ETH Library

When FPGAs Meet Regionless Explicit MPC: An Implementation of Long-horizon Linear MPC for Power Electronic Systems

Conference Paper

Author(s):
Jeong, Min; Fuchs, Simon; Biela, Jürgen

Publication date:
2020

Permanent link:
https://doi.org/10.3929/ethz-b-000498634

Rights / license:
In Copyright - Non-Commercial Use Permitted

Originally published in:
https://doi.org/10.1109/iecon43393.2020.9254277

This page was generated automatically upon download from the ETH Zurich Research Collection.
For more information, please consult the Terms of use.

When FPGAs Meet Regionless Explicit MPC: An Implementation of Long-horizon Linear MPC for Power Electronic Systems

Min Jeong
High Power Electronic Systems (HPE)
ETH Zürich
jeong@hpe.ee.ethz.ch

Simon Fuchs
High Power Electronic Systems (HPE)
ETH Zürich
fuchs@hpe.ee.ethz.ch

Jürgen Biela
High Power Electronic Systems (HPE)
ETH Zürich
jbiela@ethz.ch

***Abstract*—This paper presents a novel real-time implementation of linear model predictive control (MPC) schemes based on region-less explicit MPC (RL-EMPC). RL-EMPC is a recently proposed explicit MPC solution for achieving a low memory footprint compared to other explicit MPC methods. Thus RL-EMPC can effectively handle systems with high sampling rates even for large-size MPC problems, i.e. systems with many states and/or constraints. An architecture for an implementation on an FPGA device is presented and validated with two application examples (buck-boost converter, modular multilevel converter). Simulation results on FPGA hardware-level demonstrate that long-horizon linear MPC can be implemented on low-cost FPGAs with the proposed architecture for various power electronic systems.**

***Index Terms*—continuous control set (CCS) MPC, modulated MPC, modular multilevel converter (MMC, M2C), Buck-Boost-converter**

I. Introduction

Optimal design procedures for power electronic systems are usually based on the steady-state behaviour (losses, maximum voltages/currents, voltage/current ripples, etc) of the converters. There, margins for dynamic control that are added on top of the steady-state system trajectories can significantly decrease the power density of the converter. Consequently, reducing or even omitting these margins can be a crucial factor to increase the power density and thereby reduce the converter volume, weight, and cost without a reduction in efficiency. However, this is not always possible with classical control methods (e.g. cascaded PI-controllers), because most control techniques do not take the system constraints into account. Therefore, both academia and industry have drawn attention to model predictive control (MPC) for power electronic systems since the system constraints can be regarded with MPC [1], [2]. As a result, power converters can be designed optimally while still exploiting the full dynamic potential to achieve the fastest possible response to reference changes and disturbances even for multi-input-multi-output (MIMO) systems. For example, in [3] MPC is used to fully exploit the installed energy storage of a modular multilevel converter (MMC) resulting in savings of about $40\,\%$ of the module capacitance value, while preserving excellent dynamic performance.

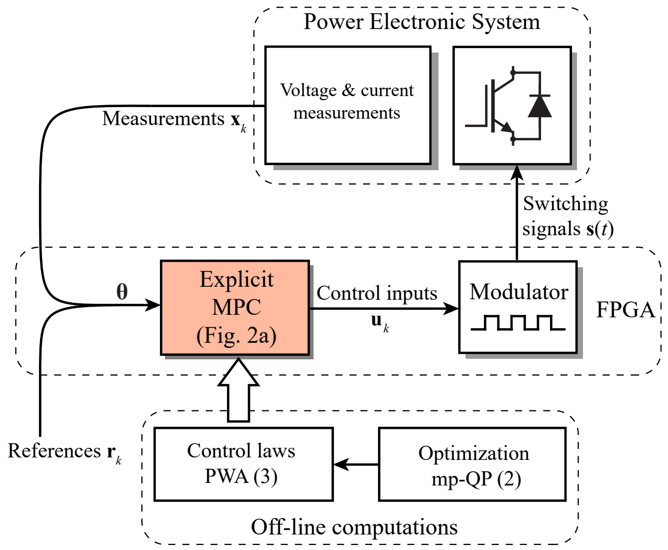

Power Electronic System
Voltage & current measurements
Measurements $\mathbf{x}_k$
Switching signals $\mathbf{s}(t)$
$\boldsymbol{\theta}$
Explicit MPC (Fig. 2a)
Control inputs $\mathbf{u}_k$
Modulator
FPGA
References $\mathbf{r}_k$
Control laws PWA (3)
Optimization mp-QP (2)
Off-line computations

Fig. 1. Typical block diagram of power electronic system with continuous control set (CCS) explicit model predictive controller (EMPC). The modulator represents a PWM or SVM modulation. The optimization solving the MPC problem is computed off-line. This paper deals with the FPGA implementation of the explicit MPC block marked in red for long horizon MPC for complex power electronic systems e.g. MMCs (cf. Sec. IV-B).

Recent advances of computational power in embedded systems further facilitate the utilization of MPC as a promising method for a wide range of converters. Nonetheless, the implementation of MPC for power electronic systems is still a challenging task as an optimization problem has to be solved within every sampling interval. This is especially problematic because power electronic applications often require a high sampling rate. Particularly for systems with tight constraints, MPC with a long prediction horizon is required to guarantee performance and stability [4]. Therefore, the computational burden of the optimization problem can be very demanding. Indirect MPC, also known as continuous control set (CCS) MPC, has recently attracted attention in such context. With CCS-MPC, the converter is modelled with continuous inputs such that the MPC problem results in a continuous-set optimization problem. This not only simplifies the underlying optimization problem to be solved in real-time, but also allows the system to operate at a fixed switching frequency as a modulator is employed to generate switching signals of power semiconductors. In case of a linear system representation for the prediction model (linear MPC), the underlying optimization problem usually results in a quadratic programming (QP) problem [2], for which approaches for realizing real-time implementations exist [5].

The methods for realizing the real-time implementation of CCS-MPC can be grouped into two sets: implicit MPC and explicit MPC. Implicit MPC solves the optimization problem on-line at each sampling instant, whereas explicit MPC (EMPC) handles the optimization process off-line. Therefore, EMPC accomplishes a simple on-line computation and can be implemented even on low-cost embedded systems at very fast sampling rates. Despite such characteristics make EMPC very suitable to realize MPC for power converters, EMPC has been applied to only a few applications [6], [7]. This is because memory requirements of general EMPC methods grow exponentially with the problem size, limiting the practical usage of the methods to only small problems.

