Embedded Model Predictive Control for an Electric Submersible Pump on a Programmable Logic Controller*

B. J. T. Binder[1], D. K. M. Kufoalor[1], A. Pavlov[2], T. A. Johansen[1]

Abstract— **Electric Submersible Pumps (ESP) can be installed in oil wells to create artificial lift, in order to increase recovery from the reservoir, and boost the production rate. A Model Predictive Controller (MPC) for an ESP, based on linearized models, is developed and designed using SEPTIC (Statoil's in-house software tool for MPC), and implemented on a Programmable Logic Controller (PLC). Automatic code generators have been utilized to generate an application-specific MPC controller in ANSI C code. Hardware-in-the-loop simulation results show that the control performance of the PLC implementation is comparable to the original SEPTIC controller.**

I. INTRODUCTION

MPC is widely used in the onshore petroleum industry, such as refineries. Such applications are typically implemented on top of low-level decentralized controllers, where the low-level (monovariable) controllers address basic requirements regarding safety and stability, while the MPC controller improves the performance of the overall multivariable system. The basic controllers are typically implemented on ultra-reliable hardware, such as Programmable Logic Controllers (PLCs), while MPC applications are usually implemented in a PC/server-based environment [1], [2].

Though MPC is very common in the onshore petroleum industry, it is not common offshore, where safety and reliability are of the essence, and ultra-reliable hardware and software is required. Due to space limitations, offshore equipment tend to be of a considerable smaller scale, leading to much faster dynamics. Better control performance may be possible to achieve by letting fast MPC optimize the plant directly, bypassing low-level control. However, such an MPC implementation must not only operate on a higher update frequency, but also fulfill hard real-time requirements, as well as safety and reliability requirements. The use of ultra-reliable hardware and firmware is thus required.

PLCs are commonly used in the industry, due to their reliability and robustness properties, but the computationally demanding MPC algorithm introduces challenges regarding the limited computational resources usually found in such hardware. However, significant progress has recently been made in the area of embedded MPC. Implementation aspects for predictive control on a PLC has been considered in [3] and [4]. Important contributions include exploiting the MPC problem structure and computational architectures [5], [6]. This has resulted in a variety of software tools targeting embedded platforms, with efficient and portable software implementations of algorithms, such as FORCES [6], qpOASES [7], FiOrdOs [8], and CVXGEN [9].

We have not found many references addressing dynamic control of ESP-lifted wells. The complexity of operating ESP-lifted fields is shown in [10], where improving and automating the workflow for production operations to improve the asset performance, including optimization and diagnostics based on real-time data, is discussed. In [11], optimization of ESP-lifted oil fields based on steady-state optimization and simple PI-control is considered. However, as the focus is on steady-state optimization, the dynamics that would arise e.g. from external disturbances are not considered. Also, perfect system knowledge is assumed. In [12], [13], a model of an ESP-lifted well is developed, the control challenges related to ESPs are discussed, and automatic control of an ESP installation is implemented and tested in a large-scale industrial test facility. The results motivate the use of MPC to provide optimal dynamic control.

Whereas dynamic control is the focus in [13], the focus in the present paper is on the controller implementation aspects. Feasibility of embedded MPC is investigated for a production well incorporating an Electric Submersible Pump (ESP) for artificial lift. The considered system is described in Section II, together with a mathematical model of the system developed by Statoil in [12], [13], which is implemented as a simulator in MATLAB.

An MPC controller for this system is developed using SEPTIC, Statoil's in-house software for MPC. The controller development and configuration is described in Section III. An embedded version of the developed controller is then implemented on a PLC, as described in Section IV. Hardware-in-the-loop simulation results are presented and discussed in Section V, and concluding remarks are given in Section VI.

II. SYSTEM DESCRIPTION

The considered system consists of one oil well with an Electric Submersible Pump (ESP) and a production choke valve, as seen in Fig. 1. The ESP is used for artificial lift, with the purpose of reducing the hydrostatic pressure from the fluid in the well and the production line, thus reducing the bottom hole pressure, increasing the inflow from the reservoir and the production from the well.

