J. A. Stiver and P. J. Antsaklis, "A Novel Discrete Event System Approach to Modeling and Analysis of Hybrid Control Sysytems," P roc. o f t he T wenty-ninth A nnual A llerton C o nference o n C ommunication, C ontrol a nd C omputing , Univ. of Illinois at Urbana-Champaign, October 2-4, 1991.

A NOVEL DISCRETE EVENT SYSTEM APPROACH TO MODELING AND ANALYSIS OF HYBRID CONTROL SYSTEMS

James A. Stiver and Panos J. Antsaklis
Dept. of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556
Phone: (219) 239-6916, (219) 239-5792
E-mail: jstiver@bach.helios.nd.edu, flxfsn@irishmvs.cc.nd.edu

1. INTRODUCTION

In this paper, we present a framework for the modeling and analysis of Hybrid Control Systems (HCS). For our purposes, an HCS is a system consisting of a time-driven, continuous-state system, which is being supervised and controlled by a discrete-state symbolic system. In particular, a hybrid control system consists of two rather distinct parts: a system adequately described by a set of difference or differential equations, which shall be called the plant, and a higher level decision making controller, modeled as a discrete-event system. Note that the use of the term "hybrid" is quite distinct from its common use in the control field referring to systems with both digital and analog components. Hybrid control systems are quite common in practice. A familiar hybrid control system is the temperature control of a typical home, if the furnace and air conditioner are modeled as continuous-time systems as they are being controlled by a thermostat which can be regarded as a decision making system. Recently, attempts have been made to study hybrid control systems in a unified, analytical framework [4, 12, 17, 18, 24].

2. AN APPROACH TO MODELING HYBRID CONTROL SYSTEMS

The difficulty in modeling and analysis of HCS arises from their heterogeneous nature. The plant is a continuous-state system which evolves in real time, while the controller is an event driven system which evolves in logical time. In order for the plant and controller to interact, the HCS must contain an additional part which serves as an interface and allows an exchange of information between the plant and controller. The three parts of a hybrid control system, controller, interface, and plant, can be envisioned as a hierarchical system with the controller as the highest layer, the interface in the middle, and the plant as the lowest layer.

The model described here, and used throughout this work, is a refinement of the above three layer hierarchy made up of four interacting blocks. Two of these blocks, the plant and controller, relate to the corresponding parts of the hierarchy, and the remaining two blocks, the actuator and aggregator, each represent a portion of the interface. The actuator relays commands from the controller to the plant, and the aggregator carries information from the plant to the controller. Figure 2.1 shows the arrangement of these four blocks as well as the names and locations of the signals which provide communication between the blocks. Each of the blocks and its associated signals will be described now.

2.1 Plant

The plant is modeled as a time-invariant, continuous-time system (CTS). This can be used to represent any type of system which evolves in real time and can be suitably approximated by ordinary differential equations. Formally the plant is specified by the following equations:

$$\dot{x}(t) = f(x(t), r(t)) \quad (1)$$
$$z(t) = g(x(t)) \quad (2)$$

• Proc. of Allerton Conf., October 1991

J. A. Stiver and P. J. Antsaklis, "A Novel Discrete Event System Approach to Modeling and Analysis of Hybrid Control Sysytems," P roc. o f t he T wenty-ninth A nnual A llerton C o nference o n C ommunication, C ontrol a nd C omputing , Univ. of Illinois at Urbana-Champaign, October 2-4, 1991.

