processes

Review

Embedded Control in Wearable Medical Devices: Application to the Artificial Pancreas

Stamatina Zavitsanou, Ankush Chakrabarty, Eyal Dassau * and Francis J. Doyle III

Harvard John A. Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138 , USA; szavitsanou@g.harvard.edu (S.Z.); achakrabarty@seas.harvard.edu (A.C.); frank_doyle@seas.harvard.edu (F.J.D.)

* Correspondence: dassau@seas.harvard.edu; Tel.: +1-617-496-0358

Academic Editor: B. Wayne Bequette

Received: 14 July 2016; Accepted: 15 September 2016; Published: 23 September 2016

Abstract: Significant increases in processing power, coupled with the miniaturization of processing units operating at low power levels, has motivated the embedding of modern control systems into medical devices. The design of such embedded decision-making strategies for medical applications is driven by multiple crucial factors, such as: (i) guaranteed safety in the presence of exogenous disturbances and unexpected system failures; (ii) constraints on computing resources; (iii) portability and longevity in terms of size and power consumption; and (iv) constraints on manufacturing and maintenance costs. Embedded control systems are especially compelling in the context of modern artificial pancreas systems (AP) used in glucose regulation for patients with type 1 diabetes mellitus (T1DM). Herein, a review of potential embedded control strategies that can be leveraged in a fully-automated and portable AP is presented. Amongst competing controllers, emphasis is provided on model predictive control (MPC), since it has been established as a very promising control strategy for glucose regulation using the AP. Challenges involved in the design, implementation and validation of safety-critical embedded model predictive controllers for the AP application are discussed in detail. Additionally, the computational expenditure inherent to MPC strategies is investigated, and a comparative study of runtime performances and storage requirements among modern quadratic programming solvers is reported for a desktop environment and a prototype hardware platform.

Keywords: embedded control systems; artificial pancreas; software architecture; model predictive control (MPC); safety-critical applications

1. Introduction

Embedded electronics are widely incorporated in medical devices for diagnosis, prognosis and monitoring of a disease. Examples of such devices, generally implemented with humans in the decision-making process, include: digital thermometers, blood pressure monitors, blood glucose monitors, ECG (Electrocardiography), radiography, MRI (Magnetic Resonance Imaging), CT (Computed Tomography) and PET (Positron Emission Tomography). The pressing need to develop fully-automated therapeutic devices (therefore, removing the human-in-the-loop formalism) for active decision-making, coupled with demands for portability and cost-effectiveness, has prompted the progress of embedded control systems for medical applications.

A medical embedded controller is a miniaturized, automated system, informed by measurements, i.e., feedback, from the patient and his or her external environment, designed to perform a dedicated function related to immediate treatment or overall management of patient health. Sensors, transducers and human-computer interactions are used for the embedded controller to interact with the environment, and processing units such as microcontrollers/microprocessors are typically employed to execute decision-making strategies based on the acquired measurement information. Additional design specifications, such as, for example, (i) runtime constraints imposed by the implementation platform; (ii) reliable operation in the presence of exogenous disturbances and potential system failures; (iii) optimal power management for the prevention of overheating and to enhance device longevity and (iv) security regulations to guarantee patient privacy, are also considered while designing the embedded controller.

An extremely important application that requires the successful integration of feedback control mechanisms embedded in medical devices is the artificial pancreas (AP) for people with type 1 diabetes mellitus (T1DM). T1DM patients suffer from an auto-immune disorder that prevents the pancreas from secreting insulin, a hormone essential for the regulation of blood glucose levels. Dysregulation of blood glucose concentrations typically leads to serious medical conditions, such as hyperglycemia- and hypoglycemia-induced health complications that can increase mortality rates by 1.5 fold in patients with diabetes [1]. Consequently, exogenous insulin must be delivered to maintain the blood glucose concentration within a safe range (70–180 mg/dL). The AP delivers personalized insulin infusions in a closed-loop, that is the magnitude of insulin infused is computed based on blood glucose measurements. Currently, the most advanced treatment of T1DM employs a continuous glucose monitoring system (CGM) that takes measurements of glucose concentration at regular intervals. These CGM measurements are leveraged by a control algorithm in the AP, which (without the patient's active involvement) computes an appropriate magnitude of insulin infusion. The infusion is actuated by an insulin pump, also referred to as a continuous subcutaneous insulin infusion (CSII) system.

Several control algorithms have been considered for an AP system [2] and have been clinically evaluated, including: fuzzy logic-based control (FLC) [3,4], proportional-integral-derivative (PID)-type control [5,6] and model predictive control (MPC) [7–10]. The results indicate that a reliable AP system can be developed using any of these controller variants [11]. Currently, the control algorithms for the AP are implemented on CPU-based devices, such as PCs, laptops and tablets, with a recent shift towards smaller processing units, such as those found in smartphones. It is necessary, however, to search for alternative control strategies or to modify existing controllers to warrant implementation on miniaturized devices with limited memory capacity operating at ultra-low power levels.

The inherent robustness to model mismatch and delays, the design flexibility, the ability to handle multivariable systems, as well as safety and reliability constraints and the flexible use of the system model make MPC an attractive control algorithm for complex system. The MPC algorithm has been successfully implemented in industrial safety-critical process control applications, such as oil and gas, chemicals, aerospace, food and processing [12] and in the automotive and aircraft domains [13,14]. State-of-the-art embedded MPC on platforms like programmable logic controllers (PLCs) [15] and digital distributed control systems (DCSs) has motivated the consideration of embedded MPC for the AP system. Therefore, in this study, emphasis is placed on the MPC implementation requirements for an embedded AP. It is expected that an embedded AP with MPC in the decision-making module can be replaced by computationally simpler algorithms like PID or fuzzy-logic based controllers if required.

The objective of this paper is to explore current and future trends in the AP technology with a focus on architecture and algorithms amenable to low-complexity miniaturized devices. We discuss available hardware and software configurations and provide insight into the configuration for embeddable AP. We also propose multiple control algorithms for embedded AP, with special emphasis on model predictive control and its variants due to several important qualities: namely, design flexibility in terms of complexity reduction and available resource constraints (memory/time/power) along with the inherent embedding-agnostic advantages discussed above. We also compare different optimization methods and solvers to demonstrate this design flexibility of the MPC. However, the controller has to be built around peripheral layers that are responsible for ensuring the security, privacy and integrity of the patient. To this end, we provide a detailed list of potential embedded AP hazards and methods in the current literature to mitigate these issues. It is important to clarify that the AP system is considered as an exemplar system in this work; however, the configurations, algorithms and safety protocols covered are generalizable to embedded/wearable medical devices across a wide range of applications.

The rest of this paper is organized as follows. Section 2 presents the AP system configurations for the implementation on a smartphone or on-chip. Design requirements for the software architecture based on the specifications of the available hardware resources are discussed. Section 3 lists the available control algorithms used in modern AP systems that can potentially be embedded in low complexity hardware platforms. Multiple available MPC solvers are evaluated, and a comparative study of their runtimes, code size and control performance is made both on a desktop and on a prototyping hardware platform with representative numerical examples. Challenges associated with the design, system prototyping and verification methods are discussed in Section 4. Safety requirements that must be guaranteed in order to develop a reliable AP system are described in Section 5. Some concluding remarks on the development of embedded AP systems are presented in Section 6.

2. AP System Architecture

2.1. Components of the AP System

The AP system consists of the following three components: a continuous glucose monitor (CGM), a control algorithm and a pump for delivering insulin (and glucagon in bi-hormonal therapy). In this work, the single hormone approach will be discussed because it is commonly available. The CGM provides the controller with glucose measurements, typically every 5 min for currently available CGM devices. Subsequently, the controller processes the information reported by the CGM, and the appropriate insulin dosage is computed. The calculated control action is then signaled to the insulin pump to execute the control action and actuate the prescribed insulin dose. An AP consisting of the three necessary elements of CGM, CSII pump and control law regulates glucose in a closed-loop manner, without necessitating the active participation of the patient in the decision-making process.

2.2. Hardware Configurations of the AP System

The AP system architecture has been clinically evaluated for three potential configurations, based on the hardware platform where the controller was implemented, specifically: computer/laptop [7,16–18], tablet [8,19,20] and smartphone [21]. In this study, we examine two possible AP hardware configurations; we refer to them as Configuration A and Configuration B. In Configuration A, the control unit is an application installed on a commercially available handheld device. A future design aim is that the application can be employed concurrently with other applications run by the user. To this end, a smartphone-based AP platform (currently, for the development phase of the Diabetes Assistant (DiAS) platform, the smartphone is employed as a dedicated controller, and no other phone-related functions are permitted) (DiAS; University of Virginia, Charlottesville, VA, USA) has been developed [22] and evaluated in outpatient studies [10,23–26]. Extended studies involving a two-month evening and night closed-loop control study have also been performed [27]. Results from these studies indicate that the developed platform is reliable in terms of computational efficiency and safety. An iPhone 4S was used in [28] to run a control algorithm for bi-hormonal therapy that was evaluated in an outpatient study of preadolescent children with T1DM. A hand-held computer [29] was used to run the control algorithm in [30,31]. A free-living conditions clinical trial was performed using a smartphone that implemented the controller [32]. A hybrid closed-loop delivery system implemented on a cellular device was evaluated in a supervised outpatient study [33]. A wearable device that integrates the CGM, CSII, a glucagon pump, the control algorithm and wireless transmitters was presented and evaluated in a pilot at home clinical study [34]. A bio-inspired AP system implemented on a miniature silicon microchip within a portable hand-held device is evaluated in [35]. In Configuration B, on the other hand, the controller is embedded directly into the pump. In this case, no additional peripheral devices (such as smartphones) are required to run the control algorithm.

Configurations A and B are presented in a schematic; see Figure 1. There are two main components: the data acquisition unit and the control unit. Possible inputs to the data acquisition unit include: (i) the real-time glucose concentration values sent wirelessly from the sensor; (ii) discrete finger stick values used for the calibration of the sensor; and (iii) manually = provided patient information, such as announced meals, that may permit tighter control by feed-forward action. A transmitter that is operatively coupled with a microprocessor is configured to communicate with the sensor and the user input. The microprocessor in the data acquisition unit is used to transmit control signals to the receiver during operation, and to control the retrieval and processing of stored data. The data acquisition unit can be either a standalone monitoring or displaying device (as for example seen in Figure 1, Configuration A) or it can be integrated into the insulin pump (see Configuration B). In Configuration A, the data acquisition unit wirelessly sends information to the handheld controller [22]. The measured glucose values and insulin treatment can be saved on a secure virtual cloud [36,37] for analysis and evaluation by the patient and their healthcare provider.

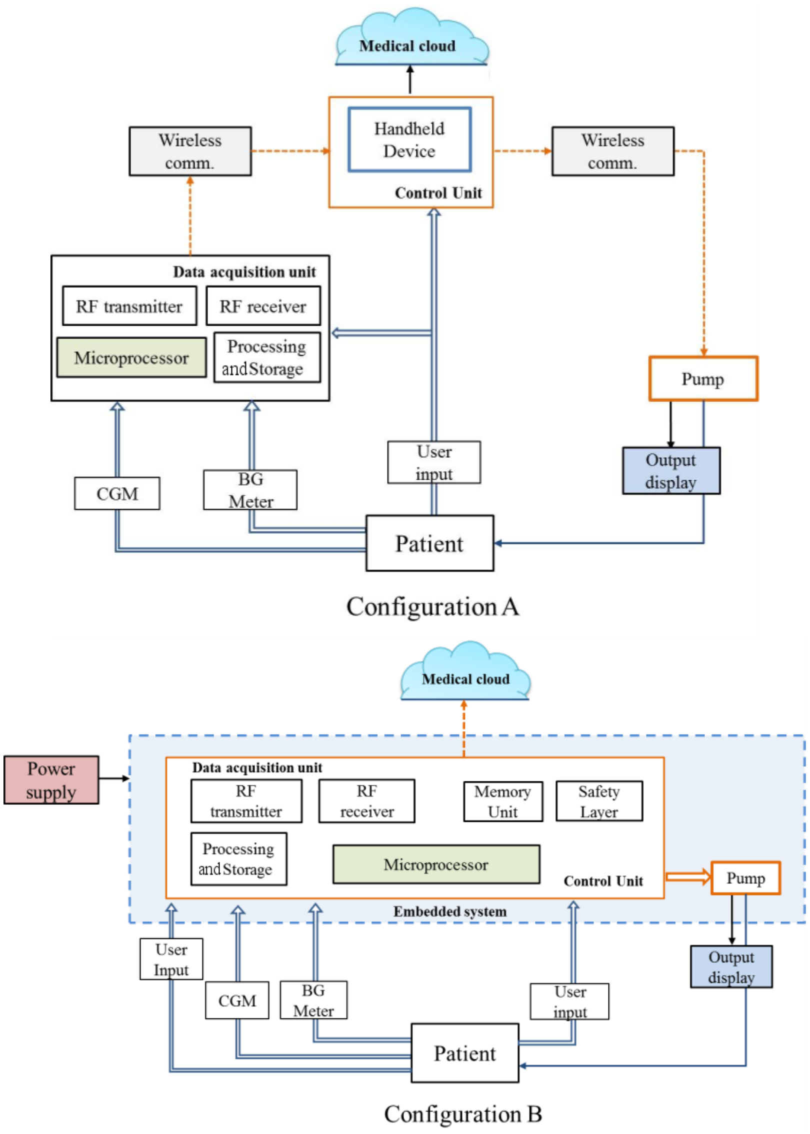

Figure 1. Schematic artificial pancreas (AP) system architecture. In Configuration A (top), the control unit is an application installed on a handheld device. In Configuration B (bottom), the embedded controller is integrated into the actuating insulin pump (BG = blood glucose; CGM = continuous glucose monitoring; RF = radio frequency).

In Configuration B, enclosed within the blue dashed line in Figure 1, the control unit is built into the insulin pump. The ultimate objective for the AP system is to integrate the data acquisition unit into the pump. Therefore, the two components are combined into a single dedicated device to enable secure resource sharing at higher speeds.

2.2.1. Configuration A

Due to recent advances in mobile computing, the smartphone (handheld device) has been established as a useful medium for implementing a myriad range of applications. In the case of an AP system, the control unit can be implemented as an application on a smartphone. The supporting hardware specifications for the running AP control application are obtained using ARM Cortex-A15 as a reference microprocessor that is comparable with microprocessors commonly used in modern smartphone technology (Qualcomm, Samsung, NVidia).