Recently, the region-less explicit MPC (RL-EMPC) method is proposed in [8], [9] to overcome the main drawback of the memory requirements and it shows a possibility to handle large-size MPC problems with EMPC. However, only few implementation examples have been demonstrated yet and no attempts have been made for fast dynamic systems like power converters. Moreover, none of them investigates implementations on embedded systems nor analyzes the complexity in terms of resource usage and on-line evaluation time. Since RL-EMPC requires more on-line computations to achieve the low memory requirements, the on-line evaluation time, which is achievable with reasonable computational resources, is a critical factor to determine the feasibility of the method.

This paper presents an FPGA implementation of the RL-EMPC approach to accelerate on-line evaluation processes and to parallelize access to external memory with high-bandwidth. With the proposed implementation, the realization of EMPC can be applied to a broader spectrum of systems, such that it can even be used for power electronic systems with many states and/or a large number of constraints (which typically grow with the prediction horizon). Even though the proposed method requires more computational power compared to other EMPC methods, recent advances in FPGA technology enable the implementation on low-cost FPGA devices. The FPGA implementation is validated with two application examples, and simulation results on FPGA hardware-level are demonstrated along with their resource usages and required clock cycles for the on-line evaluation.

The paper is organized as follows: the general concept of explicit MPC is reviewed in section II to compare different implementation approaches. Section III presents an architecture for implementing RL-EMPC on FPGAs. Finally, in section IV, RTL level simulation results for two power electronic applications are demonstrated and analyzed before concluding in section V.

II. Explicit Model Predictive Control

The concept of explicit model predictive control (EMPC) is proposed in [10] based on multi-parametric quadratic programming (mp-QP) to address linear MPC problems. Nonetheless, there are several ways to implement EMPC and each method has different features, such as optimality of control laws, latency, memory requirement, and scalability. Therefore, in this section, a formulation of linear MPC and its relation to mp-QP to pre-compute optimal control laws off-line is reviewed, followed by representative approaches to implement the on-line part of EMPC.

A. General Formulation of Linear MPC

MPC computes an optimal control input of a constrained control problem by solving an optimization problem. When MPC is applied to linear systems with linear constraints, i.e., box-constraints and/or polytopic constraints, the optimization problem can be written as

$$\min_{\mathbf{U}_k} \sum_{l=0}^{N_\mathrm{p}-1} \left\|\mathbf{x}_{k+l+1} - \mathbf{x}_{\mathrm{ref},k+l+1}\right\|_{\mathbf{Q}}^2 + \left\|\mathbf{u}_{k+l} - \mathbf{u}_{\mathrm{ref},k+l}\right\|_{\mathbf{R}}^2 \tag{1a}$$

$$\begin{aligned} s.t.\;\; & \mathbf{x}_{k+l+1} = \mathbf{A}_{k+l}\mathbf{x}_{k+l} + \mathbf{B}_{k+l}\mathbf{u}_{k+l}, && \forall l \in \mathcal{I} && \text{(1b)}\\ & \mathbf{u}_{\min} \le \;\; \mathbf{u}_{k+l} \;\; \le \mathbf{u}_{\max}, && \forall l \in \mathcal{I} && \text{(1c)}\\ & \mathbf{x}_{\min} \le \;\; \mathbf{x}_{k+l+1} \;\; \le \mathbf{x}_{\max}, && \forall l \in \mathcal{I} && \text{(1d)}\\ & \mathbf{G}_\mathrm{p}\mathbf{U}_k \le \mathbf{w}_\mathrm{p} + \mathbf{K}_\mathrm{p}\mathbf{X}_k, && && \text{(1e)} \end{aligned}$$

where $\mathbf{U}_k = [\mathbf{u}_k^\mathrm{T}, \;\cdots\;, \mathbf{u}_{k+N_\mathrm{p}-1}^\mathrm{T}]^\mathrm{T} \in \mathbb{R}^{m \cdot N_\mathrm{p}}$ is the complete control input vector, $\mathbf{X}_k = [\mathbf{x}_k^\mathrm{T}, \;\cdots\;, \mathbf{x}_{k+N_\mathrm{p}}^\mathrm{T}]^\mathrm{T} \in \mathbb{R}^{n \cdot (N_\mathrm{p}+1)}$ is the complete state vector, N_p is the prediction horizon, $\mathbf{Q} \ge 0$ and $\mathbf{P} \ge 0$ are weighting matrices, $\mathcal{I} = \{0, 1, \cdots, N_\mathrm{p} - 1\}$, and $\left\|\mathbf{z}\right\|_{\mathbf{Q}}^2$ denotes a 2-norm with the weighting matrix $\mathbf{Q}$. $\mathbf{x}_{\mathrm{ref},k+l+1}$ and $\mathbf{u}_{\mathrm{ref},k+l}$ are the l-th step reference values for the states and inputs, and $\mathbf{A}_{k+l}$ and $\mathbf{B}_{k+l}$ are the l-th discrete state space matrices for general linear systems. Note that the formulation can be applied to time-varying systems, such as periodic systems [3], if discrete state-space models can be achieved.

B. Multi-parametric Quadratic Programming

The given linear MPC can be reformulated as a condensed quadratic programming (QP) problem by substituting the future states as a function of the future control inputs and the current states ($\mathbf{X}_\mathrm{k} = \mathbf{S}_\mathrm{k}\mathbf{U}_\mathrm{k} + \mathbf{T}_\mathrm{k}\mathbf{x}_\mathrm{k}$) based on the dynamic model of the linear system [4]. Furthermore, the optimization problem can be converted to a general mp-QP by choosing parameters, denoted as θ, such that the optimization problem can be solved over the domain of the parameters. For simplicity, subscript k is omitted in the following, and the multi-parametric optimization can be written as

$$\min_{\mathbf{U}} \;\; \frac{1}{2}\mathbf{U}^\mathrm{T}\mathbf{H}\mathbf{U} + (\mathbf{F}\theta + \mathbf{f})^\mathrm{T}\mathbf{U} \tag{2a}$$

$$s.t.\;\; \mathbf{G}\mathbf{U} \le \mathbf{w} + \mathbf{K}\theta. \tag{2b}$$

The size of the optimization parameters, $\theta \in \mathbb{R}^p$, depends on the number of variables that need to be updated at each sampling time to solve the linear MPC. For example, in trajectory tracking problems, the references ($\mathbf{X}_{\text{ref}}$, $\mathbf{U}_{\text{ref}}$) can be included into θ as well as the current system states ($\mathbf{x}_{\text{k}}$) to track varying references. However, the size of the parameters is one of the critical factors that determine the complexity of solving the mp-QP problem. Therefore, if possible, new representative variables like the power level of the system should be included in the parameters to represent such varying reference trajectories with a smaller number of parameters. This could dramatically reduce the complexity of the mp-QP problem.

