

# ANALYSIS AND DESIGN OF REAL TIME SCHEDULING IN CONTROL SYSTEMS

**Pau MARTÍ**

Automatic Control Dept., Universitat Politècnica de Catalunya. C/ Pau Gargallo 5, 08028 Barcelona, SPAIN.

Tel. +3493 4011679, Fax. +3493 4017045

E-mail: [pmarti@esaii.upc.es](mailto:pmarti@esaii.upc.es)

**Abstract.** *Real time computing is considered an important and enabling technology for many application areas such as control systems. Its main characteristic is to provide predictability; i.e. it should be possible to prove that requirements are met, in accordance to any assumptions made. Regarding timing requirements, real time scheduling is the main concern. An increasing number of complex systems relying, in part or completely, on control systems, need real time computing. It is known that control applications constitute real time systems due to their strict timing constraints. Nevertheless, it has to be pointed out that control systems have been mostly treated by control engineering and real time systems have been mostly treated by computer engineering. Therefore, there is a gap between the above disciplines due to their different theoretical/practical backgrounds when dealing with real time control systems. Further research is needed in this area in order to supply knowledge transfer between control and computer engineering. The aim of this project is to minimise this gap, providing analysis and design of real time scheduling in the control systems scenario. Real time scheduling has been widely treated, especially from a computer science point of view. As a result, many scheduling algorithms have been presented. However, the suitability of these algorithms regarding control systems timing behaviour and requirements has not been extensively studied and analysed.*

## 1. INTRODUCTION

Real time computing has been widely used in many areas during the last three decades, playing a key role in our society. There are many definitions of real time computing, but basically, the main idea is that in such a computing system, the correctness of the system depends not only on the logical results of the computations but also on the time at which the results are produced [16].

From this definition, it is important to stress that the main objective of real time computing is not fast computing, it is predictability [17]. Fast computing objective is to minimise the average response time of a given set of tasks. However, the objective of real time computing is to meet the individual timing requirement of each task. Therefore, in real time computing, functional and timing behaviour of the system should be as deterministic as necessary to satisfy system specifications.

In real time systems, scheduling theory addresses, by means of algorithms, the problem of meeting the specified timing requirements in order to have an understandable, predictable and maintainable system timing behaviour. Therefore, scheduling involves the allocation of resources and time in such a way that certain performance requirements are met [11].

Despite the fact that the application area of real time systems is endless, the common feature is that all are relying on control systems. It is well accepted that control systems are real time systems due to their strict timing constraints. However, control systems have not received much attention from real time practitioners [22]. Surprisingly, control theory and real-time scheduling theory have been relatively independent research areas.

Recently, some works have appeared in the literature which address important issues in real time distributed control systems. [9] discusses some problems that can be found in real time control, [24] revises fundamental aspects for implementing real time control applications in distributed control systems, and [12] and [13] study mono-rate and multirate control systems and scheduling issues.

## 2. CONTROL SYSTEMS TIMING ANALYSIS

A generic control system has basically three main sub systems: sensory system, control system and actuation system. The aim of these three sub systems working together is to perform some control actions by means of a control procedure, on a natural or artificial environment or plant (see Fig. 1).

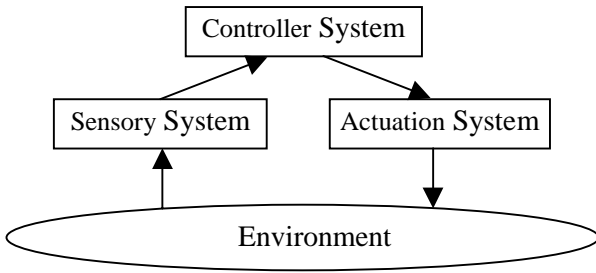


Figure 1. Generic control system

The general functionality of a control system can be described as follows: firstly, the sensory system collects data from the environment. Secondly, the control system, by means of a control algorithm, processes this data and derives the needed actuation. Finally, the actuation system performs the actuation to the environment.

The previous functional scheme can be split into more detailed activities that have different degrees of strictness according to system timing behaviour. On one hand, sampling data, control algorithm computation time and sampling actuation can have stringent timing constraints. On the other hand, monitoring or human-machine interaction can have more relaxed timing constraints.

