# Real-Time Control System for Improved Precision and Throughput in an Ultrafast Carbon Fiber Placement Robot Using a SoC FPGA Extended Processing Platform

Gilberto Ochoa-Ruiz, Romain Bevan, Florent de Lamotte, Jean-Philippe Diguet, Cheng-Cong Bao

Hindawi
International Journal of Reconfigurable Computing
Volume 2017, Article ID 3298734, 20 pages
https://doi.org/10.1155/2017/3298734

*Research Article*

# Real-Time Control System for Improved Precision and Throughput in an Ultrafast Carbon Fiber Placement Robot Using a SoC FPGA Extended Processing Platform

**Gilberto Ochoa-Ruiz,[1] Romain Bevan,[2] Florent de Lamotte,[2] Jean-Philippe Diguet,[2] and Cheng-Cong Bao[3]**

[1]*CONACYT-Universidad Autonoma de Guadalajara, Guadalajara, JAL, Mexico*
[2]*Lab-STICC-CNRS/ComposiTIC, Lorient, France*
[3]*Coriolis Composites, Lorient, France*

Correspondence should be addressed to Gilberto Ochoa-Ruiz; gilberto.ochoa@edu.uag.mx

Received 19 December 2016; Revised 8 May 2017; Accepted 22 May 2017; Published 5 July 2017

Academic Editor: Bibhu P. Panigrahi



We present an architecture for accelerating the processing and execution of control commands in an ultrafast fiber placement robot. The system consists of a robotic arm designed by Coriolis Composites whose purpose is to move along a surface, on which composite fibers are deposed, via an independently controlled head. In first system implementation, the control commands were sent via Profibus by a PLC, limiting the reaction time and thus the precision of the fiber placement and the maximum throughput. Therefore, a custom real-time solution was imperative in order to ameliorate the performance and to meet the stringent requirements of the target industry (avionics, aeronautical systems). The solution presented in this paper is based on the use of a SoC FPGA processing platform running a real-time operating system (FreeRTOS), which has enabled an improved comamnd retrieval mechanism. The system's placement precision was improved by a factor of 20 (from 1 mm to 0.05 mm), while the maximum achievable throughput was 1 m/s, compared to the average 30 cm/s provided by the original solution, enabling fabricating more complex and larger pieces in a significant fraction of the time.

## 1. Introduction

Given the continuous advances in communication networks and the increasing sophistication of embedded systems, there has been a widespread interest in recent years in the creation of distributed control systems (DCS) based on novel architectural paradigms [1]. In particular, distribution and reconfiguration are considered essential features for such new devices, typically known as intelligent electronic devices (IED). Such capabilities are indispensable in today's increasingly and ever-changing manufacturing settings for enabling fast prototyping and quick adaptation to emerging needs at the plant level [2].

Such DCS rely heavily on various industrial fieldbuses and protocols, and thus the performance of the IEDs and the correct behavior of the overall application strongly depend on the networking and processing capabilities integrated in the control, measurement, and I/O devices. However, real-time constraints limit the applicability of many solutions, due principally to the stringent time constraints inherent to the most demanding applications in which they are to be integrated [3].

Therefore, in order to be viable, modern DCS need to meet these demanding real-time capabilities and support event-based execution policies [4], which are difficult to attain due to nondeterministic latencies in the communication networks and, to a great extent, due to the computational capabilities of today's programmable devices (PLCs), which still rely on the sequential processing paradigm [5].

This problem has been recognized by the academia and the industry at large, which have tackled the problem by proposing new programming and architectural paradigms aiming at improving the performance of automation and

manufacturing processes, as well as their resilience and adaptability and the distribution of various complex control algorithms over several nodes to improve the data efficiency. Many of these approaches point towards the introduction of specialized processing units, in the form of microprocessors [6], and increasingly as customized computing machines implemented in FPGAs [7].

In this manner, the more demanding components of the application can be decentralized, improving the performance and reliability of the subprocesses, while decreasing the computational load in the main PLCs and the communication bandwidth requirements of the overall system. Moreover, the complexity of the verification process can be significantly reduced due to an improved separation of concerns, leading to more dependable, fault-tolerant, and maintainable systems [8].

In this paper, we present a case study of a DCS in the context of an advanced fiber placement platform. Such a system encompasses a large number of physically distant subcomponents due to the dimensions of the controlled plant (the robot is used for 3D print large pieces for avionics and nautical applications) and for safety reasons. The original system made use of a decentralized architecture, consisting of several PLCs for controlling the fiber feeding subsystem, the robot itself, and a very large number of I/Os integrated in the robot's head, which severely limited the attainable performance of the system.

These limitations stemmed from the manner in which the control commands were sent to the ultrafast fiber placement head (via Profibus-DP by the main PLC), thus limiting the reaction time and, in consequence, the precision of the fiber placement and the maximum attainable deposition throughput. In this paper, we present a novel, custom real-time, and distributed solution, which has been integrated directly in the robot head to accelerate the control of very fast actuators and is based on novel SoC FPGA devices.

The rest of this paper is organized as follows: in the next section, we provide more ample motivations for the implementation of IEDs, as well as other Industrial Process Measurement and Control Systems (IPMCS) using reconfigurable devices, based on the limitations of current DCS design approaches and programming paradigms. In Section 3, we delve into discussion of the successful application of FPGA devices for implementing IMPCS platforms, which will serve as the stepping stone for the rest of the article. Subsequently, we embark into a description of the application context and, in particular, of the specific challenges for the design of a novel control system for the robotized fiber placement unit at the heart of the article. Thereafter, in Section 5, we embark in the description of the proposed architecture and we discuss the rationale behind the use of SoC FPGAs and the real-time operating system solution. Finally, in Section 6, we compare the proposed solution with the original PLC-based implementation and with a second MCU-based embedded system in terms of the achieved precision, performance, and response time, as well as the attainable fabrication time.

## 2. Context and Motivation

Today's fast-changing manufacturing markets are forcing a paradigm shift in the associated fabrication processes. These tight demands are leading to an increased complexity of the industrial environments and associated equipment as well as a shift in the conception of the underlying control settings. Hence, in order to cope with these emerging necessities (i.e., improved fault-tolerance, online monitoring, and prognosis), new manufacturing infrastructures, production facilities, and operation/control methods are very much required, accompanied by new standards and devices to support them.

Therefore, in recent years, a great deal of research has been conducted to improve the capabilities of the manufacturing control systems, mainly based on the concept of distributed intelligent control, which aims at bridging the gap among different domain practices, with special emphasis on improving systems integration and coordination. The resulting automation models are underpinned by wide networks of devices, known as distributed control systems (DCS), interconnected through field area networks and industrial fieldbuses, as depicted in Figure 1. Such systems of systems are encompassed by heterogeneous sensors, actuators, and local regulators/controllers, typically attached to a locally central unit, implementing one or a set of control/monitoring algorithms.

These subsystems are typically known as Remote Terminal Units, which might exchange information with other controllers over the network (i.e., Programmable Logic Controllers, PLC) in order to synchronize their operations and carry out a complex process. The possibility of implementing DCS with local intelligence and distributed control, enabling fully monitoring a plant, is becoming more attractive but increases exponentially the complexity of such systems. For instance, DCS pose very specific requirements in terms of the latency, reliability, and availability of the control system. Moreover, a distributed architecture must deal with safety issues such as redundancy, data validation, fault isolation, and tolerance [9].

