EMBEDDED CONTROL SYSTEMS: SOME ISSUES AND SOLUTIONS[1]

P. Albertos*, A. Crespo§, M. Vallés*, I. Ripoll§

* *Department of Systems Engineering and Control (DISA)*
§ *Department of Computer Engineering (DISCA)*
Universidad Politécnica de Valencia
P.O.Box. 22012, E-46071, Valencia, Spain.
pedro@isa.upv.es, alfons@disca.upv.es, mvalles@isa.upv.es, iripoll@disca.upv.es

Abstract. Embedded control systems are becoming ubiquitous in control applications. They combine the properties of computer embedded system with newly designed complex controllers where flexible, safe and reconfigurable operations are required. The aim of this paper is to grasp the main features of these systems, analyse the main problems and present some solutions already developed by the authors. A new concept on the control kernel of an application is also introduced and some conclusions are drafted.

Keywords. Real-time control, embedded systems, optional tasks, alternative controllers, integration of control design and implementation.

1. INTRODUCTION

The strong increasing presence of embedded systems (ES) in products and services creates huge opportunities for the future in different areas such as industrial control systems, avionics, health care, environment, security, mechanics, … (Chinook, 2004). Thus, there is a growing scientific interest on conceptual and practical tools for their development (Dreamtech, 2002). In particular, their use in control applications is becoming very popular.

RT control applications on ES require the best use of the available computation resources. Among the main advantages they offer are the reduced price and size, broadening the scope of possible applications: mass-production systems due to the cost reduction and specific accurate applications for their reduced size and high performances. But the most important problem is the limited computational capabilities they can use because it is well known that, in general, short sampling periods and non-delayed control actions allow for better control performances.

So, one of the most important issues related with ES in control applications is related with the reliable and optimal use of their computational resources and what the resource shortage involves in the design and implementation of the control algorithms. For these applications, it is not always possible to implement the control by using general purpose operating systems because of the particular requirements in terms of delays and jitter limitation. Thus, the control computations should be implemented as real-time tasks being executed under a specific real-time scheduling policy.

Many works related to embedded control systems (ECS) simplify their treatment in two different senses, depending on the framework. In the computer systems arena it is common to consider them just like a new, and fundamental, application field of

[1] This project has been partially granted by the CICYT project number DPI2002-04432

computer ES and they focus on the general hardware and software issues of ES in general. See, for instance (Henzinger, 2003) and (Ledin, 2004). On the other hand, authors in the control side may consider than the hybrid character of these systems is the kernel of the problems and immediately concentrate the discussion on the study of hybrid control systems, (Tiwary, 2003).

There is a strong group of researchers involved in the interactive design of the control algorithms and their RT implementation. In the IEEE Control system Magazine special issue (Sanz and Arzen, 2003) an overview of some of the related problems is presented and it is a good reference for newcomers in the field. The authors also belong to this group and, as latter described, have made some contributions to the topic.

Thus, it seems interesting to extract the main characteristics of ES in general, those specific of ECS and the problems they pose from the points of view of computers, communications and control. Moreover, the interplay between all the components to get the best performances of the whole system should be considered.

This paper is organised as follows. In the next section, the main concepts behind the idea of ES as well as the specific requirements for their application in ECS are summarised. Some options in the design of ECS architectures are discussed in Section 3. As already mentioned, some of the specific problems related to the design and implementation of ECS have been already treated by the authors and these results are summarised in Section 4. The new idea of control nucleus, as a basis for essential control design is introduced in Section 5 and some conclusions are drafted in the last section.

2. EMBEDDED CONTROL SYSTEMS

Several roadmaps on ES state the trends and more relevant topics (ERS 2002). In (ARTIST 2003) the status and future of a set of selected areas are described. The basic characteristics that can be found in ES can be summarized as: compact and reduced size, autonomy, reconfigurability, safety, fault-tolerant and capable to work under missing data operation.

The above characteristics highly condition their use in control applications and a number of problems arise. The general lay-out of an ECS is depicted in fig. 1, where different sensors, actuators and interaction with the environment are shown. Its main characteristics are the autonomy and the limitation in the resources. One CPU, with its own power supply, must control a number of variables in an uncertain environment. The problems to be considered are related to their implementation, the computational load and resources sharing and the control performance degrading.