2.2.2. Configuration B

This configuration entails that the control unit is integrated directly into the insulin pump. Although the exact specifications of the microcontrollers used in commercial insulin pumps are not openly available, the standard requirements for portable medical devices are met with the low-power microcontrollers, such as STM32 [38], which uses a Cortex M3 processor core based on the 32-bit ARMv7 architecture, MAXQ2010 [39] and 8-bit S08 [40].

A comparative overview of microcontrollers used in smartphone-based AP systems and commercial insulin pumps is presented in Table 1.

Table 1. Hardware specifications for various AP (artificial pancreas) configurations. Comparison metrics include: processing unit, memory capability, clock frequency, arithmetic logic unit (ALU) and availability of floating point units (FPU).

Hardware Specifications				
Configuration	**(A) Smartphone**	**(B) Insulin Pump**		
Processor Core	ARM Cortex-A15	ARM Cortex-M3	MAXQ2010	8-bit S08
Memory	2 GB RAM	96 KB on-chip RAM	64 KB Flash	2–128 KB Flash
			2 KB RAM	0.128–12 KB RAM
CPU Clock Frequency	1.5–1.7 GHz	72 MHz	10 MHz	16–50 MHz
Integer ALU	32-bit	32-bit	16-bit	8-bit
FPU	Optional	✘	✘	✘

2.3. *Software Architecture*

Given the available hardware resources, the software architecture for a chosen control algorithm affects the dependability of the embedded AP system, while conforming to expected operational and technological specifications. The concept of dependability, presented in [41], integrates the following attributes: (i) reliability; (ii) availability; (iii) safety; (iv) confidentiality; (v) integrity; and (iv) maintainability.

For a safety-critical application such as an embedded medical system, the most important attribute to be met during the design stage is safety. System failure can jeopardize the patient's health, and, therefore, must be prevented under any circumstances. The other attributes are fundamental to the design and development of any generic embedded medical device and will be briefly discussed in the rest of this study.

Sensor dropouts and attenuation, insulin infusion failure or erroneously announced meals are common issues encountered during typical AP operation [42]. The need to fulfill strict safety requirements has been evident in most AP designs and alternative software architectures have been evaluated. Modular architecture, including a control, a safety supervision and a range correction module, has been developed [43]. A health monitoring system is introduced in [44] as a safety layer in the AP device system. A meal detection algorithm either based on the calculation of the glucose rate of change [45,46] or model based [47] is proposed to prevent postprandial hyperglycemia for the case of unannounced meals. A low-glucose suspend system includes a threshold-suspend feature of sensor-augmented insulin pumps that forces the pump to shut off when CGM values drop below a certain threshold [48]. This feature is now commercially available, and it is the first step towards the development of a safe portable AP system.

The AP software architecture is structured in layers and presented in Figure 2a.

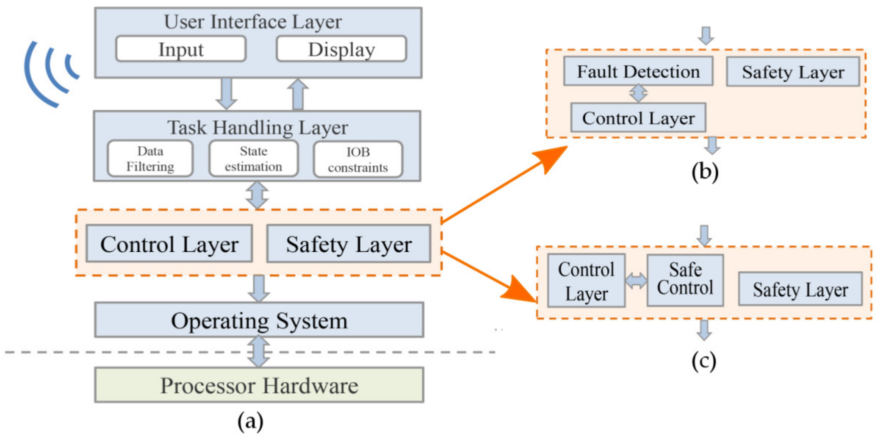

Figure 2. Software architecture for AP. (**a**) Overview of the layered software architecture. The control and safety layer enclosed in the orange dashed line is enlarged and presented in (**b**,**c**); (**b**) architecture with fault detection layer; and (**c**) architecture with control and safe control layers. IOB, insulin-on-board.

2.3.1. User Interface Layer

The user interface layer (UI) is used for the communication between the user and the embedded control system. This layer receives instructions from the patient, for example: the announcement of a meal, report of exercise, request for a manually-determined insulin bolus; and, along with the CGM data, this information is directed to the next layer. The UI layer is also responsible for displaying the returning information from the embedded system to the user. Appropriate functioning of the embedded control system is dependent on the receipt of good data from the patient, and a well-designed UI is crucial for this, making it as difficult as possible for the user to make mistakes in entering data that should be entered mistake free. On the other hand, effective communication and presentation of data to the subject is imperative for AP users to understand the inner workings of the control algorithm in order to improve their personal health management.

2.3.2. Task Handling Layer

The task handling layer is responsible for defining, executing and deleting a task. The tasks involve the filtering and coordination of data [43], the estimation of the metabolic state, the definition of operational and safety constraints, such as insulin-on-board (IOB) constraints, and the calculation of time varying glucose targets. Depending on the exact implemented control algorithm, different tasks may be created.

2.3.3. Control and Safety Layer

The next two layers, the control and safety layers, are the kernel of the AP system. The control layer is responsible for the calculation of the optimal insulin infusion to be delivered by the pump. Alternative control strategies have been suggested for the AP system [2], whose implementation in embedded platforms is discussed in detail in Section 3. The safety layer acts as a supervision layer that is responsible for evaluating the quality of the input data and detecting any faults originating in the control layer. In such scenarios, the safety layer overrides the suggested control action and signals the lower levels to perform an alternative, safer action. Such an action could be a low risk insulin infusion or the generation of an alarm that warns the patient to intercede. The interplay between the controller and the safety layer is as follows. The control is designed to produce insulin infusions with the end goal of maintaining safe levels of blood glucose. To this end, it is accessorized with basic safety constraints. However, in the event of controller failure, it is crucial to have a reliable secondary level of protection for the patient. This is the role of the safety layer: a dedicated function to prevent life-threatening failures.

As argued in [49], for medical embedded control, the highest priority is to maintain reliability and safe functionality (fault tolerance, disturbance rejection) while running at optimal operation levels. Different safety protocols can be implemented in conjunction with the core control strategy, as presented in Figure 2b,c. For example, in Figure 2b, a distinct fault detection layer is included [50,51] that identifies if a fault has occurred in the control layer and adapts the control parameters according to the dynamics of the faulty system. Another way to reconfigure the controller is to switch between pre-designed controllers depending on the specific fault. However, issues of stability, performance degradation and increase in storage requirements may arise when switching between controllers, which is not desirable in medical embedded control. An alternative configuration based on the principle of functional redundancy has been suggested in [49,52] and is presented in Figure 2c. This architecture considers two or more control alternatives that run in parallel, and based on pre-specified criteria, the best solution is selected. However, increased computational complexity and storage requirements should be taken into consideration. If the control layer cannot provide the optimal solution within the timing constraints imposed by the hardware, the safe control layer provides a feasible solution that satisfies basic safety requirements. To address this issue, in [52], the safe control layer solves an iterative optimization procedure utilizing Hildreth's algorithm [53] in order to fulfil the strict timing deadlines of real-time systems while maintaining a minimal set of safety constraints. However, it is not clear that the solution to runtime constraints lies only in the selection of suitable solvers. For example, in [49], the authors propose a safe controller that uses the same solver as the control layer, but with settings tailored towards reducing memory or timing overheads. Such a method may involve warm-start initialization, where the tail of the previous control trajectory is used as an initial guess for computing the next control iteration. The overarching objective of this "safe" controller is to construct a solution that prioritizes the safety of the next control action in spite of the sudden appearance of failure modes. A detailed description of potential faults in an AP system is presented in Section 6.

2.3.4. Operating System and Hardware Layers

The suggested control solution is then passed to the operating system, which acts as the communication channel between the upper layers and hardware. It is responsible for scheduling tasks, memory management, parallel executions and communication with peripherals.

Finally, the processor hardware layer includes the firmware of the microprocessor that controls directly the operation of the hardware by managing the operating system. It is stored in non-volatile memory, and its main responsibility is to load the operating system's kernel into primary memory. In small embedded devices, the last two layers may be combined.

3. Control Algorithms for the Artificial Pancreas

Mainly three control design approaches have been suggested for the regulation of blood glucose within a closed-loop framework: PID, fuzzy logic and MPC [2]. PID controllers operate on the difference between the measured and the desired glucose levels and calculate a control signal (insulin infusion) that is then applied on the system (patient). The relatively simple implementation and the argument that the PID algorithm is expected to emulate and replace the biological pancreas action well [54,55] have established PID controllers as a significant candidate for the AP system [56,57]. The wealth of clinical experience behind the decision-making for glucose regulation has, on the other hand, motivated the consideration of fuzzy logic control as a potential candidate [58]. Fuzzy logic aims to solve a problem by mimicking the way a person would make decisions, by introducing heuristic variables. Finally, the inherent complexity of the system with hard constraints, time delays, metabolic uncertainty and variability has encouraged the use of MPC. MPC uses a mathematical model of patients with type 1 diabetes to predict the future effect of control action (insulin/and glucagon) on their blood glucose concentration. The model is used to formulate an optimization problem that minimizes an appropriately chosen performance objective that is solved to obtain the optimal control trajectory. Several formulations of the MPC problem have been proposed for the AP system, depending on the controller objectives, the type of the mathematical model used and the control specifications. For example, the desired blood glucose can be of a constant or time-varying value [59,60] or a euglycemic zone may be formulated that can also be constant [61] or time varying [62]. The mathematical model can be physiological or empirical, time invariant or adaptive.

In this section, the requirements in terms of design, implementation and computation of the alternative control algorithms that need to be considered in order to develop an embedded AP control system are discussed. First, a brief description of PID and fuzzy logic control implementation in embedded systems is provided, and then, the requirements for embedded MPC development are discussed in detail.

3.1. Proportional Integral Derivative Control

The PID controller consists of three components: the proportional (P), the integral (I) and the derivative (D). The control update equation of the PID controller [63] is given by:

$$u\left(t\right)=K_P\left[e\left(t\right)+\frac{1}{\tau_I}\int_0^t e\left(s\right)ds+\tau_D\frac{de\left(t\right)}{dt}\right],\ e\left(t\right)=r\left(t\right)-y\left(t\right) \tag{1}$$

where $u(t)$ is the control output, $e(t)$ is the error signal, represented by a difference between the reference signal $r(t)$ and the measured output $y(t)$. The scalars K_P, τ_I, τ_D, denote the controller parameters. The PID update Equation (1) has to be discretized in order to be implemented in digital platforms [64,65], given by:

$$u_k=K_P\left[e_{k-1}+\frac{\Delta t}{\tau_I}\sum_{s=1}^{k-1}e_s+\frac{\tau_D}{\Delta t}\left(e_{k-1}-e_{k-2}\right)\right],\ e_k=r_k-y_k \tag{2}$$

where Δt is the sampling time. Once the discrete PID control algorithm is obtained, anti-windup algorithms should be considered to account for inaccuracies and oscillations imposed by saturating constraints on the actuator. Then, the aim is to efficiently implement the controller with respect to minimizing the execution time and the required memory size to meet the hardware requirements. Issues of finite word length and precision, signal conditioning, pre-filtering and computational delays will arise when PID is implemented on digital hardware. Methods for addressing some of these issues are discussed, for example, in [65]. The optimal use of the hardware resources has been recognized to play an important role in complexity reduction and speed increase in PID implementation. An efficient FPGA (field programmable gate array) design scheme for PID implementation is proposed in [66].

3.2. Fuzzy Logic Control

Fuzzy logic can be employed to construct rule-based fuzzy logic control methods. The original model-free nature of fuzzy logic control has gradually changed to model-based FLC [67] with the introduction of Takagi–Sugeno (T-S) fuzzy models [68]. The dynamics of nonlinear systems can be expressed using a T-S fuzzy model built on fuzzy rules. These rules denote local input-output dynamics and are expressed by local linear models; see, for example [69]. The fuzzy inference process consists of four steps: initialization, fuzzification, fuzzy inferencing and defuzzification. The first step involves the definition of the linguistic variables and terms, the construction of the membership functions (MF) and the rule base. During the fuzzification step, membership functions are used to convert the crisp data to fuzzy data, which are then used to evaluate the rules in the rule base using a fuzzy inference system (FIS). The results are combined, and finally, the output data are converted to non-fuzzy values during the defuzzification step. The key points that need to be considered for small footprint hardware implementations of fuzzy logic are [70]: (i) evaluation of the contribution of the active rules only; (ii) restriction of the shapes of the membership functions; and (iii) simplification of the defuzzification methods. A single-input FLC is proposed in [71] to simplify the involved evaluation rules, and with an appropriate choice of the MF shape and fuzzification/defuzzification methods, the proposed embedded scheme achieved reduced memory needs while maintaining the performance requirements. In [70], a hybrid software/hardware approach was proposed for FLC hardware implementation that resulted in improved performance and flexibility perfectly suitable for low-cost and rapid implementations in robotics.