It has been difficult to meet the aforementioned requirements with existing technologies, which are based on traditional sequential controllers, limiting the response time and their deployment in many demanding applications. Thus, control engineers have sought new manners to implement high-performance systems, often based on devices such as FPGAs [10], especially in niche areas such as motion control and voltage regulators and also in the creation of intelligent sensors and other intelligent electronic devices [11]. Moreover, the introduction of new technological features, such as the extended embedded platforms (with offers from all FPGAs vendors), containing high-end processors and reconfigurable logic, seems like the next logical step for the creation of plants-on-a-chip [12].

In this paper, we posit that such SoC FPGA platforms represent an excellent technological choice for implementing heterogeneous, customizable, scalable, and reconfigurable DCS, a fact that has been widely recognized in the industrial community [13]. Moreover, reconfigurable devices have the added benefit of fostering interoperability by easily customizing the supported fieldbus protocols on the field and remotely

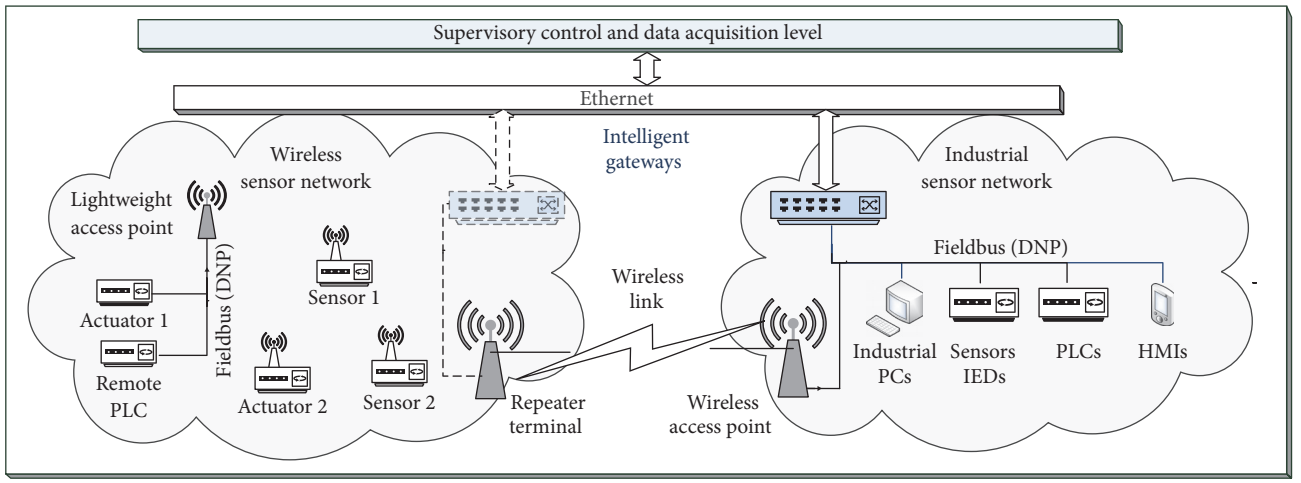



Figure 1: Typical IPMCS hardware architecture for factory floor automation.

with the possibility of using partial reconfiguration techniques, being even promoted as intelligent and upgradeable gateways, seeking to harmonize the current limitation in the industrial ecosystem [14]. In the next section, we analyze some of the current efforts in the creation of intelligent electronic devices and control subsystems in reconfigurable platforms for a variety of applications; some emphasis is given at the end of the section as well not only to the substitution of PLC devices by FPGAs [15] but also to more general applications as digital controllers for a variety of industrial applications.

Furthermore, we analyze the current divide between the existing literature and the need for actual, interconnected, and distributed IEDs and controllers, which we argue could be alleviated through the use of SoC FPGA devices and the associated resources. For this purpose, in Section 4, we present a case study for a distributed control system based on Xilinx Zynq Extended Processing Platform, which has been deployed in industrial environment for 3D printing and fiber carbon reinforcement.

## 3. Use of FPGAs in Industrial Applications

In this section, we discuss some of the limitations of the current approaches for distributed control systems. Afterwards, we briefly describe some successful uses of FPGA in the implementation of IEDs and various types of control algorithms. In the same vein, some initial efforts for the integration of PLC-like type of functionalities within the programmable fabric will be addressed. Finally, in the last subsection, we will discuss how more advanced type of reconfigurable devices (such as SoC FPGAs) could help bridge the gap between the endeavors carried out within the control and automation domain and embedded design communities.

*3.1. Limitations of Current DCS Paradigms.* Most of the current process automation and control platforms, globally known as Industrial Process Measurement and Control Systems (IPMCS), are built around traditional PLC architectures, which are generally oriented towards centralized applications, in which several nodes retrieve data from the plant and react to external events according to a main application running on a central controller. These solutions are not well suited for implementing complex DCS for many reasons, the most important being that existing approaches are designed under the execution constraints imposed by the cycle-scan nature of the PLCs [16]. The performance of such solutions is rather limited in very demanding applications, such as highly dynamic and ultrafast processes (such as electronic drives and ultrafast robotics), as depicted in Figure 2. Thus, practitioners in the domain have been looking for effective manners to respond to these heterogeneous and stringent constraints.

In recent years, there has been a trend towards the use of intelligent electronic field devices (IEDs) or Intelligent Mechatronic Components (IMCs) [17], which contain a certain amount of embedded computing power, in tandem with communication and monitoring capabilities. Complex algorithms can be distributed over such smart devices, resulting in the reduction of the computing load for the main PLCs and of the required communication bandwidth of the overall application.

The use of such intelligent components promises essential benefits for the design and reconfiguration of automated production systems due to encapsulation and reuse of a great deal of intellectual property modules. However, the design patterns promoted by current programming and architectural standards (such as the IEC 61131-3 [18]) do not conceptually support the capabilities that are necessary to fulfill those promises [19]. Moreover, the design of control systems by the practitioners is still very much linked to a PLC-programming mindset, which, as discussed above, does not lend very well to the implementation of more sophisticated decentralized applications.

Some initial efforts towards a more varied ecosystem have been observed in recent years, with an increased inclusion of various types of microcontroller units (MCUs) into IEDs, as well as their deployment as the technology of choice for new developments in the automation domain, as

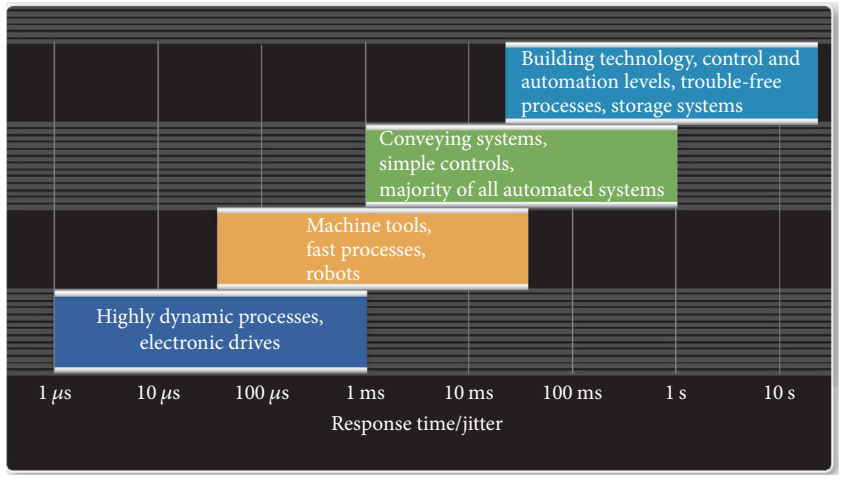



Figure 2: Real-time classes and application areas (IAONA classification).

those represented by the IEC 61499 standard for distributed control systems [20], which is based on an event-driven approach. However, the existing development tools and the supported architectures do not fully implement the standard as originally intended and are still very much bounded by the sequential cycle-scan of the underlying platforms.