The process of solving the mp-QP problem is conducted off-line and results in a function of the optimization parameters (θ). This function is in the form of a piecewise affine (PWA) function, as follows:

$$\mathbf{U}^* = \begin{cases} f_1(\theta) & \text{if } \theta \in \mathcal{D}_1 \\ \vdots & \\ f_R(\theta) & \text{if } \theta \in \mathcal{D}_R \end{cases} \tag{3}$$

where $\mathcal{D}_i$ are R nonoverlapping regions, i.e., $\mathcal{D}_i \cap \mathcal{D}_j = \emptyset$ for $i \neq j$. Each region has a different set of active constraints and applies a different affine law (f_i) [10]. There are few algorithms to construct the regions and they differ in approaches to explore what are possible combinations of active constraints: geometric methodologies, combinatorial methods, and connected-graph approaches [11]. In the context of MPC, such an mp-QP solution is equivalent to a feedback control law and denoted as an explicit MPC solution.

Consider the set of active constraints of the i-th region $\mathcal{A}_i \subseteq \{1, \cdots, n_{\text{c}}\}$, where n_{c} is the number of rows of the constraint matrix $\mathbf{G}$ in (2b). Then, the constraint equations can be divided into two groups,

$$\mathbf{G}\mathbf{U} \leq \mathbf{w} + \mathbf{K}\theta \quad \Rightarrow \quad \begin{cases} \mathbf{G}_{\mathcal{A}_i}\mathbf{U} = \mathbf{w}_{\mathcal{A}_i} + \mathbf{K}_{\mathcal{A}_i}\theta \\ \mathbf{G}_{\mathcal{N}_i}\mathbf{U} < \mathbf{w}_{\mathcal{N}_i} + \mathbf{K}_{\mathcal{N}_i}\theta \end{cases},$$

where $\mathbf{G}_{\mathcal{A}_i}$ is obtained by stacking the rows of $\mathbf{G}$ indexed by $\mathcal{A}_i$ and $\mathbf{G}_{\mathcal{N}_i}$ is the rest of the rows. The relations between the set of active constraints and the explicit MPC solution can be demonstrated by applying the Karush-Kuhn-Tucker (KKT) conditions to (2), such that

$$\mathbf{H}\mathbf{U}^* + (\mathbf{F}\theta + \mathbf{f}) + \mathbf{G}_{\mathcal{A}_i}^{\text{T}}\boldsymbol{\lambda}^* + \mathbf{G}_{\mathcal{N}_i}^{\text{T}}\boldsymbol{\mu}^* = 0, \tag{5a}$$
$$\mathbf{G}_{\mathcal{A}_i}\mathbf{U}^* = \mathbf{w}_{\mathcal{A}_i} + \mathbf{K}_{\mathcal{A}_i}\theta, \tag{5b}$$
$$\mathbf{G}_{\mathcal{N}_i}\mathbf{U}^* < \mathbf{w}_{\mathcal{N}_i} + \mathbf{K}_{\mathcal{N}_i}\theta, \tag{5c}$$
$$\boldsymbol{\lambda}^*,\ \boldsymbol{\mu}^* \geq 0, \tag{5d}$$
$$\boldsymbol{\lambda}^{*\text{T}}(\mathbf{G}_{\mathcal{A}_i}\mathbf{U}^* - \mathbf{w}_{\mathcal{A}_i} - \mathbf{K}_{\mathcal{A}_i}\theta) = 0, \tag{5e}$$
$$\boldsymbol{\mu}^{*\text{T}}(\mathbf{G}_{\mathcal{N}_i}\mathbf{U}^* - \mathbf{w}_{\mathcal{N}_i} - \mathbf{K}_{\mathcal{N}_i}\theta) = 0. \tag{5f}$$

Equation (5c) and (5f) lead to $\boldsymbol{\mu}^* = 0$, and (5a) as well as (5b) can be rewritten as

$$\mathbf{U}^* = -\mathbf{H}^{-1} \cdot (\mathbf{F}\theta + \mathbf{f} + \mathbf{G}_{\mathcal{A}_i}^{\text{T}}\boldsymbol{\lambda}^*), \tag{6a}$$
$$\boldsymbol{\lambda}^* = \mathbf{Q}(\mathcal{A}_i)\theta + \mathbf{q}(\mathcal{A}_i), \tag{6b}$$

where

$$\mathbf{Q}(\mathcal{A}_i) = -(\mathbf{G}_{\mathcal{A}_i}\mathbf{H}^{-1} \cdot \mathbf{G}_{\mathcal{A}_i}^{\text{T}})^{-1} \cdot (\mathbf{K}_{\mathcal{A}_i} + \mathbf{G}_{\mathcal{A}_i}\mathbf{H}^{-1}\mathbf{F}),$$
$$\mathbf{q}(\mathcal{A}_i) = -(\mathbf{G}_{\mathcal{A}_i}\mathbf{H}^{-1} \cdot \mathbf{G}_{\mathcal{A}_i}^{\text{T}})^{-1} \cdot (\mathbf{w}_{\mathcal{A}_i} + \mathbf{G}_{\mathcal{A}_i}\mathbf{H}^{-1}\mathbf{f}).$$

As a result, the affine law (f_i) is calculated by (6), and the region ($\mathcal{D}_i$) is defined by (5c) and (5d). The main differences between various implementation strategies result from which matrices are stored in memory and which computations are performed on-line to identify the proper region and compute the optimal control law.

C. Implementation Strategies of Explicit MPC

1) Region-based Method: The most straightforward method to apply EMPC is region-based EMPC, and both affine laws and regions are computed thoroughly off-line in a direct affine function of the optimization parameters (θ). The affine laws are obtained as

$$f_i = \mathbf{T}(\mathcal{A}_i)\theta + \mathbf{t}(\mathcal{A}_i), \tag{8}$$

where $\mathbf{T}(\mathcal{A}_i)$ and $\mathbf{t}(\mathcal{A}_i)$ are achieved by substituting (6b) into (6a) as

$$\mathbf{T}(\mathcal{A}_i) = -\mathbf{H}^{-1} \cdot (\mathbf{F} + \mathbf{G}_{\mathcal{A}_i}^{\text{T}}\mathbf{Q}(\mathcal{A}_i)), \tag{9a}$$
$$\mathbf{t}(\mathcal{A}_i) = -\mathbf{H}^{-1} \cdot (\mathbf{f} + \mathbf{G}_{\mathcal{A}_i}^{\text{T}}\mathbf{q}(\mathcal{A}_i)). \tag{9b}$$