It is important to stress that strictness in control systems usually means that actions perform at short time intervals and missing an action result is not important. However, in real time systems, strictness means that actions have to complete their computations before a time constraint. Missing an action result may imply severe consequences to the system.

Moreover, it has to be pointed out that control, from theory to practice, can follow many paradigms such as continuous or discrete control; direct, feedback or feed-forward control; mono-rate or multirate control; classic, adaptive or fuzzy control; centralised or distributed control. One or many of these paradigms are found in any control system, implying different timing behaviours. For example, a classic feedback sampled control system can have a bounded computation time of its control algorithm. However, an adaptive feedback sampled control system may not have a time bounded control algorithm.

Therefore, time is an important issue in control systems regarding timing behaviour and schedulability as well as performance and quality of service. Control theory assumes a highly deterministic timing of an implementation [2]. Consequently, little work has treated deficiencies in the computer system implementation of the control system with respect to time-variations and time-restrictions.

Nevertheless, interest in timing problems on control systems can be found in the literature. [25] investigated

the effects of time varying delays on control system stability and performance. It is shown that time varying delays can cause instability and deteriorate performance. [26] deals with time varying delays on control networked systems. [14] and [25] gives an interpretation of time varying delays as computer induced disturbances. Some stability proofs for time varying delays exist [27]. Models for sampled control systems timing behaviour are developed by [22]. [23] derives timing requirements and constraints for implementation of multirate control applications. [10] discusses modelling and analysis of real-time control systems subject to random time delays in the communication network.

Regarding control systems timing analysis, the aim of this project is to derive and specify timing behaviour of control systems in order to study the suitability of real time scheduling. Therefore, it is out of the scope of this project to study the effect of timing on control performance or control stability.

Lets take a closer look at the significant timing behaviour of a simple sampled computerised feedback control system in order to have a deeper understanding about which timing specifications should be derived from control systems (see Figure 2).

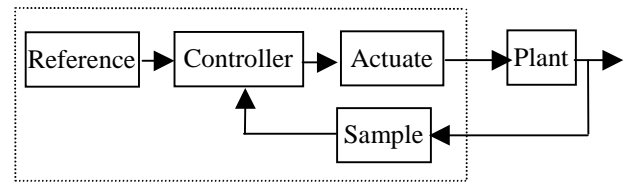


Figure 2. Simple computerised control system

It consists of four main nodes, the reference signal generator, the sampler node, the controller node and the actuator node. In the typical control scenario, the following functionality is expected: the sampler samples the plant with a fixed period  $T_{sample}$ . The time of the  $n^{th}$  sampling is given by

$$t_{sample}(n) = t_{sample}(n-1) + T_{sample} \quad (1)$$

When the sampler has collected the data it is forwarded to the controller, introducing a communication delay  $D_{sc}$ . The  $n^{th}$  controller start execution time is given by

$$t_{control}(n) = t_{sample}(n) + D_{sc} \quad (2)$$

The controller executes the control procedure in order to derive the actuation signal(s), introducing a compute delay  $T_c$ . The controller will forward the actuation signals to the actuator, introducing another communication delay  $D_{ca}$ . Finally the actuator perform the actuation at time given by

$$t_{actuate}(n) = t_{control}(n) + T_c + D_{ca} \quad (3)$$

The reference signal generator generates reference signals every  $T_{ref}$ , which usually is larger than  $T_{sample}$ .

In this short timing description, it is shown that the controller executes every  $T_{sample}$  and introduces a delay compute delay  $T_c$ . It is supposed that sampling and actuation perform without any computation time cost at a given time. It may be necessary to introduce an acceptable deviation (tolerance) from the given time of the sampling ( $tol_s$ ) and of the actuation ( $tol_a$ ) in order to be more realistic, modifying (1) and (3) as in (4) and (5)

$$t_{sample}(n) = t_{sample}(n-1) + T_{sample} \pm tol_s \quad (4)$$

$$t_{actuate}(n) = t_{control}(n) + T_c + D_{ca} \pm tol_a \quad (5)$$

Moreover, the controller timing analysis could be more precise as well as complicated if release time ( $T_{rel}$ ), start time ( $T_{start}$ ), jitter (varying or fixed difference between  $T_{rel}$  and  $T_{start}$ ,  $jit$ ) and completion time ( $T_{complet}$ ) of the controller activity were introduced, modifying (2) and obtaining (6).

$$t_{control}(n) = t_{sample}(n) + D_{sc} + jit \quad (6)$$

The analysed system has two inter-task communication procedures that introduce delays (fixed or varying) at the timing behaviour.