*This work was supported by the Research Council of Norway and Statoil

[1]Department of Engineering Cybernetics, Norwegian University of Science and Technology, O.S. Bragstads plass 2D, NO-7491 Trondheim, Norway. {benjamin.binder, kwame.kufoalor, tor.arne.johansen}@itk.ntnu.no

[2]Statoil ASA Research and Development Center, P.b. 1004, NO-3905 Porsgrunn, Norway alepav@statoil.com

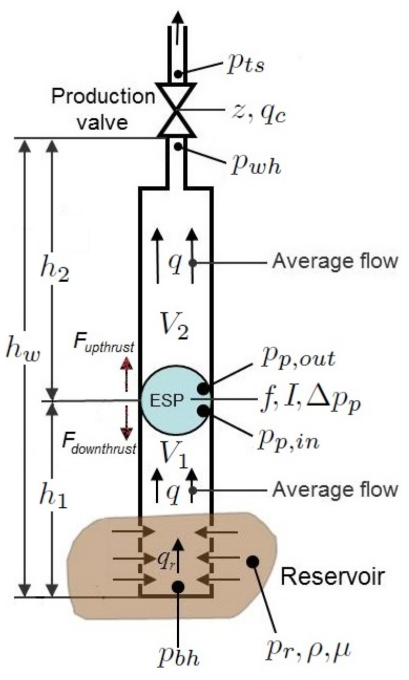


Fig. 1. Well with ESP

A. Control targets

There are a number of variables that affect the life-time of ESP installations, such as power consumption, flow rate, pressure, temperature, thrust forces and vibration. Operation outside of certain limits on these variables may lead to failure or reduced life-time of the ESP, which has a huge economic impact, both due to the costs of replacing the pump, and the loss of production. Thus, the main control priority in this system is to maintain good operating conditions for the ESP. This study focuses on keeping a constraint on the electric current, which is proportional to the power consumption of the pump, and keeping constraints on the upthrust and downthrust forces acting on the ESP shaft.

The next priority is to ensure acceptable operational conditions for the well. This includes keeping the pump inlet pressure within certain constraints, and to avoid backflow into the well by keeping the wellhead pressure higher than the topside pressure.

Less prioritized, but still important control targets, are to maximize the pump efficiency, and optimize certain production condition parameters. This includes keeping the inlet pressure at a certain setpoint, and minimize the electric current flow in the ESP motor. Another control target is to avoid high wellhead pressure relative to the topside pressure; a high pressure difference means there is a large power loss in the topside choke, which is a waste of power provided by the ESP. Choking the well may be necessary to keep the ESP within its operational limits, but is otherwise not desired.

B. Third order model

A simple, third order model of an ESP lifted well is developed by Statoil in [12], [13]. A similar model is also developed in [14], which is used for optimization in [11]. The simulator implemented in MATLAB is based on the model found in [13], and the model is repeated here for convenience. All model variables and parameters are given in SI units, and all well parameters, such as reservoir pressure p_r and well productivity index PI, are considered known constants.

1) Model equations: The system model is given by the following differential equations:

$$\dot{p}_{bh} = \frac{\beta_1}{V_1}\left(q_r - q\right) \tag{1a}$$

$$\dot{p}_{wh} = \frac{\beta_2}{V_2}\left(q - q_c\right) \tag{1b}$$

$$\dot{q} = \frac{1}{M}\left(p_{bh} - p_{wh} - \rho g h_w + \Delta p_p - \Delta p_f\right) \tag{1c}$$

and the following algebraic equations:

$$q_r = \mathrm{PI}\left(p_r - p_{bh}\right) \tag{2a}$$

$$q_c = C_c\sqrt{p_{wh} - p_{ts}}\, z \tag{2b}$$

$$\Delta p_p\left(f, q\right) = \rho g H_0\left(q_0\right)\left(\frac{f}{f_0}\right)^2 \tag{2c}$$

In this model, ESP characteristics for head and brake horsepower (BHP) at a reference frequency f_0 are assumed to be available, denoted $H_0(q)$ and $P_0(q)$, respectively. The affinity laws found e.g. in [15] describe how flow, head and BHP are related to the pump frequency. As the available pump characteristics are only valid at the reference frequency, the affinity laws are used to calculate these variables at other frequencies. The affinity laws are also used to calculate the (theoretical) flow q_0 that the pump would provide at the reference frequency, given by:

$$q_0 = q\frac{f_0}{f} \tag{3}$$

Some key variables in the model are further defined below. See also Fig. 1.

2) Key Variables:

- State variables:

p_{bh}	bottom hole pressure in the well
p_{wh}	wellhead pressure
q	average flow in the well

- Control inputs (manipulated variables, MVs):

f	ESP motor frequency (pump speed)
z	production choke valve opening

- Disturbance (disturbance variable, DV):

p_{ts}	topside pressure

- Available measurements (controlled variables, CVs):

p_{ts}	topside pressure
p_{wh}	wellhead pressure
$p_{p,in}$	ESP inlet pressure
q	flow from the well
I	electric ESP motor current

- Other:

Δp_f	pressure drop due to friction (cf. [13])

The variables q_0 and $\Delta p_c = p_{wh} - p_{ts}$ are easily calculated from these measurements, and are also assumed to be available measurements. While the other measured variables are present in the third order model, models for the inlet pressure and the electric current are stated below.