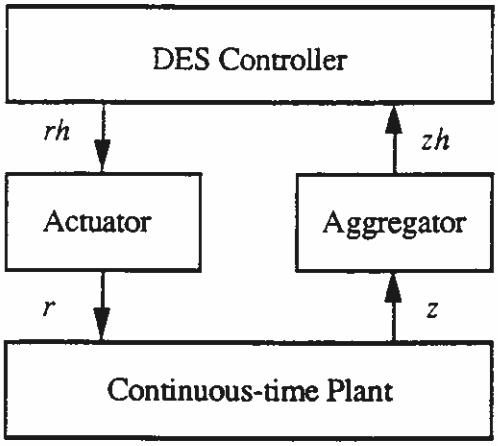

Figure 2.1: Hybrid Control System Model

where $x(t) \in \mathbf{R}^n$ is the plant state, $r(t) \in \mathbf{R}^m$ is the input, and $z(t) \in \mathbf{R}^p$ is the measured output. For the purposes of this work we assume that $z(t) = x(t)$. Note that a discrete-time system (DTS) plant can also be used by replacing equation (1) with $x(k+1) = f(x(k), r(k))$.

2.2 Aggregator

The aggregator contains the portion of the interface responsible for relaying information from the plant to the controller. The input to the aggregator is the measured output, *z*, and the output is *zh*. The task of the aggregator, therefore, is to convert the continuous-time signal, *z*, to a discrete-event signal, *zh*. To accomplish this the aggregator partitions the state space of the plant into a number of regions, where each region is associated with an event, $e_i \in E$, where E is a set of possible events. It is possible that more than one region can be associated with the same event. Whenever the state of the plant moves from one region to another, the aggregator outputs the event associated with the new region. The aggregator then remains silent until the plant state moves into another new region.

Mathematically the aggregator can be specified by a function, α, which maps each point in $\mathbf{R}^n$ to an event in E. α is not an injective function, so it induces equivalence classes in $\mathbf{R}^n$. These equivalence classes form the aforementioned regions in $\mathbf{R}^n$ which partition the state space, and are called *aggregator regions*.

Definition 2.1: The *aggregator regions* generated by the function α in the space $\mathbf{R}^n$ form a partition of $\mathbf{R}^n$ such that two points, $r_1, r_2 \in \mathbf{R}^n$, are in the same aggregator region iff there exists a curve in $\mathbf{R}^n$ from r_1 to r_2 such that $\alpha(r) = \alpha(r_1) = \alpha(r_2)$ for any point, r, on the curve. The set of aggregator regions shall be denoted by **A**. □

Thus

$\alpha : \mathbf{R}^n \rightarrow E$	aggregator function
$z(t) = x(t) \in \mathbf{R}^n$	aggregator input
$zh_i \in E$	aggregator output

The action of the aggregator can be written as

$$zh = sp[\alpha(x)] \tag{3}$$

where we abuse notation slightly by allowing α to operate on a signal. *sp* is a function which suppresses repeated events so that the aggregator generates an event only when the plant state

first enters a region. The function α is intended to induce equivalence classes which generate reasonably large regions in the state space, large enough such that the plant state can evolve for a non-zero amount of time and remain in the same region. As a result, *zh* will be a discrete and asynchronous signal which has a value only at the times that x moves from one equivalence class of α to another. The model for the aggregator may seem a bit restrictive, but its ability to handle a variety of systems will be discussed later.

2.3 DES Controller

The controller is discrete-event system, meaning it reacts to discrete symbolic input. In this case, the input and output consist of an asynchronous sequence of symbols which are called events. Several modeling strategies have been developed for discrete-event systems. Here the controller is modeled by a deterministic automaton which may have an infinite number of states. The automaton is specified by a quintuple, $\{S, E, C, \delta, \phi\}$, as follows

$S = \{s_1, s_2, s_3, \ldots\}$	set of controller states
$E = \{e_1, e_2, e_3, \ldots\}$	set of events (controller inputs)
$C = \{c_1, c_2, c_3, \ldots\}$	set of commands (controller outputs)
$\delta : S \times E \rightarrow S$	state transition function
$\phi : S \rightarrow C$	output function

As mentioned previously, the automaton may have an infinite number of states, in fact it may have an infinite number of events and commands as well. This model can be used to represent a wide variety of discrete-event systems, it follows the model found in [14] and is similar to those found in [20, 26].

