2005 American Control Conference
June 8-10, 2005. Portland, OR, USA

Real-Time Implementation of Model Predictive control

Leonidas G. Bleris and Mayuresh V. Kothare

Abstract— **A real-time implementation of Model Predictive Control (MPC) is presented in this paper. MPC, also known as receding horizon control and moving horizon control, is widely accepted as the controller of choice for multivariable systems that have inequality constraints on system states, inputs and outputs. For processes with slow dynamics and low sampling rates, MPC is typically implemented on a dedicated computer. For systems with fast dynamics such as those in MEMS, a hardware embedded MPC would be an appropriate controller implementation since the size and the application precludes the use of a dedicated computer. Recent manufacturing advances have opened the path for the fabrication of micromechanical devices and electronic subsystems under the same manufacturing and packaging process, thereby opening the path for the use of advanced control algorithms towards systems-on-chip applications.**

I. Introduction

Micro-Electro-Mechanical Systems (MEMS) integrate mechanical elements, sensors, actuators, and electronics, commonly on silicon substrates through micro-fabrication techniques. MEMS have enjoyed substantial growth in the past decade. Recent studies have estimated the market for intelligent micromachine based systems to be around 100$ Billion/year [1]. This is due to its wide application range that includes medicine and bioengineering (DNA and genetic code analysis and synthesis, drug delivery, diagnostics and imaging), automotive systems (transducers and accelerometers), manufacturing and fabrication (microscale smart robots), etc.

A major technological challenge is the combination of micromechanical devices with an electronic sub-system in the same manufacturing and packaging process; viz, the realization of autonomous Systems-on-a-Chip (SoC) applications. The incorporation of new types of functionality onto the chip will enable the creation of monolithic devices able to sense, think (carry out advanced algorithms), act and communicate as well. The goal is that these SoC devices have improved performance and robustness but also low manufacturing and packaging costs. In this direction new manufacturing processes are investigated [2]. Some applications that illustrate the viability and commercial potential of this integration have been already reported. An accelerometer [3] has been developed and marketed by Analog Devices. The fabrication was made by customizing their manufacturing processes in order to produce the micromechanical devices using the same processes that produce the integrated electronics. In another approach, researchers [4] have developed a modular integrated approach in which the aluminum metallization of CMOS is replaced with tungsten to enable the CMOS to withstand subsequent micromechanical processing. Additionally, a European IST project SiGeM is under way [5], where an integrated gyroscope is developed by post-processing thick poly-SiGe surface micro-machined structures on top of CMOS circuitry. This integrated gyroscope is expected to have a higher performance compared to the state-of-the-art two-chip solutions due to a drastic reduction in parasitic capacitance. This project should also open up opportunities for using poly-SiGe for other MEMS. CMOS integration of sensors is an important step towards system miniaturization and improved performance. Post-CMOS integration of MEMS allows the use of conventional CMOS processes and fairly independent optimization of the CMOS and MEMS processes. However, it dictates a lower thermal budget for MEMS processing. Poly-SiGe provides properties required for MEMS applications at significantly lower temperatures compared to poly-Si ($= 800^oC$). In the past year some processes have been developed for depositing poly-SiGe films at CMOS-compatible temperatures ($< 450^oC$ and even $< 400^oC$ for compatibility with low-permittivity dielectrics) [6].

Parallel to these fabrication and integration research efforts, modeling and control of MEMS has gained significant importance, due to the direct connection to the functionality of such devices. Given the ability to obtain mathematical models of MEMS, often in the form of nonlinear differential equations [7], [8], [9], a logical next step is applying feedback control. While an introduction to control of MEMS is provided in [7](Chapter VII) the area is relatively new [10], [11], [12].

Our research effort is centered on developing and embedding model predictive control [13], [14] for microchemical system applications. Microchemical systems are a new generation of miniature chemical systems that carry out chemical reactions and separations in precisely fabricated three dimensional microreactor configurations in the size range of a few microns to a few hundred microns. Applying control in a microchemical system may include efficient mixing of different laminar streams, manipulating microflows and adjusting the temperature distribution.