3) Electric current: Using the affinity laws and equation (5.14) in [15], the electric current consumption of the ESP motor is modeled as:

$$I(f,q) = \frac{I_{np}}{P_{np}} P_0(q_0) \left(\frac{f}{f_0} \right)^3 \tag{4}$$

where I_{np} and P_{np} are the so-called nameplate ratings of the ESP motor.

4) Inlet pressure: The ESP inlet pressure is modeled as:

$$p_{p,in} = p_{bh} - \rho g h_1 - F_1(q) \tag{5}$$

where h_1 is the height from the reservoir to the pump, and $F_1(q)$ is the frictional pressure drop in that section.

5) Thrust forces: The upthrust and downthrust forces acting on the ESP are important control targets, but are not modeled directly. These forces depend on the pump frequency and flow. This means that, for a given pump frequency, constraints on the flow may be imposed to avoid upthrust or downthrust conditions. In this model, such constraints are assumed to be provided at the reference frequency, denoted $q_{0,min}$ and $q_{0,max}$. Constraints on the flow q at other frequencies may be calculated using the affinity laws, but this would make these constraints proportional to the pump frequency, which complicates the controller implementation. Instead, as the theoretical flow q_0 is easily calculated from (3), the real flow q is replaced as an output by q_0, and the following constant constraints are imposed:

$$q_{0,min} < q_0 < q_{0,max} \tag{6}$$

C. Limits

It is assumed that the ESP can only operate with a frequency between 35 and 65 Hertz, and the frequency can not change faster than 0.5 Hertz per second. The production choke valve opening z is also limited between 0 and 100 percent, and the rate of change is limited to 0.5 percent per second. These limits must be respected by the controller.

III. CONTROLLER DESIGN

The system to be controlled, described in Section II, is a multivariable control problem with multiple control targets and priorities, which motivates using Model Predictive Control (MPC), see e.g. [2]. Considering the dynamics of the system, a sampling rate of at least 1 Hertz (one control action per second) is considered a requirement for the controller.

A. SEPTIC

The controller in this study is developed and configured using SEPTIC (Statoil Estimation and Prediction Tool for Identification and Control), which is Statoil's in-house software tool for MPC. SEPTIC has been in use since 1997, with many running applications [16]. The main advantage of using SEPTIC (or a similar tool) is that the software is field-proven, and has many built-in features to ease the process of developing and configuring the controller to obtain the desired performance. The main considerations in the controller configuration is outlined in this section. Details regarding the implementation aspects of SEPTIC can be found in [16].

B. Control targets

In SEPTIC, three (optional) control targets may be specified for each output variable (CV, controlled variable): high limit, low limit and setpoint. These limits are not hard constraints, instead slack variables and penalties (see e.g. [2]) are used in the implementation. For each input variable (MV, manipulated variable), hard constraints are implemented both for high and low limits, and the rate of change. In addition, a setpoint (denoted ideal value, IV) may be assigned, and changing the input variables may be penalized to reduce the control action.

C. Internal models

To obtain internal models (models used for prediction in the controller), steps in the inputs z and f and the disturbance p_{ts} are applied to the simulator, and the response of the variables I, q_0, $p_{p,in}$ and Δp_c are measured. Linear SISO (single input, single output) step response models are then generated for each input/output pair, using built-in features in SEPTIC. In the controller, predictions of the multivariable system dynamics are made based on the superposition principle [16].

D. Priorities and weights

One of the features of SEPTIC is that, in addition to weighting of the different variables, the relative priority of each control target can be assigned explicitly. This is implemented as a sequence of steady-state quadratic programs that are solved to respect the specifications in the order of decreasing priority, each stage respecting the achievements from earlier stages. When all achievable steady-state targets are calculated, the dynamic optimization problem is adjusted accordingly before it is solved.

This feature has proven its value in many applications [16], and it makes the SEPTIC implementation of the specifications described in section II-A relatively straight-forward, with quite little effort on tuning. In this application, most of the control decisions are based on priorities, though the weighting of the variables is not superfluous. In transients (e.g. due to large disturbances), the weighting of the variables will be of great importance. Loosely speaking, the priorities are decisive for steady-state conditions, while the weights determine the transients.