The input signal to the controller is *zh* and the output signal is *rh*. The names reflect the fact that they are the high level signals corresponding to the continuous-time signals z and r. Whenever an event is received by the controller on *zh*, the controller immediately undergoes a state transition determined by the state transition function,

$$t_i = \delta(t_{i-1}, zh_i) \tag{4}$$

where $t_i \in S$, $zh_i \in E$, and i is a time index. With each state transition, the controller generates an output by the output function,

$$rh_i = \phi(t_i) \tag{5}$$

where $rh_i \in C$. Note that since the state transitions and outputs of the DES controller occur immediately upon receiving an input event, there is no need to include time in the model. The delay between successive state transitions and outputs is determined completely by the input signal, *zh*. Also note that the subscript on a signal, e.g. zh_i, is used to indicate the logical order of the event, state, or command carried on that signal. The subscript on a set member, e.g. e_i, however, refers to a particular member of the set.

2.4 Actuator

In this model the actuator receives a discrete-event signal, *rh*, carrying commands from the controller. The unit in turn produces a continuous-time signal, r, which serves as the input to the plant. To produce r, the actuator must be programmed with a method to translate each command in C into a numeric value for r. The signal, r, is piecewise constant for it depends only on the most recent controller command and will not change until the next command is received. In this way the actuator operation is similar to a hold circuit. The action of the actuator is determined by the function γ, where

$$\gamma : C \rightarrow \mathbf{R}^m \qquad \text{actuator function}$$

$r(t) \in \mathbf{R}^m$ actuator output

$rh_i \in C$ actuator input

and described by

$$r = \text{hold}[\gamma(rh)] \tag{6}$$

where we allow γ to operate on a signal, and *hold* is a function which retains the current value until a new argument appears.

2.5 Example: Double Integrator

As an example of an HCS we will consider a system consisting of a double integrator and a simple automaton. The double integrator forms the plant, and it is specified by the equation

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r. \tag{7}$$

The aggregator partitions the state-space into four regions corresponding to the four quadrants, such that $\alpha(x) = e_i$ if x lies in quadrant i. The controller consists of a three state automata as shown in Figure 2.2 where c_i is the output associated with state s_i. Finally, the actuator maps the three commands, c_1, c_2, and c_3, to -10, 0, and 10 respectively. The behavior of this system is to drive the state of the plant to zero, as can be seen in Figure 2.3 which shows various state trajectories.

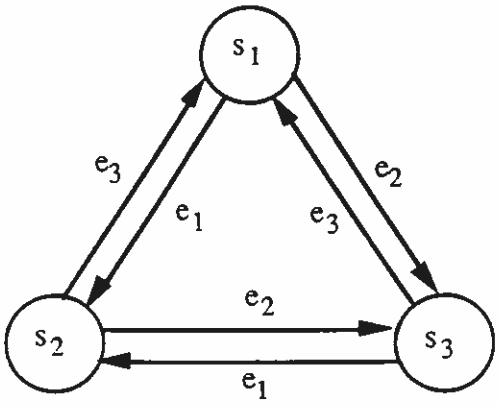

Figure 2.2: DES Controller for Double Integrator

3. A DISCRETE EVENT SYSTEM MODEL FOR HYBRID CONTROL SYSTEMS

The difficulty in analysis presented by Hybrid Control Systems is due to the incompatibility between the models employed in the various blocks which make up the HCS, namely the DES controller and the CTS or DTS plant. A scheme in which a single modeling strategy is used would facilitate analysis. In this section, a method for representing the entire HCS as two interacting DES's will be described. The benefits and limitations of the new model will be described as well as the relationship between the results obtained from the new model and the nature of the original model.