This paper is organized as follows. Section II contains a literature review of real-time MPC implementations, a brief theoretical introduction on MPC, and information on the

Partial financial support for this research from the US National Science Foundation under grants CTS-9980781 (“Engineering Microsystems: XYZ-on-a-chip” program) and CTS-0134102 (CAREER program) is gratefully acknowledged.

Leonidas G. Bleris is with the Department of Electrical and Computer Engineering, Lehigh University, Bethlehem, PA, 18015 leb3@lehigh.edu

Mayuresh V. Kothare is with the Faculty of Chemical Engineering, Lehigh University, Bethlehem, PA, 18015 mayuresh.kothare@lehigh.edu

general purpose processor we use for this implementation. A case study is examined in Section III, where we include the experimental results. In Section IV we provide the results of profiling the performance of the embedded MPC. We conclude the paper with remarks on our research results.

II. Embedded Model Predictive Control

Model predictive control originated in the chemical process industries. The main advantages of MPC are the ability to handle constraints and its applicability to multivariable nonlinear processes. Because of the computational requirements of the optimizations associated with MPC, it has primarily been applied to plants in the process industry, with slow dynamics. Furthermore, existing implementations of MPC typically perform numerical calculations using workstations in 64-bit Floating Point (FP) arithmetic, which is too expensive, power demanding and large in size.

The application of real-time embedded model predictive control for microscale devices presents new technological challenges, and some initial results have been reported in this direction. A real-time multiprocessing system is provided in [15], where the author proposes a programming procedure that results in the fastest implementation of the core calculations of model predictive control algorithms. The combination of Field-Programmable Analog Arrays (FPAAs) with Field-Programmable Gate Arrays (FPGAs) was proposed in [16] for the development of dynamically reconfigurable analog/digital hardware, capable of handling MPC computation requirements. A feedback scheduling strategy for multiple MPCs is proposed in [17], where the scheduler allocates CPU time to the tasks according to the current values of the cost functions. Since the MPC algorithm is iterative, the feedback scheduler may also abort a task prematurely to avoid excessive input-output latency. Finally, a Multi-Parametric (MP) programming method was proposed [18], [19] to solve off-line the quadratic optimization problem associated with MPC. The constrained quadratic optimization is shown to be piecewise affine, using partitions of the state space determined by the constraints. The feedback laws are precomputed and the online calculations consist of a table-lookup in memory and an affine transformation. However, for an input constrained problem, increasing the number of controlled variables (thus the number of constraints), yields prohibitive memory requirements for multivariable constrained systems with fast dynamics.

Alternatively, we have proposed [20], [13] a parametric methodology based on emulations, for reducing the precision of the microprocessor to the minimum while maintaining optimal control performance. By reducing the precision we are increasing the optimization speed while reducing the power consumption and the overall chip area. A Logarithmic Number System (LNS) based microprocessor architecture is used, providing further energy and computational cost savings.

A. Theoretical background

Mathematical models of different complexity can be developed for MEMS applications. Partial differential equations are commonly used to describe their dynamical behavior. Nevertheless techniques of reduced order modeling can be applied leading often to state-space formulations [7] of the form:

$$x(k+1) = Ax(k) + Bu(k) \quad (1)$$

$$y(k) = Hx(k) \quad (2)$$

where x represents the states, y the output, and u the actuation. The problem can be with input or output constraints. The matrices A, B and H depend upon the actual examined MEMS. A schematic of the implementation of MPC is given in the following Figure 1.

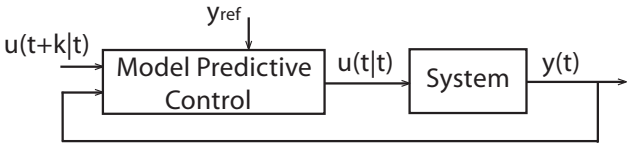

Fig. 1. Block diagram of MPC.