3.3. Model Predictive Control

At each sampling instant, MPC solves an on-line optimization problem in order to find the optimal control action that minimizes the objective over a specified time horizon and also satisfies constraints on inputs (insulin/and glucagon), states (physiological/artificial) or outputs (glucose). When the optimal sequence of the future control actions $\{u_i, u_{i+1}, \dots, u_{i+N_c-1}\}$ is obtained, only the first value u_i is applied to the system, and the optimization problem is then reformulated and solved at the next time instant, when new measurement information (namely, y_{i+1}) is available. The optimization problem that yields the optimal zone-MPC control action, described in [62,72], is given by,

$$\begin{gathered}
\min_{\hat{u}_k,\check{u}_k} Q \sum_{k=1}^{N_p} (z_k)^2 + \sum_{k=0}^{N_c-1} (\hat{R}\hat{u}_k^2 + \check{R}\check{u}_k^2), \\
\text{subject to :} \\
x_{k+1} = Ax_k + Bu_k \\
y_k = Cx_k \\
z_k = Z(y_k, \overline{y}_{k+i}, \underline{y}_{k+i}) \\
-\underline{u}_{k+i} \leq u_k \leq \overline{u}_{k+i} \\
\hat{u}_k = \max\,(u_k, 0) \\
\check{u}_k = \min\,(u_k, 0)\,,
\end{gathered} \tag{3}$$

where i denotes the current sampling instant and y_i is the current measurement output. With glucose excursions z_k deviating from the euglycemic zone defined with the zone-excursion function Z, where,

$$Z\left(y, \overline{y}, \underline{y}\right) = \begin{cases} y - \overline{y}, \text{ if } y > \overline{y} \\ \underline{y} - y, \text{ if } y \leq \underline{y} \\ 0, \text{ otherwise} \end{cases}$$

the scalar u denotes the control input, y is the measured scalar output, $\overline{y}$ and $\underline{y}$ are the upper and lower bounds of the euglycemic zone and A, B, C are system matrices. The matrix Q penalizes the glucose deviation from the target zone, and the matrices $\hat{R}$ and $\check{R}$ denote the control input weights above and below the basal insulin level, respectively. The positive integers N_p and N_c, with $N_c \leq N_p$, denote the prediction and control horizons of the MPC. The successful implementation of MPC in real-time applications relies on balancing the different aspects of controller design, such as constraint satisfaction, optimality, stability and complexity reduction. The repetitive solution of the optimization problem (3) is computationally intensive and possibly prohibitive for low-memory and low-power hardware implementation. In real-time applications with hard constraints on execution time and storage capabilities, it is very challenging to compute an optimal solution in actual time. To overcome this challenge, alternative approaches that can be employed are discussed in this section.

3.3.1. Explicit/Multi-Parametric MPC

An approach to overcome implementation complications related to demanding online computations of traditional MPC is the so-called explicit/multi-parametric MPC (mp-MPC) that has been proposed in [73,74]. The online optimization problem solved in MPC is now solved offline by leveraging multi-parametric programming to obtain the optimal control laws as explicit functions of the states/outputs (defined as the parameters of the system) and the "critical regions" where these functions are valid. Therefore, the online optimization involved in MPC is replaced by simple function evaluations in the form of a look-up table that leads to a significant decrease in the on-line computational effort. Hence, mp-MPC is regarded as a potential alternative for low-power hardware implementations [75,76]. The idea of an explicit MPC-based AP system has been proposed in [16,77,78] with simplified computational models.

Increasing the complexity of the system's description produces a significant increase in memory demands that could be prohibitive for hardware implementation. In fact, increasing the dimensions of the states and inputs, the length of the prediction horizon or the number of constraints of the optimization problem can lead to very expensive off-line computations and an increase in the number of critical regions. The increased complexity of the solution and the large number of critical regions can consequently affect the efficiency of the online search algorithm. The disproportionate scaling of the memory, the off-line computational demands and the complexity of the online search algorithm with the increase of the problem's complexity has been recognized widely as a barrier for further consideration of the mp-MPC approach in real-time applications [79–81]. Many studies have tried to address this problem and suggested countermeasures based on the problem's formulation and solution. The common rule is to reduce the dimension of the states and prediction horizon or the number of dynamics in the case of hybrid systems. However, this could lead to significant loss of optimality and, potentially, the infeasibility of the computed control action. An effective solution with direct applicability is the inclusion of manipulated variable blocking [82,83]. In essence, the manipulated variables remain constant during specified intervals, which implies that the number of optimization variables is decreased. Therefore, the computational complexity of the corresponding mp-MPC problem is also decreased. Complexity reduction of the calculated explicit solution is suggested in [84,85], where the number of critical regions is decreased, but the optimality of the simplified feedback is maintained. However, due to their post-processing nature, these approaches may be inappropriate for certain applications where the offline computation time is critical. An alternative approach is the development of approximation designs and search algorithms that produce approximate descriptions of the explicit sub-optimal solutions for reducing both online and offline computations [86–89]. An alternative, effective way to decrease storage costs and reduce complexity in explicit model predictive controllers (EMPCs) is by the characterizing the EMPC surface using regression or interpolation methods [90–93].

Therefore, the successful implementation of explicit MPC on embedded platforms for a particular application depends on the system formulation (prediction horizon, number of states and inputs), the available resources and, if necessary, the applicability of a potential approximation algorithm.

3.3.2. Modified On-Line MPC for Embedded Control Systems

Alternative approaches for online MPC implementation have been developed with the main objective being to decrease the required time for online execution of the computations in order to fulfil the time constraints imposed by the hardware resources. The efficiency of implementing online MPC in microscale devices is dependent not only on the optimization problem inherent to the MPC algorithms, but also on the choice of the optimization solvers. In [94], a direct application of Newton's method is considered that results in solving an unconstrained nonlinear problem and presents direct applicability to different problem formulations. Then, the precision of the arithmetic operations of the optimization problem is reduced [95] using a logarithmic number system (LNS) microprocessor-based architecture [96]. LNS demonstrates a significant decrease in memory requirements and an increase in optimization speed compared to arithmetic units. Another common approach is the implementation of the warm-start procedure. The previously computed control trajectory for the next control horizon is suitably shifted to be used as a starting point for computing the current optimal controller. This leads to a reduction in the number of iterations in comparison to cold-start, with some smoothness assumptions on the optimal MPC space. Alternatively, an option could be to terminate the optimization procedure early, so long as the optimization method produces feasible solutions at each iteration. The early termination of an appropriate interior-point method resulted in high quality control, as reported in [80]. Similar to the explicit MPC, manipulated variable blocking is an efficient method to reduce the complexity of the computations. Additionally, evaluation points can be introduced to evaluate the control variables at specific time instants and not over the entire prediction horizon [83]. Another approach that can be employed for complexity reduction is linear interpolation between precomputed predicted trajectories with desirable characteristics [97,98]. Each predicted trajectory is tuned to achieve different objectives, i.e., optimal performance or maximum feasibility. Then, interpolation between the predictions is performed to obtain the best performance subject to feasibility [99]. This method achieves larger feasibility regions for the same computational load and a performance close to the exact MPC, but with reduced computational burden. A conceptually similar idea that aims to meet the performance objectives within a limited time frame is to derive alternative control laws off-line and then online to choose the one that gives the best performance using a scheduler or supervisor [100]. A feedback scheduler is suggested in [101,102] that uses feedback information from the optimization algorithm to dynamically allocate CPU time to tasks and to determine when to terminate the optimization problem and output the control signal. A hybrid logic MPC is suggested in [94] to decrease further the computational complexity. A less expensive control-scheme is employed when the output reaches the set-point. In this particular example, the optimizations are aborted when the output lies within a range. Other approaches based on suboptimal MPC strategies are investigated in [103,104] that reduce the online computational effort of MPC.

3.3.3. Optimization Algorithms and Solvers

As discussed in the previous section, the most demanding part of MPC is the repetitive solution of the optimization problem (3). Therefore, the choice of the optimization algorithm and solver for embedded control applications is crucial. The solver’s efficiency is directly related to the real-time specifications of the available hardware platform as discussed in [105]. In essence, the runtime specifications impose a limit on the number of iterations that can be performed by the solver, which subsequently affects the level of achieved optimality. Moreover, reduced solver runtimes imply decreased power consumption that can considerably contribute to extending the life of the battery that powers the system. Two of the most commonly-used optimization algorithms that are briefly discussed here are active-set and interior point.

3.3.3.1. Active-Set Method

This is a very popular method that is based on defining (at each time step) a set of constraints that are active at the solution and treated as equalities. The active-set method iteratively adjusts the working set, i.e., an estimate of the active set, since constraints may become active or inactive at every time step. The process terminates when the current iterate minimizes the cost function over the working set and all Lagrange multipliers related to the constraints of the working set are non-negative. The major advantage of the method is the fast solution of the optimization problem since only active constraints are considered for every iteration. However, since the active set varies from step to step, the structure of the problem may also change.

3.3.3.2. Interior Point Method

The notion of the interior point method (IPM) involves posing the inequality-constrained optimization problem as a sequence of a system of linear equations [106] that can be solved efficiently by using Cholesky factorization or exploiting the sparsity of the structure of matrices involved. The interior point method provides an attractive alternative to the active-set algorithm because the dimension and structure of the system of linear equations is invariant across iterations [107]. In [108], a comparison of the active-set method (ASM) and IPM performance when implemented on an FPGA was made. It is suggested that ASM performs better for small-scale applications, while IPM should be preferred in the case of larger applications. It was also shown that the storage requirements increase exponentially with the number of inequality constraints for ASM compared to IPM, which demonstrates a linear storage requirement trend.

Open-source software packages can be used to implement the afore-mentioned optimization algorithms in embedded platforms. These software tools can generate custom optimization codes using high-level programming languages, such as C/C++ or Python, that are suitable for real-time applications. The solvers interfacing with C that were used in this work are briefly presented in the following.

(1) *qpOASES* (Quadratic Programming Online Active Set Strategy)

qpOASES [109] is an open source software package that implements a parametric active-set method. This method is inspired by parametric quadratic programming, and it is built on the idea that the active set does not change much from one quadratic programming problem (QP) to the next; since the move from the old QP to the new is made in a convex set, the feasibility of all QPs is ensured [110]. *qpOASES* was originally implemented in C++, but a C version can now be used. It is programmed generally to handle any problem specifications, and therefore, the storage requirements do not change with the problem size.

(2) *CVXGEN* (Code Generator for Embedded Convex Optimization)

CVXGEN [111] is an open source software tool that implements the Mehrotra predictor-corrector IPM. The advantage of *CVXGEN* is that it generates a custom library-free solver while taking into account the problem structure and performing pre-conditioning accordingly to boost solution speed [111]. Additionally, its implementation is straightforward and tailored for seamless integration into the MPC algorithm. Although *CVXGEN* works well for small-sized problems with up to 4000 KKT (Karush–Kuhn–Tucker) coefficients in the constraints and objective, it cannot handle larger problems, for example an MPC with a large predictive horizon.

(3) *ECOS* (embedded conic optimization solver)

ECOS [112] is an open-source numerical software package that implements a standard Mehrotra predictor-corrector interior-point algorithm. It is designed to compute the solution of second-order cone optimization problems (SOCP). A quadratic programming MATLAB interface rewrites the QP through an epigraph reformulation of the objective function as a SOCP to be further handled by *ECOS*. *ECOS* code is short and simple with a low footprint, making it suitable for embedded applications.

(4) *QPC* (Quadratic Programming in C)

QPC [113] is a quadratic programming solver library written in C interfaced with MATLAB. It provides the option to use ASM or IPM algorithms to obtain the optimal solution. In particular, *qpas* (quadratic programming active set) uses a dual active-set algorithm and *qpip* (quadratic programming interior point) a primal-dual interior-point algorithm to solve strictly convex QPs.

(5) *FORCES* (Code Generator for Fast Optimization) *Pro*

FORCES [114] is an extension of *ECOS* designed specifically for MPC applications to linear and nonlinear systems. The toolbox enables automatic code generation for embedded design, making it an attractive proposition for integration into the AP system. However, since it is a specialized MPC solver and not a generic QP solver, a comparison to the other QP solvers detailed above is not a fair comparison. Therefore, *FORCES Pro* is not considered in the comparative study that follows.

(6) *CVXOPT* (Python Software for Convex Optimization)

CVXOPT [115] is an open source software package for convex optimization written in Python. The implemented algorithm is a standard Mehrotra predictor-corrector interior-point algorithm.

(7) *CVXPY* (Python-Embedded Modeling Language for Convex Optimization Problems)

CVXPY [116] is an open source software package that converts the optimization problem, specified by the user, into a standard, conic form and interfaces with a solver to obtain the optimal solution. The solvers called by *CVXPY* are *CVXOPT* and *ECOS* for small–medium-scale problems and *SCS* (splitting conic solver) [117] for large-scale problems.

3.3.4. Performance Comparison

The performance of the solvers is evaluated for zone-MPC as described in [62] and presented in (3) for the AP system. The zone-MPC incorporates a hybrid logic, which essentially states that the controller is activated when the output lies outside a zone that can periodically change. The zone represents the clinically safe range of glucose concentration.

We present the following methodological framework (Figure 3) that describes the steps towards the implementation of our optimization problem in an embedded setting. Our objective is to determine which algorithm (not necessarily which solver) performs best with small runtimes and memory footprint and is therefore suitable for embedded implementation. Initially, we perform a series of simulation studies to evaluate the performance of the previously presented QP-solvers in a desktop environment using MATLAB (2015b, The MathWorks Inc., Natick, MA, USA). This step is a high-level assessment of the available solvers based on their suitability for embedded implementation. For the field of AP, this step usually is the starting point for any control design evaluation, since the Universities of Padova/Virginia metabolic Simulator (UVA/Padova) [118], an FDA-accepted tool, is implemented in MATLAB. According to Figure 3, we propose an intermediate step for the transition between desktop implementation and dedicated hardware design. This intermediate step is the use of an embedded prototype platform, in our case a Raspberry Pi Zero and a Raspberry Pi 3 Model B (https://www.raspberrypi.org/). The advantage of this step is that we will not focus on the explicit design of the system on chip as it is readily available; instead, we focus on deploying our control algorithm from this embedded device and address implementation challenges, such as fixed-point/floating precision-related numerical errors and data communication. Therefore, the performance of the QP solvers is evaluated in preliminary hardware-in-the-loop simulation studies using the Raspberry Pi as a development tool. The final step is the evaluation of the solver that the previous steps indicated as the most appropriate in a dedicated system-on-chip (ASIC) that meets the requirements of the AP application. This final step has not been investigated thoroughly and is the subject of future work.

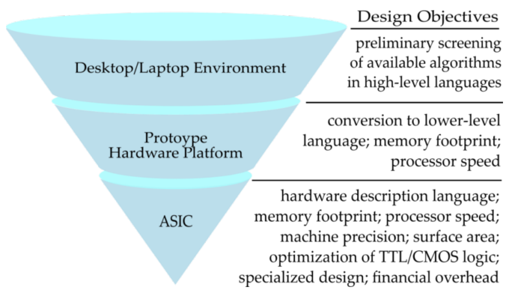
Design Objectives
Desktop/Laptop Environment
preliminary screening of available algorithms in high-level languages
Protoype Hardware Platform
conversion to lower-level language; memory footprint; processor speed
ASIC
hardware description language; memory footprint; processor speed; machine precision; surface area; optimization of TTL/CMOS logic; specialized design; financial overhead

Figure 3. Methodological framework towards an Application-Specific Integrated Circuit (ASIC) implementation.