The regions are achieved by inserting (6b) and (8) into (5c) and (5d), resulting in:

$$\underbrace{\begin{bmatrix} \mathbf{G}_{\mathcal{N}_i}\mathbf{T}(\mathcal{A}_i) - \mathbf{K}_{\mathcal{N}_i} \\ -\mathbf{Q}(\mathcal{A}_i) \end{bmatrix}}_{\mathbf{A}(\mathcal{A}_i)} \cdot \theta \leq \underbrace{\begin{bmatrix} -\mathbf{G}_{\mathcal{N}_i}\mathbf{t}(\mathcal{A}_i) + \mathbf{w}_{\mathcal{N}_i} \\ \mathbf{q}(\mathcal{A}_i) \end{bmatrix}}_{\mathbf{b}(\mathcal{A}_i)}, \tag{10}$$

where $\mathbf{A}(\mathcal{A}_i)$ and $\mathbf{b}(\mathcal{A}_i)$ form a polyhedral, known also as a critical region. Consequently, the region-based method stores the matrices of $\mathbf{T}(\mathcal{A}_i)$, $\mathbf{t}(\mathcal{A}_i)$, $\mathbf{A}(\mathcal{A}_i)$, and $\mathbf{B}(\mathcal{A}_i)$, and is realized by checking constraints sequentially to evaluate an optimal affine control law as

$$\mathbf{U}^* = \begin{cases} \mathbf{T}(\mathcal{A}_1)\theta + \mathbf{t}(\mathcal{A}_1) & \text{if } \mathbf{A}(\mathcal{A}_1)\theta \leq \mathbf{b}(\mathcal{A}_1) \\ \vdots & \vdots \\ \mathbf{T}(\mathcal{A}_R)\theta + \mathbf{t}(\mathcal{A}_R) & \text{if } \mathbf{A}(\mathcal{A}_R)\theta \leq \mathbf{b}(\mathcal{A}_R) \end{cases}. \tag{11}$$

2) Variations of Region-based Method: The sequential search of the region-based EMPC method is intuitive, yet it is not regarded as the most efficient strategy. As can be seen from the structure of the matrices in (10), the size of each matrix is equivalent to the number of all constraints. Even after removing redundant rows to minimize the size of $\mathbf{A}(\mathcal{A}_i)$ and $\mathbf{b}(\mathcal{A}_i)$, the sequential search method results in mainly two problems. First, the worst-case search time can become very long since all constraints should be checked for all regions. Second, the memory requirements for storing all matrices can be demanding.

Many variations have been proposed to improve such drawbacks, for example, the binary search tree method [12] for shorter search time as well as low-complexity methods

[13] for lower memory requirements. Refer to [14] for a comprehensive comparison between different methods. However, most methods were developed targeting small size problems, and the complexity increases exponentially when the number of constraints and/or the dimension of the parameters becomes large.

3) Region-less Explicit MPC: The region-less method was proposed mainly to reduce the burden of memory requirements at the cost of more demanding on-line computations [8]. Instead of saving all matrices, given in (10), the region-less method only saves fundamental blocks from the optimization problem itself, e.g., $\mathbf{H}^{-1}$, $\mathbf{F}$, $\mathbf{f}$, $\mathbf{G}$, $\mathbf{w}$, and $\mathbf{K}$. These blocks can be re-utilized at all regions to check the region conditions with (5c) and (5d) and evaluate the affine law with (6) through on-line computations. Only matrices of duality conditions, $\mathbf{Q}(\mathcal{A}_i)$ and $\mathbf{q}(\mathcal{A}_i)$, are required for each region, thus a huge reduction of memory requirements can be achieved. Note that the notion of "region", where same active constraints are shared, still exists in the region-less method, yet it is named as "region-less" because it does not require matrices of "critical regions" as given in (10).

The biggest advantage of the region-less method is that it is a general solution to shrink the memory requirements regardless of the number of constraints or the dimension of parameters. Therefore, in [9], an implementation of explicit MPC for a large problem is demonstrated, where the parameter dimension is $\theta \in \mathbb{R}^{13}$ and the number of regions is $R = 1095$.

III. FPGA Structure for Region-less Explicit MPC

Based on the region-based method and its variations, which are reviewed in the previous section, many implementations are introduced with detailed embedded architectures (cf. [14], [15]). For both optimal and sub-optimal solutions, extensive research has been conducted to realize the explicit controllers focusing on diverse aspects, from optimality and latency to required hardware resources. Moreover, toolboxes, e.g., MOBY-DIC [16], enabled the automatical generation of VHDL files to realize the efficient implementation of EMPC on FPGAs. However, in most cases, the implementations were applied in limited cases, where the dimension of the parameters and/or the number of the constraints is restricted. This is because a memory requirement is the bottleneck of the mentioned EMPC implementations.

As described in the previous section, the region-less MPC method is an intriguing method to tackle the bottleneck and realize EMPC for general linear MPC problems. Therefore, in this section, a detailed system structure is presented to implement RL-EMPC on FPGAs. First, functional blocks are introduced to parallelize the computations of the region-less method. Then, a timing structure for interfacing to external memory will be illustrated, followed by a method to calculate timing issues, such as system sampling frequency and worst-case latency.

Algorithm 1 Lagrangian-Multiplier check (LM in Fig. 2)

```
// for-loop can be paralleled at a functional level
// arithmetic operations can be paralleled at RTL level
for $i \in \{1, \dots, R\}$ do
    Compute $\boldsymbol{\lambda}_i = \mathbf{Q}(\mathcal{A}_i)\theta + \mathbf{q}(\mathcal{A}_i)$
    if $\boldsymbol{\lambda}_i \geq 0$ then
        Write $\boldsymbol{\lambda}_i$ and $\mathcal{A}_i$ to FIFO
    end if
end for
```

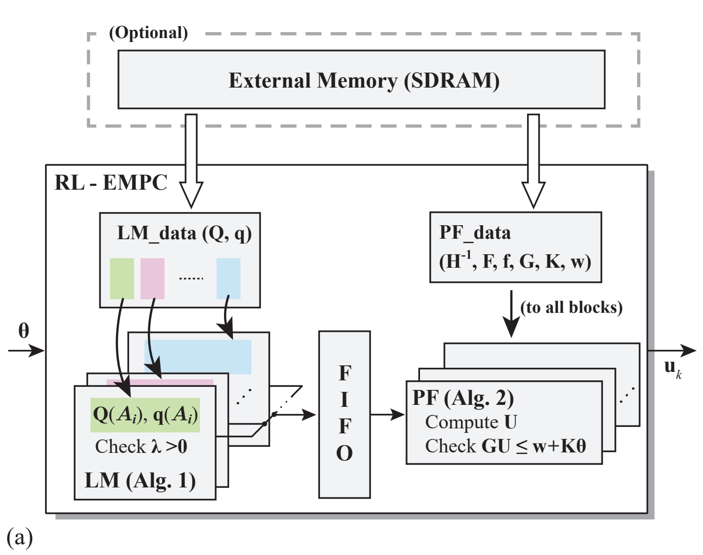
Algorithm 2 Primal feasibility check (PF in Fig. 2)

```
// while-loop can be paralleled at a functional level
// arithmetic operations can be paralleled at RTL level
while FIFO $\neq \emptyset$ do
    Read $\boldsymbol{\lambda}_i$ and $\mathcal{A}_i$ from FIFO
    Compute $\mathbf{U} = -\mathbf{H}^{-1} \cdot (\mathbf{F}\theta + \mathbf{f} + \mathbf{G}^{\mathrm{T}}_{\mathcal{A}_i}\boldsymbol{\lambda}_i)$
    if $\mathbf{G}\mathbf{U} \leq \mathbf{w} + \mathbf{K}\theta$ then
        $\mathbf{U}^* \leftarrow \mathbf{U}$
        $\mathbf{u}_k$ is the first $m$ entries of $\mathbf{U}^*$
        return $\mathbf{u}_k$
    end if
end while
```