The steps for the application of MPC are described with the following:

1. Initially the future outputs are calculated at each sample interval over a predetermined horizon N, the prediction horizon, using the process model. These outputs $y(t+k|t)$ for $k = 1...N$ depend up to the time t on the past inputs and on the future signals $u(t+k|t)$, $k = 0...N-1$ which are those to be sent to the system.

2. The next step is to calculate the set of future control moves by optimizing a determined criterion in order to keep the process as close as possible to a predefined reference trajectory. This criterion is usually a quadratic function of the difference between the predicted output signal and the reference trajectory. In some cases the control moves $u(t+k|t)$ are included in the objective function in order to minimize the control effort:

$$J_P(k) = \sum_{k=0}^{P} \{[y(t+k|t) - y_{ref}]^2 + Ru(t+k|t)^2\} \quad (3)$$

$$|u(t+k|t)| \leq b \quad , \quad k \geq 0 \quad (4)$$

where $y(t+k|t)$ are the predicted outputs, y_{ref} is the desired set reference output, $u(t+k|t)$ the control sequence and R is the weighting on the control moves, a design parameter. This system is subject to input constraints given by the vector b.

3. Finally, the first control move $u(t|t)$ is sent to the system while the rest are rejected. This is because at the next sampling instant the output $y(t+1)$ is measured by the system and the procedure is repeated with the new values so that we get an updated control sequence.

B. Emulated Optimization

The optimization is a fundamental part of MPC since it results in optimal control inputs for the process. The computational effort required in an embedded real-time MPC derives almost entirely from the optimization algorithm. Logically the choice of the optimization technique [21] is decisive to the performance of the controller. It has to be noted that from the architecture perspective the operations that are responsible for the majority of instructions that are executed in the processor are matrix inversions and abundant dot products. The algorithm that is implemented for the minimization of the cost function is a direct application of Newton's method, incorporating the constraints in the cost function, using barrier functions for the inequality constraints, and penalty functions for equality constraints, defined as:

$$d_i(u) = \mu_i (a_i^T u - b_i)^2 \,, \quad i \in E \tag{5}$$

$$d_i(u) = \mu_i \log(a_i^T u - b_i) \,, \quad i \in I \tag{6}$$

resulting in the unconstrained non-linear problem:

$$\underset{u}{minimize} \quad f(u) = \frac{1}{2} u^T G u + g^T u + \sum_i d_i(u) \tag{7}$$

The problem of equation (7) can be solved numerically approximating $f(u)$ by a quadratic function around u, obtaining the gradient $\nabla f(u)$ and Hessian $H(u)$, and iterating:

$$u^{(k+1)} = u^{(k)} - H^{-1}(u^{(k)}) \cdot \nabla f(u^{(k)}) \tag{8}$$

The predicted output can be computed explicitly, at each sampling time, using $u(t+k|t)$ and the state space model of (4). The complexity of this algorithm is dominated by the computation of H^{-1}, which requires $O(n^3)$ operations, where n is the number of variables in the optimization. The advantage of formulating the on-line MPC problem in this fashion is that it directly extends to other objective functions, as opposed to the MP approach which applies to QPs [18], [19].

C. Hardware implementation of MPC

For the real-time implementation of model predictive control we use the high-performance single board computer phyCORE-MPC555, illustrated in Figure 2. This boards packs the power of Motorola's embedded 32-bit MPC555 microcontroller within a miniature footprint. The MPC555 is a high-speed 32-bit Central Processing Unit that contains a 64-bit floating point unit designed to accelerate the advanced algorithms necessary to support complex applications. All signals and ports of the MPC555 extend to two Molex high density (0.635 mm pitch) 160 pin header connectors. These high density pins allow it to be plugged like a "big chip" into user target hardware.