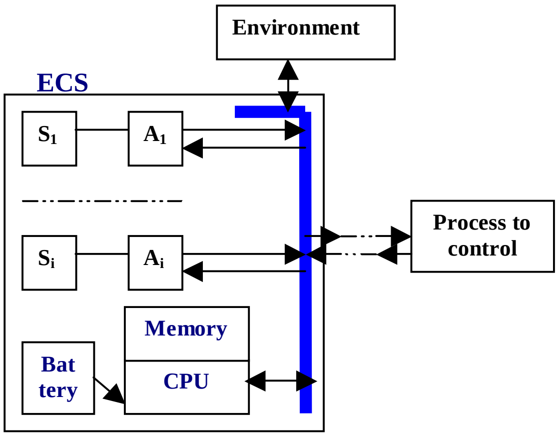

Figure 1. ECS structure

I.- From the implementation point of view:

i) The same resource must be shared between different tasks. As a result of this competition for the CPU use, the timing of the tasks is not fully determined and the time delays and jitter should be taken into account.

ii) Alternative control algorithms should be ready to get the control of the process. Working in a changeable environment, the control goals and options may change and the control algorithms should be adequate to new scenarios.

iii) Working conditions, such as priority, allocated time and memory or signals availability may change. Thus, complexity, structure and basic properties of the control system will change.

iv) Variable delays should be considered, (Cervin, 2003). The synchronicity of signals cannot be ensured anymore.

v) Validation and certification. Any embedded control system should be proved to be reliable and safety operation should be ensured.

II.- From the computational point of view:

i) Economic algorithms should be designed. To save as much as possible computation time.

ii) Easy update of information should be provided. To use the shortest time in controller changes.

iii) Hybrid systems should be considered. Logic, discrete time and continuous time information should be merged.

iv) CPU use measurement and optimisation. Being a scarce resource, the current use of the CPU should be measured and corrective actions should be applied to optimise its use.

v) Optional tasks must be considered. Control algorithms should be split into mandatory and optional parts, the last ones being only run if time is available.

vi) On-line scheduling. Based on the current control goals as well as the availability of resources the required tasks should be scheduled in the most performing way.

vii) Memory saving. Being also a limited resource, computation tasks should rely on few data, also providing a reduced number of intermediate results to reduce memory storage capacity and accessing time.

viii) Economic hardware redundancy. To increase the reliability, hardware redundancy may be implemented if its cost, size and involved complexity is affordable.

ix) Fault detection and isolation As well as reconfigurability, to allow an autonomous behaviour.

III.- From the control algorithm point of view, specific design methodologies should be used:

i) Reduced order models. The complexity of the controller is directly related to the complexity of the plant. More properly, it is related to the complexity of the model used in the design. Thus model reduction techniques should be considered to simplify either the plant model or the designed controller.

ii) Decision and supervisory control. Changes in the environment and availability of resources should be monitored and changes in the operation mode should be decided by an upper level of control.

iii) Hybrid control systems. There is also a lot of research on this topic. It is interesting to point out that some authors dealing with ECS only focus on the hybrid character of the controller as there is the need of combining discrete and continuous time dynamics. But this is just one, although fundamental, property of the ECS.

iv) Multimode control. A control problem cannot be approached in the same way if the environmental conditions change. Thus, the control strategy must consider changes in the controller and caution should be taken to guarantee the correct operation (stability is the minimum) of the controlled system.

v) Sampling rate changes. It is well known that, in general, the quality of the digital control decreases if the sampling period increases. Thus, if resources are available, the control tasks period should be reduced and this implies changes in the controller parameters as well as in the stored information.

vi) Non-conventional sampling and updating patterns. The signals sampling and the control action delivering time instants should be appropriated for the required controlled plant dynamics and the available resources. This implies the option of different sampling rates (multirate control) and to consider the lost of synchronicity.

vii) Missing data control. Data availability is not at all guaranteed in any operating condition. Thus, control algorithms should cope with missing data situations.

viii) Event-triggered control. Most control algorithms are time driven. The sampling-updating pattern is regular and synchronous, with a constant time interval between actions (periodic actions). But dealing with harsh and uncertain environments, some activities are triggered by external events. And these events may happen randomly and at any time.

ix) Degraded and back-up (safe) control strategies. Different control requirements should be considered, according to the operating conditions. In emergency cases, a degraded behaviour should be accepted, ensuring the safety of the operation.

x) Fault-tolerant control. There is a lot of research and literature on this topic. It is crucial for ECS as there is no option to externally reconfigure the control structure. Thus, supervisory control involving fault detection and isolation as well as a decision system to select the most suitable controller must be considered. Alternatively, time invariant fault tolerant controllers can be considered if a degrading of performances is allowed in faulty conditions.

xi) Battery monitoring and control. Any autonomous system should adapt its activities to the available power. The monitoring and control of the on-board batteries will result in forced changes in goals and control structures.