The implemented controller configuration is provided in Table I. The first column states the explicitly assigned priority, while the last column states the assigned weight or penalty. The weights for the inputs f and z specify the penalization of change in the inputs; the limits are hard constraints. Of practical reasons (scaling of the variables), the units used in the controller are different from the SI-units in the model, as seen in the table.

TABLE I
MPC CONTROLLER CONFIGURATION

Prio	Var.	Unit	Low	Setp.	High	W
1	I	A			65	100
2	q_0	m^3/h	53.2		80.7	25
3	Δp_c	bar	1			25
3	$p_{p,in}$	bar	40		70	20
4	$p_{p,in}$	bar		50		10
5	I	A		0		1
5	q_0	m^3/h		65.9		1
6	Δp_c	bar			10	1
-	f	Hz	35		65	1*
-	z	$\%$	0		100	1*

*) Penalty on change

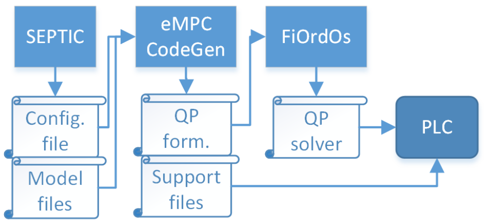


Fig. 2. Embedded implementation process

IV. EMBEDDED IMPLEMENTATION

The process of transferring the SEPTIC implementation to a PLC is outlined in Fig. 2, and is further described in this section.

A. Target platform

The target hardware platform is the Programmable Logic Controller (PLC) ABB AC500 PM592-ETH. This PLC has support for ANSI C89 and C99, and significant computational power. It has a G2_LE core implementation of the Freescale MPC603e microprocessor running at 400 MHz. This is a low-power RISC CPU with a dedicated hardware floating point unit (FPU), and the PLC can achieve a cycle time per floating-point instruction of 0.004 µs (minimum) [17]. The PLC is equipped with 4 MB RAM for user program memory, and 4 MB integrated user data memory.

The software package ABB PS501 Control Builder Plus (version 2.3.0), which is based on the CoDeSys automation platform technology, is used to integrate the MPC controller (C code) into a PLC software/runtime environment. The support offered for ANSI C code includes a restricted set of standard library functions, and linking against external libraries is not supported. Thus, the implemented C code must be self-contained (library-free).

The GNU GCC 4.7.0 compiler toolchain is used to compile the C code. The supported compiler optimization levels are limited to –O1 and –O2, with limited compiler flag options. In this study, the –O1 optimization level is used.

B. QP formulation

Due to the limited hardware resources and runtime support on the PLC (compared to a PC), a custom light-weight QP solver is implemented to solve the control problem. To achieve this, the SEPTIC configuration is first translated into a sparse QP problem formulation, including slack variables for the soft constraints. An automatic code generator tailored for SEPTIC, was developed for this purpose in [18] (referred to as the eMPC code generator in the sequel). It extracts information from the internal data structures in SEPTIC and generates the QP formulation based on the existing SEPTIC configuration. and also provides some general functionality for an embedded MPC implementation.

C. The QP solver

As mentioned in the introduction, many code generators are now available to generate fast and efficient QP-solvers, with modest memory requirements, making embedded implementations practically feasible. In this study, a preliminary release of the MATLAB toolbox FiOrdOs[1] has been used to generate the custom solver. The generated solver is implemented in ANSI C code, and solves the parametric QP problem:

$$\min_{x \in \mathbb{X}} \frac{1}{2} x^T H x + g^T x \tag{7a}$$

$$\text{subject to: } A_i x \leq b_i,\ A_e x = b_e \tag{7b}$$

The solver is based on the primal-dual first-order method proposed in [19], originally developed for image processing applications. Unlike the other methods in FiOrdOs, which require a positive definite Hessian H, this method allows a positive *semi*-definite Hessian, which is the case for the QP problem generated by the eMPC code generator. Promising results using this method for embedded MPC has been shown in [20].

For our application, the decision variable vector x contains predictions of future inputs and measurements, and the slack variables. The current state of the controlled system and previous control actions enter the QP problem in the vector b_e, which makes this parametric. The other matrices and vectors are constant, and are constructed using the problem formulation generated by the eMPC code generator, as in [20] where more details are provided. To achieve this efficiently, the eMPC code generator has been modified to also generate the problem formulation in MATLAB format, and a MATLAB script has been written to automatically construct the necessary matrices and vectors, and generate the solver using the FiOrdOs code generator.