(1) MATLAB implementation

All numerical examples are performed using MATLAB (2015b, The MathWorks Inc., Natick, MA, USA) running on a computer with two Intel Xeon ES-2623 processors with 32 GB of RAM (Intel, Santa Clara, CA, USA). A 30-h closed-loop in silico experiment is run using the UVA/Padova metabolic Simulator (The Epsilon Group, Charlottesville, VA, USA) [118], with unannounced meal disturbances of 50 g and 70 g of carbohydrates at 17:00 p.m. and 11:00 a.m., respectively. The zone-MPC problem leverages a discrete-time linear model containing three states, one control input and one control output, which is an abstraction of the T1DM patient. Two sets of time horizons are evaluated, namely (i) N_p = 20, N_c = 10 and (ii) N_p = 12, N_c = 5, to investigate the solvers' performance for different problem sizes. The resulting quadratic program, which is to be solved every 5 min, involves $4N_c + 2(N_p - N_c)$ optimization variables and $8N_c + 4(N_p - N_c)$ inequality constraints.

Table 2 summarizes the execution time required for each of the previously-described C solvers to solve the QP problem (3) for both sets of prediction and control horizons. Multiple runs of the in silico experiment are performed, and 4000 time points are generated in total. The average runtime, the standard deviation and the worst case runtime are presented for every solver. The runtime indicates the computational effort required for the solver to provide the optimal solution. The possibility of the warm-start procedure for each solver is reported and compared with cold-start. MATLAB's native quadratic programming solver *quadprog* is also tested to compare its performance with the C-based solvers and to note the acceleration in runtime provided by the C solvers. It should be highlighted that all solvers converge to the same optimal insulin dosing trajectory during multiple runs of the experiment with both short and longer predictive horizons and their corresponding control horizons.

Table 2. Solver runtime performance computed for 4000 time points; t_{aver}, $t_{\max}$ and SD (standard deviation) are the average, the maximum and the standard deviation of the solver runtime in ms observed over the duration of the experiment; IPM (interior point method) and ASM (active-set method) are the interior point method and the active-set method, respectively, and Y/N indicate the inclusion or not of the warm-start procedure. In detail, the examined solvers are Quadprog (quadratic programming) of Matllab, *qpOASES* (quadratic programming online active set strategy), *CVXGEN* (code generator for embedded convex optimization), *QPC* (quadratic programming in C) using either *qpas* (quadratic programming active set) or *qpip* (quadratic programming interior point) and *ECOS* (embedded conic optimization solver).

Solver	***Quadprog***	***Quadprog***	***Quadprog***	***qpOASES***	***qpOASES***	**CVXGEN**	**QPC (*qpas*)**	**QPC (*qpip*)**	***ECOS***
Algorithm	IPM	ASM	ASM	ASM	ASM	IPM	ASM	IPM	IPM
Warm-start?	N	N	Y	N	Y	N	N	N	N
Case 1: Np = 20, Nc = 10									
t_{aver} (±SD) (ms)	8.6 (±3.3)	12.9 (±20.6)	9.9 (±14.5)	13.7 (±6.9)	12.7 (±6.1)	N/A	**0.4 (±0.3)**	4.3 (±1.9)	7.2 (±2.5)
t_{max} (ms)	34	114	93	49	48	N/A	**2**	15	30
Case 2: Np = 12, Nc = 5									
t_{aver} (±SD) (ms)	7.9 (±2.8)	6.6 (±7.6)	5.8 (±6.0)	2.5 (±1.4)	2.5 (±1.3)	3.0 (±0.7)	**0.2 (±0.15)**	1.6 (±0.7)	6.6 (±2.3)
t_{max} (ms)	33	46	53	22	10	8	**2**	5	39

It can be deduced from Table 2 that for the problem with longer horizons, the most efficient solver is *qpas* with an average runtime equal to 0.4 ms with a standard deviation of 0.3 ms. The solvers *qpip* and *ECOS* follow with computation times of 4.3 (±1.9) and 7.2 (±2.5) ms, respectively. For this case, *CVXGEN* is not able to generate a solver because the problem exceeds the number of KKT multipliers the software can handle. *qpOASES* requires the most time to compute the solution with an average runtime of 12.7 (±6.1) ms even with warm-start.

For the case of smaller time horizons, again *qpas* and *qpip* perform very efficiently with an average runtime of 0.2 (±0.15) and 1.6 (±0.7) ms, respectively. For *qpOASES* with warm-start, the runtime reduced by 80% compared to the larger-sized problem without a large degradation in control performance. Once more, the warm-start procedure does not provide any noticeable improvement in runtime characteristics compared to cold-start. A reason for this is that for most of the simulation time, the glucose is within the prescribed zone, and so, the optimal insulin to be applied is known to be the basal rate, thereby obviating the need for warm-starting in the zone-MPC formulation. *CVXGEN* also presents a very robust performance considering that it provides a small runtime value in the worst case and the standard deviation is relatively small. In the case of quadprog, IPM outperforms ASM in the case of the larger-sized problem, whereas in the smaller-sized problem, ASM presents better runtime characteristics. For both cases, the warm-start procedure for ASM improves the performance of the solver.

Since the most appropriate values of N_p and N_c for this application are those of the second case as also reported in [62], we can conclude that the best choice for further consideration is *qpas* for the active-set algorithm if runtime minimization is the highest priority during design. The rationale behind the study of runtimes is to obtain an insight into the complexity of the solvers' algorithms. We expect that the ratio of runtimes provides an estimate of complexity, with small runtimes indicating lower computational resource requirements. From Table 2, we conclude that *qpas* performs satisfactorily with small runtimes, indicating that this implementation of the active-set algorithm may be tractable for the embedded implementation. An additional metric of the algorithms' capabilities and limitations is the resulting code-size/storage requirements, presented in Table 3. We infer that for the smaller time horizon problem, the solver *ECOS* has the minimum memory footprint, while the *CVXGEN* and *qpOASES* code size can be prohibitive for the available hardware platforms.

Table 3. Solvers' storage requirements for the examined problem characteristics.

Solver	*qpOASES*	*CVXGEN*	*ECOS*	*qpas*	*qpip*
Code size (KB)	835	1800	7	25	43

(2) Raspberry Pi implementation

Tables 2 and 3 indicate that *ECOS* is an efficient solver that performs well for the examined optimization problems while it has the minimum memory footprint. This solver is based on *CVXOPT* [112], which is a Python solver implementing an IPM algorithm. *CVXPY*, on the other hand, uses either *ECOS* or *CVXOPT* for small-scale problems to obtain the optimal solution. Therefore, the performance of the zone-MPC that leverages *CVXOPT* and *CVXPY* solver protocols is evaluated with preliminary hardware-in-the-loop simulation studies to provide estimates of runtime and energy consumption on an embedded device. The previously-defined experiment of two unannounced meals of carbohydrates is used. The zone-MPC algorithm is deployed from the Raspberry Pi Zero and Raspberry Pi 3 Model B. The operating system of both models is a dedicated version of Linux known as Raspbian. The RPi 3 has 1 GB of RAM, while the RPi Zero 512 KB of RAM. The zone-MPC code is translated from MATLAB to Python, since the Raspberry Pi compiles Python code. A TCP/IP connection is used to receive virtual glucose data (CGM) wirelessly from the simulator that runs on a desktop and sends the suggested control action (u) back to the simulator, as presented in Figure 4.

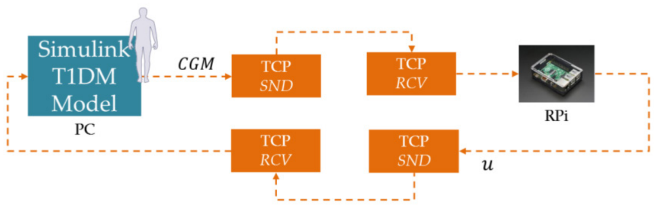

Figure 4. TCP (Transmission Control Protocol) communication protocol for RPi (Raspberry Pi) and PC. RCV and SND indicate receive and send data respectively.

The code is designed to internally set the floating point precision to 32 and 64 bits for the construction of the QP parameters, MPC constraint updates and the communication between MATLAB and Raspberry Pi. An estimate of the average energy in mAh consumed over the duration of the experiment is calculated using a DROK USB 2.0 digital multimeter (DROK, Service@droking.com) to measure the average current (in mA) entering the hardware platform. An estimate of the total energy consumed is found by multiplying the average current by the total simulation time. The results of runtime and energy consumption are summarized in Table 4. Results of glucose concentration profiles and the corresponding insulin actions obtained by the zone-MPC when implemented on the PC, the Raspberry Pi 3 and the Raspberry Pi Zero (blue, yellow and red lines) are presented in Figure 5.

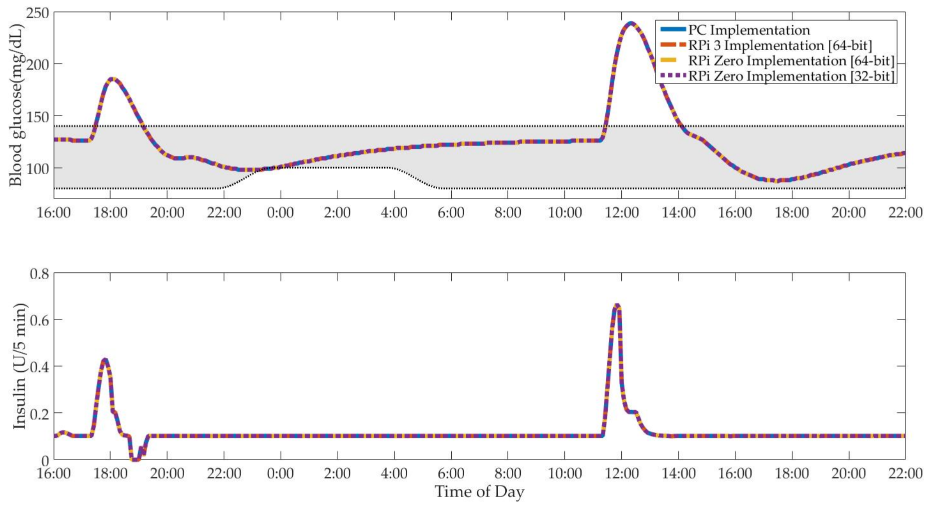

Figure 5. (Top) Comparison of glucose regulation with zone-MPC implemented on a PC, on Raspberry Pi 3 and Raspberry Pi Zero for 50-g and 70-g unannounced meal (blue, red, yellow lines). Comparison of glucose regulation with zone-MPC implemented on Raspberry Pi Zero with multiple bit precisions for quadratic programming problem (QP) solving and associated computations (yellow and purple lines). (Bottom) Comparison of optimal insulin levels computed by the different implementations.

Figure 5 demonstrates that the solvers implemented on Raspberry Pi 3 and Raspberry Pi Zero converged to the same optimal solution that is found in the MATLAB implementation, and therefore, identical glucose profiles are obtained. It should be noted that for this comparison study, 64-bit precision is considered.

Table 4. Solver runtime performance computed for 4000 time points; t_{aver}, t_{max} and SD are the average, the maximum and the standard deviation of the solver runtime in ms observed over the duration of the experiment; the average and the standard deviation of the total energy in mAh consumed over the duration of the experiment are reported for both platforms.

Prototype Platform	**Raspberry Pi Zero**		**Raspberry Pi 3**	
Solver	***CVXOPT***	***CVXPY***	***CVXOPT***	***CVXPY***
		Case 1: $N_p = 20$, $N_c = 10$		
$t_{\text{aver}} \pm$ SD (ms)	128.8 ± 14.1	189.7 ± 16.3	50.0 ± 10.2	73.7 ± 11.7
t_{max} (ms)	196.8	268	104.8	143.2
Energy Consumed (mAh)	39.9 ± 0.2	59.9 ± 0.4	16.5 ± 0.0	25.1 ± 0.1
		Case 2: $Np = 12$, $Nc = 5$		
$t_{\text{aver}} \pm$ SD (ms)	56.9 ± 9.2	104.9 ± 11.3	22.4 ± 7.8	41.2 ± 9.6
t_{max} (ms)	110.3	269	40.2	98.2
Energy Consumed (mAh)	16.5 ± 0.2	32.4 ± 2.0	6.7 ± 0.4	13.2 ± 0.1

Table 4 shows that for both hardware testing platforms, *CVXOPT* computes optimal MPC actions almost twice as fast as *CVXPY*. As expected, the execution times obtained from the Raspberry Pi Zero are slower than those reported for the Raspberry Pi 3 because Raspberry Pi Zero has a single-core CPU, whereas Raspberry Pi 3 a quad-core. The energy consumed (measured via the aforementioned multimeter) over the duration of the experiment is decreased for *CVXOPT* by approximately 50% and 95% for the long and short prediction horizons respectively in both platforms.

It can also be deduced from Table 4 that the Raspberry Pi Zero requires more energy than the Raspberry Pi 3 for longer simulation periods. This is because of the increased execution time of the Raspberry Pi Zero and also because the average current entering the device does not differ significantly from the Raspberry Pi 3 due to significant power used in communication and peripheral computing (for example, updating the state, computing the MPC constraints).

Additionally, we performed a preliminary robustness testing of the glucose regulation performance on the Raspberry Pi 3 using *CVXOPT* with 32- and 64-bit precision levels that are supported by the *numpy* package in Python. From Figure 5, we can see that glucose and insulin profiles obtained with floating point precision set to 32 bits are identical to those obtained with the precision of 64 bits (yellow and purple lines). Therefore, a floating point precision of 32 bits can be considered for safe glycemic regulation with our current precision-agnostic design; there is no need to have a 64-bit design since that has a higher memory requirement without offering an ascertainable improvement in performance.

As illustrated in the above screening process, the intermediate step of hardware implementation on a prototyping platform such as the Raspberry Pi provides keener insights into embedded implementation as compared to a strictly high-level implementation on a desktop. The comparison of 32- and 64-bit precision indicated that the current implementation of our control problem can be supported by 32-bit precision resulting in decreased storage requirements and, therefore, memory savings. Conversely, choosing a microcontroller offering a 64-bit implementation will increase the manufacturing cost without providing better control performance. A potential hardware that enables low floating point precision (16 or eight bits) requires a revised design approach of our MPC control problem (for example, alteration of the MPC parameters) to ensure safe glucose regulation. The end goal is to find a compromise to the inherent trade-off between hardware selection and controller performance implementation. Other trade-offs include: selection of ROM and RAM memory, wireless communication methods, controller design flexibility, available hardware platforms and economic constraints, to name a few. Our proposed intermediate step leverages a suitable prototyping environment to investigate these objectives in order to inform the next (and perhaps the most important) step of the embedding process: the selection of the chip for deploying the AP.