And altogether, the integration of control design and computer control implementation is crucial to achieve the best control performances.

3. ECS ARCHITECTURES

In industry, most of the ES are based on microcontrollers and PLC's (Programmable Logic Controllers). Microcontrollers provide most of the basic features to implement basic control systems (processor, input/output, converters) but normally they provide low computation level and application. They often have no operating system, or a specialized embedded kernel (often a real-time operating system), or the programmer is assigned to port one of these to the new system. Small kernels with specific services are used for these kind of systems to develop several tasks. Its capacity of adaptation and reconfiguration are very limited and its use is constrained to well very known and simple control systems (printers, modems …).

The need of a higher computation power can be improved by the use of DSPs. These microcomputers whose hardware, software, and instruction sets are optimized for high-speed numeric processing applications play an essential role for processing digital data in real time.

Most of the industry control applications use PLCs which provide a wide range of input/output and communication protocols. However, the programming languages have not evolved as in other software technology and there are many different ladder diagram languages to develop applications. In general, these languages provide poorly programming structures and the kernels do not provide most of the features offered by the real-time operating systems. Some initiatives (PLCOpen), focused around the standard IEC 61131-3, are trying to adopt a norm in the design and operation of the programming interface.

However, the new embedded systems are characterized by growing software complexity and functionalities where embedded software dominates the development cost and schedule. The old way of developing software for each embedded project from scratch is giving way to the need to reuse software, and build on existing software wherever possible.

Also the embedded (real-time) operating systems are providing more and more new services to fulfil the new needs. Examples of these embedded operating systems are: embedded Linux (several distributions), Windows CE, VxWorks, QNX, OS-9, etc. The diversity of operating environments and platforms poses a real challenge in deploying software across multiple platforms and configurations. The use of embedded operating systems providing POSIX interface facilitates the portability and reusability of the applications.

The embedded operating system usually should have the following characteristics (Ripoll 2002):

- *Configurable and scalable*: the design of operating systems based on components permits the selection of the appropriated ones to build the specific kernel for the embeddable application.
- *Innovative techniques* in scheduling to select the most efficient for the application combining different policies at several levels of scheduling.
- *Resource management* allowing the use of Quality of Service techniques improving the efficiency and flexibility of a real-time system. An efficient approach to real-time resource management consists in applying feedback control theory to real-time scheduling. So, a scheduling strategy naturally adapts to the application needs and the resources allocated to an application are automatically updated to its needs.
- *Fault-tolerance mechanisms* which are able to make the results available on a timely basis even if this fact could lead to a functioning in "degraded mode". This is a pre-required quality for real-time embedded systems.
- *Small footprint* to embed the system in different configuration.
- *Multi-platform*. The embedded operating system has to be portable or recompilable for different processor architectures.
- *Power aware techniques*. Development of systems supporting dynamic power management strategies based on dynamic voltage/frequency scaling for tuning the power-performance trade-off to the needs of the application.
- *Efficient memory management* including special forms of memory management that limit the possibility of memory fragmentation, and assure a minimal upper bound on memory allocation and de-allocation times.

4. SOME PREVIOUS RESULTS

There is a lot of literature providing solutions to some of the issues above. In particular, the authors have presented alternatives to some of the previously discussed problems. Due to the lack of space, the readers are addressed to these papers and the references herein.

- Controller design under non-conventional sampling patterns. Improvements in the control performances are achieved by taking into account the sampling pattern when designing the controllers (Albertos, 1999, 2001).
- Controller parameters and data updating under sampling rate changes. As below described, changes in the CPU load demand changes in the control tasks periodicity. (Albertos, 2003a).
- On-line re-scheduling. To take into account changes in the control relevance of some actions (Albertos, 2000).
- Missing data control. Using reliable output estimators (virtual sensors) to cope with scarce or missing data, due to communication congestion or data acquisition systems failures in uncertain environments (Sanchis, 2002).
- Scheduling policies to enhance quality of service (Hassan, 2002), (Mazario, 2004).
- Real-time task model to minimise the output jitter in control tasks (Balbastre, 2000, 2004).
- Algorithms for dynamic memory allocation in real-time systems (Masmano, 2004).
- Development of open components for real-time embedded operating systems (OCERA, 2002).

5. CONTROL KERNEL

As previously discussed, changes in the operating conditions require changes in the controller. Under any working condition, the system must keep some basic properties. If this is not the case, the emergency routines should take care of the system by moving it to a safe, even shut-down, situation.