3.1 DES Equivalent Plant

Notice that from the point of view of the DES controller the remainder of the HCS appears as a DES. This is because the input and output of the actuator, plant, and aggregator,

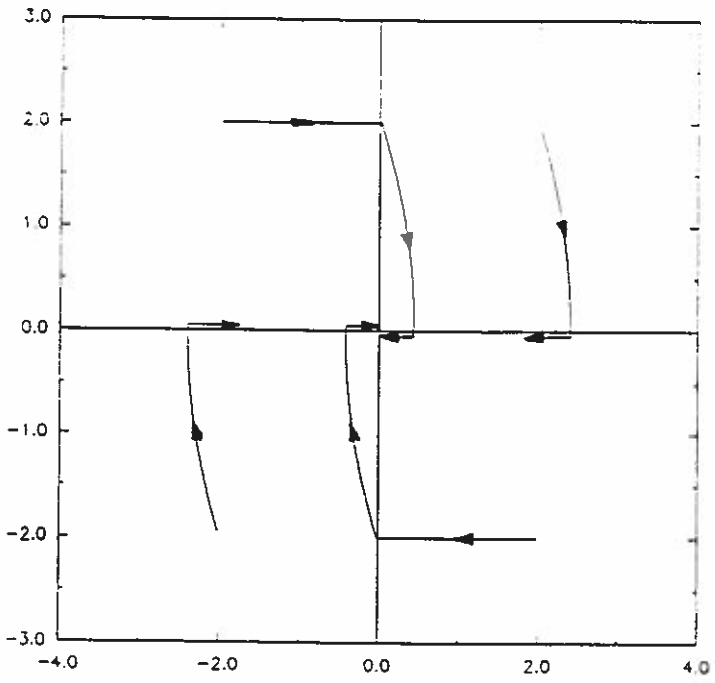

Figure 2.3: Various State Trajectories of the Double Integrator

taken together, are discrete-event signals. This allows the HCS model described in Section 2 to be converted into a discrete-event system model by combining the actuator, plant, and aggregator into a single block. The discrete-event controller remains the same in the new model and the newly formed block constitutes another DES which shall be referred to as the *DES equivalent plant*. Like the controller, the DES equivalent plant is modeled by an automaton, but unlike the controller this automaton may not be deterministic, and the state transitions do not occur instantly when an input is received. However, one can determine exactly when the transitions occur because of the time dependence of the underlying system. The DES equivalent plant is specified by the quintuple $\{P, C, E, \psi, \phi_p\}$ as follows

$P = \{p_1, p_2, p_3, \ldots\}$	the set of states
$C = \{c_1, c_2, c_3, \ldots\}$	the set of inputs (commands)
$E = \{e_1, e_2, e_3, \ldots\}$	the set of outputs (events)
$\psi : P \times C \rightarrow \mathbf{P}(P)$	the state transition function
$\phi_p : P \rightarrow E$	the output function

where $\mathbf{P}(P)$ denotes the power set of P. Each state in P is uniquely associated with exactly one of the aggregator regions, i.e. there is a one-to-one correspondence between the members of the set P and the members of set **A**. This correspondence is such that when the state of the real-time plant is in particular member of **A**, the state of the DES equivalent plant will be the corresponding member of P. The function ϕ_p plays the same role as α because it maps the plant state to an event in E. Notice that the state transition function, ψ, maps to an element of the power set of P, indicating that the automaton is nondeterministic in general. The function ψ is also a partial function in general.

We now express the state transition function, ψ, for the case of a DTS plant. We will use $xh_i \in P$ to denote the state of the DES equivalent plant at time i.

$$\psi(xh_i, rh_i) = \phi_p{}^{inv}(\alpha(f(\alpha^{inv}(\phi_p(xh_i)), \gamma(rh_i)))) - \{xh_i\} \qquad (8)$$

J. A. Stiver and P. J. Antsaklis, "A Novel Discrete Event System Approach to Modeling and Analysis of Hybrid Control Sysytems," P roc. o f t he T wenty-ninth A nnual A llerton C o nference o n C ommunication, C ontrol a nd C omputing , Univ. of Illinois at Urbana-Champaign, October 2-4, 1991.

