NON-PREEMPTIVE SCHEDULING FOR CRITICAL AND HARD REAL-TIME APPLICATIONS ON EMBEDDED PLATFORMS

Mihai V. Micea

Department of Computer Science and Engineering (DCSE),
"Politehnica" University of Timisoara,
2, Vasile Parvan Bv., 300223 – Timisoara, Romania
micha@dsplabs.cs.utt.ro

Abstract: The problem of providing high predictability when scheduling critical and hard real-time applications on embedded and DSP-based platforms is studied in this paper. A model of hard real-time tasks, the ModX (executable module), is presented and a set of non-preemptive scheduling techniques are discussed, based on this model. Extensive evaluation tests have been performed to simulate and analyze the proposed scheduling algorithms and their comparative performance. The main evaluation results are also discussed in this work.

Keywords: Non-preemptive, scheduling, hard real-time, embedded.

INTRODUCTION

Digital control is a topic of major interest in today's engineering and research activities. Embedded systems and digital signal processing (DSP) systems [Cre03, Gro04] are widely used in digital control applications, requiring in most cases hard real-time behavior of the hardware-software components. There are two essential characteristics an embedded platform has to meet to provide correct operation results for critical applications [Mic04a, Ste01, Gai02]: (a) the entire process of system development should integrate the time coordinate, and (b) the system must provide maximum of predictability for the hard real-time tasks.

Our current research focuses on developing suitable methodologies and architectures that enable hard real-time systems to meet the two basic requirements stated here. The approach is based on studying and integrating proper models of time, signals and tasks, emphasizing on non-preemptive scheduling techniques.

HARD REAL-TIME TASK MODEL

A *ModX* (*executable module*) is defined [Mic04b] as a periodic, modular, hard real-time task, with strict temporal specifications, scheduled and executed in non-preemptive context:

$$M_i \equiv \langle \boldsymbol{T}, \boldsymbol{P}, \boldsymbol{S}, \boldsymbol{F} \rangle \qquad (1)$$

where: $\boldsymbol{P} = \{\boldsymbol{P_{IN}}, \boldsymbol{P_{OUT}}, \boldsymbol{P_{GLB}}\}$ is the set of input, output and global parameters of M_i, respectively; $\boldsymbol{S} = \{\boldsymbol{S_{IN}}, \boldsymbol{S_{OUT}}\}$ is the set of input and output signals M_i interacts with; $\boldsymbol{F}$ is the task's instruction set (its functional specification); and:

$$\boldsymbol{T} = \left\{ T_{pr}^{M_i}, T_{ex}^{M_i}, T_{dl}^{M_i}, T_{dy}^{M_i}, N^{M_i} \right\} \qquad (2)$$

represents the set of temporal parameters of M_i, in their respective order: period, execution time, deadline, delay of execution during each period, and execution count (Figure 1).

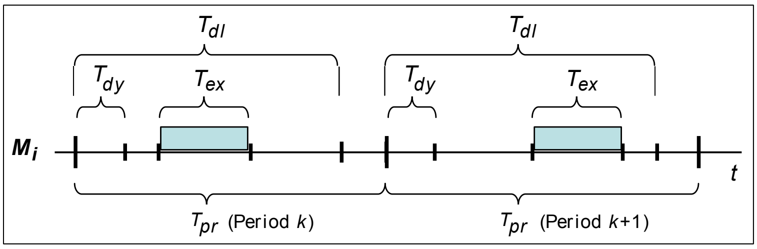

Figure 3. Temporal parameters of ModX M_i

NON-PREEMPTIVE SCHEDULING ALGORITHMS

Two main dynamic non-preemptive scheduling algorithms, considered as most efficient in the literature [Geo96, Kan98, Jef91], have been adapted to our task model: *MLFNP* (Minimum Laxity First Non-Preemptive) and *EDFNP* (Earliest Deadline First Non-Preemptive). A set of lemmas and theorems are introduced to discuss the particularities of these algorithms and some of the schedulability tests which apply to our ModX model.