*3.2. Trends in the Use of FPGAs for Implementing IEDs.* In recent years, researchers and industry have been looking for means to overcome the aforementioned limitations. For instance, the study of the possibilities of FPGAs for implementing IEDs and PLC-based platforms has been a very active area of research [21]. The most important benefits of using FPGAs for implementing complex control systems are related to performance in terms of the execution speeds that can be attained by massively parallel architectures, in tandem with significantly larger I/O processing capabilities.

Reconfigurable devices could enable improved control systems, where designers can combine one or several RISC processors with dedicated computing hardware accelerators [10] implementing control algorithms, while enabling the integration of communication blocks (to support the field-bus protocols) and other specialized peripherals. On the other hand, designers can also design custom hardware architectures for stringent applications in terms of performance, which when coupled with the embedded processors could help harness IP reuse and product diversification [7]. Furthermore, other advantages of FPGAs over competing technologies are their programmability on the field, customization through programmable logic, and the ability to tailor the communication protocols to a particular system configuration, among others.

A good introduction to the advantages of the use of FPGAs in industrial settings can be found in [5], where an account and analysis of the benefits of using these devices are presented. As noted by the authors, industrial robot control systems in particular represent an especially interesting application scenario, given that such systems have evolved from open-loop to closed-loop, adaptive controlled systems. This evolution entails a dramatic technological shift from relatively simple architectures to more complex platforms integrating DSP functions, ADC/DAC converters, along with Pulse Width Modulation (PWM) generators and the underlying logic resources of the FPGA for implementing the computation for the control algorithms. The next generation of control systems needs to be able to cope with increasingly faster real-time responses, and thus FPGAs are regarded as ideal candidates for implementing these demanding applications [12].

It has been argued that, when compared with their analogue counterparts, a digital system that could execute quasi-instantaneously a control algorithm should be of great interest by cumulating the advantages of both worlds [22]. This has led to a third category of control devices: the quasi-analog controllers by digital means. FPGA devices represent a good choice for this new category of controllers, since they incorporate various heterogeneous resources (i.e., BRAM and DSP blocks) within the reconfigurable fabric, along with the necessary logic for implementing a great variety of algorithms. Moreover, FPGAs enable integrating not only traditional control algorithms but also other more PLC-oriented functions that can coexist in the same device [23], offering a very high level of integration and computational heterogeneity.

It is not the goal of this section to delve into a detailed state of the art of this area, which has been profusely done in the papers cited above. However, we can briefly mention some successful applications of FPGA devices, for instance, as power conversion controllers (as pulse width-modulation (PWM) inverters [24, 25] and multilevel converters [26, 27]). Uses can also be found in the control of electrical machines and in robotics applications (induction machine drives [28, 29] and motion control [30, 31]). More recently, reconfigurable devices have been increasingly finding applications as means for implementing hardware-in-the-loop platforms [32, 33] for debugging purposes principally but increasingly as a means to emulate subcomponents of larger systems.

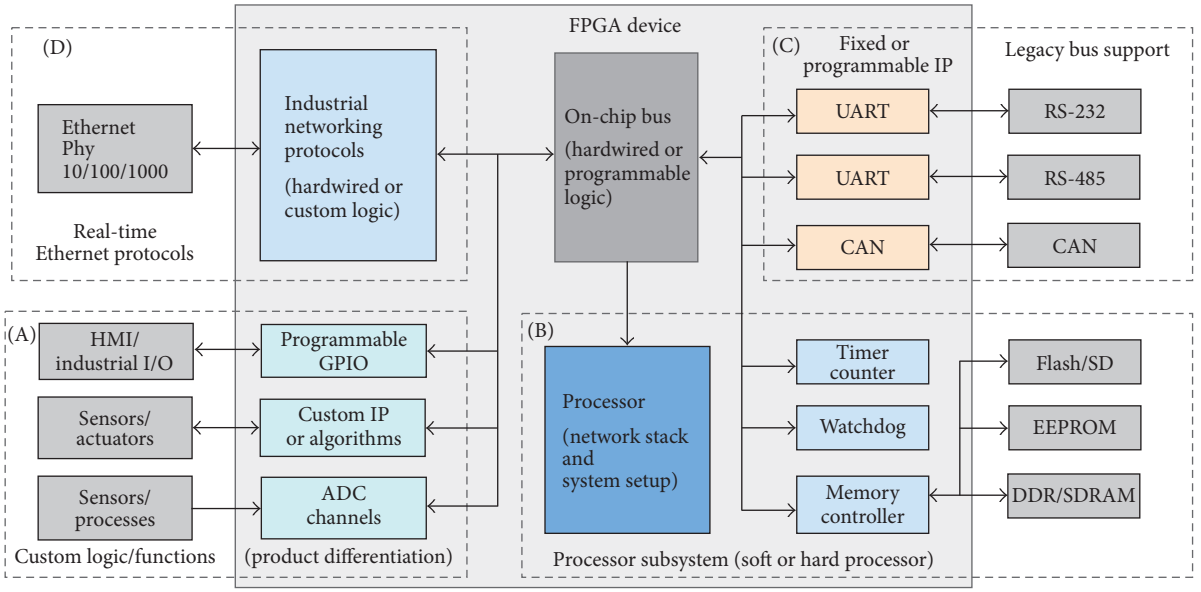



Figure 3: A possible architecture for a SoC FPGA-based networked industrial intelligent electronic device.

*3.3. Trends in the Use of FPGAs for Implementing IEDs.* In the industrial control domain, there are a number of reasons for the use and integration of novel architectures and industrial-optimized semiconductor devices. In the first place, there is a shift from point-to-point data communication towards network-based solutions, as profusely discussed in Section 2.

Second, in many application domains, this allows system integrators to build larger, scalable, upgradeable, and more cost-effective systems. Factory automation and control systems could benefit from the expandability that network communication and control offer, such as easily adding and upgrading equipment that is connected using standardized protocols [34].

Third, another major trend is represented by a shift towards the miniaturization of the application processing systems. Many factory equipment suppliers have learned that, by incorporating sophisticated motor-control algorithms, they can use low-cost motors, while reducing power consumption and improving reliability and safety. The same applies for many of the applications and control algorithms briefly described above: indeed, many of the applications described at the end of the previous section have clearly shown that parallel implementation of many control algorithms can attain significantly greater performances compared with other competing approaches, such as MCU and DSP devices [35].

These major trends have been driven by a need for high-performance, Ethernet-ready, low-power semiconductor devices to control the next generation of industrial machines. In particular, low cost is an important priority in many very specific and high-end applications; FPGAs and SoC FPGAs address this need by enabling differentiation via custom algorithms and functions tightly integrated in a single device, reducing BOM costs by integrating specialized ASIC components, DSP processors, and industrial buses and protocols into a single device, as depicted in Figure 3.