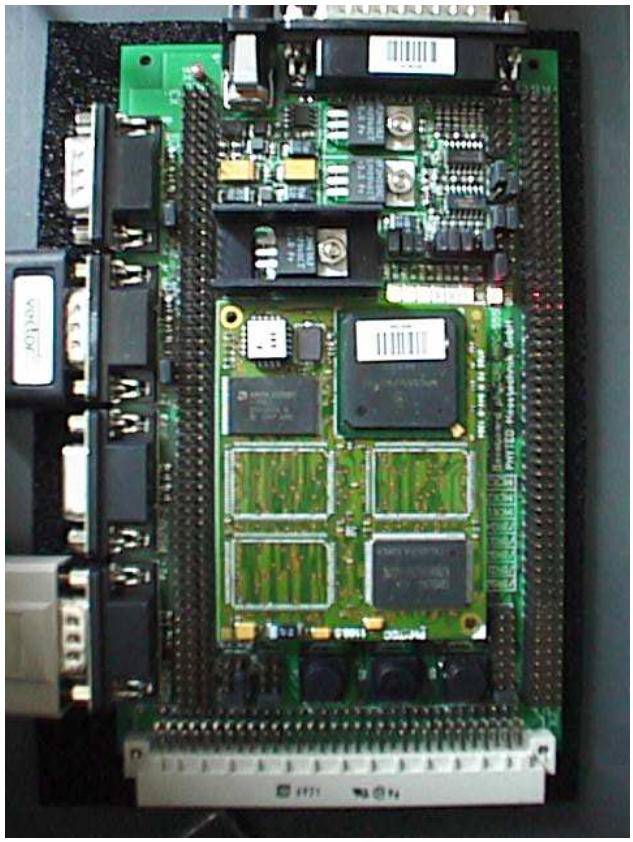

Fig. 2. PhyCORE-MPC555 board.

III. Case Study

The problem examined has been adapted from [22]. It is a state-space problem that describes the dynamics of a rotating antenna at the origin of the plane (driven by an electric motor). The control problem is to use the input voltage to the motor (u V) to rotate the antenna so that it always meets the a predefined objective (i.e. points towards a moving object in the plane). We assume that the angular positions of the antenna and the moving object (θ and θ_r rad respectively) and the angular velocity of the antenna ($\dot{\theta}$ rad/sec) are measurable. The motion of the antenna can be described by the following discrete-time equations obtained from their continuous-time counterparts by discretization using a sampling time of 0.1 s and Euler's first-order approximation for the derivative:

$$x(k+1) = \begin{bmatrix} 1 & 0.1 \\ 0 & 1 - 0.1a(k) \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 0.1k \end{bmatrix} u(k)$$

$$y(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(k)$$

The parameter $a(k)$ is proportional to the coefficient of viscous friction in the rotating parts of the antenna and for the simulations we assume that $a(k) = 1$.

In order to test the performance of MPC on the embedded target we use Processor-In-the-Loop (PIL) techniques. As

Fig. 3. The Motorola MPC555 processor on the top right. The size of the chip is about 2cm×2cm.

illustrated in Figure 4, the board is connected to a host workstation. In PIL mode, a plant model runs in non-real-time on the host workstation in Simulink. Meanwhile, generated code running on the MPC555 exchanges signals via RS232 serial communication with the Simulink simulation running on your workstation. At each sample interval, Simulink performs model updates and sends output signal data via RS232 to the MPC555. We currently examine FEMLAB based MEMS model as the actual controlled system by the embedded model predictive controller.