Finally, the whole timing analysis, from the sampling to the actuation of any instance of the simple control system under study, control delay, can be given by

$$ControlDelay(n) = t_{actuate}(n) - t_{sample}(n) \quad (7)$$

A full-specified illustration of the previous analysis can be seen in the Fig.3

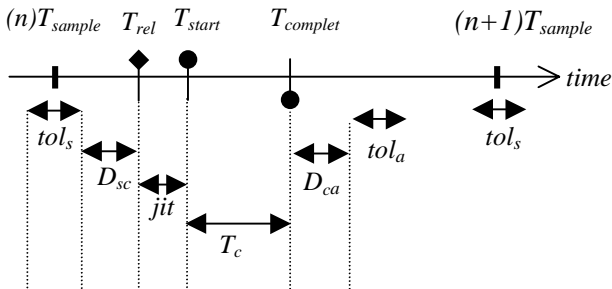


Figure 3. Timing diagram of a typical control loop

Similar detailed analysis can be found in [12] and [7].

Furthermore, this system timing behaviour sets precedence relations between the different activities (time triggered activity when trigger by period, event triggered actions when trigger by the completion of other activities) as well as resource allocation that will be used when scheduling the system.

One of the goals of the present project will be to provide timing analysis of control systems in order to gather all

timing requirements and specifications that any control system can have and it should meet.

### 3. REAL TIME SCHEDULING BASIC CONCEPTS

Real time scheduling is one of the main research area within real time systems according to the number of research papers that can be found. In this section, basics scheduling concepts will be described before going through the real time scheduling revision.

Real time systems are characterised by a set of executing concurrent tasks that have deadlines, which represent the time at which the task should complete its execution without causing damage to the system.

Depending on the consequences of a missed deadline, real time tasks, i.e. a computation that is executed in a sequential way, can be split into:

- Hard real time tasks: the completion of the task must be within a bounded interval, otherwise catastrophic consequences are said to occur
- Soft real time tasks: a task is said to be soft if missing its deadline decreases the performance of the system but not has catastrophic consequences.

From the previous classification among tasks, two types of real time systems can be differentiated. Hard real time systems are those where it is absolutely imperative that responses occur within the specified deadline while soft real time systems are those where response times are important but the system will still function if deadlines are occasionally missed. In fact, it has to be pointed out that there is a current debate on the previous definitions. Sometimes, if the consequences of missing a deadline are catastrophic, the target system is called critical (or safety critical) real time system instead of hard real time system.

Another timing characteristic that can be specified on a real time task concerns the regularity of its activation. Depending on it, a task is defined as a periodic or aperiodic task. A task that is invoked at regular intervals is a periodic task. On the other hand, a task that is not invoked at regular intervals is an aperiodic task. Aperiodic tasks characterised by a minimum inter-arrival time are called sporadic.

In general, a real time task  $\tau_i$  can be characterised by several of the parameters described in the following table:

Parameter		Description
Criticalness	hard/ soft	Parameter related to the consequences of missing the deadline

Worst case computation time	$C_i$	An estimation of the maximum time required by the processor for executing the task without interruption
Period	$T_i$	Time within consecutive tasks releases
Deadline	$D_i$	Time before which the task should be completed
Priority	$P_i$	Relative importance (as far as scheduling is concern) of the task with respect to the other tasks in the system
Start time	$s_i$	Time at which a task really starts its execution
Finishing time	$f_i$	Time at which a tasks finishes its execution
Worst case blocking time	$B_i$	Maximum time during which the task is blocked by lower priority tasks when accessing a resource
Interference	$I_i$	Maximum time that the task will have to wait due to preemption from higher priority tasks
Processor utilisation	$U_i$	Processor utilisation of the task (equal to $C_i/T_i$ )
Worst case response time	$R_i$	Longest time duration between the invocation of a task and the time that the task completes its bounded amount of computation time
Release time	$r_i$	Time at which a task becomes ready for its execution
Jitter	$J_i$	Time that a task spends from its release until its start time
Lateness	$L_i$	Delay of a task completion with respect to its deadline
Tardiness (exceeding time)	$E_i$	Time during which a task stays active after its deadline
Laxity (slack time)	$X_i$	Maximum time a task can be delayed on its activation to complete within its deadline

Some of the parameters of a real time task are illustrated in Figure 4.