A. Functional blocks and parallelism for FPGAs

In RL-EMPC, two functional blocks are required to evaluate the KKT conditions on-line: Lagrangian-multiplier (LM) checker and primal feasibility (PF) checker. The LM block, equivalent to (5c), examines whether the current parameter (θ) can satisfy a Lagrangian-multiplier condition (dual feasibility) at a given region, and the PF block, equivalent to (5d), checks whether all constraints are satisfied in the region with the calculated Lagrangian multiplier .

These two functional blocks can be computed in parallel to maximize the advantage of FPGAs using a first-in first-out (FIFO) block as proposed in Algorithm 1 and 2. While the PF checker is running for positive results of the LM checker, the LM checker can already proceed with the next region. Moreover, further parallelizations can be achieved as a trade-off between speed (latency) to compute control laws and resource usages on FPGAs. Two different levels of parallelism can be carried out. First, at a functional level, multiple functional blocks (LM and PF checkers) can be employed in parallel as shown in Fig. 2(a) to perform multiple LM and PF checks at the same time. Depending on applications, the complexity of LM and PF differs significantly, and thus the number of parallel blocks can be adapted flexibly to the applications and the required sampling rate of the systems. Note that the number of parallel LM checkers does not have to be same as the number of parallel PF checkers. Second, at a register-transfer level (RTL), each block can be realized with a different amount of resources. For example, one can parallelize multiplications to speed up the computations inside the LM and PF checkers depending on available resources on FPGAs. Proper pipelining can increase the efficiency of resources to realize blocks, but will also lead to longer computation times.



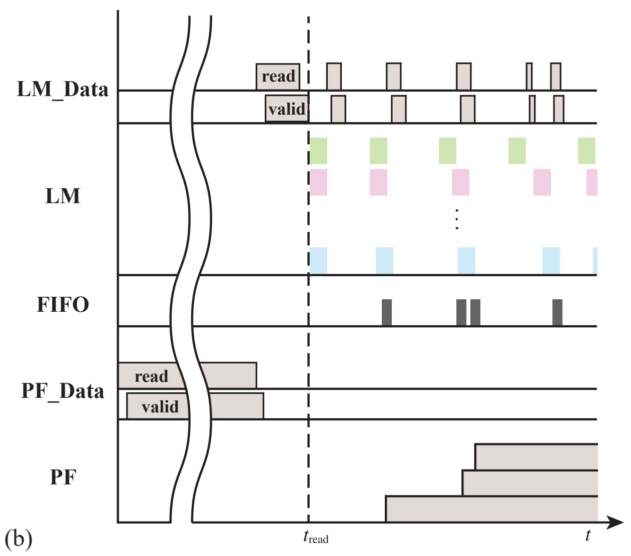


Fig. 2. (a) Block scheme of the proposed architecture for RL-EMPC. (b) Timing diagram of the proposed architecture with the external memory. The LM blocks start to check duality conditions in parallel after an initialization process of reading data (t_{read}). The PF blocks begins to check the constraint conditions when the FIFO block is filled.

B. Hardware Structures

For systems with a small number of parameters, RL-EMPC dramatically reduces the memory requirements - see section IV-A for detailed comparisons. Therefore, all data can be stored in on-chip memory blocks and the implementation can be carried out in a simple order. However, when the number of parameters becomes large or many constraints need to be applied to the system, embedded memory components cannot handle all data. Therefore, external memory, such as SDRAM, could be utilized to solve the problem as shown in Fig. 2(a). In general, the amount of data required for the PF blocks is small compared to the LM blocks. Therefore, when RL-EMPC is initialized, the PF blocks read the complete required information and save it into the memory blocks in the FPGA. Unlike that, the LM blocks require a large data, and only a share of the data is stored temporarily on-chip, and new data are requested whenever the LM blocks check a specific duality condition. Fig. 2(b) shows a timing diagram of the complete process to parallelize access to external memory. Note that even though the FPGAs can request and access the data from the external memory in parallel, the overall performance may be limited by the bandwidth of memory access when the sampling frequency of the system is very high.

C. System Sampling time and Worst-case Latency

One major advantage of explicit MPC is that the on-line computation time can be analyzed thoroughly, such that the system sampling rate can be decided based on the computation time of the worst-case scenario and the optimality of the control laws can be guaranteed. In general, the worst-case computation time can be calculated by summing the required time of three different functions: data read, LM blocks, and PF blocks. Even though the LM and the PF blocks operate in parallel in practice, for the worst-case computation time, it is assumed that all FIFO blocks might be inserted at the last part of LM checkers. The worst-case time heavily depends on the application, and the system sampling rate can be selected based on the worst-case time by adding additional timing margins for communication and modulator computation times.

IV. Application Examples

In order to demonstrate the effectiveness of the proposed RL-EMPC on FPGAs, this section compares EMPC solutions and presents simulation results for two power electronic applications. The first application is a buck-boost converter, where the system is relatively simple with 2 states and only a few constraints are applied. Nevertheless, since the required sampling rate is high ($\geq 100\,\text{kHz}$), the implementation of long-horizon MPC is not trivial especially when using low-cost FPGA devices. The second system is a modular multilevel converter, where the system model consists of 11 states and 6 inputs as well as many linear constraints. Such systems are usually regarded as unpractical/infeasible to apply EMPC.

For both systems, PWM is used to translate control problems with continuous inputs, and indirect MPC methods are applied to formulate linear MPC problems. After converting the linear MPC problems into mp-QP problems, the solutions of the mp-QP problems are computed with the Multi Parametric Toolbox (MPT) [17]. The proposed circuit architecture is developed with the DSP Builder for Intel FPGAs and implemented targeting low-cost FPGAs (e.g., Intel Max 10 or Cyclone V). The circuits are implemented in fixed-point using the fixed-point converter in Matlab. The HDL co-simulation was carried out in Matlab/Simulink with Mentor Graphics' Questa Sim to verify the generated HDL files in a closed loop manner.