Features and functions supported by an FPGA can be updated long after deployment [36]. In areas such as industrial networking, where the protocols and standards are shifting and changing, the programmability of FPGAs versus fixed-logic devices (e.g., ASICs and ASSPs) saves migration costs and makes the solutions easier to maintain and scale. The same applies for the control algorithms as well, which can be updated when needed, taking advantage of the fast-prototyping capabilities afforded by reconfigurable devices, minimizing the cost and the time to market. As depicted in block (A) of Figure 3, the vast majority of today's FPGAs and SoC FPGAs incorporate programmable ADC channels, which in tandem with logic, memory, and DSP resources present in the reconfigurable fabric enable implementing complete custom functionalities, such as those described in the previous section. In the same vein, FPGAs integrate up to hundreds of programmable and customizable general-purpose I/O pins, which could be used to interface an FPGA-based IED and the associated custom functions with other industrial sensors and actuators.

Moreover, the possibility of integrating embedded processors (block (B) in the figure) within the device fosters improved HW/SW partitioning design strategies, helping designers to achieve a better compromise in the implementation of the constituent blocks of a control algorithm. The use of control functions into tightly integrated SoC FPGA-based IED entails many other benefits as well, which will be described as follows. First and foremost, the algorithms described before cannot function in isolation: an IED must be capable of performing system setup and managerial and monitoring operations in order to guarantee the correct operation of the controlled plant or process. These tasks are typically catered by a tightly coupled embedded processor, which either can be implemented in the FPGA resources or be a hardwired CPU. Secondly, the processor takes care as well

of scheduling the execution and the communication among cooperating control functions, making the implementation of the control easier to maintain and validate at each stage of the development process. Finally, the performance of the control algorithms as well as the communication among the various control modules, sensors, and actuators is drastically reduced due to shortest response times of the underlying hardware implementation.

Process level communication is facilitated as well by the use of the embedded MCUs, which are responsible for federating peer-to-peer communication, both at the device level and among IEDs and other equipment. This communication is achieved through legacy buses (block (C)), which can either be present in the device or specifically mapped for a given application, taking advantage of the available logic resources and programmable GPIO. The FPGA can act as a bridge between industrial Ethernet protocols to RS-232, RS-485, and CAN, still widely used by many vendors for actuators and sensors alike. The main benefit of using FPGAs in this regard is that system can be tailored for the specific needs of the application, mapping only the required IP modules for supporting a given legacy protocol, saving valuable resources, and reducing the footprint of the hardware solution in a cost-effective manner.

In the same vein, the highest level of communication is the use of Ethernet, which provides the largest data bandwidth and distance to provide communication between various factory sites. Fieldbus communications employ RS-485, RS-422, and RS-232 as the physical layer interface, with protocols specified by the IEC 61158 standard, for example, DeviceNet, CANopen, and Profibus. However, as Ethernet matures in the SCADA segment, many fieldbus installations are being replaced or redesigned with real-time Ethernet protocols augmented with deterministic communication profiles and mechanisms. In order to meet the real time, low latency, and the deterministic capabilities required for industrial applications, many of these Ethernet protocols use specialized Ethernet Media Access Control (MAC) modules (hardware accelerators present in the FPGA, Block (D)), in tandem with specialized data packaging stacks for high-speed encoding and decoding. Reconfigurable devices support many of these protocols, which can be mapped onto the programmable logic and easily accessed and controlled by the processor through a lightweight IP stack or be present as hardened modules.

As briefly discussed at the end of Section 1, in this paper, we present the development of IED for controlling, in real time, the deposition of fibers by Automated Fiber Placement Robot to be discussed in the next section. The proposed solution had a set of constraints that were especially well suited for implementation in an embedded device: the available space was reduced, it had to meet real-time constraints, at both the control and communication levels, in order to maximize the attainable throughput and achievable precision, and, finally, it needed to be adaptable to any protocol changes in the rest of the DCS platform where it is to be integrated. Some implementation choices were initially explored, finally settling into a SoC FPGA from Xilinx (the Zynq EPP 7000), which enables many of the features described above, as we will discuss in a subsequent section.

## 4. Application Context: The Control AFP Robot

Composite materials are increasingly used by the automotive, aerospace, and nautical industries to manufacture complex structures in terms of shape and also with the aim of addressing stringent requirements such as to lighten the weight of the vehicles while maintaining other properties such as resistance and reliability. Today, it is beyond dispute that three-dimensional (3D) fabric preforms can produce high-performance composite parts in sizes ranging from small to gigantic.

But, for high-volume industries, such as the automotive sector, 3D preforming processes have been, thus far, too time-consuming and, therefore, too expensive to be a serious materials/process option for producing cars. However, these limitations have been steadily overcome in recent years with the introduction of emerging technologies that offer the opportunity to automate the time-consuming and labor-intensive hand layup of 3D preforms, based on robotized fiber placement systems.

Robots have long been used to perform a variety of manufacturing tasks, but their use in the field of composites has been limited. In some cases, end-of-arm equipment has been used for water-jet cutting, drilling and tapping, material-handling, assembly, and fiber-placement applications [37]. Lasers have assisted greatly in verifying material location and orientation for manual layup and water-jet systems can generate holes and cutouts after layup is complete and the part is cured. Moreover, significant improvements in $X$-$Y$ cutting systems and associated software have expedited material profiling [38].

Robotized Automated Tape Laying (ATL) and Automated Fiber Placement (AFP) are two emerging technologies for the production of a large variety of composites parts in the aeronautic industry. Their advantage towards fabric or large tape manual layup consists mainly of the ability to place consistently the fiber at the right place with the correct orientation in order to achieve the mechanical characteristics demanded by primary load-bearing structures [39].

Automation also promotes consistency in the quality of the produced parts, often obtained in a fraction of the time, compared to manual methods. The possibility of producing larger components, such as aircraft fuselages, is another advantage of these methods together with the ability to achieve near net-shape preforms, reducing material wastage and, hence, costs.

All these aspects make ATL and AFP ideal candidates for the production of helicopter panels and blades, tail cones, components for business jets, short and long range civil aircraft, military aircraft, engine nacelles, fan blades, and components for the automotive industry. However, such systems have some limitations regarding the speed and precision at which the fibers can be deposed, as we will see in the following subsections.

(a) (b)

Figure 4: (a) An example of a fiber placement robot and (b) of a piece for the automotive industry.

*4.1. The Fiber Placement Robot.* The need for flexibility and modularity has led to the development of new systems mostly based on polyarticulated robots, which are able to be adapted or reprogrammed to different processes and different applications. These units are able to handle a variety of raw materials and to provide high production rates while working on complex and challenging structures.

Nowadays, these standard off-the-shelf polyarticulated robots are widely developed and have been produced for many years for the requirements of the automotive industry. They have reached a very high level of reliability and appear to be ideally suited for use within an AFP system able to satisfy all the requirements listed above. These polyarticulated robots have payloads ranging from 6 kg to 1 ton and can be combined with linear axis up to 60 m and spindle axis up to 40 m.