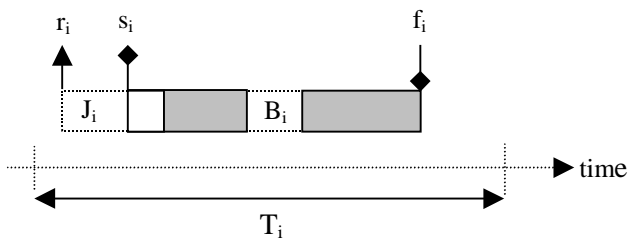


Figure 4. Parameters for a real time task  $\tau_i$

To characterise all tasks in any distributed real time system, the above parameters should be combined with other models in order to specify resource constraints, concurrency relations, precedent relations, communication patterns and placement constraints that any set of real time tasks may have.

#### 4. REAL TIME SCHEDULING

In this section a general revision of scheduling algorithms will be discussed. For further reading, see [15], [1], [3], [4], [20] and [21].

In the scheduling problem it is need to specify three sets: a set of  $n$  tasks  $T = \{\tau_1, \tau_2, \dots, \tau_n\}$ , a set of  $m$  processors  $P = \{P_1, P_2, \dots, P_m\}$  (that may include communication nodes and networks), and a set of  $s$  type of resources  $R = \{R_1, R_2, \dots, R_s\}$ . Precedence relations among tasks can be specified in a directed acyclic graph and timing constraints can be associated with each task.

In this scenario, scheduling means to allocate tasks to processors and resources in order to complete all tasks under the imposed constraints. This problem, in general, belongs to the classes of NP-complete problems [6][5][8]. However, for a limited number of cases polynomial time solutions exist.

In [21] the scheduling approach is broken down into three main activities: off-line configuration, run-time dispatching, and a priori analysis. The off-line configuration generates the static information that will be used for the run-time dispatching. The run-time dispatching deals with the switching between computations for different events at run-time. The priori analysis examine the system and tell us if all the timing requirements will we met according to the off-line configuration information and the behaviour of the run-time dispatching algorithm. Scheduling approaches differ on how much effort is done in every activity.

The priori analysis is used to determine by means of tests the feasibility of scheduling approaches, i.e. whether the temporal constraints of all processes will be met at run time. The analysis is said to be sufficient and necessary when a task set will meet all it deadlines if, and only if, it passes the test. If a task set passes the test then it will meet all its deadlines at run-time, but if it fails does not imply that it will not meet all its deadlines. Then the analysis is said to be sufficient and not necessary [1].

Among the great variety of real time scheduling approaches [4], the following main classes can be identified (obviously, any scheduling algorithms belong to more than one of identified classes):

- Off-line vs. on-line scheduling: off-line scheduling is done when scheduling tasks can be performed before starting the execution, storing the schedule in a table that will be used for the dispatcher at run-time. On-line scheduling is done when scheduling tasks is performed during the execution.
- Preemptive vs. non-preemptive scheduling: with preemptive scheduling, the running task can be interrupted at any time by another task, according

to some predefined scheduling policy. In non-preemptive scheduling, a task, once started, is executed by a processor until its completion unless it gets blocked over a resource.

- Static vs. dynamic scheduling: static scheduling algorithms are those in which scheduling decisions are based on fixed parameters, assigned to tasks before their activation. Dynamic scheduling, decisions are based on dynamic parameters that may change during system execution.
- Optimal vs. heuristic scheduling: an algorithm is said to be optimal if it may fail to provide a feasible schedule only if no other algorithms of the same class can meet it. An algorithm is said to be heuristic, if it tends toward but does not guarantee to find the optimal schedule.

The different scheduling approaches belonging to the previous classes, should be comprehensive enough to handle with [18]:

- Preemptable and nonpreemptable tasks
- Periodic and nonperiodic tasks
- Tasks with multiple level of importance
- Group of tasks with a single deadline
- End-to-end timing constraints
- Precedence constraints
- Communications requirements
- Resource requirements
- Placement constraints
- Fault tolerant needs
- Tight and loose deadlines
- Normal and overload conditions

Therefore, the solutions should be integrated enough to handle the interfaces between:

- CPU scheduling and resource allocation
- I/O scheduling and CPU scheduling
- CPU scheduling and real time communications scheduling
- Local and distributed scheduling
- Scheduling of safety critical tasks and scheduling of less critical tasks

Real time scheduling has provided general-purpose algorithms that use general task models to express timing requirements. Control systems have specific timing requirements that should be possible to correctly express with these task models. Hence, further research is needed in order to study the suitability of real-time general-purpose algorithms when expressing control systems timing requirements.