Note that the notation α^{inv} and $\phi_p{}^{inv}$ is used to denote the preimage of α and ϕ_p. The DES equivalent plant is non-deterministic in general because α^{inv} maps a single event to a subset of $\mathbf{R}^n$

$$\alpha^{inv} : E \rightarrow \mathbf{P}(\mathbf{R}^n)$$

which will generally result in a set, containing more than one state, as the value of $\psi(xh_i, rh_i)$. In addition, $\phi_p{}^{inv}$ may map a single event to several DES equivalent plant states

$$\phi_p{}^{inv} : E \rightarrow \mathbf{P}(P)$$

resulting in an even larger set for $\psi(xh_i, rh_i)$.

The DES equivalent plant represents the extent to which the DES controller in a particular hybrid control system can regulate the plant. The control of the plant is limited by two factors. First, the plant input is limited to those set points which can be generated by the actuator. This limitation reduces the number of possible state trajectories which can be realized in the plant, and this is reflected in the DES equivalent plant by a reduced number of possible state transitions.

The second limitation on the control of the plant is caused by the aggregator. The aggregator does not provide the DES controller with the exact state of the plant, but rather with a sort of approximation. Thus, based solely on information from the aggregator, the response of the plant to a given input is not known precisely. This effect appears in the DES equivalent plant as nondeterministic state transitions. That is, for a given input and state, the DES equivalent plant may have several possible transitions. The example in Section 3.3 clearly shows this effect. First however, we derive the conditions for determinism.

3.2 Determinism in the DES Equivalent Plant

We would like to know what restrictions can be placed on the hybrid system to ensure that the DES equivalent plant can be represented by a deterministic automaton. This is because the exact behavior of the system can only be determined if the system is deterministic.

Definition 3.1 : A DES equivalent plant is said to be deterministic iff $\forall\ p_i \in P, c_j \in C$, we have $\psi(p_i, c_j) \in \{p_l, \text{void}\}$ where $p_l \in P$. □

If ψ is to represent a deterministic automaton, then the expression on the right side of equation (8) must take on values which are sets with only one member.

Theorem 3.1 : The DES equivalent plant will be deterministic iff

$$\forall\ a_1 \in \mathbf{A}, rh_i \in C\ \exists\ a_2 \in \mathbf{A}\ \text{s.t.}\ x(k+1) \in a_1 \cup a_2$$

where

$$x(k) \in a_1$$

and

$$x(k+1) = f(x(k), \gamma(rh_i)).$$

Proof

Recall that $\mathbf{A}$ is the set of aggregator regions and that there is a one-to-one correspondence between the aggregator regions and the states of the DES equivalent plant. Thus by making the correspondence

$$p_i \Leftrightarrow a_1 \text{ and } p_j \Leftrightarrow a_2,$$

the theorem can be rewritten as

$$\forall\ p_i \in P, rh_i \in C\ \exists\ p_j \in P\ \text{s.t.}\ p \in \{p_i, p_j\}$$

where p_i and p are the DES equivalent plant states at time(k) and time(k+1) respectively. Thus Theorem 3.1 is a basically a restatement of the definition a deterministic DES equivalent plant, saying that if the state changes, there is a unique state to which it will change. □

The restriction of Theorem 3.1 guarantees that if the current value of *zh* and *rh* are known, then the next value of *zh* can be determined uniquely and thus ψ is a function in the conventional sense because it returns only one value. Another way of stating the theorem is that, under any given value of *rh*, all real-time plant trajectories in any given aggregator region lead to the same subsequent aggregator region.