4. Brief Overview on Design Approaches

The design of an embedded AP system is a complex and challenging procedure. Firstly, the requirements of the system in terms of performance, manufacturing cost, power, size and weight need to be determined. Depending on the type of insulin pump considered, i.e., tubed or patch, different requirements must be met for the underlying system-on-a-chip. The tubed-pump can last up to five years, and therefore, the control unit should be designed with a long-term perspective; conversely, patch pumps are replaced every three days, and hence, the design specifications are completely different. Consecutively, these requirements must be translated to detailed and formal specifications that will then be used to determine the appropriate architecture design for both software and hardware. It is crucial that the chosen architecture satisfies the constraints imposed by both the functional and the non-functional requirements [119]. Safety is an additional, but crucial design parameter. A safety-aware design procedure is developed as such to guarantee system safety in spite of exogenous disturbances or unexpected failure modes.

When the various components are combined together, the compliance with the overall requirements of the integrated system is evaluated. The validation process evaluates whether the developed system performs exactly as planned. For safety-critical systems, it is of paramount importance that the final product integrating both hardware and software is formally certified [120,121]. An overview of the design, verification and validation methods for medical devices can be found in [122–124]. The performance of different control algorithms and formulations can be evaluated using a prototyping environment before testing with the actual device, to ensure that the suggested control method is suitable for the application [125]. In fact, hardware-in-the-loop (HIL) simulation has been gradually considered as the necessary step from pure virtual simulation to full physical prototyping [126] for many safety-critical applications. Field programmable gate arrays (FPGAs) have been considered as very useful platforms for MPC hardware prototyping, especially for the development of customized boards. The flexibility to design the entire circuit including the processor and peripherals provides the designer with the ability to create an application-specific optimized design at both the hardware and software level. On the other hand, the advent of flash memory in modern microcontrollers has enabled their use as prototyping platforms.

An open source FPGA prototyping toolbox for embedded optimization has been recently developed [127] to allow fast designing, validation and prototyping of algorithms written in C or MATLAB on FPGA platforms. An FPGA-in-the-loop simulation experiment was performed in [128] to evaluate the performance of an MPC controller for an aircraft application with a relatively large number of inputs and constraints. The problem involved not only an MPC solver based on primal-dual IPM, but also a target calculator using a fast gradient method. The required custom solutions for the implementation of MPC solvers such as ASM, IPM or fast gradient method (FGM) on FPGAs have been exploited in [108,129,130]. In [131] the authors propose a co-design methodology for MPC implementation on a chip that resulted in a dual architecture consisting of a general purpose microcontroller and a matrix coprocessor. The matrix coprocessor is responsible for all of the computationally-demanding operations. The final design was downloaded into an FPGA for verification and validation. An explicit-MPC scheme for the control of intravenous anesthesia was implemented and prototyped into an FPGA [132,133]. The performance of the controller was validated with HIL simulation. Moreover, many low-power microcontrollers have been used for controller-in-the-loop hardware prototyping. For example, a LEGO NXT testing platform was used in [134] for the implementation of MPC based on the fast gradient method to control the position of a Segway-like mobile robot. An Arduino-based microcontroller was used in [135] to implement an optimized PID that uses MPC as a reference governor; where the reference governor optimization problem is solved parametrically in an effort to decrease the online computational effort; and finally, in [136], MPC implementation requirements were exploited using the Cortex-A family of ARM-based microcontrollers.

5. Key Challenges of Embedded Artificial Pancreas Systems

5.1. Known and Foreseeable Hazards Associated with the Operation of the AP System

The identification and analysis of known and foreseeable hazards is critical for the design of a safe and reliable system. The potential hazards that can occur during the operation of the AP system are illustrated in Figure 6 and explained in Table 5. Both Configurations A and B (Figure 6a,b) are considered. A failure of the AP system can be defined as either: extreme and prolonged hypoglycemia (denoted as C1 in Figure 6) or extreme and prolonged hyperglycemia (denoted as C2 in Figure 6). These are two life-threatening medical states, and, therefore the system must prevent them from occurring. Although the hypo- or hyper-glycemic conditions are the most unsafe, there are numerous other faults and errors that potentially lead to failure modes. Safety issues related to each component and the complete AP system are described in [137]. Faults related specifically to the insulin pump operation and communication channels have been studied in [138–140].

Here, we will focus on potential faults due to MPC implementation on limited power resources, and we will compare the two different configurations. In the case of MPC, either online or explicit, it should be guaranteed that a feasible solution can always be obtained; otherwise the control unit would fail. This can be accomplished during the control design phase by modifying the optimization problem to ensure recursive feasibility. As described in Section 3.3, strict runtime deadlines imposed by the hardware can restrict the number of iterations that can be performed, and so, suboptimal solutions are obtained. The level of potential performance degradation that can be allowed has to be carefully considered in order not to promote the hyper- or hypo-glycemic state.

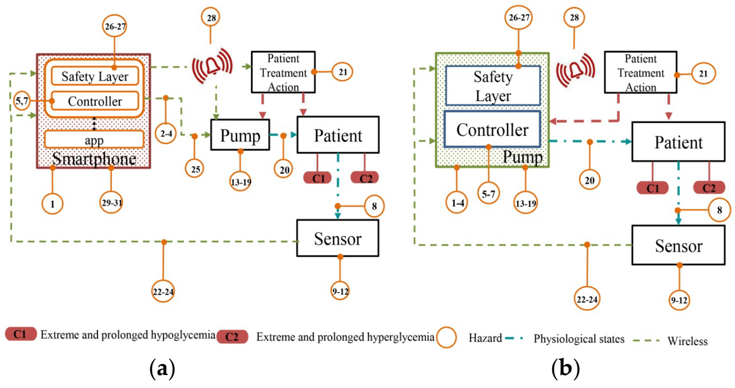

Figure 6. Potential hazards in the AP system: (**a**) controller as a smartphone app; (**b**) controller embedded in the pump.

Table 5. Foreseeable hazards in the AP system.

Cause	**Hazard**
Control Unit	(1) Battery failure
	(2) No value displayed
	(3) Over/under delivery of insulin
	(4) Over delivery of insulin
Control Algorithm	(5) Numerical inaccuracies
	(6) Resource limitation-suboptimal control action
	(7) Software aging
CGM (continuous glucose monitoring)	(8) No signal
	(9) Under/over reading
	(10) No glucose display
	(11) No communication
	(12) Receiver displays failure
Insulin Pump	(13) Failure in reservoir
	(14) Failure in delivery line
	(15) Battery failure
	(16) Mechanical/electrical failure
	(17) Request for new delivery while delivering previous bolus
	(18) Insulin delivered at maximum infusion rate (60 U/h)
	(19) Over/under/no delivery of insulin
User	(20) Infusion set disconnected
	(21) Over/under delivery of insulin
Communication	(22) No communication between CGM and transmitter
	(23) No communication between CGM and control unit
	(24) Loss of communication between CGM and insulin pump
	(25) No communication between control unit and insulin pump
Safety Layer	(26) Algorithmic fault
	(27) No detection made
	(28) Failure to trigger an alarm
Handheld Device	(29) Operating system upgrade
	(30) Interference with other applications
	(31) Cybersecurity vulnerabilities

The numerical performance of the iteration algorithms used to solve the optimization problem depends on whether floating or fixed point arithmetic is used. The floating point arithmetic employed by the smartphone configuration is subject to numerical round-off errors and accumulation of these errors that should be accounted for [141]. The use of fixed-point precision in efficient embedded systems results in quantization errors that could lead to degradation in closed-loop performance. A problem of interest, therefore, is to develop fast MPC methods robust to quantization effects. This is a relatively open problem, with only a few methods proposed in the literature to design MPC for systems with quantized inputs, or equivalently, systems with finite/discrete input alphabet. An early attempt at guaranteeing the stability of receding horizon control of linear time-invariant (LTI) systems with quantized inputs is presented in [142], by estimating a region of attraction of the quantized MPC (Q-MPC). A terminal state penalty term formulation based on an algebraic Lyapunov equation is proposed in [143] for open-loop-stable LTI systems for analyzing the closed-loop stability of the Q-MPC-controlled system. Modern Q-MPC methodologies pose the quantization effects as an additive disturbance term, and robust control methods are leveraged to provide closed-loop stability guarantees; see, for example, [144]. Robust tube-based MPC formalisms, such as those studied in [145,146], can then be used to construct Q-MPCs for nonlinear systems that ensure ultimate boundedness or robust stability in a closed-loop. Recently, the authors in [147] demonstrated that such tube-based Q-MPCs can result in conservative controllers because the worst-case quantization error is propagated to every step of the predictive horizon during control action computation. To ameliorate this issue, a practical stability approach with one-step MPC was proposed in [148] yielding less conservative ultimate bounds on the closed-loop trajectories. Although not verified, we believe the method in [93] could be extended seamlessly to construct explicit MPCs for systems with quantized inputs.

Another challenge relevant to embedded MPC is 'software aging' that is originated by memory bloating and leaking, data corruption or storage space fragmentation [49] and can progress with the extensive usage of the system, resulting possibly in system failure.

The resource capabilities of a smartphone motivate the development and implementation of advanced controllers that can provide optimal blood glucose regulation. However, the inclusion of an additional device accompanies the development of new hazards and potential errors in the AP system, as seen in Figure 6a. Battery failure, loss of wireless communication with the other AP components, compatibility issues during software upgrades, interference with other apps and vulnerability to network threats are some of the possible hazards that should be taken into consideration for the development of a reliable system. On the other hand, the insulin pump system-on-a-chip provides a more robust configuration because communication issues are restricted since the number of involved devices is reduced and also the system is treated as a stand-alone medical device whose reliability and integrity will be formally certified by the manufacturer. However, issues of battery drain, power and runtime capacities should be considered when advanced algorithms are used. The following subsection discusses in more detail challenging issues related to system communication in terms of security and confidentiality.

5.2. Communication: Security and Confidentiality

The communication between the involved components of an AP system introduces significant challenging issues and potential vulnerabilities in the system. The continuous blood glucose monitor communicates wirelessly with the pump to transmit the current glucose concentration values. The security and integrity of these "data in motion" is of high importance, especially in a closed-loop system because the patient's treatment is based on the reliable transmission of glucose data. Moreover, the AP system is integrated wirelessly with peripheral devices, such as a remote control, a PC, a mobile phone or a tablet, where the history of the patient's data (glucose values, insulin infusions, meal announcements and other "data at rest") is gathered and used as a helpful tool for the patients to acquire familiarity with their blood glucose regulation and as the physician's support platform for future alterations in patient's treatment. The peripheral devices can pose additional security challenges in the AP system [149].

In [150,151] it was demonstrated that malicious attacks can compromise the privacy and integrity of the AP system. The wireless channel for communicating between the insulin pump and the CGM, the remote control and peripheral devices is vulnerable to seemingly innocuous attacks that can be launched using public-domain information and off-the-shelf hardware. As a result, erroneous insulin treatment can arise that jeopardizes the patient's health. The development of responsive medical devices to cybersecurity vulnerabilities is now more important than before because the once isolated and stand-alone medical devices are integrated with networking, software, operational systems that are widely exposed to threats [152,153]. A brief overview of some challenges in designing "high-confidence" medical devices is given in [154].

Other sensitive data include device information (model, serial numbers), physicians and health team center identification, patient identity information, medical condition, health history and treatment history. Access to this sensitive and confidential data should only be allowed to the patient or their legal guardian and their physicians. Attacks that threaten the secure transmission of private data and therefore violate the patient's medical record confidentiality are referred as eavesdropping. Impersonation, on the other hand, refers to failure to authenticate the operation of a medical device that can lead to malfunctions caused by viruses and threats or deliberate illegitimate usage of the device. Finally, jamming attacks refer to attacks that aim for the denial of service (DoS). In [155], potential methods to mitigate these types of threats are discussed for implantable medical devices that include: cryptography [156], either regular or physiological signal (PV) based, anomaly detection, external device deployment and direct sequence-spread spectrum (DSSS) or frequency hopping (FH).

Therefore, AP-specific security guarantees need to be defined, and information security requirements need to be expressed and enforced in order to develop secure and trustworthy AP systems. Each of the AP devices should be secure at the hardware level to ensure that the computations will not be corrupted and at the operational system level to ensure that the system will not be vulnerable to viruses and threats. Two alternative options have been proposed in [150] to defend a system against the risks: (i) a rolling code based on the cryptographic method; and (ii) a defense based on body-coupled communication. In [151], a personalized patient infusion pattern-based access control scheme for a wireless insulin pump is proposed, which can identify abnormal insulin overdose attacks using patient-specific insulin infusion patterns.

Another risk that must be considered is that the system's security can be also threatened through physical access, since the pump's settings can be easily changed through the user interface. An unauthorized alteration in the pump's setting while the system is unattended can pose a serious threat to the security of the system that can put the patients' health at risk. In [149], potential options for risk mitigation are proposed: (i) a fail-safe physical interface for the patient to acquire the device control in case of a lost or stolen remote; and (ii) the ability to disable wireless communication.

6. Conclusions

Embedded control for medical applications is an emerging field that is driven by the need to develop wearable medical devices that actively control the patient's treatment and health management. The miniaturization of the available hardware resources, induced by the required flexibility of a wearable device, imposes stringent design constraints on the system (power, memory, execution time).

We have focused in this paper on the regulation of glucose using embeddable algorithms. In this context, the advantages and the wide use of MPC for glucose regulation within an AP system motivate the development of lightweight MPC adequate for hardware implementation. Approximation techniques, appropriate problem formulation and computational considerations for the optimization algorithms [80,134,157] can be employed to derive low footprint, but efficient MPC controllers. For such safety-critical applications, the controller must be integrated in a system that is able, in the case of external causes that lie outside the area of the controller's responsibilities, to detect any abnormal behavior and to respond adequately. Thus, the system's architecture design at the software and hardware level has to be developed in a safety-aware manner with a high degree of redundancy. Safety is the most crucial parameter that imposes hard constraints on the system requirements and must be satisfied even at the cost of performance degradation. Safety constraints should be strongly connected to and included within the control algorithm. The fact that MPC has been demonstrated to work with safety constraints makes it a great candidate in the context of embedded medical controllers.

In general, a holistic approach that involves engineering, safety and security requirements should be aimed at the design of a system architecture that ensures the reliability of the embedded medical device.

Acknowledgments: The authors would like to thank Gondhalekar, Ravi (Harvard John A. Paulson School of Engineering and Applied Sciences, Harvard University) for helpful discussions. We acknowledge the National Institutes of Health (Grants DP3DK104057, DP3DK101068 and DP3DK094331) and the Juvenille Diabetes Research Foundation (JDRF) (Grant 1-SRA-2016-147-M-R) for their financial support.