Moreover, control problems arise when control systems are implemented in distributed real time systems [9]. In these cases, timing analysis has to cover a broader set of

aspects, such as allocation of tasks to processors, task synchronisation (precedence and resource constraints) and task communication over field-buses or networks.

Finally, it has been stressed [19] that further research is also needed on open real-time systems and globally distributed real time systems, as well as, on embedded systems. Economic and safety issues will be considered. Therefore, complex control systems, from its embedded implementations to highly distributed implementations will have to look close on real time research improvements.

## 5. FUTURE RESEARCH

In this section, some key research issues are identified when trying to bridge the gap between real time systems and control systems. As far as I know, further research is needed in the following topics:

- The jitter problem: most real time scheduling algorithms are concerned with meeting temporal constraints, such as deadlines. An earlier than required completion is acceptable. Control applications, however are concerned with the completion time itself, e.g., by requiring completion within a time interval or bounding the differences in starting or completion times.
- Real time period vs. control period: related to the jitter problem, there is difference between period interpretation in real time systems and control systems. In the former, the period is the time between consecutive task activations, during which some operation must be performed. Moreover, from the first activation, the following activations will be multiples of the first one, at constant periods. In control systems, the period is the time between consecutive performed operations. Therefore, adding the jitter problem to the control systems period interpretation, it can be said that the period in control systems is not as constant as in real time systems because it can accumulate, from one period to the following, little delays.
- Interaction between tasks with different periods: in the real time scheduling scenario, usually it is supposed that when two or more task interact, they have the same period. However, it is known that control systems have tasks that perform at different periods and have tightly interactions.
- Distributed precedence constraints: to schedule tasks with precedence constraints in a distributed real time system is not already solved. Moreover, precedence relations have been modelled using directed acyclic graphs. However, control systems have per nature cyclic precedence relationships between tasks, e.g., in a feedback control loop.

- Communication / precedence relations: real time scheduling has considered that communication between real time tasks implies precedence relations. However, control systems may have communication patterns that do not necessarily imply precedence relations. Therefore, communication and precedence analysis should be carried out separately.

The above-identified issues show some misunderstanding or differences between real time systems and control systems. Therefore, real time scheduling and the corresponding schedulability analysis should be extended in order to incorporate the previous issues.

## 6. CONCLUSIONS

In this state of the art summary, it has been pointed out that there is a gap between real time systems and control systems theory and practise. Although few works in the literature deal with this problem, there is still a need for further research.

A way to overcome this gap can be modelling control systems temporal behaviour and deriving their timing requirements in order to prove the suitability of real time scheduling current trends.