A. Buck-Boost Converter

The non-inverting buck-boost converter is a simple, low cost, and efficient converter suitable for many DC-DC applications. However, due to its non-minimum phase characteristic, designing a high-bandwidth controller can be a troublesome task to fulfill the desired transient response. Applying MPC helps to significantly improve the control performance regarding disturbance rejection and reference tracking [6]. Here, the method described in [6] is utilized, and the same parameters are used to generate the presented EMPC solutions and simulation results.

The buck-boost converter has nonlinear dynamics, and proper modelling is a critical factor for the complexity of the MPC formulation. In [6], the converter is represented as a PWA model with 2 states and 1 input given as a function of the supply voltage as

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{A}_i \mathbf{x}_k + \mathbf{b}_i u_k + \mathbf{f} i_{\text{load}} \\ d_{2,k} = c_i \end{cases}, \quad \text{if } v_{\text{s},k} \in \mathbb{V}_i,$$

where $\mathbf{x} = [v_{\text{C}},\ i_{\text{L}}]^{\text{T}}$, $u = d_1 v_{\text{s}}$, $\mathbf{A}_i \in \mathbb{R}^{2\times2}$, $\mathbf{b}_i \in \mathbb{R}^{2\times1}$, $\mathbf{f} \in \mathbb{R}^{2\times1}$, and the supply-voltage range is divided into six intervals given as $\mathbb{V}_i$ to decide c_i, $i \in \{1, \cdots, 6\}$. By assuming that the duty-cycle of the boost stage (d_2) a piecewise-constant function, the nonlinear converter dynamics are translated into a linear model. Note that the supply voltage is assumed to stay in one interval throughout the prediction horizon, and thus d_2, $\mathbf{A}_i$, and $\mathbf{b}_i$ are all also assumed to be constant based on the value of $v_{\text{s},k}$ at each sampling instant. Based on the linear model, a linear MPC can be formulated with two box constraints on states, which limit the minimum and maximum values of the capacitor voltage (v_{C}) and the inductor current (i_{L}), and one polytopic constraint on the input to limit the duty cycle of the buck state $d_1 \in [0, 1]$. The chosen buck-boost converter is designed to operate with an arbitrary supply voltage (v_{s}) and load current (i_{load}) within the operation range. Therefore, the optimization parameters are chosen as $\theta = [v_{\text{C}},\ i_{\text{L}},\ i_{\text{load}},\ v_{\text{s}}]^{\text{T}} \in \mathbb{R}^4$ to solve the mp-QP problem.

TABLE I
EXPLICIT MPC RESULTS OF BUCK-BOOST CONVERTER

	Region-based	Binary-tree	Region-less
Memory Size ($N_{\text{p}} = 2,\ R = 156$)	59.52 kB	60.24 kB[a]	7 kB
Memory Size ($N_{\text{p}} = 4,\ R = 1{,}248$)	525.36 kB	2.67 MB[b]	80 kB

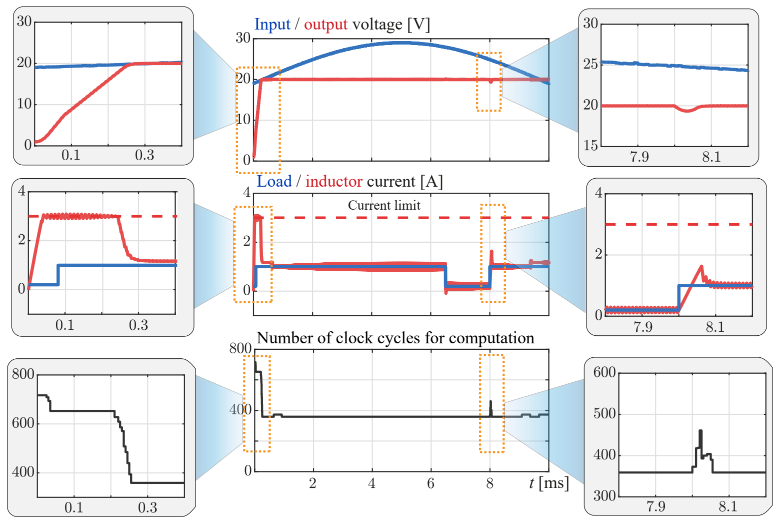
[a] : Depth (9), number of nodes (828)
[b] : Depth (18), number of nodes (45,510)

TABLE II
FPGA RESOURCE UTILIZATION OF RL-EMPC[a]

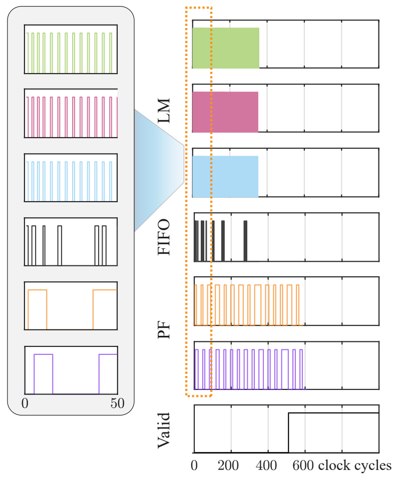
	ALM [K]	Memory [Kb]	DSP & Multiplier
LM[c]	2.20	778	16
PF[c]	8.5	113	124
Total[b,c]	14.82 (37%)	989 (77%)	220 (88%)
Available[d]	40	1,290	250

[a] : $N_{\text{p}} = 4$
[b] : Three LM blocks and two PF blocks in parallel
[c] : Numbers for LM and PF taken for a fit of only one LM/PF block on the whole device. Numbers for Total are the result of compiler/fitter optimization and smaller than the sum.
[d] : Intel Max 10 (10M40SAE144C8G), no external memory

The results in Tab. I show the memory requirements of the EMPC controllers when prediction horizon lengths of 2 and 4 are applied. In this example, the length of the prediction horizon does not bring substantial performance improvement, yet it is demonstrated to compare the memory requirements and the feasibility of the proposed implementation. Based on the proposed architecture, the RL-EMPC can be implemented on a low-cost FPGA (Max 10) device for a prediction horizon



(a)



(b)

Fig. 3. (a) RTL level co-simulation results of the buck-boost converter using the RL-EMPC solutions from Tab. I on a Max 10 device. For control performance and a comparison with a linear controller, refer to [6]. (b) RTL level co-simulation results of the RL-EMPC at $t = 0.21$ ms. The region is detected after 508 clock cycles, when the valid signal becomes high. Note that the complete computation of LM and PF checkers finishes after 609 clock cycles, which is denoted as the number of clock cycles for computation in (a).

of 4. The resulting resource usage as calculated by the Intel Quartus Fitter is presented in Tab. II, and simulation results are shown in Fig. 3. All data is stored in on-chip blocks due to the low amount of required memory, and the co-simulation results show that the proposed method can successfully compute control laws at the sampling rate of $100\,\mathrm{kHz}$ with the FPGA device operating at $125\,\mathrm{MHz}$. At the given frequencies, the simulation results show that the number of clock cycles for computations (< 800) is much smaller than the maximum allowed clock cycles (1250). When external memory is utilized, even a longer prediction horizon can be applied.