Author Contributions: S.Z. and A.C. conceived of and performed the simulation experiments and analyzed the resulting data. S.Z., A.C., E.D. and F.J.D. contributed to writing and correcting the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. National Center for Chronic Disease Prevention and Health Promotion. National Diabetes Statistics Report. 2014. Available online: https://www.cdc.gov/diabetes/pubs/statsreport14/national-diabetes-report-web.pdf (accessed on 5 May 2016).
2. Doyle, F.J., III; Huyett, L.M.; Lee, J.B.; Zisser, H.C.; Dassau, E. Closed-loop artificial pancreas systems: Engineering the algorithms. *Diabetes Care* **2014**, *37*, 1191–1197. [CrossRef] [PubMed]
3. Nimri, R.; Phillip, M. Artificial pancreas: Fuzzy logic and control of glycemia. *Curr. Opin. Endocrinol. Diabetes Obes.* **2014**, *21*, 251–256. [CrossRef] [PubMed]
4. Mauseth, R.; Lord, S.M.; Hirsch, I.B.; Kircher, R.C.; Matheson, D.P.; Greenbaum, C.J. Stress testing of an artificial pancreas system with pizza and exercise leads to improvements in the system's fuzzy logic controller. *J. Diabetes Sci. Technol.* **2015**, *9*, 1253–1259. [CrossRef] [PubMed]
5. Ly, T.T.; Roy, A.; Grosman, B.; Shin, J.; Campbell, A.; Monirabbasi, S.; Liang, B.; Von Eyben, R.; Shanmugham, S.; Clinton, P.; et al. Day and night closed-loop control using the integrated medtronic hybrid closed-loop system in type 1 diabetes at diabetes camp. *Diabetes Care* **2015**, *38*, 1205–1211. [CrossRef] [PubMed]
6. Pinsker, J.E.; Lee, J.B.; Dassau, E.; Seborg, D.E.; Bradley, P.K.; Gondhalekar, R.; Bevier, W.C.; Huyett, L.; Zisser, H.C.; Doyle, F.J., III. Randomized crossover comparison of personalized MPC and PID control algorithms for the artificial pancreas. *Diabetes Care* **2016**, *39*, 1135–1142. [CrossRef] [PubMed]
7. Harvey, R.A.; Dassau, E.; Bevier, W.C.; Seborg, D.E.; Jovanovič, L.; Doyle, F.J., III; Zisser, H.C. Clinical evaluation of an automated artificial pancreas using zone-model predictive control and health monitoring system. *Diabetes Technol. Ther.* **2014**, *16*, 348–357. [CrossRef] [PubMed]
8. Dassau, E.; Brown, S.A.; Basu, A.; Pinsker, J.E.; Kudva, Y.C.; Gondhalekar, R.; Patek, S.; Lv, D.; Schiavon, M.; Lee, J.B.; et al. Adjustment of open-loop settings to improve closed-loop results in type 1 diabetes: A multicenter randomized trial. *J. Clin. Endocrinol. Metab.* **2015**, *100*, 3878–3886. [CrossRef] [PubMed]
9. Stewart, Z.A.; Wilinska, M.E.; Hartnell, S.; Temple, R.C.; Rayman, G.; Stanley, K.P.; Simmons, D.; Law, G.R.; Scott, E.M.; Hovorka, R.; et al. Closed-Loop Insulin Delivery during Pregnancy in Women with Type 1 Diabetes. *N. Engl. J. Med.* **2016**, *375*, 644–654. [CrossRef] [PubMed]
10. Del Favero, S.; Place, J.; Kropff, J.; Messori, M.; Keith-Hynes, P.; Visentin, R.; Monaro, M.; Galasso, S.; Boscari, F.; Toffanin, C.; et al. Multicenter outpatient dinner/overnight reduction of hypoglycemia and increased time of glucose in target with a wearable artificial pancreas using modular model predictive control in adults with type 1 diabetes. *Diabetes Obes. Metab.* **2015**, *17*, 468–476. [CrossRef] [PubMed]
11. Nimri, R.; Phillip, M. Toward Automation of Insulin Delivery—Management Solutions for Type 1 Diabetes. *Endocr. Dev.* **2016**, *30*, 1–13. [PubMed]

12. Qin, S.J.; Badgwell, T.A. A survey of industrial model predictive control technology. *Control Eng. Pract.* **2003**, *11*, 733–764. [CrossRef]
13. Joosten, D.A.; Van Den Boom, T.J.J.; Lombaerts, T.J.J. Fault-tolerant control using dynamic inversion and model-predictive control applied to an aerospace benchmark. In Proceedings of the 17th IFAC World Congress, Seoul, Korea, 6–11 July 2008; pp. 12030–12035.
14. Hrovat, D.; Di Cairano, S.; Tseng, H.E.; Kolmanovsky, I.V. The development of Model Predictive Control in automotive industry: A survey. In Proceedings of the 2012 IEEE International Conference on Control Applications, Dubrovnik, Croatia, 3–5 October 2012; pp. 295–302.
15. Valencia-Palomo, G.; Rossiter, J.A. Efficient suboptimal parametric solutions to predictive control for PLC applications. *Control Eng. Pract.* **2011**, *19*, 732–743. [CrossRef]
16. Dassau, E.; Zisser, H.; Harvey, R.A.; Percival, M.W.; Grosman, B.; Bevier, W.; Atlas, E.; Miller, S.; Nimri, R.; Jovanovič, L.; et al. Clinical evaluation of a personalized artificial pancreas. *Diabetes Care* **2013**, *36*, 801–809. [CrossRef] [PubMed]
17. Hovorka, R.; Elleri, D.; Thabit, H.; Allen, J.M.; Leelarathna, L.; El-Khairi, R.; Kumareswaran, K.; Caldwell, K.; Calhoun, P.; Kollman, C.; et al. Overnight closed-loop insulin delivery in young people with type 1 diabetes: A free-living, randomized clinical trial. *Diabetes Care* **2014**, *37*, 1204–1211. [CrossRef] [PubMed]
18. El-Khatib, F.H.; Russell, S.J.; Magyar, K.L.; Sinha, M.; McKeon, K.; Nathan, D.M.; Damiano, E.R. Autonomous and continuous adaptation of a bihormonal bionic pancreas in adults and adolescents with type 1 diabetes. *J. Clin. Endocrinol. Metab.* **2014**, *99*, 1701–1711. [CrossRef] [PubMed]
19. Gondhalekar, R.; Dassau, E.; Doyle, F.J., III. Periodic zone-MPC with asymmetric costs for outpatient-ready safety of an artificial pancreas to treat type 1 diabetes. *Automatica* **2016**, *71*, 237–246. [CrossRef]
20. Jacobs, P.G.; El Youssef, J.; Castle, J.; Bakhtiani, P.; Branigan, D.; Breen, M.; Bauer, D.; Preiser, N.; Leonard, G.; Stonex, T.; et al. Automated control of an adaptive bihormonal, dual-sensor artificial pancreas and evaluation during inpatient studies. *IEEE Trans. Biomed. Eng.* **2014**, *61*, 2569–2581. [CrossRef] [PubMed]
21. Kovatchev, B.P.; Renard, E.; Cobelli, C.; Zisser, H.C.; Keith-Hynes, P.; Anderson, S.M.; Brown, S.A.; Chernavvsky, D.R.; Breton, M.D.; Mize, L.B.; et al. Safety of outpatient closed-loop control: First randomized crossover trials of a wearable artificial pancreas. *Diabetes Care* **2014**, *37*, 1789–1796. [CrossRef] [PubMed]
22. Keith-Hynes, P.; Mize, L.B.; Robert, A.; Place, J. The diabetes assistant: A smartphone-based system for real-time control of blood glucose. *Electronics* **2014**, *3*, 609–623. [CrossRef]
23. Kovatchev, B.P.; Renard, E.; Cobelli, C.; Zisser, H.C.; Keith-Hynes, P.; Anderson, S.M.; Brown, S.A.; Chernavvsky, D.R.; Breton, M.D.; Farret, A.; et al. Feasibility of outpatient fully integrated closed-loop control. *Diabetes Care* **2013**, *36*, 1851–1858. [CrossRef] [PubMed]
24. Anderson, S.M.; Raghinaru, D.; Pinsker, J.E.; Boscari, F.; Renard, E.; Buckingham, B.A.; Nimri, R.; Doyle, F.J., III; Brown, S.A.; Keith-Hynes, P.; et al. Multinational home use of closed-loop control is safe and effective. *Diabetes Care* **2016**, *39*, 1143–1150. [CrossRef] [PubMed]
25. Renard, E.; Farret, A.; Kropff, J.; Bruttomesso, D.; Messori, M.; Place, J.; Visentin, R.; Calore, R.; Toffanin, C.; Di Palma, F.; et al. Day and Night Closed-Loop Glucose Control in Patients with Type 1 Diabetes under Free-Living Conditions: Results of a Single-Arm 1-Month Experience Compared with a Previously Reported Feasibility Study of Evening and Night at Home. *Diabetes Care* **2016**, *39*, 1151–1160. [CrossRef] [PubMed]
26. Huyett, L.M.; Ly, T.T.; Reuschel-DiVirgilio, S.; Clay, S.M.; Bevier, W.C.; Gondhalekar, R.; Dassau, E.; Forlenza, G.P.; Doyle, F.J., III; Pinsker, J.E.; et al. Outpatient closed-loop control with unannounced moderate exercise in adolescents using zone model predictive control. *Diabetes Technol. Ther.* **2016**, *18*, A24.
27. Kropff, J.; Del Favero, S.; Place, J.; Toffanin, C.; Visentin, R.; Monaro, M.; Messori, M.; Di Palma, F.; Lanzola, G.; Farret, A.; et al. 2 Month evening and night closed-loop glucose control in patients with type 1 diabetes under free-living conditions: A randomised crossover trial. *Lancet Diabetes Endocrinol.* **2015**, *3*, 10–15. [CrossRef]
28. Russell, S.J.; Hillard, M.A.; Balliro, C.; Magyar, K.L.; Selagamsetty, R.; Sinha, M.; Grennan, K.; Mondesir, D.; Ehklaspour, L.; Zheng, H.; et al. Day and night glycaemic control with a bionic pancreas versus conventional insulin pump therapy in preadolescent children with type 1 diabetes: A randomised crossover trial. *Lancet Diabetes Endocrinol.* **2016**, *4*, 233–243. [CrossRef]
29. Elleri, D.; Allen, J.M.; Biagioni, M.; Kumareswaran, K.; Leelarathna, L.; Caldwell, K.; Nodale, M.; Wilinska, M.E.; Acerini, C.L.; Dunger, D.B.; et al. Evaluation of a portable ambulatory prototype for automated overnight closed-loop insulin delivery in young people with type 1 diabetes. *Pediatr. Diabetes* **2012**, *13*, 449–453. [CrossRef] [PubMed]

30. Thabit, H.; Tauschmann, M.; Allen, J.M.; Leelarathna, L.; Hartnell, S.; Wilinska, M.E.; Acerini, C.L.; Dellweg, S.; Benesch, C.; Heinemann, L.; et al. Home use of an artificial beta cell in type 1 diabetes. *N. Engl. J. Med.* **2015**, *373*, 2129–2140. [CrossRef] [PubMed]
31. Thabit, H.; Lubina-Solomon, A.; Stadler, M.; Leelarathna, L.; Walkinshaw, E.; Pernet, A.; Allen, J.M.; Iqbal, A.; Choudhary, P.; Kumareswaran, K.; et al. Home use of closed-loop insulin delivery for overnight glucose control in adults with type 1 diabetes: A 4-week, multicentre, randomised crossover study. *Lancet Diabetes Endocrinol.* **2014**, *2*, 701–709. [CrossRef]
32. Tauschmann, M.; Allen, J.M.; Wilinska, M.E.; Thabit, H.; Stewart, Z.; Cheng, P.; Kollman, C.; Acerini, C.L.; Dunger, D.B.; Hovorka, R. Day-and-night hybrid closed-loop insulin delivery in adolescents with type 1 diabetes: A free-living, randomized clinical trial. *Diabetes Care* **2016**, *39*. [CrossRef] [PubMed]
33. Grosman, B.; Ilany, J.; Roy, A.; Kurtz, N.; Wu, D.; Parikh, N.; Voskanyan, G.; Konvalina, N.; Mylonas, C.; Gottlieb, R.; et al. Hybrid closed-loop insulin delivery in type 1 diabetes during supervised outpatient conditions. *J. Diabetes Sci. Technol.* **2016**, *10*, 708–713. [CrossRef] [PubMed]
34. Blauw, H.; Van Bon, A.; Koops, R.; DeVries, J. Performance and safety of an integrated bihormonal artificial pancreas for fully automated glucose control at home. *Diabetes Obes. Metab.* **2016**, *18*, 671–677. [CrossRef] [PubMed]
35. Reddy, M.; Herrero, P.; Sharkawy, M.; Pesl, P.; Jugnee, N.; Pavitt, D.; Godsland, I.; Alberti, G.; Toumazou, C.; Johnston, D.; et al. Metabolic control with the bio-inspired artificial pancreas in adults with type 1 diabetes: A 24-hour randomized controlled crossover study. *J. Diabetes Sci. Technol.* **2015**, *17*, 405–413. [CrossRef] [PubMed]
36. Capozzi, D.; Lanzola, G. A generic telemedicine infrastructure for monitoring an artificial pancreas trial. *Comput. Methods Programs Biomed.* **2013**, *110*, 343–353. [CrossRef] [PubMed]
37. Place, J.; Robert, A.; Ben Brahim, N.; Keith-Hynes, P.; Farret, A.; Pelletier, M.-J.; Buckingham, B.; Breton, M.; Kovatchev, B.; Renard, E. DiAs web monitoring: A real-time remote monitoring system designed for artificial pancreas outpatient trials. *J. Diabetes Sci. Technol.* **2013**, *7*, 1427–1435. [CrossRef] [PubMed]
38. Altia Insulin Pump Starts GUI Revolution. Available online: http://www.altia.com/downloads/case_studies/tandem_case_study.pdf (accessed on 6 February 2016).
39. Mossman, J. Insulin pumps: Design basics and tradeoffs. Available online: http://www.eetimes.com/document.asp?doc_id=1278073 (accessed on 2 January 2016).
40. NXP and Insulet Corporation to Help Improve Diabetes Care. Available online: http://www.nxp.com/about/our-customers/nxp-and-insulet-corporation-to-help-improve-diabetes-care:CASE_STUDY_INSULET (accessed on 6 February 2016).
41. Avizienis, A.; Laprie, J.; Randell, B. Fundamental Concepts of Dependability. In Proceedings of the 3rd IEEE Information Survivability Workshop (ISW-2000), Boston, MA, USA, 24–26 October 2000; pp. 7–12.
42. Bequette, B.W. Challenges and recent progress in the development of a closed-loop artificial pancreas. *Annu. Rev. Control* **2012**, *36*, 255–266. [CrossRef] [PubMed]
43. Patek, S.D.; Magni, L.; Dassau, E.; Toffanin, C.; De Nicolao, G.; Del Favero, S.; Breton, M.; Dalla Man, C.; Renard, E.; Zisser, H.; et al. Modular closed-loop control of diabetes. *IEEE Trans. Biomed. Eng.* **2012**, *59*, 2986–2999. [CrossRef] [PubMed]
44. Harvey, R.A.; Dassau, E.; Zisser, H.; Seborg, D.E.; Jovanovič, L.; Doyle, F.J., III. Design of the health monitoring system for the artificial pancreas: Low glucose prediction module. *J. Diabetes Sci. Technol.* **2012**, *6*, 1345–1354. [CrossRef] [PubMed]
45. Dassau, E.; Bequette, B.W.; Buckingham, B.A.; Doyle, F.J., III. Detection of a meal using continuous glucose monitoring. *Diabetes Care* **2008**, *31*, 295–300. [CrossRef] [PubMed]
46. Turksoy, K.; Samadi, S.; Feng, J.; Littlejohn, E.; Quinn, L.; Cinar, A. Meal-detection in patients with type 1 diabetes: A new module for the multivariable adaptive artificial pancreas control system. *IEEE J. Biomed. Health Inform.* **2016**, *20*, 47–54. [CrossRef] [PubMed]
47. Chen, S.; Weimer, J.; Rickels, M.; Peleckis, A.; Lee, I. Towards a Model-based Meal Detector for Type I Diabetics. In Proceedings of the 6th Medical Cyber-Physical Systems Workshop, Seattle, WA, USA, 13–16 April 2015.
48. Bergenstal, R.M.; Klonoff, D.C.; Garg, S.K.; Bode, B.W.; Meredith, M.; Slover, R.H.; Ahmann, A.J.; Welsh, J.B.; Lee, S.W.; Kaufman, F.R. Threshold-based insulin-pump interruption for reduction of hypoglycemia. *N. Engl. J. Med.* **2013**, *369*, 224–232. [CrossRef] [PubMed]