The implementation of a particular scheduling algorithm on a target embedded or DSP-based platform requires the additional execution of an *online scheduler*, M_S, along with the application ModXs. A typical online scheduling and execution architecture includes a *Dispatch Table* ("*HDis_Tab*") of a bounded length λ and a *Dispatcher* system task, in addition to the online scheduler M_S (see Figure 2). We propose two approaches: an online scheduler with constant number of ModX executions during each scheduling cycle – *CEC-NPOS* (Constant Execution Cycles – NP Online Scheduler), and a cyclic (periodic) online scheduler – *PC-NPOS* (Periodic Cycles – NPOS, see also Figure 2). Additional necessary conditions imposed by the *CEC-NPOS* and *PC-NPOS*, as in (3), are derived and demonstrated.

$$\sum_{i=1}^{n} \left\lceil 2\left(T_{pr}^{M_S} - T_{ex}^{M_S} \right) \cdot 1 \big/ T_{pr}^{M_i} \right\rceil \le \lambda \qquad (3)$$

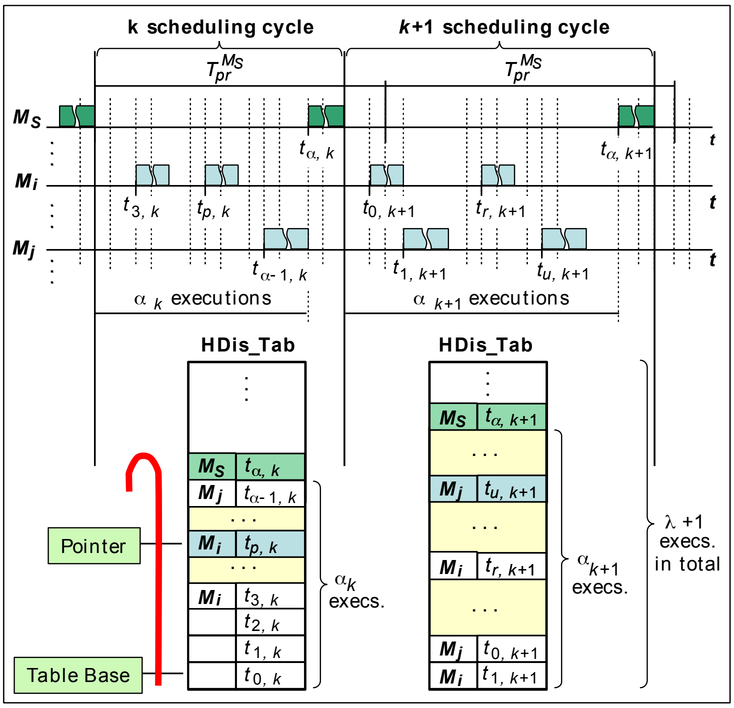

Figure 4. Example of two consecutive scheduling cycles, for the *PC-NPOS* algorithm

PERFORMANCE OF THE NON-PREEMPTIVE ALGORITHMS

A comparative evaluation of the *MLFNP* and *EDFNP* offline algorithms and of the *CEC-NPOS* and *PC-NPOS* online schedulers has been performed, using 12 workstations. More than 59000 tests have been accomplished to calculate the *schedulability ratio* (*SR*) for the two pairs of algorithms, as a function of the following additional parameters: the total number of ModXs in the sets, the processor utilization *PU* and the ModX periods, randomly generated using the uniform and normal distributions (Figure 3).

