993.

[77] J. M. Maciejowski. *Predictive Control with Constraints*. Pearson Prentice Hall, 1 edition, 2002.

[78] S. Mahajan, B.V. Rao, and B.A. Peters. A retrieval sequencing heuristic for miniload end-of-aisle automated storage/retrieval systems. *International Journal of Production Research*, 36(6):1715–1731, 1998.

[79] K.I.M. McKinnon and H.P. Williams. Constructing integer programming models by the predicate calculus. *Annals of Operations Research*, (21):227–246, 1989.

[80] S. Miremadi, B. Lennartson, and K. Åkesson. A BDD-based approach for modeling plant and supervisor by extended finite automata. *IEEE Transactions on Control Systems Technology*, 20(6):1421–1435, 2012.

[81] M. Mori, M. Fujishima, Y. Inamasu, and Y. Oda. A study on enrgy efficiency improvement for machine tools. *CIRP Annals − Manufacturing Technology*, 60:145–148, 2011.

[82] E. Müller, T. Stock, and R. Schilling. A Method to Generate Energy Value-Streams in Production and Logistics in Respect of Time- and Energy-Consumption. *Production Engineering*, (8):243–251, 2014.

[83] N.Groot, B. De Schutter, and H. Hellendoorn. Integrated Model Predictive Traffic and Emission Control Using a Piecewise-Affine Approach. *IEEE Transactions on Intelligent Transportation Systems*, 14(2):587–598, jun 2013.

[84] Institute of Industrial Technology and Automation (ITIA) National Research Council (CNR). The de-manufacturing pilot plant. 11 Jun 2013.

[85] Organization for Economic Cooperation and Development. OECED Key Enviromental Indicators. 2004.

[86] J. Gómez Ortega and E.F. Camacho. Mobile robot navigation in a partially structured environment using neural predictive control. *Control Engineering Practice*, 4:1669–1679, 1996.

[87] C. Khiang Pang, C. Vinh Le, O. Peen Gan, X. Min Chee, D. Hong Zhang, M. Luo, H. Leng Chan, and F. Lewis. Intelligent energy audit and machine management for energy-efficient manufacturing. In *5th IEEE International Conference on Cybernetics and Intelligent Systems*, 2011.

[88] E. Perea-López, B. E. Ydstie, and I. E. Grossmann. A model predictive control strategy for supply chain optimization. *IEEE Computers and Chemical Engineering*, (27):1201–1218, Jun 2003.

[89] A. Perizzato and R. Scattolini. "Genomic Model Predictive Control Tools for Evolutionary Plants" project - Deliverable D5.1 - Test-bed Simulation Model. Technical report, Milan, Oct 2014. Italian project: Progetto Bandiera "La fabbrica del futuro".

[90] M.L. Pinedo. *Scheduling*. Springer, 2008.

[91] S.J. Qin and T.A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764, 2003.

[92] G. Rabadi, R.J. Moraga, and A. Al–Salem. Heuristics for the Unrelated Parallel Machine Scheduling Problem with Setup Times. *Journal of Intelligent Manufacturing*, 17(1):85–97, 2006.

[93] R. Raman and I. E. Grossmann. Relation between MILP modelling and logical inference for chemical process synthesis. *Compurers and Chemical Engineering*, 15(2):73–84, 1991.

[94] R. Raman and I. E. Grossmann. Integration of logic and heuristic knowledge in MINLP optimization for process synthesis. *Compurers and Chemical Engineering*, 16(3):155–171, 1992.

[95] J. B. Rawlings and D. Q. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, Madison, Wisconsin, USA, 1 edition, Aug 2009.

[96] J. Richalet. Industrial applications of model based predictive control. *Automatica*, 9(5):1251–1274, 1993.

[97] J. Richalet, A. Rault, J.L. Testud, and J. Papon. Model predictive heuristic control: application to industrial processes. *Automatica*, 14(2):413–428, 1978.

[98] K.J. Roodbergen and I.F.A. Vis. A survey of literature on automated storage and retrieval systems. *European Journal of Operational Research*, 194(2):343–362, 2009.

[99] K. Salonitis and P. Ball. Energy efficient manufacturing from machine tools to manufacturing. In *46th CIRP Conference on Manufacturing Systems*, volume Procedia CIRP 7, pages 634–639, 2013.

[100] R. Scattolini. Architectures for distributed and hierarchical model predictive control – a review. *Journal of Process Control*, 19:723–731, 2009.

[101] V. Schmid, K. F. Doerner, and G. Laporte. Rich routing problems arising in supply chain management. *European Journal of Operational Research*, (224):435–448, 2013.

[102] G. Seliger. *Wealth distribution and use productivity. Ed. Sustainability in manufacturing*. Springer, 2007.

[103] P. Senthilkumar and S. Narayanan. Literature Review of Single Machine Scheduling Problem with Uniform Parallel Machines. *Intelligent Information Management*, 2(8):457–474, 2010.

[104] Y. Seow and S. Rahimifard. A framework for modelling energy consumption within manufacturing systems. *CIRP Journal of Manufacturing Science and Technology*, 4:258–264, 2011.