49. Johansen, T.A. Toward dependable embedded model predictive control. *IEEE Syst. J.* **2014**, *PP*, 1–12. [CrossRef]
50. Colnaric, M. Design of embedded control systems. *IEEE Int. Conf. Ind. Technol.* **2003**, *1*, 22–30.
51. Barroso, J. Modeling and Intelligent Control of a Distillation Column. Ph.D. Thesis, Technical University of Lisbon, Lisbon, Portugal, October 2009.
52. Ng, K.C.; Wang, L.; Peake, I.D. Safety-Critical Multi-Core Software Architecture for Model Predictive Control. In Proceedings of the Australian Control Conference, Melbourne, Australia, 10–11 November 2011; pp. 434–439.
53. Hildreth, C. A quadratic programming procedure. *Nav. Res. Logist. Q.* **1957**, *4*, 79–85. [CrossRef]
54. Steil, G.M.; Grodsky, G.M. The artificial pancreas: Is it important to understand how the β cell controls blood glucose? *J. Diabetes Sci. Technol.* **2013**, *7*, 1359–1369. [CrossRef] [PubMed]
55. Steil, G.M. Algorithms for a Closed-Loop Artificial Pancreas: The case for proportional-integral-derivative control. *J. Diabetes Sci. Technol.* **2013**, *7*, 1621–1631. [CrossRef] [PubMed]
56. Steil, G.M.; Palerm, C.C.; Kurtz, N.; Voskanyan, G.; Roy, A.; Paz, S.; Kandeel, F.R. The effect of insulin feedback on closed loop glucose control. *J. Clin. Endocrinol. Metab.* **2011**, *96*, 1402–1408. [CrossRef] [PubMed]
57. Weinzimer, S.A.; Steil, G.M.; Swan, K.L.; Dziura, J.; Kurtz, N.; Tamborlane, W.V. Fully automated closed-loop insulin delivery vs semi-automated hybrid control in pediatric patients with type 1 diabetes using an artificial pancreas. *Diabetes Care* **2008**, *31*, 934–939. [CrossRef] [PubMed]
58. Atlas, E.; Nimri, R.; Miller, S.; Grunberg, E.A.; Phillip, M. MD-Logic Artificial Pancreas System: A pilot study in adults with type 1 diabetes. *Diabetes Care* **2010**, *33*, 1072–1076. [CrossRef] [PubMed]
59. Magni, L.; Raimondo, D.M.; Bossi, L.; Dalla Man, C.; De Nicolao, G.; Kovatchev, B.; Cobelli, C. Model predictive control of type 1 diabetes: An in silico trial. *J. Diabetes Sci. Technol.* **2007**, *1*, 804–812. [CrossRef] [PubMed]
60. Soru, P.; De Nicolao, G.; Toffanin, C.; Dalla Man, C.; Cobelli, C.; Magni, L. MPC based Artificial Pancreas: Strategies for individualization and meal compensation. *Annu. Rev. Control* **2012**, *36*, 118–128. [CrossRef]
61. Grosman, B.; Dassau, E.; Zisser, H.C.; Jovanovič, L.; Doyle, F.J., III. Zone model predictive control: A strategy to minimize hyper- and hypoglycemic events. *J. Diabetes Sci. Technol.* **2010**, *4*, 961–975. [CrossRef] [PubMed]
62. Gondhalekar, R.; Dassau, E.; Zisser, H.C.; Doyle, F.J., III. Periodic-Zone Model Predictive Control for Diurnal Closed-Loop Operation of an Artificial Pancreas. *J. Diabetes Sci. Technol.* **2013**, *7*, 1446–1460. [CrossRef] [PubMed]
63. Seborg, D.E.; Edgar, T.F.; Mellichamp, D.A.; Doyle, F.J., III. *Process Dynamics and Control*; Wiley: Hoboken, NJ, USA, 2010.
64. Krasňanský, R.; Dvorščák, B.; Kozák, Š. Hardware Realization of Embedded Control Algorithm on FPGA. In Proceedings of the 5th International Conference on Computational Logics, Algebras, Programming, Tools, and Benchmarking, Venice, Italy, 25–29 May 2014; pp. 13–18.
65. Astrom, K.J.; Wittenmark, B. *Computer-Controlled Systems Theory and Design*, 3rd ed.; Prentice Hall: Upper Saddle River, NJ, USA, 1997.
66. Chan, Y.F.; Moallem, M.; Wang, W. Efficient implementation of PID control algorithm using FPGA technology. In Proceedings of the 43rd IEEE Conference on Decision and Control (CDC), Atlantis, Paradise Island, Bahamas, 14–17 December 2004; Volume 5, pp. 4885–4890.
67. Feng, G. A survey on analysis and design of model-based fuzzy control systems. *IEEE Trans. Fuzzy Syst.* **2006**, *14*, 676–697. [CrossRef]
68. Takagi, T.; Sugeno, M. Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. Syst. Man Cybern.* **1985**, *15*, 116–132. [CrossRef]
69. Teixeira, M.C.M.; Zak, S.H. Stabilizing controller design for uncertain nonlinear systems using fuzzy models. *IEEE Trans. Fuzzy Syst.* **1999**, *7*, 133–142. [CrossRef]
70. Sánchez-Solano, S.; Cabrera, A.J.; Baturone, I.; Moreno-Velo, F.J.; Brox, M. FPGA implementation of embedded fuzzy controllers for robotic applications. *IEEE Trans. Ind. Electron.* **2007**, *54*, 1937–1945. [CrossRef]
71. Taeed, F.; Salam, Z.; Ayob, S. FPGA implementation of a single-input fuzzy logic controller for boost converterwith the absence of an external analog-to-digital converter. *IEEE Trans. Ind. Electron.* **2012**, *59*, 1208–1217. [CrossRef]

72. Gondhalekar, R.; Dassau, E.; Doyle, F.J., III. MPC Design for Rapid Pump-Attenuation and Expedited Hyperglycemia Response to Treat T1DM with an Artificial Pancreas. In Proceedings of the American Control Conference (ACC), Portland, OR, USA, 4–6 June 2014; pp. 4224–4230.
73. Bemporad, A.; Morari, M.; Dua, V.; Pistikopoulos, E.N. The explicit linear quadratic regulator for constrained systems. *Automatica* **2002**, *38*, 3–20. [CrossRef]
74. Pistikopoulos, E.N.; Georgiadis, M.; Dua, V. *Multi-Parametric Model-Based Control Theory and Applications*; Wiley: Weinhaim, Germany, 2007.
75. Dua, P.; Kouramas, K.; Dua, V.; Pistikopoulos, E.N. MPC on a chip—Recent advances on the application of multi-parametric model-based control. *Comput. Chem. Eng.* **2008**, *32*, 754–765. [CrossRef]
76. Pistikopoulos, E.N. From multi-parametric programming theory to MPC-on-a-chip multi-scale systems applications. *Comput. Chem. Eng.* **2012**, *47*, 57–66. [CrossRef]
77. Dua, P.; Doyle, F.J., III; Pistikopoulos, E.N. Model-based blood glucose control for Type 1 diabetes via parametric programming. *IEEE Trans. Biomed. Eng.* **2006**, *53*, 1478–1491. [CrossRef] [PubMed]
78. Percival, M.W.; Wang, Y.; Grosman, B.; Dassau, E.; Zisser, H.; Jovanovič, L.; Doyle, F.J., III. Development of a multi-parametric model predictive control algorithm for insulin delivery in type 1 diabetes mellitus using clinical parameters. *J. Process Control* **2011**, *21*, 391–404. [CrossRef] [PubMed]
79. Huyck, B.; Callebaut, L.; Logist, F.; Ferreau, H.J.; Diehl, M.; De Brabanter, J.; Van Impe, J.; De Moor, B. Implementation and Experimental Validation of Classic MPC on Programmable Logic Controllers. In Proceedings of the 20th Mediterranean Conference on Control & Automation (MED), Barcelona, Spain, 3–6 July 2012; pp. 679–684.
80. Wang, Y.; Boyd, S. Fast model predictive control using online optimization. *IEEE Trans. Control Syst. Technol.* **2010**, *18*, 267–278. [CrossRef]
81. Pannocchia, G.; Rawlings, J.B.; Wright, S.J. Fast, large-scale model predictive control by partial enumeration. *Automatica* **2007**, *43*, 852–860. [CrossRef]
82. Cagienard, R.; Grieder, P.; Kerrigan, E.C.; Morari, M. Move blocking strategies in receding horizon control. *J. Process Control* **2007**, *17*, 563–570. [CrossRef]
83. Kufoalor, D.K.M.; Aaker, V.; Johansen, T.A.; Imsland, L.; Eikrem, G.O. Automatically generated embedded model predictive control: Moving an industrial PC-based MPC to an embedded platform. *Optim. Control Appl. Methods* **2015**, *36*, 705–727. [CrossRef]
84. Takács, B.; Kvasnica, M.; Di Cairano, S. Nearly-Optimal Simple Explicit MPC Regulators with Recursive Feasibility Guarantees. In Proceedings of the 52nd IEEE Conference on Decision and Control, Florence, Italy, 10–13 December 2013; pp. 7089–7094.
85. Kvasnica, M.; Fikar, M. Clipping-Based Complexity Reduction in Explicit MPC. *IEEE Trans. Autom. Control* **2012**, *57*, 1878–1883. [CrossRef]
86. Johansen, T.A.; Grancharova, A. Approximate explicit model predictive control implemented via orthogonal search tree partitioning. *IEEE Trans. Autom. Control* **2003**, *48*, 810–815. [CrossRef]
87. Bemporad, A.; Filippi, C. Suboptimal explicit receding horizon control via approximate multiparametric quadratic programming. *J. Optim. Theory Appl.* **2003**, *117*, 9–38. [CrossRef]
88. Bemporad, A.; Filippi, C. An Algorithm for approximate multiparametric convex programming. *Comput. Optim. Appl.* **2006**, *35*, 87–108. [CrossRef]
89. Jones, C.N.; Morari, M. Polytopic Approximation of explicit model predictive controllers. *IEEE Trans. Autom. Control* **2010**, *55*, 2542–2553. [CrossRef]
90. Parisini, T.; Zoppoli, R. A Receding-horizon regulator for nonlinear-systems and a neural approximation. *Automatica* **1995**, *31*, 1443–1451. [CrossRef]
91. Parisini, T.; Sanguineti, M.; Zoppoli, R. Nonlinear stabilization by receding-horizon neural regulators. *Int. J. Control* **1998**, *70*, 341–362. [CrossRef]
92. Pin, G.; Filippo, M.; Pellegrino, F.A.; Parisini, T. Approximate Off-Line Receding Horizon Control of Constrained Nonlinear Discrete-Time Systems. In Proceedings of the European Control Conference (ECC), Budapest, Hungary, 23–26 August 2009; pp. 2420–2425.
93. Pin, G.; Filippo, M.; Pellegrino, F.A.; Fenu, G.; Parisini, T. Approximate model predictive control laws for constrained nonlinear discrete-time systems: Analysis and offline design. *Int. J. Control* **2013**, *86*, 804–820. [CrossRef]