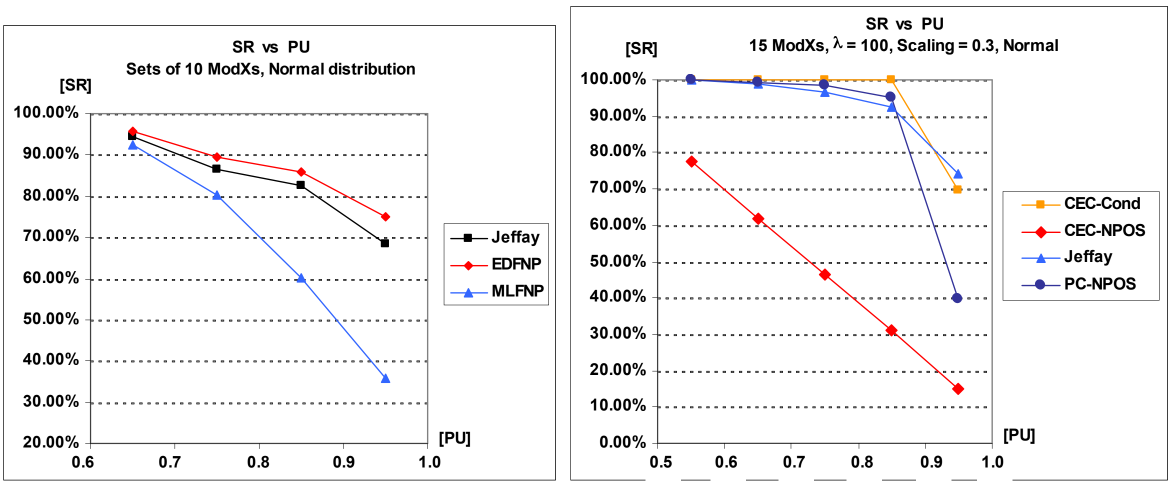

Figure 5. Performance evaluation of the offline and online non-preemptive scheduling algorithms

CONCLUSION

Critical and hard real-time applications require high operation predictability of the target system. Non-preemptive task models and scheduling techniques have been proven as a valid solution to develop and implement such applications on embedded and DSP-based platforms.

The performance evaluation tests show that *EDFNP* behaves better than *MLFNP*. Therefore, *EDFNP* has been chosen as the core of the online scheduling algorithms further developed to accommodate the realistic implementation of non-preemptive scheduling on real-time platforms (*CEC-NPOS* and *PC-NPOS*). Extensive evaluation tests also prove the efficiency of the *PC-NPOS* online scheduling technique over the *CEC-NPOS* algorithm.

The non-preemptive task model and scheduling techniques presented in this paper are currently being used in the development and implementation of a hard real-time kernel on a Motorola DSP56307 EVM platform: the HARETICK kernel [Mic04a].

REFERENCES

[Cre03] Cretu, V., Jurca, T., Micea, M. V., Sora, I., (2003), "Instrumentation and Measurement in Romania: Technical Developments at 'Politehnica' University of Timisoara", in *IEEE Instrum. Meas. Mag.*, Vol. 6, No. 3, pp. (41-47).

[Gai02] Gai, P., Abeni, L., Buttazzo, G., 2002, "Multiprocessor DSP Scheduling in System-on-a-Chip Architectures", in *Proc. of the 14th Euromicro Conf. on Real-Time Syst.* (ECRTS'02), Vienna, Austria pp. (231-240).

[Geo96] George, L., Rivierre, N., Spuri, M., 1996, "Preemptive and Non-Preemptive Real-Time Uni-Processor Scheduling", *Rapport de recherche*, Nr. 2966, Institut National de Recherche en Informatique et en Automatique, INRIA, Rocquencourt, France.

[Gro04] Groza, V., Abielmona, R., El-Kadri, M., Elbadri, M., Sakr, N., 2004, "A Reconfigurable Co-Processor for Adaptive Embedded Systems", in *Proc. of the 2nd Wshop. on Intelligent Solutions in Embedded Syst.*, WISES 2004, Graz, Austria.

[Jef91] Jeffay, K., Stanat, D., Martel, C., 1991, "On Non-Preemptive Scheduling of Periodic and Sporadic Tasks", in *Proc. of the 12th IEEE Real-Time Syst. Symp.*, San Antonio, Texas, IEEE Computer Society Press, pp. (129-139).

[Kan98] Kang, S.-I., Lee, H.-K., 1998, "Analysis and Solution of Non-Preemptive Policies for Scheduling Readers and Writers", in *ACM Operating Syst. Review*, No. 32 (3), pp. (30-50).

[Mic04a] Micea, M. V., Cretu, V., 2004, "Non-Preemptive Execution Support for Critical and Hard Real-Time Applications on Embedded Platforms", in *Proc. of the 2004 Intl. Sympos. on Signals, Syst. and Electronics*, ISSSE'04, Linz, Austria, [CD-ROM version].

[Mic04b] Micea, M. V., Cretu, V., Patcas, L. M., 2004, "Program Modeling and Analysis of Real-Time and Embedded Applications", in the *Sci. Bulletin of "Politehnica" Univ. of Timisoara*, *Trans. on Autom. Ctrl. and Comput. Sci.*, Vol. 49 (63), No. 3, Timisoara, Romania, pp. (207-212).

[Ste01] Stewart, D. B., 2001, "Twenty-five Most Common Mistakes with Real-time Software Development", in the *2001 Embedded Syst. Conf.*, Class 270, San Francisco.