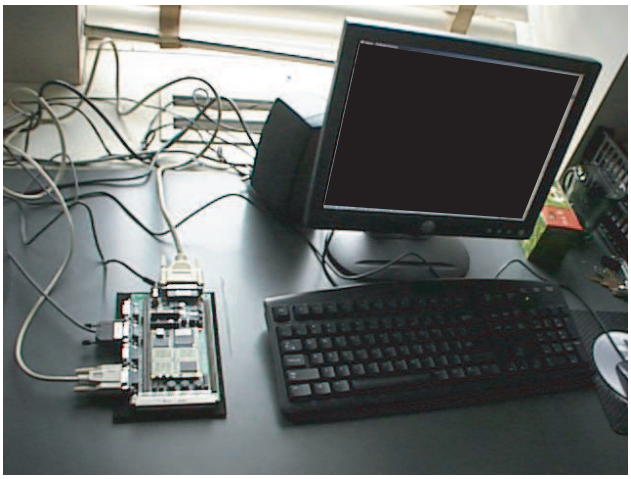

Fig. 4. Co-simulation setup

A. PIL Simulation Results

The main complexity of MPC arises from the underlying optimization algorithm, and the size of the matrices involved. Therefore these can be made arbitrarily large by increasing the control and prediction horizons, allowing us to test the proposed embedded controller performance in different computational complexity levels. The embedded controller behavior was tested for different cases, having the set-point at 2. The controlled variables are constrained between $-2 < u < 2$V.

In order to examine the performance of MPC running on the embedded board we set the number of optimizations, carried out at each sampling time, to a fixed number. Using profiling techniques, and by requiring the number of optimization to be fixed, we will be able to examine the performance of MPC on the Motorola processor for different control and prediction horizons. In Figure 5 we illustrate the output and the actuation for prediction horizon of 10 and control horizons of 3,4,6 and 8 (the output and actuation for the different cases overlap).

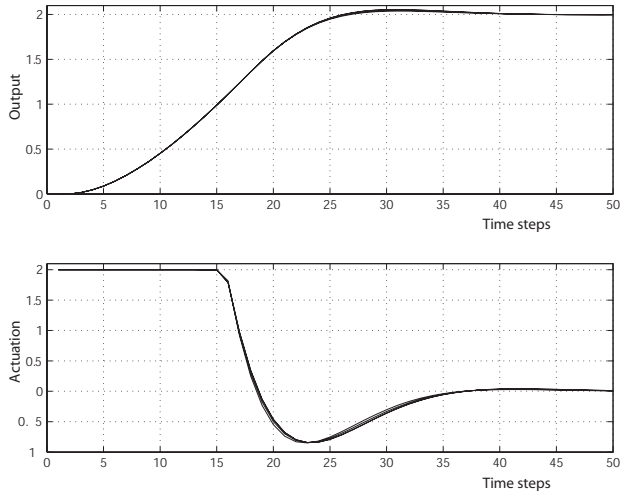

Fig. 5. Output and actuation for prediction horizon of 10 and control horizons of 3,4,6 and 8.

The output and the actuation for prediction horizon 20 and control horizons of 3,4,8 and 12 are illustrated in Figure 6. In this case, there is some deviation in the results but the performance remains satisfactory.

B. Hybrid MPC

In order to decrease the computational requirements of the optimizations associated with MPC a hybrid logic can be incorporated in the algorithm. For example, when the output reaches the set-point the MPC switches to a less expensive control scheme (i.e. gain scheduling). In this work we require that the optimizations are aborted when the updates on the actuation remain within specific bounds. In Figure 7 we illustrate the results for prediction horizon of 10 and control horizon of 6, after incorporating this logic to the algorithm. More specifically when the updates on the actuation remain within the bounds of $-0.00001 \leq u \leq 0.00001$ the optimizations are aborted.