B. Modular Multilevel Converters (MMC)

Over the last decade, the modular multilevel converter (MMC) has gained a lot of attention and is used in many applications at medium and high voltage levels. However, since the MMC is a multi-input multi-output (MIMO) system with nonlinear characteristics, controlling the MMC with conventional methods, such as cascaded PI-controllers or resonant controllers, results in limited transient performance and/or oversized module capacitors [18]. In [3], a linear MPC method is proposed to overcome these limitations, such that the MMC with very small module capacitance (reduced by 40%) can be operated with a fast transient behavior. Even though the proposed method has a relatively low sampling rate for power electronic systems ($1 \dots 1.5\,\mathrm{kHz}$), a real-time implementation of the controller is still a challenging task since the size of the optimization problem is large with many states and constraints.

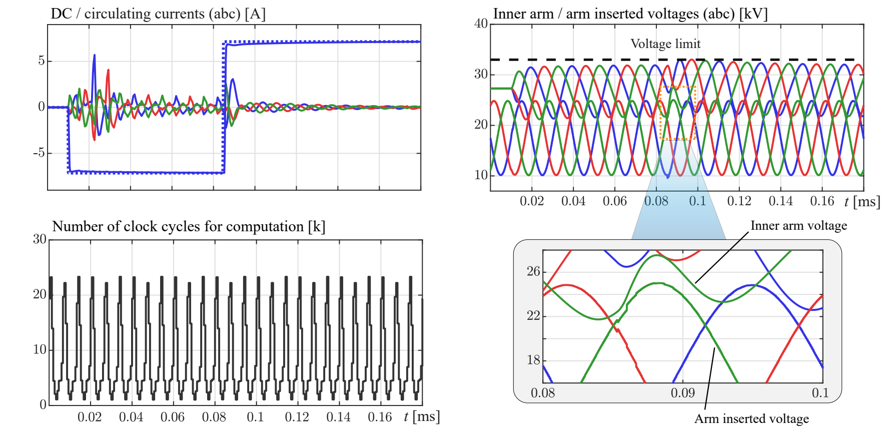
In [3], the MMC system is modelled as a periodic-time varying linear system by linearizing the energy dynamics of the MMC as

$$\underbrace{\begin{bmatrix}\mathbf{i}_{k+1}\\ \mathbf{w}_{k+1}\end{bmatrix}}_{\mathbf{x}_{k+1}} = \begin{bmatrix}\mathbf{A}_\mathrm{c} & \mathbf{0}\\ \mathbf{A}_\mathrm{w}(\varphi_\mathrm{g}) & \mathbf{0}\end{bmatrix}\cdot \underbrace{\begin{bmatrix}\mathbf{i}_{k}\\ \mathbf{w}_{k}\end{bmatrix}}_{\mathbf{x}_{k}} + \begin{bmatrix}\mathbf{B}_\mathrm{c}\\ \mathbf{0}\end{bmatrix}\cdot \underbrace{\begin{bmatrix}\mathbf{v}^{\delta}_{\mathrm{e},\alpha\beta 0,k}\\ \mathbf{v}^{\delta}_{\mathrm{a},\alpha\beta 0,k}\end{bmatrix}}_{\mathbf{u}_{k}},$$

where $\mathbf{i} \in \mathbb{R}^5$ and $\mathbf{w} \in \mathbb{R}^6$ are the current and the energy states, $\mathbf{u} \in \mathbb{R}^6$ is the control inputs, $\mathbf{A}_\mathrm{c} \in \mathbb{R}^{5\times5}$ and $\mathbf{B}_\mathrm{c} \in \mathbb{R}^{5\times6}$ are the time-invariant current system matrices, and $\mathbf{A}_\mathrm{w}(\varphi_\mathrm{g}) \in \mathbb{R}^{6\times5}$ is the periodic time-varying energy matrix depending on the grid angle (φ_g). In other words, the converter is represented as a PWA model with 11 states and 6 inputs as a function of the grid angle (φ_g). The domain of the grid angle (φ_g) is divided into N_s intervals, where N_s is the number of samples in one grid-period. Note that the MMC model is an average model, which is independent of the number of modules. The balancing of the individual module voltages can be performed using sorting based PWM methods such as presented e.g. in [19].

For each interval of the grid angle, the system dynamics are different and a different linear MPC problem is formulated. Moreover, since the control of the MMC is a trajectory tracking problem, references should be included. Instead of adding all states and input references, the power level of the system is included to represent the varying references as shown in [18], such that the complexity of mp-QP problems can be minimized. The optimization parameters are chosen as $\theta = [\mathbf{i}^\mathrm{T}, \mathbf{w}^\mathrm{T}, p]^\mathrm{T} \in \mathbb{R}^{12}$. In contrast to the solution presented in [3], the mp-QP problems are formulated only with control input constraints, such that the number of generated regions remains reasonable.

The EMPC memory requirements with a prediction horizon of 5 are given in Tab. III for the same parameters used in [3]. At each grid angle interval, many regions are generated, and the whole grid period sums up to an huge number of regions compared to other applications. Therefore, common EMPC methods cannot be applied. The region-based method requires a large amount of memory, and even with an external memory, the implementation is not feasible. In such situation, constructing a binary tree is extremely difficult and results in massive memory requirements, so that the binary tree method is not considered here. Compared to that, the RL-EMPC method leads to moderate memory requirements and can be implemented on a low-cost FPGA (Cyclone V) device with external SDRAM. The resulting resource usage is presented



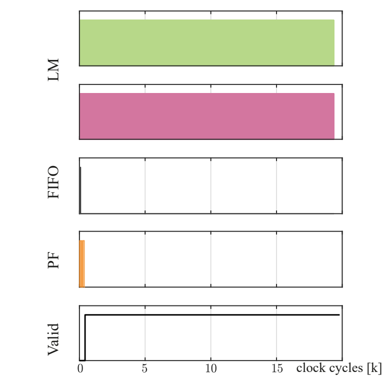


Fig. 4. (a) RTL level co-simulation results of the MMC using the RL-EMPC solutions from Tab. III on a Cyclone V SoC device. For control performance and a comparison with a linear controller, refer to [3]. (b) RTL level co-simulation results of the RL-EMPC at $t = 0.02\,\mathrm{ms}$. The region is detected after 414 clock cycles, when the valid signal becomes high. Note that the complete computation of LM and PF checkers finishes after 19374 clock cycles, which is denoted as the number of clock cycles for computation in (a).