Often the restriction of Theorem 3.1 is unnecessary. If a particular control objective is to be attained, the non-deterministic transitions may not be relevant to the design of the controller. In any case, the state transition function of a DES equivalent plant can be separated into a deterministic part and a non-deterministic part. The deterministic part will be a partial function in general because it may not be defined for all states and inputs. Let the deterministic part of the state transition function be ψ_d such that

$$\psi_d : P \times C \rightarrow P$$

where

$$\psi_d(xh_i, rh_i) \in \psi(xh_i, rh_i)$$

and ψ_d is defined only for those pairs (p_i, c_j) where $\|\psi(p_i, c_j)\| = 1$. The notation $\|X\|$ refers to the cardinality of set X. Furthermore if the restriction of Theorem 3.1 is met, then

$$\psi(xh_i, rh_i) = \{\psi_d(xh_i, rh_i)\}.$$

The deterministic portion of the DES equivalent plant, described by the function ψ_d, represents in a sense the controllable portion of the DES equivalent plant. The state transitions included in ψ_d can be enabled by the controller and there is no uncertainty in which transition will occur. If these deterministic transitions are sufficient to achieve the control objectives for the system, then the non-deterministic part of the DES equivalent plant is of no consequence.

3.3 Example: Double Integrator

The double integrator system from Section 2.5 is a case of an HCS with a non-deterministic DES equivalent plant. Figure 3.1 shows the state trajectories of the real-time plant under the three available control inputs, along with the aggregator regions. The trajectories which are roughly parallel are under the same input, *r*. As can be seen the aggregator regions this example do not lead to a deterministic DES plant. The reason for this situation is that the aggregator does not preserve enough information about the state of the actual plant. Therefore, by changing the aggregator to add more regions as shown in Figure 3.2, the system can be made to have a deterministic DES equivalent plant. The new aggregator is a refinement of the original which preserves more information about the actual plant state and leads to the deterministic DES equivalent plant shown in Figure 3.3. The original aggregator was sufficient for the stabilizing DES controller of the original example, but the new aggregator will allow any controller to be designed without the worry of unpredictable plant behavior.

J. A. Stiver and P. J. Antsaklis, "A Novel Discrete Event System Approach to Modeling and Analysis of Hybrid Control Sysytems," P roc. o f t he T wenty-ninth A nnual A llerton C o nference o n C ommunication, C ontrol a nd C omputing , Univ. of Illinois at Urbana-Champaign, October 2-4, 1991.