The AFP system deployed in our application (Figure 4(a), created by Coriolis Composites) uses the 6 motion axes of the polyarticulated robot (supplied either by ABB or by KUKA) plus two external units: a mould guiding unit and a robot positioning axis; therefore, the complete cell has 8 degrees of freedom. The robot, as well as the whole cell, complies with aeronautic specifications in terms of fiber placement and cutting accuracy as well as the repeatability of the process [40].

This flexible, compact, and versatile AFP system adapts easily to different geometries and ranges, making it suitable for manufacturing of complex parts and adaptable to any industrial settings as well as for applications in research centers, an is the case of the system presented herein. Its reliable and robust design meets the requirements of series production maintenance and high production levels are ensured through the speed of its movements. Precision and repeatability (basic criteria for the aeronautic market) are assured through a light maneuverable layup head, as shown in Figure 4(a).

*4.2. Fiber Placement Real-Time Requirements.* The AFP robot integrates an advanced fiber deposition head, a complex system in charge of feeding the carbon fibers to the deposition subsystem (where an array of actuators reside), using a roller system to move the fiber from a gantry to the mould, as depicted in Figure 5(a). The number of controllable actuators in the edge of the head determines the size and attainable throughput in the fabrication process, which involves deposing fibers over a prefabricated mould. This process is akin to 3D printing, with the main difference that the positioning process involves a stitching fibers in layups instead of melting plastics such PLA to produce the piece (see Figure 4(b)).

As discussed before, the fibers are fed to the head subsystem from the creel using a pulley system in order to avoid the burden of the extra payload and complexity in the head, thus limiting the speed and the accuracy of the process. Therefore, a mechanism to depose fibers of different sizes in a controlled manner is implemented in the robot head, as depicted in Figure 5(a); each of the fibers in the head subsystem requires 3 actuators for feeding (via a roller), clamping, and cutting individual fibers, permitting a fine-grained deposition.

These actions need to be performed while the robot head is moving along the mould in an ultrafast, precise, and synchronized manner, which entails that the trigger time needs to be very short for all the actuators (very low jitter and skew). The Profibus communication proved to be a major bottleneck for real-time performances as the number of fibers augmented, producing accumulative positioning errors as depicted in Figure 5(b). In order to attain much lower response times, the communication bottleneck had to be eliminated by distributing the real-time functionalities of the BoxPC controller, integrating a rapid action processing system in close proximity to the actuators.

This issue stems from the fact that the Profibus link limited the speed at which the triggers could operate, forcing the system to decelerate in order to wait for new commands. Therefore, a reduced response time was deemed necessary to alleviate the above-mentioned issues, leading to a higher-performance solution (in terms of the deposition precision), which could potentially help in attaining higher fabrication throughput.

A distributed and networked IED implementation for managing this process seemed like the most viable choice, since this could be optimized for coping with the real-time requirements of the application directly into the robot head, discharging the main PLC from some of the time-consuming duties and enabling a higher degree of intelligence and data efficiency.

A more detailed description of the initial distributed control architecture will be provided in the next section, pointing at its limitations and outlining the requirements for the IEC control system in more detail, in order to gain greater understanding of the benefits of using reconfigurable devices in this industrial application. Particularly, we discuss how the features of SoC FPGA described in Section 3.3 were deployed.

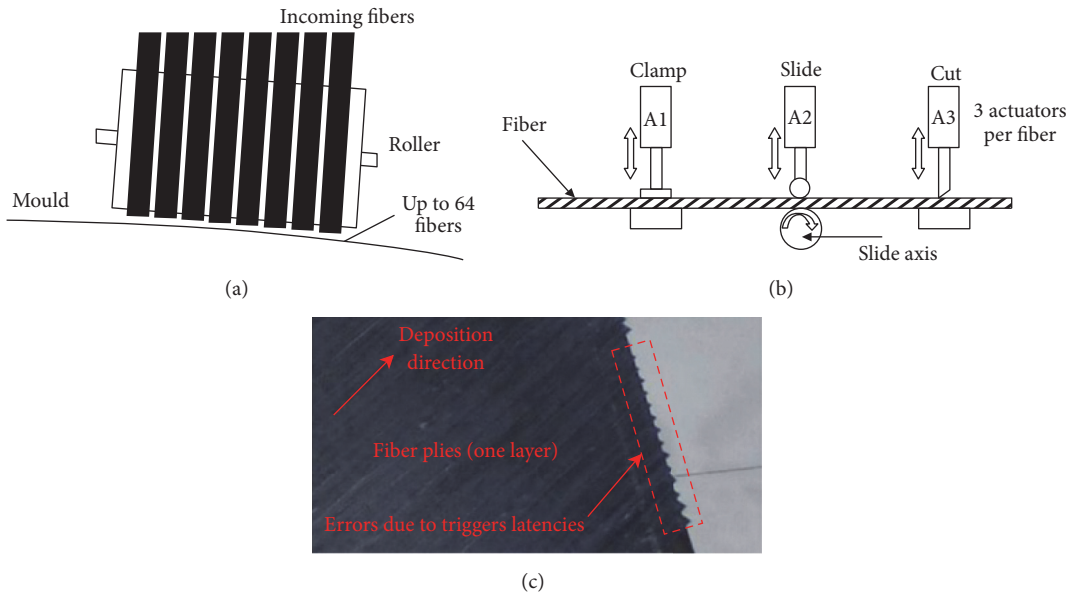



Figure 5: (a) The fiber placement subsystems in the head of the robot. (b) Actions and actuators per incoming fiber.

*4.3. Distributed Control System Architecture.* The AFP system is composed, apart from the robotic arm, of a placement head, a creel, and a tube for feeding the fibers. The creel provides all the necessary functions for unwinding the composite fiber bobbins at very high speeds with low tension and enables swift loading and unloading of the bobbins. The flexible pipes feed each fiber from the creel to the layup head, avoiding risks for twisting or damage. The system is compatible not only with preimpregnated thermoset material certified by the aeronautic industry but also with materials "of the future" such as dry fibers and preimpregnated thermoplastics.

This adaptation is made possible by a rapid change of the heating system. The unwinding, guiding, and laying up system are already adapted to these three families of materials. The entire system is controlled by a complex DCS split among different components due to the use of different vendor and proprietary subsystems. However, the fiber feeding system mounted in the robot is of paramount importance in the successful deployment of ADFP in ultrafast fabrication processes, and thus we concentrate on the specific issues of this subcomponent. In what follows, we will provide a general description of how the AFP robot is deployed within the DCS application to provide the reader with a glimpse of its complexity and of the limitations addressed in this work.

The initial DCS architecture is depicted in Figure 6, where the AFP robot (1) is the central component of the distributed system. The robot is tightly connected to Automatic Spool Frame (2), which contains the various types of fibers to manufacture a given piece and is controlled, along with the robot, by the AFP Robot Controller PLC. This PLC contains the general fiber placement software, generating the necessary commands to control the 6 axes and position the robot's head in the required coordinates. The control program, in the BoxPC (4), constantly monitors the speed and position of the robot via a high-precision encoder in order to determine at which points a set of actions over the fibers need to take place. Whenever an action has to be undertaken, the appropriate command (5) is sent to the deposition head by a real-time BoxPC (a WinAC RTX [41]) via a Profibus link.