94. Bleris, L.G.; Kothare, M.V. Real-time implementation of model predictive control. In Proceedings of the American Control Conference (ACC), Portland, OR, USA, 8–10 June 2005; pp. 4166–4171.
95. Bleris, L.G.; Kothare, M.V.; Garcia, J.; Arnold, M.G. Embedded Model Predictive Control for System-on-a-chip Applications. In Proceedings of the 7th IFAC Symposium on Dynamics and Control of Process Systems (DYCOPS-7), Boston, MA, USA, 5–7 July 2004; Volume 1.
96. Swartzlander, E.E.; Alexopoulos, A.G. The sign/logarithm number system. *IEEE Trans. Comput.* **1975**, *24*, 1238–1242. [CrossRef]
97. Rossiter, J.A.; Kouvaritakis, B.; Bacic, M. Interpolation based computationally efficient predictive control. *Int. J. Control* **2004**, *77*, 290–301. [CrossRef]
98. Ding, Y.; Rossiter, J.A. Comparisons and combinations of interpolation methods with conventional predictive control. In Proceedings of the Mediterranean Conference on Control and Automation (MED), Athens, Greece, 27–29 June 2007.
99. Rossiter, J.A.; Pluymers, B.; De Moor, B. The potential of interpolation for simplifying predictive control and application to LPV systems. In *Assessment and Future Directions of Nonlinear Model Predictive Control*; Springer: Berlin, Germany, 2007; Volume 358, pp. 63–76.
100. Valencia-Palomo, G. Efficient Implementations of Predictive Control. Ph.D. Thesis, University of Sheffield, Sheffield, UK, October 2010.
101. Henriksson, D.; Cervin, A.; Akesson, J.; Arzen, K.E. Feedback scheduling of model predictive controllers. In Proceedings of the 8th IEEE Real-Time and Embedded Technology and Applications Symposium, San Jose, CA, USA, 25–27 September 2002; pp. 207–216.
102. Marti, P.; Lin, C.; Brandt, S.A. Optimal State Feedback Based Resource Allocation for Resource-Constrained Control Tasks. In Proceedings of the 25th IEEE Real-Time Systems Symposium, Lisbon, Portugal, 5–8 December 2004.
103. Cannon, M.; Kouvaritakis, B. Efficient Constrained Model Predictive Control with Asymptotic Optimality. In Proceedings of the 41st IEEE Conference on Decision and Control, Las Vegas, NV, USA, 10–13 December 2002; pp. 2744–2749.
104. Kouvaritakis, B.; Rossiter, J.A.; Schuurmans, J. Efficient robust predictive control. *IEEE Trans. Autom. Control* **2000**, *45*, 1545–1549. [CrossRef]
105. Huyck, B.; Ferreau, H.J.; Diehl, M.; De Brabanter, J.; Van Impe, J.F.M.; De Moor, B.; Logist, F. Towards online model predictive control on a programmable logic controller: Practical considerations. *Math. Probl. Eng.* **2012**, *2012*. [CrossRef]
106. Friedlander, M.P.; Orban, D. A primal-dual regularized interior-point method for convex quadratic programs. *Math. Program. Comput.* **2012**, *4*, 71–107. [CrossRef]
107. Rao, C.V.; Wright, S.J.; Rawlings, J.B. Application of interior-point methods to model predictive control. *J. Optim. Theory Appl.* **1998**, *99*, 723–758. [CrossRef]
108. Lau, M.S.K.; Yue, S.P.; Ling, K.V.; Maciejowski, J.M. A comparison of interior point and active set methods for FPGA implementation of model predictive control. In Proceedings of the European Control Conference (ECC), Budapest, Hungary, 23–26 August 2009; pp. 156–161.
109. Ferreau, H.J.; Kirches, C.; Potschka, A.; Bock, H.G.; Diehl, M. qpOASES: A parametric active-set algorithm for quadratic programming. *Math. Program. Comput.* **2014**, *6*, 327–363. [CrossRef]
110. Ferreau, H.J.; Bock, H.G.; Diehl, M. An online active set strategy to overcome the limitations of explicit MPC. *Int. J. Robust Nonlinear Control* **2008**, *18*, 816–830. [CrossRef]
111. Mattingley, J.; Boyd, S. CVXGEN: A code generator for embedded convex optimization. *Optim. Eng.* **2012**, *13*, 1–27. [CrossRef]
112. Domahidi, A.; Chu, E.; Boyd, S. ECOS: An SOCP solver for embedded systems. In Proceedings of the European Control Conference (ECC), Zürich, Switzerland, 17–19 July 2013; pp. 3071–3076.
113. Wills, A.; Ninness, B. QPC—Quadratic Programming in C. Available online: http://sigpromu.org/quadprog/ (accessed on 10 March 2016).
114. Domahidi, A. FORCES: Fast Optimization for Real-Time Control on Embedded Systems. Available online: http://forces.ethz.ch/doku.php?id=start (accessed on 15 March 2016).
115. Andersen, M.S.; Dahl, J.; Vandenberghe, L. CVXOPT Documentation, Release 1.1.8. Available online: http://cvxopt.org (accessed on 12 March 2016).

116. Diamond, S.; Boyd, S. CVXPY: A python-embedded modeling language for convex optimization. *J. Mach. Learn. Res.* **2016**, *17*, 1–5.
117. O'Donoghue, B.; Chu, E.; Parikh, N.; Boyd, S. Conic optimization via operator splitting and homogeneous self-dual embedding. *J. Optim. Theory Appl.* **2016**, *169*, 1–27. [CrossRef]
118. Kovatchev, B.P.; Breton, M.; Dalla Man, C.; Cobelli, C. In silico preclinical trials: A proof of concept in closed-loop control of type 1 diabetes. *J. Diabetes Sci. Technol.* **2009**, *3*, 44–55. [CrossRef] [PubMed]
119. Wolf, M. *Computers as Components: Principles of Embedded Computing System Design*, 2nd ed.; Elsevier: Waltham, MA, USA, 2012.
120. Kornecki, A.J.; Zalewski, J. Hardware certification for real-time safety-critical systems: State of the art. *Annu. Rev. Control* **2010**, *34*, 163–174. [CrossRef]
121. Kornecki, A.; Zalewski, J. Certification of software for real-time safety-critical systems: State of the art. *Innov. Syst. Softw. Eng.* **2009**, *5*, 149–161. [CrossRef]
122. Fries, R.C. *Reliable Design of Medical Devices*, 3rd ed.; CRC Press: Boca Raton, FL, USA, 2012.
123. Vogel, D.A. *Medical Device Software Verification, Validation and Compliance*; Artech House: Norwood, MA, USA, 2011.
124. Wiklund, M.E.; Kendler, J.; Strochlic, A.Y. *Usability Testing of Medical Devices*; CRC Press: Boca Raton, FL, USA, 2010.
125. Ling, K.V.; Yue, S.P.; Maciejowski, J.M. A FPGA Implementation of Model Predictive Control. In Proceedings of the American Control Conference (ACC), Minneapolis, MN, USA, 14–16 June 2006; pp. 1930–1935.
126. Fathy, H.K.; Filipi, Z.S.; Hagena, J.; Stein, J.L. Review of Hardware-in-the-Loop Simulation and Its Prospects in the Automotive Area. In Proceedings of the SPIE International Society for Optical Engineering, Kissimmee, FL, USA, 18–21 April 2006; Volume 6228, pp. 1–20.
127. Suardi, A.; Kerrigan, E.C.; Constantinides, G.A. Fast FPGA prototyping toolbox for embedded optimization. In Proceedings of the European Control Conference (ECC), Linz, Austria, 15–17 July 2015; pp. 2589–2594.
128. Hartley, E.N.; Jerez, J.L.; Suardi, A.; Maciejowski, J.M.; Kerrigan, E.C.; Constantinides, G.A. Predictive control using an FPGA with application to aircraft control. *IEEE Trans. Control Syst. Technol.* **2014**, *22*, 1006–1017. [CrossRef]
129. Jerez, J.L.; Constantinides, G.A.; Kerrigan, E.C. FPGA Implementation of an Interior Point Solver for Linear Model Predictive Control. In Proceedings of the International Conference on Field-Programmable Technology, Beijing, China, 8–10 December 2010; pp. 316–319.
130. Wills, A.; Mills, A.; Ninness, B. FPGA Implementation of an Interior-Point Solution for Linear Model Predictive Control. In Proceedings of the 18th IFAC World Congress, Milano, Italy, 28 August–2 September 2011; pp. 14527–14532.
131. Vouzis, P.D.; Bleris, L.G.; Arnold, M.G.; Kothare, M.V. A system-on-a-chip implementation for embedded real-time model predictive control. *IEEE Trans. Control Syst. Technol.* **2009**, *17*, 1006–1017. [CrossRef]
132. Ingole, D.; Holaza, J.; Takács, B.; Kvasnica, M. FPGA-Based Explicit Model Predictive Control for Closed-Loop Control of Intravenous Anesthesia. In Proceedings of the International Conference on Process Control, Štrbské Pleso, Slovakia, 9–12 June 2015; pp. 42–47.
133. Ingole, D.; Kvasnica, M. FPGA Implementation of Explicit Model Predictive Control for Closed Loop Control of Depth of Anesthesia. In Proceedings of the 5th IFAC Conference on Nonlinear Model Predictive Control, Seville, Spain, 17–20 September 2015; pp. 484–489.
134. Zometa, P.; Kögel, M.; Faulwasser, T.; Findeisen, R. Implementation Aspects of Model Predictive Control for Embedded Systems. In Proceedings of the American Control Conference (ACC), Montréal, QC, Canada, 27–29 June 2012; pp. 1205–1210.
135. Kalúz, M.; Klaušo, M.; Kvasnica, M. Real-Time Implementation of a Reference Governor on the Arduino Microcontroller. In Proceedings of the International Conference on Process Control (PC), Štrbské Pleso, Slovakia, 9–12 June 2015; pp. 350–356.
136. Frison, G.; Jørgensen, J.B. MPC Related Computational Capabilities of ARMv7A Processors. In Proceedings of the European Control Conference (ECC), Linz, Austria, 15–17 July 2015; pp. 3419–3426.
137. Blauw, H.; Keith-Hynes, P.; Koops, R.; DeVries, J.H. A review of safety and design requirements of the artificial pancreas. *Ann. Biomed. Eng.* **2016**. [CrossRef] [PubMed]
138. Zhang, Y.; Jones, P.L.; Jetley, R. A hazard analysis for a generic insulin infusion pump. *J. Diabetes Sci. Technol.* **2010**, *4*, 263–283. [CrossRef] [PubMed]

139. Cope, J.U.; Samuels-Reid, J.H.; Morrison, A.E. Pediatric Use of Insulin Pump Technology: A retrospective study of adverse events in children ages 1–12 years. *J. Diabetes Sci. Technol.* **2012**, *6*, 1053–1059. [CrossRef] [PubMed]
140. Martins, L.E.G.; De Oliveira, T. A case study using a protocol to derive safety functional requirements from fault tree analysis. In Proceedings of the 22nd IEEE International Requirements Engineering Conference, Karlskrona, Sweden, 25–29 August 2014; pp. 412–419.
141. Jerez, J.L.; Goulart, P.J.; Richter, S.; Constantinides, G.A.; Kerrigan, E.C.; Morari, M. Embedded online optimization for model predictive control at megahertz rates. *IEEE Trans. Autom. Control* **2014**, *59*, 3238–3251. [CrossRef]
142. Picasso, B.; Pancanti, S.; Bemporad, A.; Bicchi, A. Receding–horizon control of LTI systems with quantized inputs. In Proceedings of the IFAC Conference on Analysis and Design of Hybrid Systems, St. Malo, France, 16–18 June 2003; pp. 295–300.
143. Quevedo, D.E.; Goodwin, G.C.; De Doná, J.A. Finite constraint set receding horizon quadratic control. *Int. J. Robust Nonlinear Control* **2004**, *14*, 355–377. [CrossRef]
144. Liberzon, D. Quantization, time delays, and nonlinear stabilization. *IEEE Trans. Autom. Control* **2006**, *51*, 1190–1195. [CrossRef]
145. Mayne, D.Q.; Seron, M.M.; Raković, S.V. Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica* **2005**, *41*, 219–224. [CrossRef]
146. Limon, D.; Alamo, T.; Salas, F.; Camacho, E.F. Input to state stability of min-max MPC controllers for nonlinear systems with bounded uncertainties. *Automatica* **2006**, *42*, 797–803. [CrossRef]
147. Aguilera, R.P.; Quevedo, D.E. On the stability of MPC with a finite input alphabet. In Proceedings of the 18th IFAC World Congress, Milano, Italy, 28 August–2 September 2011; pp. 7975–7980.
148. Aguilera, R.P.; Quevedo, D.E. Stability analysis of quadratic MPC with a discrete input alphabet. *IEEE Trans. Autom. Control* **2013**, *58*, 3190–3196. [CrossRef]
149. Paul, N.; Kohno, T.; Klonoff, D.C. A review of the security of insulin pump infusion systems. *J. Diabetes Sci. Technol.* **2011**, *5*, 1557–1562. [CrossRef] [PubMed]
150. Li, C.; Raghunathan, A.; Jha, N.K. Hijacking an Insulin Pump: Security Attacks and Defenses for a Diabetes Therapy System. In Proceedings of the 13th International Conference on e-Health Networking, Applications and Services, (Healthcom), Columbia, MO, USA, 13–15 June 2011; pp. 150–156.
151. Hei, X.; Du, X.; Lin, S.; Lee, I.; Sokolsky, O. Patient infusion pattern based access control schemes for wireless insulin pump system. *IEEE Trans. Parallel Distrib. Syst.* **2015**, *26*, 3108–3121. [CrossRef]
152. Williams, P.A.; Woodward, A.J. Cybersecurity vulnerabilities in medical devices: A complex environment and multifaceted problem. *Med. Devices Evid. Res.* **2015**, *8*, 305–316. [CrossRef] [PubMed]
153. Burleson, W.; Clark, S.S.; Ransford, B.; Fu, K. Design challenges for secure implantable medical devices. In Proceedings of the 49th Design Automation Conference (DAC), San Francisco, CA, USA, 3–7 June 2012; pp. 12–17.
154. Lee, I.; Pappas, G.J.; Cleaveland, R.; Hatclif, J.; Krogh, B.H.; Lee, P.; Rubin, H.; Sha, L. High-confidence medical device software and systems. *IEEE Comput.* **2006**, *39*, 33–38. [CrossRef]
155. Ankaralı, Z.E.; Abbasi, Q.H.; Demir, A.F.; Serpedin, E.; Qaraqe, K.; Arslan, H. A Comparative Review on the Security Research for Wireless Implantable Medical Devices. In Proceedings of the 4th International Conference on Wireless Mobile Communication and Healthcare (Mobihealth), Athens, Greece, 3–5 November 2014; pp. 246–249.
156. Fan, J.; Reparaz, O.; Rožić, V.; Verbauwhede, I. Low-energy encryption for medical devices: Security adds an extra design dimension. In Proceedings of the IEEE/ACM Design Automation Conference (DAC), Austin, TX, USA, 2–6 June 2013; pp. 1–6.
157. Johansen, T.A.; Jackson, W.; Schreiber, R.; Tøndel, P. Hardware synthesis of explicit model predictive controllers. *IEEE Trans. Control Syst. Technol.* **2007**, *15*, 191–197. [CrossRef]