D. Problem size

The QP problem size depends on the number of inputs, outputs and control targets (introducing slack variables and constraints), as well as controller configuration parameters, such as prediction horizon, input blocking for the MVs and evaluation points for the CVs [2], [16]. In this application, each of the MVs have 5 blocks, each CV have 8 or 9 evaluation points, and there are 7 soft constraints. The generated problem has 60 decision variables ($x \in \mathbb{R}^{60}$, $H \in \mathbb{R}^{60\times60}$, $g \in \mathbb{R}^{60}$), 105 inequality constraints ($A_i \in \mathbb{R}^{105\times60}$, $b_i \in \mathbb{R}^{105}$) and 43 equality constraints ($A_e \in \mathbb{R}^{43\times60}$, $b_e \in \mathbb{R}^{43}$).

[1]See http://fiordos.ethz.ch

E. PLC implementation

The eMPC code generator also provides a C code framework for the embedded MPC implementation. This framework, combined with the QP solver generated by FiOrdOs, forms the basis for the embedded implementation. Additional functionality, such as warm-starting the solver and OPC communication via Ethernet, is also implemented. In order to reduce the computational load, the explicit priorities which may be defined in SEPTIC (see Section III-D) are not implemented in the embedded controller.

V. SIMULATION RESULTS

In this section, hardware-in-the-loop simulation results are presented. The performance of the embedded controller is compared to the original SEPTIC MPC, which is considered the benchmark controller.

A. Hardware setup

The embedded controller is implemented on the PLC, and communication with the ESP simulator is established via Ethernet and an OPC server. SEPTIC is run on the same PC as the simulator, but there is no interaction except via the OPC server.

B. Benchmark scenario

To test and evaluate the performance of the controller, a benchmark simulation scenario is defined, i.e. a predefined sequence of measurements of the topside pressure, including measurement noise. To ensure comparable simulation results, the other measurements are noise-free. Thus, the scenario is exactly the same in all simulations presented in this section, the only difference is the controller.

The scenario starts at steady-state conditions, with a topside pressure of 20 bar, and the ESP controlled manually. The controller is activated after 1 minute, and is active for the remainder of the scenario. After 2 minutes, the topside pressure is increasing to 28 bar. After 4 minutes, it decreases to 12 bar. After 6 minutes, it is slowly increasing until it again reaches 20 bar.

C. Original results

Since explicit priorities are not implemented in the embedded controller, only the weights are used. Thus, the embedded controller ignoring the priorities of the SEPTIC configuration in table I will not provide the same performance as SEPTIC. As expected, hardware-in-the-loop simulations using this configuration show a poor performance of the embedded controller. E.g., tracking of the (low-priority) setpoint on $p_{p,in}$ is improved compared to SEPTIC, on the expense of large violations of the (higher-priority) low limit on q_0.

The simulations are shown in Fig. 3a and 3b. The topside pressure, defining the scenario, is given in the top plot. The next four plots are measurements of the controlled variables I, q_0, Δp_c and $p_{p,in}$, respectively. The last two plots are the pump frequency f and the choke opening z, which are the controller inputs. The constraints are also plotted (dashed lines), and the setpoint on $p_{p,in}$ (dash-dotted line at 50 bar).

TABLE II
MPC CONTROLLER CONFIGURATION WITHOUT PRIORITIES

Var.	Unit	Low	Setp.	High	W
I	A			65	200
q_0	m^3/h	53.2		80.7	50
Δp_c	bar	1			25
$p_{p,in}$	bar	40		70	1
$p_{p,in}$	bar		50		2
I	A		0		0.01
q_0	m^3/h		65.9		0.05
Δp_c	bar			10	1
f	Hz	35		65	0.02
z	$\%$	0		100	0.005

D. Implementation without priorities

To achieve acceptable performance of the embedded controller, SEPTIC is reconfigured without explicit priorities, and the desired performance is achieved through proper tuning of the weights and penalties. The new configuration is shown in Table II. Most noticeable is the greatly reduced weights on the setpoints for I and q_0, and the move penalty on the inputs.