These basic properties should be captured by a kernel representation of the system in order to apply an essential control. This control should be able to ensure the system stability under a shortage of resources. Thus, in this case the system will fall under this essential control, until more resources (options) become available.

General speaking, the kernel representation implies a reduced order model of the system as well as a mechanism to transfer from a normal, extended model, to the kernel and vice versa. Thus, a simple algorithm to transfer between models, also recovering the involved data, should be provided.

The kernel concept must consider to cope with essential goals connected to safety in the operation as well as different resources' shortage, such as lost of data or time limitation. Let us discuss the model reduction issue under time constraints.

5.1 Transfer between models

Assume a standard (Albertos, 2004) state space representation of a discrete time plant such as:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_{k+1} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_k + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u_k \qquad (1)$$
$$y_k = [C_1 \quad C_2][x_1 \quad x_2]_k^T$$

where $x_{1,k}$ represents the "fast" components of the state vector and $x_{2,k}$ the "slow" ones. Under some operating conditions constraints, a partial model is enough to compute the appropriate control action:

a. The time allocated for control computation is reduced and a safe operation (with degraded performances) is foreseen. Thus, the sampling period may be enlarged and only the slow part of the plant model could be considered.
b. Under an emergency condition, a quick reaction of the control is expected. The sampling period is reduced and the slow behaviour of the plant may be considered as unchanged.

In both cases a reduced model of the plant is desirable and a change in the model parameters should be implemented. A similar reasoning will be applied if dealing with the controller model. Different techniques can be applied to get a reduced order model. See, for instance (Albertos, 2004, p. 86).

Deleting the slow dynamics implies to assume $x_{2,k+1} = x_{2,k}$ in (1). That is:

$$x_{1,k+1} = \bar{A}_1 x_{1,k} + \bar{B}_1 u_k \,;\, y_k = \bar{C}_1 x_{1,k} + \bar{D} u_k$$

where

$$\bar{A}_1 = A_{11} + A_{12}(I - A_{22})^{-1} A_{21}$$
$$\bar{B}_1 = A_{12}(I - A_{22})^{-1} B_2 + B_1$$
$$\bar{C}_1 = C_1 + C_2(I - A_{22})^{-1} A_{21}$$
$$\bar{D} = C_2(I - A_{22})^{-1} B_2$$

In this case, to go back to the full model (1) at time *k*, the state vector should be updated to:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_k^+ = \begin{bmatrix} I \\ (I - A_{22})^{-1} A_{21} \end{bmatrix} x_{1,k} + \begin{bmatrix} O \\ (I - A_{22})^{-1} B_2 \end{bmatrix} u_k$$

On the other hand, looking for a safe slow model, the assumption of an "immediate" change in the fast modes is equivalent to assume that $x_{1,k}$ is an input to the reduced model. That is:

$$x_{2,k+1} = A_{22} x_{2,k} + A_{21} x_{1,k} + B_2 u_k$$
$$y_k = C_2 x_{2,k} + C_1 x_{1,k}$$

In this case, to go back to the full model and initialise the full state vector, the state variables should be measured or estimated through the output vector.

5.2 Changes in the sampling period

The DT model (1) of a continuous time plant corresponds to a nominal sampling period, *T*. If, for any reason, the control tasks are re-scheduled, their periodicity will be changed. It is well known (Albertos, 2000) that, for control purposes, faster sampling rates allow for better control performances than slower ones. So, according to the CPU use, the sampling rate of the control algorithms (and, in the same way that of the plant models used to compute the controller) should be adjusted.

Changes in the discretization rate of a controller requires changes in both, the controller parameters and the set of data handled by the control algorithm, in order to get bumpless commutation and a reduced degrading of control performances. Some options, as previously mentioned, are described in (Albertos, 2003b).

5.3 Scheduling support

Both previous approaches require scheduling support to allow mode changes in the system. A mode change is initiated whenever the systems detects a change in the environment or in the internal state that must drive it from one operating mode to another allowing the use of reduced models and the transfer between models. Several protocols for mode changes can be found in the literature. A survey on mode change protocols can be found in (Real, 2004). The main requirements to be achieved by the protocols are: schedulability, periodicity, promptness and consistency.

6. CONCLUSIONS

In this paper we have reviewed the main issues related to ES with special focus on ECS. The ECS are analysed from different point of views: implementation, computation and control algorithm design. It is complemented with the run-time support analysis which discusses the requirements of the real-time embedded operating system to fulfil the needs of the new generation of ES. Finally, the new idea of control kernel, as a basis for essential control design has been introduced.

The ideas here summarised are the matter of further research from both the theoretical and practical viewpoints.