[105] M. Sköldstam, K. Åkesson, and M. Fabian. Modeling of discrete event systems using finite automata with variables. In *46th IEEE Conference on Decision and Control (CDC)*, pages 3387–3392, New Orleans, LA, USA, 12-14 Dec 2007.

[106] P. Solding and P. Thollander. Increased energy efficiency in a Swedish iron foundry through use of discrete event simulation. In *Winter Simulation Conference*, pages 1971–1976, 2006.

[107] B. Stahl, M. Taisch, A. Cannata, F. Mueller, S. Thiede, C. Herrmann, A. Cataldo, and F. Cavadini. Combined Energy, Material and Building Simulation for Green Factory Planning. In *20th CIRP Conference on Life Cycle Engineering*, Singapore, 17-19 Apr 2013.

[108] M. Taisch, B. Stahl, F. Vaccari, and A. Cataldo. A production-state based approach for energy flow simulation in manufacturing systems. In *APMS 2013 Conference*, pages 227–234, 9-12 Sept 2013.

[109] A. N. Tarău, B. De Schutter, and H. Hellendoorn. Travel time control of destination coded vehicles in baggage handling systems. In *17th IEEE Int. Conf. on Control Applications*, pages 5293–298, San Antonio, Texas, USA, Sep 3-5 2008.

[110] A. N. Tarău, B. De Schutter, and H. Hellendoorn. Centralized versus decentralized route choice control in DCV-based baggage handling systems. In *IEEE Int. Conf. on Networks, Sensing and Control*, pages 334–339, Okayama, Japan, Mar 26-29 2009.

[111] A. N. Tarău, B. De Schutter, and H. Hellendoorn. Distributed route choice control in DCV-based baggage handling systems. In *18th IEEE Int. Conf. on Control Applications*, pages 818–824, Saint Petersburg, Russia, Jul 8-10 2009.

[112] A. N. Tarău, B. De Schutter, and H. Hellendoorn. Model-cased control for route choice in automated baggage handling systems. *IEEE Trans. on Systems, Man and Cybernetics - Part C: applications and reviews*, 40(3), 2010.

[113] S. Terrazas-Moreno, I. E. Grossmann, and P. Trotter. A rigorous comparative study of temporal versus spatial Lagrangean decomposition in production planning problems. In *20th European Symposium on Computer Aided Process Engineering*, volume 28, pages 1225–1230, 2010.

[114] F.D. Torrisi and A. Bemporad. HYSDEL – – – A tool for generating computational hybrid models for analysis and synthesis problems. *IEEE Trans. on Control Systems Technology*, 12:235–249, Mar 2004.

[115] F. Vaccari. *Study of the Energy Behaviours of a Manufacturing System*. Thesis, Politecnico di Milano, Facoltà di Ingegneria dei Sistemi, Corso di laurea in Ingegneria gestionale, SupervisorProf. M. Taisch, Co-supervisor B. Stahl, Co-supervisor Politecnico di Torino Prof. C. Rafele, Milan - Italy, 2012.

[116] H. van Ekeren, R. Negenborn, P. van Overloop, and B. De Schutter. Time-instant optimization for hybrid model predictive control of the Rhine-Meuse Delta. *Journal of Hydroinformatics*, 15:271–292, 2013.

[117] F. D. Vargas-Villamir and D. E. Rivera. Multilayer optimization and scheduling using model predictive control: application to reentrant semiconductor manufacturing lines. *Computers and Chemical Engineering*, 24(8):2009–2021, 2000.

[118] F. D. Vargas-Villamir and D. E. Rivera. A model predictive control approach for real-time optimization of reentrant manufacturing lines. *Computers in Industry*, 45(1):45–57, 2001.

[119] W. Wang and D.E. Rivera. Model predictive control for tactical decision-making in semiconductor manufacturing supply chain management. *IEEE Trans. on Control Systems Technology*, 16(5):841–855, Sep 2008.

[120] N. Weinert, S. Chiotellis, and G. Seliger. Methodology for planning and operating energy-efficient production system. *CIRP Annals − Manufacturing Technology*, 60:41–44, 2011.

[121] M. X. Weng, J. Lu, and H. Ren. Unrelated parallel machine scheduling with setup consideration and a total weighted completion time objective. *International Journal of Production Economics*, 70(3):215 – 226, 2001.

[122] H.P. Williams. *Model Building in Mathematical Programming. Fifth edition*. WILEY, 2013.

[123] B. Xia, C. Song, and B. Wu. An integrated state space partition and optimal control method of multi-model for nonlinear systems with state estimation. In *27th Chinese Control and Decision Conference*, pages 1604–1609, may 2015.

[124] B.E. Ydstie. Distributed decision making in complex organizations: the adaptive enterprise. *Computers and Chemical Engineering*, 29:11–27, 2004.

[125] Y. Yu and R. De Koster. Sequencing heuristics for storing and retrieving unit loads in 3d compact automated warehousing systems. *IIE Transactions*, 44:69–87, 2012.