These events (referred to as triggers thereafter) control the Head Electronic-Pneumatic (6) subsystem (containing internally a very large number of actuators (7)) used for deposing the fiber over a surface (typically a mould (8)), using the mechanism of Figure 5(a). In order to attain the highest possible throughput, while maintaining a high precision (and low jitter) in the actions, the response time between the trigger and the action must be as short as possible, typically in the order of microseconds, positioning the system in the scale of the most demanding applications depicted in Figure 2. Apart from the stringent execution requirements needed for attaining the necessary processing capabilities, the solution should meet some other requirements, which are described as follows.

*(1) Real-Time Ethernet Connectivity*. The solution needs to be able to interact with the main PLC and the associated HMI software in order to upload the deposition program over real-time Ethernet connections. Manual command settings should also be possible through the HMI for debugging purposes. The proposed solution must then implement a lightweight TCP/IP stack and client for such purposes.

*(2) Remote Storage of the Deposition Program*. In order to minimize the data transfer delays associated with the previous solution, it is desirable to be able to store large control programs, which trigger the actions of the electropneumatic actuators in the head. The program needs to be stored either

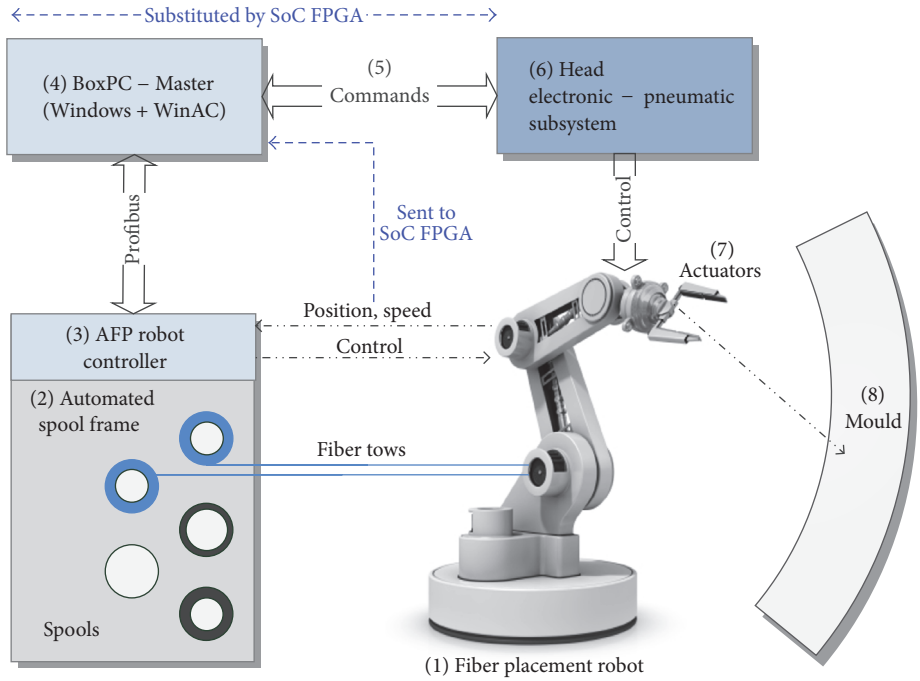



Figure 6: Previous system architecture for the fiber placement robotic platform based on a BoxPC controller.

temporally (for test purposes or small fabrication batches) or in a more long-term basis (using internal flash memory or SD cards).

*(3) Act over a Large Number of I/O in a Scalable Manner.* In order to be of any use, the system should be capable of dealing with a large number of actuators so that larger draping patterns can be handled. However, the amount of actuators for a given phase in the fabrication process changes and, thus, the system should be able to be seamlessly configured for different configurations (i.e., 8, 16, 32, and 64 tapes).

*(4) Calculate the Position and Speed Independently.* As we will see in the next section, the deposition program includes information about the precise moment at which the rolling, clamping, and cutting operations must be performed. The latter two are especially important, since they determine the precision that can be attained. Therefore, position and speed calculations should be available in real time so that triggers are activated at the optimal time, while avoiding the latencies associated with the original implementation.

*(5) Backward Compatibility and Upgradeability.* The proposed solution should be capable of interacting with existing infrastructure and equipment, as well as supporting legacy communication protocols. For instance, a previous development made use of SPI communications for the GPIO voltage scaling, as well as for interacting with visualization and storage devices.

In the next section, we will present the proposed solution, based on a SoC FPGA (Xilinx's Zynq Extended Processing Platform [42]), which enabled us to meet the constraints and requirements discussed above, in a cost-effective manner.

## 5. Proposed Real-Time Architecture Based on SoC FPGA

The PLC and BoxPC solution for controlling the deposing of fibers in the AFP robot was not able to meet the ever-increasing stringent requirements that such an application requires. For this reason, it was decided that an intelligent electronic device, closer to the deposing head, could accelerate the actuator triggering process, as well as the precision in the actions, by introducing a higher degree of intelligence and reducing the signal latency (see the dotted square on the top of Figure 6).

A first, proof of concept system was created, based on a small microcontroller running a real-time operating system (FreeRTOS [43]), which, in tandem with a Modbus stack, addressed the communication bottleneck and, furthermore, accelerated the command processing itself by storing large numbers of actions (even complete manufacturing sequences) directly in flash memory. In this manner, the main PLC was discharged of the significant computational and communication loads, while eliminating the use of the BoxPC PLC.

As we will see later in the article, this solution provided a significant speed-up over the PLC-based solution, but it quickly proved to be of limited use for systems which required deposing more than 16 fibers, as the number of signals the MCU could control is limited. This shortcoming severely limited the scalability of the deposing system, in which it is desirable to be able to program the number of fibers in real time and to be able to depose up to 48 fibers (which entails controlling $3 \times 48 = 132$ actuators). Therefore, it was decided to move to FPGA implementation to take advantage of the very large

number of programmable I/Os FPGAs provide, as well as to leverage the field-programmability of this technology to accommodate future developments and upgrades. In order to make this passage more straightforward, a SoC FPGA has been chosen (the Zynq EPP 7000) in order to port the FreeRTOS implementation previously developed while gaining in customization capabilities using programmable logic and the increased number of I/Os.

In the next subsections, we will first briefly discuss the benefits of using SoC FPGAs. Then, we will detail the proposed architecture and software implementation of the real-time control system. Finally, we will discuss initial tests performed using a mechatronic platform for validating the design before moving to the actual system. Then, in the next section, a comparison between the various systems will be carried out.

*5.1. An Introduction to SoC FPGA Heterogeneous Platforms.* The design of FPGA-based Systems-on-Chip has typically revolved around a hardware-centric view of system design, which has been deemed as too complex and technology-specific by nonspecialists, making the use of FPGA difficult beyond some niche applications in which their full potential has already been demonstrated.

Moreover, to make matters worse, the implementation of the management processing unit (MPU) of many of platforms (i.e., SoC, ASSP, or intelligent control devices) has been often carried out using the so-called soft-processors, which are mapped to the reconfigurable logic of the FPGA and usually do not have enough processing power for the most demanding applications. Furthermore, FPGA vendors have struggled to gain traction beyond some niche markets, since nonexperts regard the development flow as too complex.

For tackling the technological shortcomings briefly discussed above, the main FPGA vendors have made some major strides in adapting to the needs of the markets by introducing new capabilities, both technological and methodological [44]. Examples of the former are the increased integration of specialized functions such as Digital Signal Processing (DSP) blocks, distributed and configurable memory blocks (i.e., distributed BRAMs), and, more recently, large memory banks for data intensive applications, as well as the support of a plethora of communication protocols for moving large amounts of data.