IV. Code Profiling

A profiler can analyze the amount of time your code spends performing various tasks of the optimization and

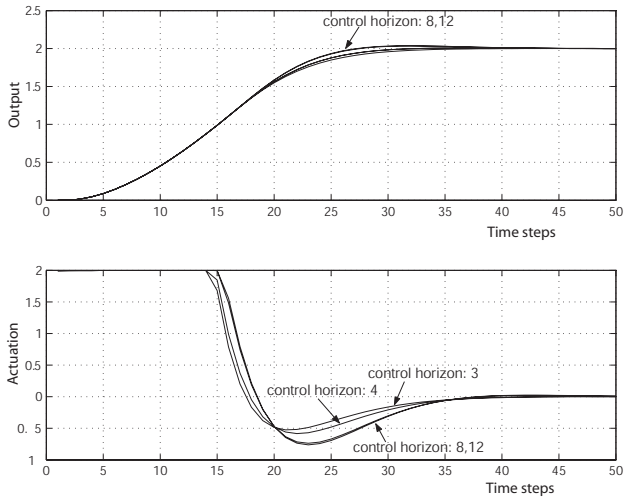

Fig. 6. Output and actuation for prediction horizon of 20 and control horizons of 3,4,8 and 12.

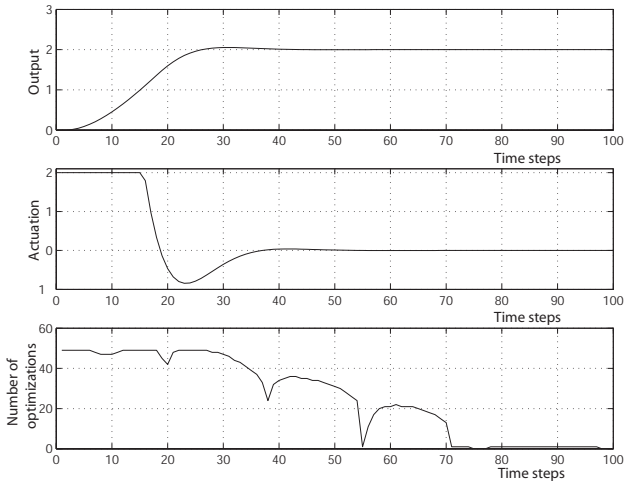

Fig. 7. Output, actuation and number of optimizations for prediction horizon of 10 and control horizon of 6.

detect bottlenecks (time consuming routines that data passes through) that are just inordinately slow. Therefore profiling should be used to improve the efficiency of core routines and improve a subsequent SoC implementation. Most of the profilers perform statistical sampling of the runtime environment; these profilers are called passive or sampling profilers. A passive profiler divides the program being profiled into evenly-sized "buckets" in memory. It then samples the processor's program counter at regular intervals to determine which bucket the counter is in. The main advantage of a passive profiler is that it requires no modification to the program under observation. You just run the profiler and tell it what program to observe. Also, passive profilers distribute the overhead that they incur evenly over time, allowing the post-processing steps to ignore it. On the other hand, they cannot sample too frequently or the sampling interrupt will overwhelm the program being sampled. In addition, because they rely on a statistical sampling technique, the program must run for a long enough period to collect a valid sample.

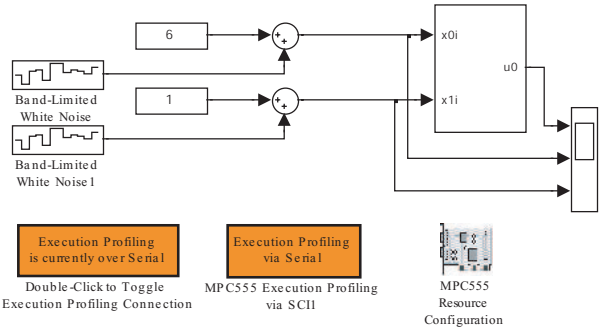

Fig. 8. Profiling setup.

In order to profile the performance of the embedded target we use the setup of Figure 8, under SIMULINK environment. The block represents MPC running on the Motorola board, and the states oscillate with the addition of bandlimited white noise, thereby requiring new optimization solutions at each sampling interval. The simulation results are provided in Figure 9.