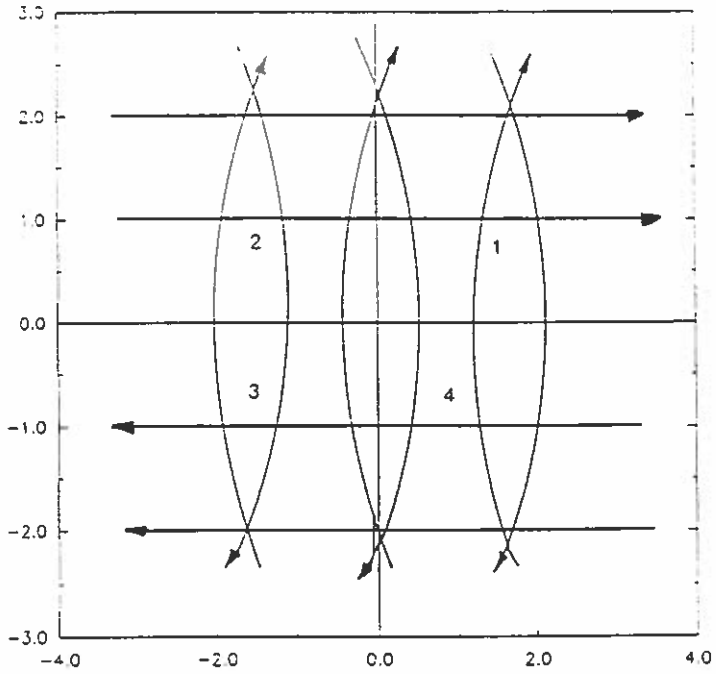

Figure 3.1: Original Aggregator Regions with State Trajectories

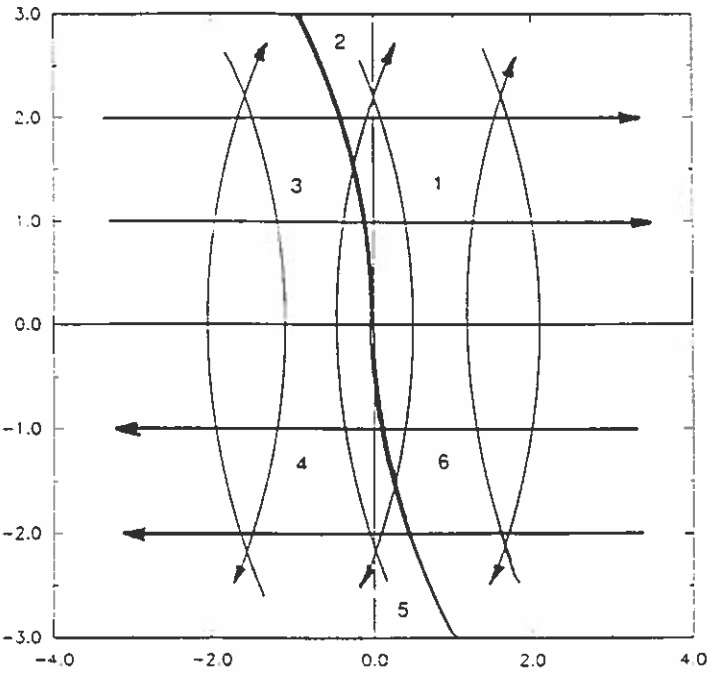

Figure 3.2: New Aggregator Regions with State Trajectories

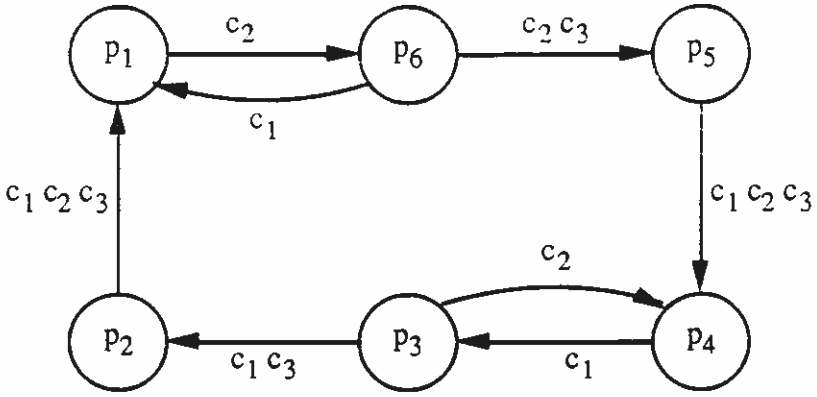

Figure 3.3: The DES Equivalent Plant for the Double Integrator

Final Note

A more thorough treatment of this research can be found in [25], which includes an application of Lyapunov stability to HCS.

7. REFERENCES

[1] Antsaklis, P. J., Passino, K., Wang, S., "Towards Intelligent Autonomous Control Systems: Architecture and Fundamental Issues", *Journal of Intelligent and Robotic Systems*, pp. 315-342, 1989.

[2] Åstrom, J., Wittenmark, B., *Computer-Controlled Systems*, Prentice-Hall, NJ, 1990.

[3] Benveniste, A., Le Guernic, P., "Hybrid Dynamical Systems Theory and Nonlinear Dynamical Systems Over Finite Fields", *New Trends in Nonlinear Control Theory*, pp. 485-495, Springer-Verlag, NY, 1989.

[4] Benveniste, A., Le Guernic, P., "Hybrid Dynamical Systems and the SIGNAL Language", *IEEE Transactions on Automatic Control*, Vol. 35, No. 5, pp. 535-546, May 1990.