REFERENCES

Albertos, P. and Crespo, (1999). "A. Real-Time Control of Non-Uniformly Sampled Data Systems". *Control Engineering Practice.*

Albertos, P., Crespo A., Ripoll, I., Vallés M., and Balbastre P. (2000) "RT control scheduling to reduce control performance degrading" *IEEE Conference on Decision and Control 2000..*

Albertos, P. and Crespo, A. (2001). "Integrated Design and Implementation of Digital Controllers. EUROCAST 2001. *Lecture Notes on Computers Science*. pp 385-392 Springer.

Albertos, P. , Vallés, M.,and Valera, A. (2003a) "Controller Updating under Operational Logic Changes". *European Control Conf.* (UK)

Albertos, P. , Vallés, M. and Valera, A. (2003b) "Controller Transfer Under Sampling Rate Dynamic Changes" IFAC Workshop on *Modelling and Analysis of Logic Controlled Dynamic Systems*. Irkutsk (Russia).

Albertos, P. and Sala, A. (2004) *Multivariable Control Systems: an engineering approach*. Springer-Verlag.

ARTIST. Advanced Real-Time Systems. Selected topics in Embedded Systems Design. European Project IST-2001-34820. Y2 Reports. (http://www.artist-embedded.org/ Roadmaps/ARTIST_Roadmaps_Y2.pdf)

Balbastre, P., I. Ripoll, A. Crespo. (2000). "Control task delay reduction under static and dynamic scheduling policies". *Proc. of the 7th Intern. Conference on Real-Time Computing Systems and Applications*.

Balbastre P., Ripoll I., Vidal J., Crespo A. (2004). "A Task Model to Reduce Control Delays". *J. of Real-Time Systems*. Vol. 27, Issue 3, pp. 215-236.

Cervin, A. Henriksson, D., Lincoln, B., Eker, J. and Arzen, K-E. (2003) "How Does Control Timing Affect Performance?" *IEEE Control System Magazine*.

Chinook.webpage: http://www.cs.washington.edu/research/chinook/links.html

Crespo, A., I. Ripoll, P. Albertos (1999) "Reducing delays in RT control: The control action interval". *Proceedings of the 14th IFAC World Congress*, pp. 257-262,

Dreamtech Software Team, (2002). *Programming for Embedded Systems: Cracking the CodeTM.* J. Wiley.

ESR: Embedded Systems Roadmap 2002. PROGRESS/STW: public version 1.0. Technology Foundation of the Netherlands (STW),(http://www.artist-embedded.org/ Intranet/Roadmaps/STWroadmap.pdf).

Hassan H., Simó J., Crespo A. (2002). "Enhancing the Flexibility and the Quality of Service of Autonomous Mobile Robotic Applications". *ECRTS 2002*: 213-219

Henzinger, T, Horowitz, B. and Kirsch, C. (2003) "Embedded Control Systems Development with Giotto" in *Software-Enabled Control: Information Technology for Dynamical Systems* (T. Samad, G. Balas, eds.), IEEE Press and Wiley-Interscience, pp. 123-146.

Ledin, J. (2004). *Embedded Control Systems in C/C++***.** CMP Books.

Marzario L., Lipari G., Balbastre P., Crespo A. (2004). "IRIS: A New Reclaiming Algorithm for Server-Based Real-Time Systems." IEEE *Real-Time and Embedded Technology and Applications Symposium*. pp. 211-218

Masmano M., Ripoll I., Crespo A. (2004). "TLSF: A New Dynamic Memory Allocator for Real-Time Systems". *ECRTS 2004*. Catania (Italy).

OCERA: Open Components for Real-Time Embedded Applications . IST 35102 European Project. European Commission. (http://www.ocera.org). 2002-05.

Real J., Crespo A. (2004). "Mode Change protocols for Real-Time Systems: A Survey and a New Proposal". *Journal of Real-Time Systems*. Vol 26, Issue 2, March pp. 161-197

Ripoll, I., Pisa, P., Abeni L., Lanusse A, Saez, S. and Crespo, A. "RTOS State of the Art Analysis". OCERA white paper: www.ocera.org

Sanchis, R. and P. Albertos (2002). "Recursive identification under scarce measurements. Convergence analysis". *Automatica* 38, 535–544.

Sanz, R. and Årzén K-E. (2003)."Trends in Software and Control".*IEEE Control Systems Magazine* June 2003. pp 12-15

Tiwary, A., Shankar, N., and Rushby, J. (2003). "Invisible Formal Methods for Embedded Control Systems". *Proc. of the IEEE*. Vol 91. N.1, pp 29-39.