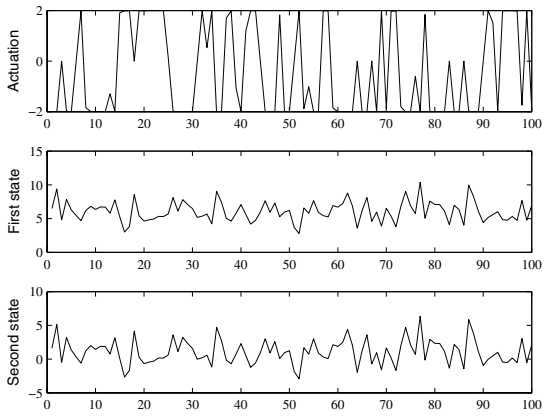

Fig. 9. Simulation results of the profiler

To examine the influence of the control horizon on the computational performance of MPC running on the MPC555 processor we set the number of optimizations at a fixed number. The results are given in Figure 10 for fifty and twenty optimizations. Additionally, in Figure 11 we provide the performance of MPC for different number of optimizations keeping a control horizon of 3 and prediction horizon of 10. As expected the time grows linearly with the optimizations for fixed prediction and control horizons. An interpolation shows that for the particular problem and

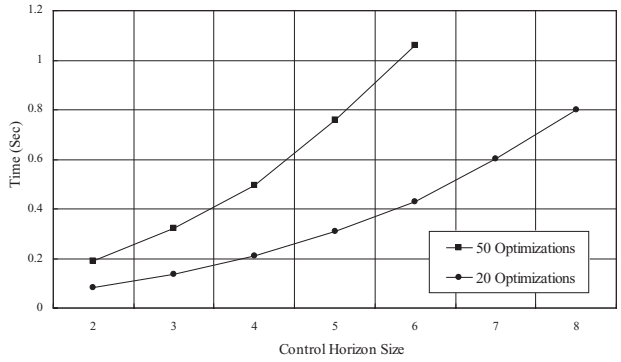

Fig. 10. Profiling results for different control horizons with 20 and 50 optimizations.

MPC parameters, we spend approximately 15 milliseconds for each optimization.

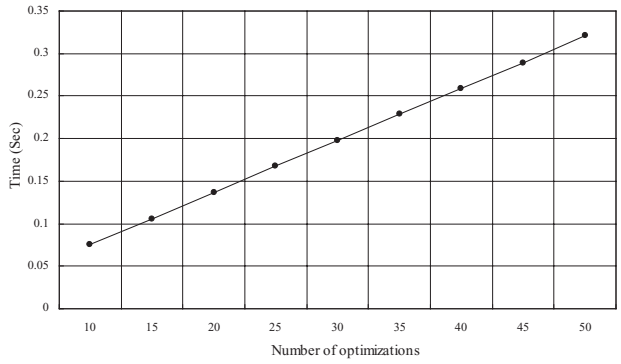

Fig. 11. Profiling results for control horizon 3 and prediction horizon of 10, for different number of completed optimizations.

V. Concluding Remarks

Manufacturing advances have opened the path for the fabrication of micromechanical devices and electronic subsystems under the same manufacturing and packaging process. It is safe to assume that the cost of such manufacturing process will decrease with time, thereby opening the path for the use of advanced control algorithms towards systems-on-chip applications. The advantages of MPC such as the ability to handle constraints, the applicability to nonlinear processes and to multivariable problems, will make this control method a necessary choice for many MEMS applications.

In [20] we proposed a methodology for designing an Application Specific Instructor Processor (ASIP) optimized for the operations associated with model predictive control. In this paper we presented an implementation of model predictive control using a general purpose processor. With the use of PIL simulations we were able to customize the optimization algorithm used and choose the optimal design parameters, improving the algorithm prior to proceeding to fabrication. Furthermore, profiling results allow us to obtain information on the performance of this processor, and make the necessary comparisons with our proposed ASIP.

References

[1] Sandia National Laboratories. Vision for MEMS. http://www.sandia.gov/mstc/technologies/micromachines/vision.html.