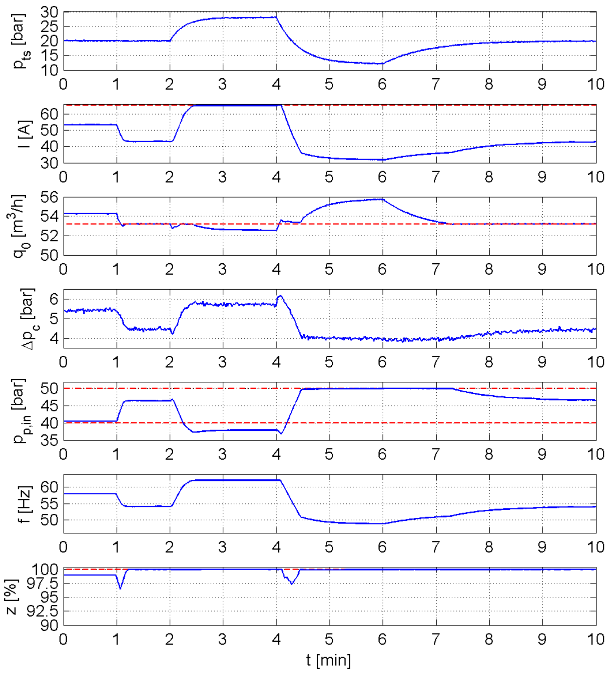
Hardware-in-the-loop simulations with this configuration are shown in Fig. 3c and 3d. The performance of the embedded controller is quite comparable to the performance of the original SEPTIC implementation. However, there are some differences. Most noticeable is the use of the control input z (production choke valve) between 4 and 7 minutes in the embedded implementation. The SEPTIC MPC keeps the choke fully open in this interval, while the choke is only about 90 percent open in the embedded implementation. This, however, has little effect on the performance. The mean power consumption (electric current, I) is only slightly increased compared to the original SEPTIC implementation, from 44.42 Ampere to 44.86 Ampere, a 1 percent increase (see Section V-E).

E. Control performance

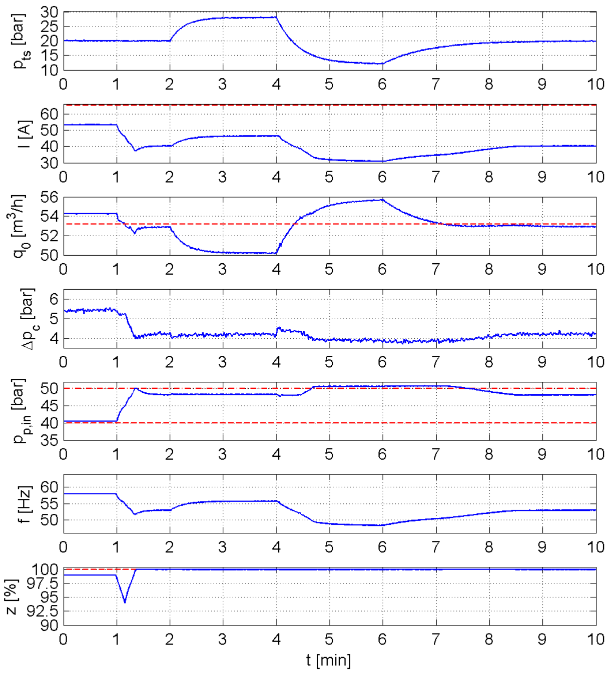
For comparison of controller performance, the error (constraint violation or deviation from setpoint) on certain control targets are presented in Table III. The errors are calculated for different time intervals, stated in the table below each control target (in minutes).

F. Computation time and memory

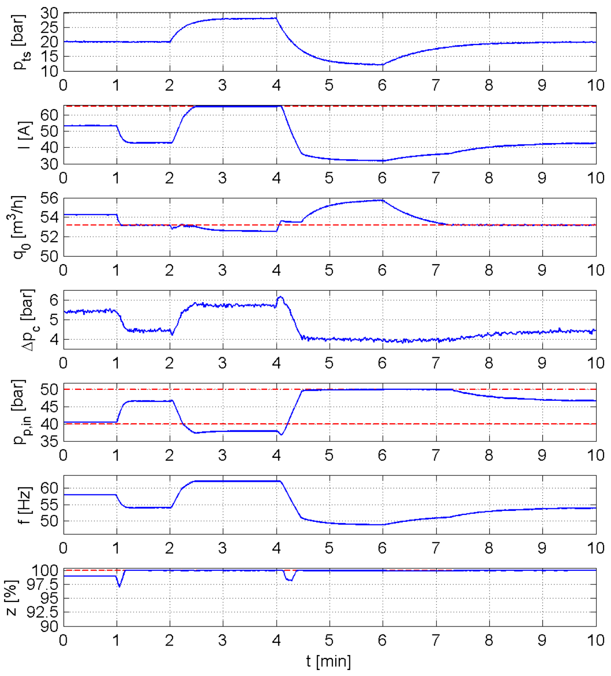
In the simulations, the average computational time of the embedded controller was 124.09 ms, which is well within the required sampling rate of 1 Hz. The FiOrdOs generated solver has a small footprint. The program code size downloaded to the PLC was 179kB ($<4\%$ of its capacity), and the program data size 108kB ($\sim 2\%$). As the number of iterations in the solver is fixed (100 iterations was used in the simulations presented here), the computational time of the solver is also very predictable, which is crucial in real-time applications. The worst-case computational time in this implementation was measured to be 124.89 ms, which is only 0.64% above the average time.