On the other hand, and in order to address the hardware/software divide typically associated with FPGA-based SoCs, FPGA vendors have made major strides in introducing application grade processors, such as the ARM Cortex A9, capable of running full operating systems such as Linux, with the aim of simplifying the specification, implementation, and validation of heterogeneous embedded systems. This new kind of devices (which can be dubbed SoC FPGAs and depicted in Figure 7) couples a pair of high-performance ARM processors with a programmable logic extension block to promote a software-centric approach first and foremost.

Following this rationale, these Extensible Processing Platforms (as Xilinx has named their Zynq devices [42]) take a processor-first approach, in which the ARM processor development flow is emphasized over the traditional FPGA-based design approaches. This entails that software designers can start developing new applications right away, using the well-known and well-regarded ARM Cortex architecture, taking advantage of a fixed number of standards modules and interfaces, which were briefly discussed in Section 3.3 and shown in Figure 3.

In many instances, when developing a new product or project, the first step entails developing a proof of concept. Thus, the designers are thus less concerned about customizing the system requirements for specific customer or niche market. The most important concern at this phase is to have the maximum amount of flexibility to determine which functions are needed for the basic prototype in terms of the constituent components required for the embedded application.

Then, in a second phase, the design team can fine-tune the application to meet specific constraints (i.e., power consumption and real-time performance) by using profiling tools, which can help them to decide whether any segments of the applications can be sped up exploiting hardware implementation, discharging the main processor of some time-consuming processes. SoC FPFAs like the Zynq integrate a tightly coupled programmable logic extension block that allows designers to partition their hardware and software functions based on system requirements and to customize the device for a given application scenario [45]. They can implement functions in the programmable logic extension block to create their own application-specific, highly optimized Systems-on-Chips (SoCs), with the additional advantages of reducing chip-count and the complexity of the circuit board, as well as avoiding signal integrity issues.

It is at this hardware specialization phase that the methodological aspects hinted above come to the fore. In order to accelerate the integration of complex SoC and simplify the design process for nonexperts, FPGA vendors offer nowadays a variety of means for translating and implementing application-specific functions into hardware accelerated functions. The functions can be written using Hardware Description Languages (HDLs) or translated to RTL using High-Level Synthesis (HLS) techniques and then wrapped by bus interfaces for promoting IP reuse and taking advantage of the HW/SW interface and associated application-programming interface (API), which makes the call and use of the function easier from the application development perspective [46].

In order to make the communication between the processor and the programmable logic more efficient, the architecture of SoC FPGAs such as the Zynq is completed by industry standard AXI interfaces, which provide high-bandwidth, low-latency connections between the two parts of the device. This means that the processor and logic can each be used for what they do best, without the overhead of interfacing between two physically separate devices.

In this paper, we leverage the capabilities of these newly introduced heterogeneous Extended Processing Platforms for implementing the control system described in Section 4 and depicted in Figure 6. The main goal of this hardware implementation is to overcome the limitations of the previous

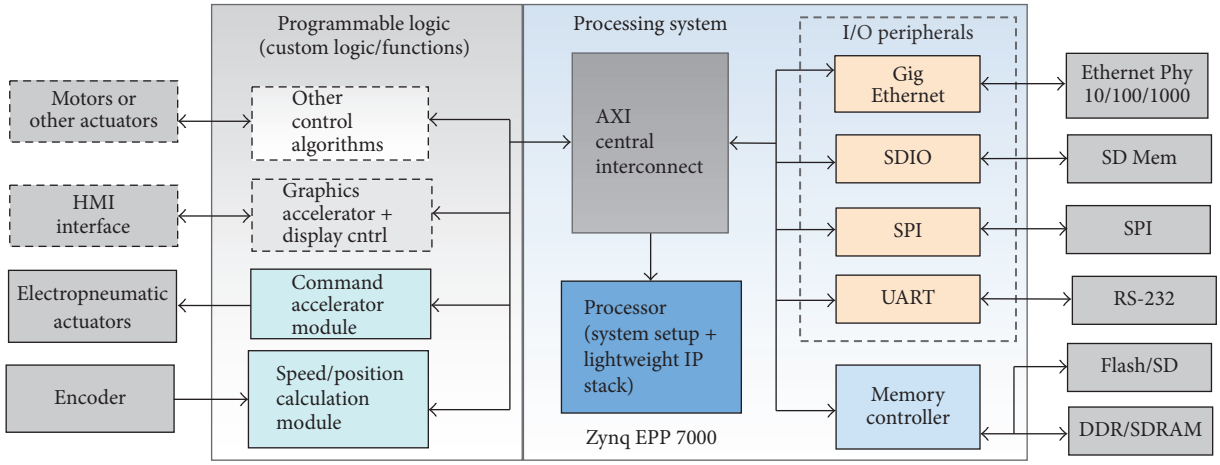



Figure 7: Architecture of the implemented SoC FPGA-based industrial intelligent electronic device.

PLC and MCU-based solutions and to meet the stringent constraints and capabilities introduced in the previous section.

*5.2. Intelligent Electronic Controller for Fiber Placement.* In this section, we introduce the proposed architecture (Figure 7) for speeding up the deposition of carbon fibers in the system described above, as well as the integration of the solution in the overall design chain of Coriolis Composites. The main rationale of the proposed architecture is to overcome as much as possible the communication bottleneck of the original PLC-based architecture, while maintaining compatibility with the CAD tools deployed for generating the fiber placement control commands and fostering the upgradeability of the system.

The bottleneck created by the use of the BoxPC has been circumvented by implementing a TCP/IP client using a lightweight IP stack, using the Ethernet MAC module integrated in the SoC FPGA for transferring the entire control program to the IED controller. The fiber deposition path is generated using a couple of pieces of software, CATFiber and CADFiber, which in tandem produce CAD data for a workpiece. This code is encompassed, on one hand, by the orientations of the carbon plies for each layer and, on the other hand, by the actions to be performed upon each fiber (Figure 5).

The CAD tools enable importing and visualizing surface and geometry information of the manufacturing tools and jigs, allowing the generation of ply sequences (defined by their contour and a reference curve or ply direction). Moreover, the tools are tightly coupled with quality assessment analyses of draped laminated complex surfaces and augmented with fiber covering simulation tools, which cater for fiber angle deviation and steering. If necessary, a ply can be cut automatically into smaller sections in order to fulfill maximum angle deviations.

On the other hand, the Composites Manufacturing Module of the CAD tools allows the automatic generation actions for the tapes, that is, bands of several fibers that are deposed over the surface in a computer-controlled manner. Premanufacturing checks can be performed thanks to various analysis tools including fiber compaction, roller crush, and tool path viewing.

The design process for a given piece is as follows: (i) a laminate piece is designed with the required number of plies and orientations, (ii) subsequently, plies are generated for a mould surface based on the CAD data from the piece, (iii) afterwards, a deposition program for the tapes is generated depending on the number of fibers to be used for a particular scenario (current systems support 16 fibers simultaneously), and finally (iv) tool paths are created in Kuka Robot Language (KRL) as a succession of linear movements or spline based displacements.