## REFERENCES

- [1] Audsley, N.C., Burns, A., Davis, R.I., Tindell, K.W. and Wellings, J. (1995) "Fixed Priority Pre-emptive Scheduling: An historical Perspective", *Real-Time Systems, the International Journal of Time-Critical Computing Systems*, Vol. 8, Number 2/3, 1995.
- [2] Åström, K. J and Wittenmark, B. (1997) *Computer-Controlled Systems*. Third edition. Prentice Hall
- [3] Burns, A. And Wellings, A. J. (1990). *Real-Time Systems and their Programming Languages*. Second Edition. Addison- Wesley.
- [4] Buttazzo, G.C. (1997) *Hard Real-Time Computer Systems. Predictable Scheduling Algorithms and Applications*. Kluwer Academic Publishers. ISBN 0-7923-9994-3
- [5] El-Rewini, H., Ali, H.H. and Lewis T. (1995) "Task-Scheduling in Multiprocessing Systems", *IEEE Computer* vol. 28 , no. 12
- [6] Garey, M.R. and Johnson, D. S. (1979). *Computers and Intractability*. Freeman
- [7] Lönn, H. and Axelsson, J. (1999) "A Comparison of Fixed-Priority and Static Cyclic Scheduling for Distributed Automotive Control Applications", *Proc. of the 11<sup>th</sup> Euromicro Workshop on Real-Time Systems*.
- [8] Mok, A.K. and Dertouzos M.L. (1978) "Multiprocessor Scheduling in a Hard Real Time Environment", *Proc. 7<sup>th</sup> Texas Conference in Computer Systems*. IEEE 5-1:12
- [9] Nilsson, J. (1998) "Some Topics in Real-Time Control", ACC, June 24-26. Philadelphia.
- [10] Nilsson, J.,Bernhardsson, B. and Wittenmark, B. (1998) "Stochastic Analysis and Control of Real-Time Systems with Random Time Delays", *Automatica*, 34:1, p.57-64.
- [11] Ramamritham, K. and Stankovic, J.A. (1994) "Scheduling Algorithms and Operating Systems Support for Real-Time Systems", *Proceedings of the IEEE*, Vol. 82, NO.1, January 1994.
- [12] Redell O. (1998) "Global Scheduling in Distributed Real Time Computer Systems. An Automatic Control Perspective", *Technical Report TRITA-MMK 1998:6*, ISSN 1400-1179, ISRN KTH/MMK-98/6-SE. Department of Machine design, The Royal Institute of Technology, S-100 44 Stockholm. Sweden
- [13] Sandstrom, K. (1999) "Modelling and Scheduling of Control Systems", *Licenciate Tesis TRITA-MMK 1999:5*, ISSN 1400-1179, ISRN KTH/MMK/R--99/5-SE. Department of Machine design, The Royal Institute of Technology, S-100 44 Stockholm. Sweden
- [14] Shin K., and Cui X. (1996) "Computing Time Delay and its Effects on Real-Time Control Systems", *IEEE Trans. on Control Systems Technology*. Vol. 3, N. 2, p.218-224
- [15] Shin, K.G and Ramanathan, P. (1994) "Real-Time Computing: A New Discipline of Computer Science and Engineering", *Proceedings of the IEEE*, Vol. 82, NO.1, January 1994.
- [16] Stankovic, J.A. (1988) "Misconceptions About Real-Time Computing: A Serious Problem for Next Generation Systems", *IEEE Computer*. 21(10):10-19.
- [17] Stankovic, J.A., and Ramamritham, K. (1990) "Editorial: What is Predictability for Real-Time Systems?", *Real-Time Systems* 2(4):247-264.
- [18] Stankovic, J.A. and Ramamritham, K. (1993) *Advances in Real-Time Systems*. IEEE Computer Society Press. ISBN 0-8186-3792-7
- [19] Stankovic, J.A. (1998) "Real Time and Embedded Systems", *ACM*

- [20] Stankovic, J.A., Spuri, M., Ramamritham, K and Buttazzo, G. (1998) Deadline Secheduling for Real-Time Systems. EDF and Related Algorithms. Kluwer Academic Publishers. ISBN 0-7923-8269-2
- [21] Tindell K. and Hansson H. (1997) "Real Time Systems by Fixed Priority Scheduling", Technical report, Department of computer systems, Uppsala University.
- [22] Törngren M. (1995) "Modelling and Design of Distributed Real-Time Control Applications", Ph-D thesis, Dept. of Machine Design, The Royal Institute of Technology, Stockholm, Swedwn.
- [23] Törngren, M., Eriksson, C., and Sandström, K. (1997) "Deriving Timing Requirements and Constraints for Implementation of Multirate Control Applications", Research Report TRITA-MMK 1997: 1, ISSN 1400-1179, ISRN KTH/MMK/R-97/1-SE. Department of Machine design, The Royal Institute of Technology, S-100 44 Stockholm. Sweden
- [24] Törngren, M. (1998) "Fundamentals of Implementing Real-time Control Applications in Distributed Computer Systems", J. of Real-Time Systems, 14, 219-260, Kluwer Academic Publishers.
- [25] Wittenmark B., Nilsson J., Törngren M. (1995) "Timing Problems in Real-Time Control Systems: Problem Formulation", Proc. Of the American Control Conference, Seattle, Washington
- [26] Wittenmark B., Bastian, B. and Nilsson J. (1998) "Analysis of Time Delays in Sinchronous and Asynchronous Control Loops", Proc. Of the 37<sup>th</sup> Conf. On Decision and Control, Tampa, FL. USA
- [27] Voulgaris, P. (1994) "Control of Asynchronous Sampled Data Systems", IEEE Trans. on Automatic Control 39(7)