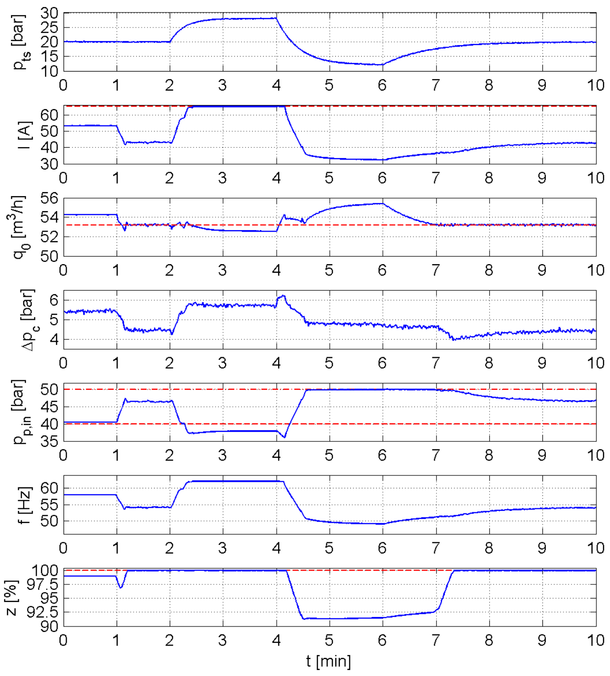
(a) SEPTIC (original)



(b) Embedded (original)



(c) SEPTIC (no priorities)



(d) Embedded (no priorities)

Fig. 3. Simulation results

TABLE III
PERFORMANCE

Target		**Original**		**No priorities**	
(Interval)	Error	SEPTIC	PLC	SEPTIC	PLC
I **high**	Max	0.1601	0	0.1625	0.1238
(2-4.5)	Mean	4.187e-3	0	4.854e-3	6.035e-3
	MSE	5.658e-4	0	5.821e-4	3.192e-4
q_0 **low**	Max	0.6623	3.0628	0.6620	0.6644
(1-4.5)	Mean	0.2749	1.7477	0.2780	0.2751
	MSE	0.1530	4.5379	0.1521	0.1531
$p_{p,in}$ **low**	Max	3.1723	0	3.1725	4.0151
(2-4.5)	Mean	1.6514	0	1.6475	1.7627
	MSE	3.6673	0	3.6500	4.1390
$p_{p,in}$ **setp**	Mean	1.7892	1.0962	1.7876	1.9679
(4-10)	MSE	9.6471	1.5508	9.7950	11.7728
I **setp**	Min	31.8312	30.8984	31.8380	32.3683
(1-10)	Mean	44.4247	38.9517	44.4056	44.8572

VI. CONCLUSION

In this paper, a feasible approach to implement model predictive control on industry-standard ultra-reliable embedded hardware was outlined. The use of established industrial MPC software tools for controller design, in combination with automatic code generators, significantly reduced the effort required for the implementation. This could be further improved by including more functionality in automatic code generators.

Some modifications to the original controller design were needed to achieve acceptable performance of the embedded controller, as explicit priorities are not included in the embedded implementation. Instead, the desired performance was achieved through tuning of weights and penalties on soft constraints. As seen in the simulation results, a performance quite comparable to the original results was achieved, however, realising all the specifications for all combinations of the control targets without the explicit priorities required more tuning effort.

A fast sampling rate (up to 8 Hz) was achievable for the system considered in this study. The control problem size (number of variables and constraints) was relatively small, but the results imply that this approach is also feasible for larger control problems. The number of iterations in the solver could be increased to improve the controller performance, or reduced to reduce the computation time.

ACKNOWLEDGMENTS

We thank Gisle Otto Eikrem and Morten Fredriksen at Statoil, for making SEPTIC and the eMPC code generator availalable for this study, and support regarding implementation. We also thank Stefan Richter for making the preliminary release of the FiOrdOs toolbox, including the new primal-dual first-order method, available for our research.