[5] Borgne, M., Benveniste, A., Le Guernic, P., "Polynomial Ideal Theory Methods in Discrete Event and Hybrid Dynamical Systems", Proceedings of the 28th Conference on Decision and Control, pp. 2695-2700, Tampa, FL, December 1989.

[6] Cao, X., Ho, Y., "Models of Discrete Event Dynamic Systems", *IEEE Control Systems Magazine*, pp. 69-76, June 1990.

[7] Carroll, J., Long, D., *Theory of Finite Automata*, Prentice-Hall, N.J., 1989.

[8] Cassandras, C., Strickland, S., "Sample Path Properties of Timed Discrete Event Systems", *Proceedings of the IEEE*, Vol. 77, No. 1, pp. 59 - 71, January 1989.

[9] Cassandras, C., Ramadge, P., "Toward a Control Theory for Discrete Event Systems", *IEEE Control Systems Magazine*, pp. 66-68, June 1990.

[10] Fishwick, P., Zeigler, B., "Creating Qualitative and Combined Models with Discrete Events", Proceedings of The 2nd Annual Conference on AI, Simulation and Planning in High Autonomy Systems, pp. 306-315, Cocoa Beach, FL, April 1991.

[11] Francis, B., Wonham, W. M., "The Internal Model Principal of Control Theory", *Automatica*, vol. 12, no. 5, pp. 457-465, May 1976.

[12] Gollu, A., Varaiya, P., "Hybrid Dynamical Systems", Proceedings of the 28th Conference on Decision and Control, pp. 2708-2712, Tampa, FL, December 1989.

[14] Passino, K., "Analysis and Synthesis of Discrete Event Regulator Systems", Ph. D. Dissertation, Dept. of Electrical and Computer Engineering, Univ. of Notre Dame, Notre Dame, IN, April 1989.

[16] Passino, K., Michel, A., Antsaklis, P. J., "Lyapunov Stability of a Class of Discrete Event Systems", Proceedings of the American Control Conference, Boston MA, June 1991. To appear.

[17] Peleties, P., DeCarlo, R., "Modeling of Interacting Continuous Time and Discrete Event Systems : An Example", Proceedings of the 26th Allerton Conference on Comunication,

Control, and Computing, pp. 1150-1159, Univ. of Illinois at Urbana-Champaign, October 1988.

[18] Peleties, P., DeCarlo, R., "A Modeling Strategy with Event Structures for Hybrid Systems", Proceedings of the 28th Conference on Decision and Control, pp.1308 - 1313, Tampa FL, December 1989.

[20] Ramadge, P., Wonham, W. M., "The Control of Discrete Event Systems", *Proceedings of the IEEE*, Vol. 77, No. 1, pp. 81 - 98, January 1989.

[22] Sain, M. K., *Introduction to Algebraic System Theory*, Acedemic Press, NY, 1981.

[23] Sontag, E. D., *Mathematical Control Theory - Deterministic Finite Dimensional Systems*, Springer-Verlag, NY, 1990.

[24] Stiver, J., "Modeling of Hybrid Control Systems Using Discrete Event System Models", M.S. Thesis, Dept. of Electrical Engineering, Univ. of Notre Dame, Notre Dame, IN, May 1991.

[25] Stiver, J., Antsaklis, P., "A Novel Discrete Event System Approach to Modeling and Analysis of Hybrid Control Systems", Control Systems Technical Report #71, Dept. of Electrical Engineering, University of Notre Dame, Notre Dame, IN, June 1991.

[26] Zeigler, B., *Multifacetted Modelling and Discrete Event Simulation*, Acedemic Press, NY, 1984.

[27] Zhong, H.,Wonham, W. M., "On the Consistency of Hierarchical Supervision in Discrete-Event Systems", *IEEE Transactions on Automatic Control,* Vol. 35, No. 10, pp. 1125-1134, October 1990.