TABLE III
EXPLICIT MPC RESULTS OF MODULAR MULTILEVEL CONVERTER[a]

	Region-based	Region-less
Memory Size[b] ($R = 3,471$)	18.50 MB	0.67 MB
Memory Size[c] ($R = 49,838$)	247.15 MB	10.35 MB

[a] : N_p = 5, f_s = 1.5 kHz
[b] : Example of one grid angle partition
[c] : All grid angle partitions with N_s = 30

TABLE IV
FPGA RESOURCE UTILIZATION OF RL-EMPC[a]

	ALM [K]	Memory [Kb]	DSP & Multiplier
LM[c]	9	~ 0	14
PF[c]	21	767	468
Total[b,c]	33 (59%)	813 (12%)	468 (100%)
Avaialable[d]	56	7,025	468

[a] : N_p = 5
[b] : Two LM block and one PF block in parallel
[c] : Numbers for LM and PF taken for a fit of only one LM/PF block on the whole device. Numbers for Total are the result of compiler/fitter optimization and smaller than the sum.
[d] : Intel Cyclone V SoC (5CGXFC7D6F31C6) with external memory

in Tab. IV, and simulation results are shown in Fig. 4. The enlarged simulation result illustrates that the constraints of keeping the arm inserted voltage lower than the inner arm voltage are satisfied even when the power flow reverses. Therefore, the MMC can operate close to its physical limits, resulting a compact realization of the MMC with a very small module capacitance value. The co-simulation is carried out at the sampling rate of 1.5 kHz with the FPGA device operating at 250 MHz. At the given frequencies, the simulation results show that the number of clock cycles for computations ($<$ 25k) is much smaller than the allowed clock cycles (166k).

V. CONCLUSION

For power electronic systems, linear MPC is an attractive solution to exploit components up to their physical limits and achieve an optimal converter design as well as fast transient performance. This paper presents a method for a real-time implementation of linear MPC using FPGA devices based on RL-EMPC. Two examples are demonstrated to prove the feasibility of the proposed implementation for general linear MPC problems along with analyses about their resource usages and on-line computation times. First, for the buck-boost converter, RL-EMPC reduces the memory requirements by 87 % compared to the region-based method and realizes the real-time implementation for a prediction horizon of 4 with an Intel Max 10 device, while other EMPC methods are limited to a prediction horizon of 2. If external memory is utilized, RL-EMPC can be realized even for a longer prediction horizon. Second, for the MMC, RL-EMPC reduces the memory requirement by 96 % compared to the region-based method and enables a realization based on a Cyclone V device. This cannot be achieved with other EMPC approaches and shows the effectiveness of the proposed method. Results show that the proposed approach allows a simple and low-cost implementation to compute optimal control laws in real-time even for larger-size MPC problems.

REFERENCES

[1] S. Vazquez, J. Rodriguez, M. Rivera, L. G. Franquelo, and M. Norambuena, “Model predictive control for power converters and drives: Advances and trends,” *IEEE Trans. on Ind. Electron.*, vol. 64, no. 2, pp. 935–947, 2017.

[2] D. E. Quevedo, R. P. Aguilera, and T. Geyer, “Model predictive control for power electronics applications,” in *Handbook of Model Predictive Control*. Springer International Publishing, sep 2018, pp. 551–580.

[3] S. Fuchs, M. Jeong, and J. Biela, “Long horizon, quadratic programming based model predictive control (MPC) for grid connected modular multilevel converters (MMC),” in *Conf. of the IEEE Ind. Electron. Soc. (IECON)*, Oct 2019.

[4] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017, pp. 33–70.

[5] I. McInerney, G. A. Constantinides, and E. C. Kerrigan, “A survey of the implementation of linear model predictive control on FPGAs,” *IFAC-PapersOnLine*, 2018, 6th IFAC Conf. on Nonlinear Model Predictive Control NMPC.

[6] V. Spinu, A. Oliveri, M. Lazar, and M. Storace, “FPGA implementation of optimal and approximate model predictive control for a buck-boost DC-DC converter,” in *IEEE Int. Conf. on Control Appl.*, Oct 2012, pp. 1417–1423.

[7] S. Mariéthoz, S. Almér, and M. Morari, “Optimal control of a two control input buck-boost converter,” in *Proc. of the IEEE Conf. on Decision and Control (CDC)*, 2009, pp. 6575–6581.

[8] M. Kvasnica, B. Takács, J. Holaza, and S. Di Cairano, “On region-free explicit model predictive control,” in *54th IEEE Conf. on Decision and Control (CDC)*, 2015.

[9] J. Drgoňa, M. Klaučo, F. Janeček, and M. Kvasnica, “Optimal control of a laboratory binary distillation column via regionless explicit MPC,” *Computers and Chemical Engineering*, vol. 96, pp. 139–148, 2017.

[10] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, “The explicit linear quadratic regulator for constrained systems,” *Automatica*, vol. 38, no. 1, 2002.

[11] V. M. Charitopoulos, *Uncertainty-aware Integration of Control with Process Operations and Multi-parametric Programming Under Global Uncertainty*. Cham: Springer International Publishing, 2020.

[12] P. Tøndel, T. Johansen, and A. Bemporad, “Evaluation of piecewise affine control via binary search tree,” *Automatica*, vol. 39, no. 5, pp. 945 – 950, 2003.

[13] M. Kvasnica, J. Holaza, B. Takács, and D. Ingole, “Design and verification of low-complexity explicit MPC controllers in MPT3,” in *Europ. Control Conf. (ECC)*, 2015.

[14] A. Gersnoviez, M. Brox, and I. Baturone, “High-speed and low-cost implementation of explicit model predictive controllers,” *IEEE Trans. on Control Systems Technology*, vol. 27, no. 2, pp. 647–662, 2019.

[15] A. Oliveri, C. Gianoglio, E. Ragusa, and M. Storace, “Low-complexity digital architecture for solving the point location problem in explicit model predictive control,” *Journal of the Franklin Institute*, vol. 352, no. 6, pp. 2249 – 2258, 2015.

[16] A. Oliveri *et al.*, “MOBY-DIC: A MATLAB toolbox for circuit-oriented design of explicit MPC,” *IFAC Proc. Volumes*, vol. 45, no. 17, 2012.

[17] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, “Multi-Parametric Toolbox 3.0,” in *Proc. of the Europ. Control Conf.*, July 2013, pp. 502–510, https://www.mpt3.org/.

[18] M. Jeong, S. Fuchs, and J. Biela, “High performance LQR control of modular multilevel converters with simple control structure and implementation,” in *22th Europ. Conf. on Power Electron. and Appl. (EPE ECCE Europe)*, 2020, in press.

[19] S. Fuchs, S. Beck, and J. Biela, “Analysis and reduction of the output voltage error of PWM for modular multilevel converters,” *IEEE Trans. on Ind. Electron.*, vol. 66, no. 3, March 2019.