[2] J. H. Smith, S. Montague, J. J. Sniegowski, J. R. Murray, and P. J. McWhorter. Embedded micromechanical devices for the monolithic integration of MEMS with CMOS. In *IEEE International Electron Devices Meeting (IEDM)*, pages 609–612, Albuquerque, NM, June 1995.

[3] W. Kuehnel and S. Sherman. A surface micromachined silicon accelerometer with on-chip detection circuitry. *Sensors and Actuators*, 45:7–16, 1994.

[4] W. Yun, R. Howe, and P. Gray. Surface micromachined, digitally force-balanced accelerometer with integrated CMOS detection circuitry. In *IEEE Solid-State Sensor and Actuator Workshop*, page 126, Boston, MA, December 1992.

[5] Information Society Technologies. Poly-SiGe for CMOS backend integration of MEMS. http://www.cordis.lu/ist/.

[6] A. Witvrouw. Poly Si-Ge, a superb material for MEMS. In *MRS fall meeting symposium: Micro and Nanosystems*, Boston, MA, December 2003.

[7] S. E. Lyshevski. *MEMS and NEMS*. CRC Press, 2002.

[8] N. Maluf. *An Introduction to Microelectromechanical Systems Engineering*. Artech House MEMS Library, Norwood, MA, 2000.

[9] T. T. Hsu. *MEMS and Microsystems - Design and Manufacture*. McGraw-Hill, 2002.

[10] A. M. Shkel. Smart MEMS: Micro-structures with error-suppression and self-calibration control capabilities. In *Proceedings of the 2001 American Control Conference*, pages 1208–1213, Arlington, VA, June 2001.

[11] E. Grayver and R. T. M'Closkey. Automatic gain control ASIC for MEMS gyro applications. In *Proceedings of the 2001 American Control Conference*, pages 1219–1222, Arlington, VA, June 2001.

[12] M. Napoli, B.Bamieh, and M.Dahleh. Optimal control of arrays of microcantilevers. *ASME Journal of Dynamic Systems Measurement and Control*, 121:686–690, 1999.

[13] L. G. Bleris, M. V. Kothare, J. G. Garcia, and M. G. Arnold. Towards embedded model predictive control for system-on-a-chip applications. *Accepted: Journal of Process Control*, 2004.

[14] L. G. Bleris, J. G. Garcia, and M. V. Kothare. Model predictive hydrodynamic regulation of microflows. In *Accepted: 2005 American Control Conference*, Portland, OR, July 2005.

[15] George Hassapis. Implementation of model predictive control using real-time multiprocessing computing. *Microprocessors and Microsystems*, 27:327–340, 2003.

[16] O. A. Palusinski, S. Vrudhula, L. Znamirowski, and D. Humbert. Process control for microreactors. *Chemical Engineering Progress*, pages 60–66, 2001.

[17] D. Henriksson, A. Cervin, J. Akesson, and K. Arzen. Feedback scheduling of model predictive controllers. In *Proceedings of the 8^{th} IEEE Real-Time and Embedded Technology and Applications Symposium*, San Jose, CA, September 2002.

[18] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, January 2002.

[19] E. N. Pistikopoulos. On-line optimization via off-line optimization! - a guided tour to parametric programming and control. Invited plenary lecture at the 7^{th} IFAC Symposium on Dynamics and Control of Process Systems (DYCOPS-7), Boston, MA, July 2004.

[20] L. G. Bleris, M. V. Kothare, J. G. Garcia, and M. G. Arnold. Embedded model predictive control for system-on-a-chip applications. In *Proceedings of the 7^{th} IFAC Symposium on Dynamics and Control of Process Systems (DYCOPS-7)*, Boston, MA, July 2004.

[21] R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, 1987.

[22] M. V. Kothare, V. Balakrishnan, and M. Morari. Robust constrained model predictive control using linear matrix inequalities. *Automatica*, 32(10):1361–1379, October 1996.