REFERENCES

[1] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Eng. Practice*, vol. 11, no. 7, pp. 733–764, 2003.

[2] J. M. Maciejowski, *Predictive Control: with Constraints*. Pearson and Prentice Hall, 2002.

[3] G. Valencia-Palomo and J. A. Rossiter, "Efficient suboptimal parametric solutions to predictive control for PLC applications," *Control Eng. Practice*, vol. 19, no. 7, pp. 732–743, 2011.

[4] B. Huyck, H. J. Ferreau, M. Diehl, J. D. Brabanter, J. F. M. V. Impe, B. D. Moor, and F. Logist, "Towards online model predictive control on a programmable logic controller: Practical considerations," *Math. Problems in Eng.*, vol. 2012, pp. 1–20, 2012.

[5] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Trans. Control Syst. Technol.*, vol. 18, no. 2, pp. 267–278, 2010.

[6] A. Domahidi, A. Zgraggen, M. Zeilinger, M. Morari, and C. Jones, "Efficient interior point methods for multistage problems arising in receding horizon control," in *Proc. 51st IEEE Conf. Decision and Control (CDC 2012)*, Maui, HI, USA, Dec. 2012, pp. 668 – 674.

[7] H. J. Ferreau, H. G. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit MPC," *Int. J. Robust Nonlinear Control*, vol. 18, pp. 816–830, July 2008.

[8] F. Ullmann, "A Matlab toolbox for C-code generation for first order methods," M.S. thesis, Eidgenössische Technische Hochschule, Zürich, Switzerland, 2011.

[9] J. Mattingley and S. Boyd, "CVXGEN: A code generator for embedded convex optimization," *Optimization and Eng.*, vol. 13, no. 1, pp. 1–27, 2012.

[10] A. Al-Jasmi, H. Nasr, H. K. Goel, G. Moricca, G. Carvajal, J. Dhar, M. Querales, M. Villamizar, A. Cullick, J. Rodriguez, G. Velasquez, Z. Yong, F. Bermudez, and J. Kain, "ESP "Smart Flow" Integrates Quality and Control Data for Diagnostics and Optimization in Real Time," in *SPE Digital Energy Conf.* The Woodlands, TX, USA: Soc. of Petroleum Engineers, 2013.

[11] R. Sharma and B. Glemmestad, "Optimal control strategies with nonlinear optimization for an electric submersible pump lifted oil field," *Modeling, Identification and Control*, vol. 34, no. 2, pp. 55–67, 2013.

[12] A. Pavlov and V. Alstad, "Modelling, simulation and automatic control of ESP lifted wells," Statoil ASA, Norway, Tech. Rep., 2010.

[13] A. Pavlov, D. Krishnamoorthy, K. Fjalestad, E. Aske, and M. Fredriksen, "Modelling and model predictive control of oil wells with electric submersible pumps," in *Proc. 2014 IEEE Conf. Control Applications (CCA 2014)*, Antibes, France, Oct. 2014.

[14] R. Sharma and B. Glemmestad, "Modeling and simulation of an electric submersible pump lifted oil field," *Int. J. Petroleum Sci. and Technol.*, vol. 8, no. 1, pp. 39–68, 2014.

[15] G. Takacs, *Electrical Submersible Pumps Manual: Design, Operations, and Maintenance*, ser. Gulf Equipment Guides. Gulf Professional Publishing, March 2009.

[16] S. Strand and J. R. Sagli, "MPC in Statoil – advantages with in-house technology," in *Int. Symp. Advanced Control Chemical Processes (ADCHEM)*, Hong Kong, 2003, pp. 97–103.

[17] ABB Automation products AC500, CP400, CP600, DigiVis 500, Wireless. [Online]. Available: www.abb.com

[18] V. Aaker, "Embedded MPC for a subsea separation process," M.S. thesis, Dept. Eng. Cybern., Norwegian Univ. Sci. and Technol. (NTNU), Trondheim, Norway, 2012.

[19] T. Pock and A. Chambolle, "Diagonal preconditioning for first order primal-dual algorithms in convex optimization," in *Proc. 2011 IEEE Int. Conf. Comput. Vision (ICCV 2011)*, Barcelona, Spain, Nov. 2011, pp. 1762–1769.

[20] D. K. M. Kufoalor, S. Richter, L. Imsland, T. A. Johansen, M. Morari, and G. O. Eikrem, "Embedded model predictive control on a PLC using a primal-dual first-order method for a subsea separation process," in *Proc. 22nd IEEE Mediterranean Conf. Control and Automation (MED 2014)*, Palermo, Italy, 2014.