Once a piece has been created and simulated, the program is stored in an XML format and the program is executed by the BoxPC module described in Section 4.3 (Figure 6). As discussed above, the control for the robot is independent of that for the pneumatic actuators in the deposing head: the main PLC caters for positioning the robot along the trajectories generated by CADFiber, while a real-time OS running on the BoxPC controls the actions associated with the fiber deposing subsystem, introducing the previously discussed performance bottleneck.

Indeed, in spite of the capabilities of the BoxPC, a control loop encompassed by the position and speed signals coming from the robot limited the attainable speed at which the robot head could react, since this information is vital to trigger the control signals for the actuators in the head (Figure 5(a)). This is because the code generated by CADFiber for triggering the actuators in the head is dependent upon the robot position information, and in order to improve the accuracy of the finished pieces, all the encompassing actuators per fiber have to react very fast in a highly synchronized and repeatable manner. Additionally, the internal calculations by the BoxPC and the latencies introduced by the Profibus link only made matters worse.

Therefore, it was decided early on to substitute the BoxPC with a dedicated real-time embedded controller, which could satisfy the requirements briefly discussed at the end of Section 4.3. First, real-time connectivity is achieved by

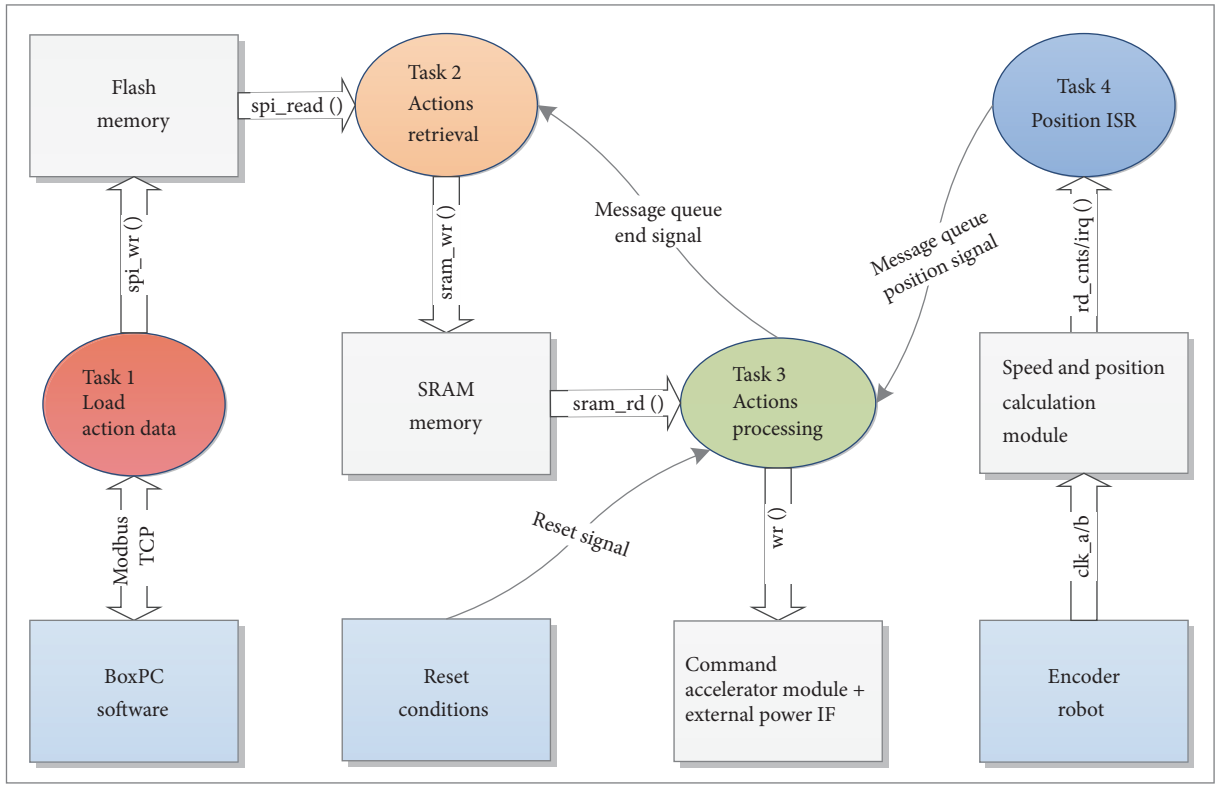



Figure 8: Architecture of the real-time control application based on FreeRTOS.

implementing an Ethernet client on the hybrid reconfigurable device, taking advantage of the integrated MAC controller. In this manner, we are able to communicate order to the deposing system, but the totality of the program resides now in the embedded platform, reducing the communication latency. The program can be remotely sent by the PLC via Modbus and stored in internal flash memory or controlled from the HMI for testing purposes by modifying sections of a Modbus stack. A real-time operating system (FreeRTOS) running on the ARM processor caters for the timely execution of the various components in the application (which is shown in Figure 8) and fosters reuse and maintainability of the developed code, which can be easily migrated to other platforms as well. We harness as well the capabilities of the Zynq heterogeneous platform for maintaining back compatibility with previous solutions by supporting legacy buses such as I2C, SPI, and RS-232, but the main advantages are the increased I/O capabilities and customization afforded by using the programmable section of the device.

As can be seen in the left side of Figure 7, the programmable logic of the heterogeneous platform has been used to move some of the functionalities that were missing in the BoxPC implementation, while overcoming the limitations of more constrained embedded devices (i.e., microcontroller units or MCUs). First of all, a specialized circuit has been implemented for decoding the quadrature encoder signals (a pulse detection circuit) and for computing the position and speed of the robot in real time without the intervention of the MPU. Secondly, a configurable expansion port is used for configuring the number of fibers to be deposed at any given point, according to the specifications of the CAD program stored in the internal memory; since this module has been mapped to the programmable logic, it can be upgraded to accommodate future developments.

The programmable section of the hybrid device contains enough resources to accommodate extra functionalities as can be observed in the top left corner of Figure 7.

Such functionalities could include an independent controller and graphics accelerator for implementing an HMI for monitoring and testing purposes (i.e., using a customized programmable accelerator, such as the Xylon IP). Furthermore, as we will discuss in Section 6.1, the proposed architecture has been first integrated in a test bench platform before moving to the final implementation in the actual robot; the test bench platform incorporates a motor that is currently controlled separately, complicating the testing process. Incorporating the control algorithms and electronic drive for the motor directly in the device could make a seamless demonstration and learning tool.

*5.3. Hardware/Software Architecture: RTOS and Custom IPs.* In this section, we will briefly discuss how the real-time embedded application has been conceived and implemented in the ARM processor integrated in the heterogeneous all programmable Zynq platform. As described before, such platforms foster a processor-first approach, in which the

ARM boots first, performing subsequently duties such as system initialization and configuration. Afterwards, the processor retrieves the configuration data for bootstrapping the programmable configurable logic, effectively fostering fail-safe strategies and avoiding some of the pitfalls of reconfigurable devices in control systems. Once the device and peripherals are up and running, the ARM processor takes a more managerial role, federating the proper execution of the overall application, catered in this case by a real-time operating system (RTOS). We have chosen FreeRTOS for a number of reasons: the code is open source and widely used and documented. Furthermore, the RTOS supports a wide range of microcontrollers and MPUs and has been especially designed for medium range devices, albeit consuming a very small memory footprint.

The task diagram for the application as implemented using FreeRTOS is shown in Figure 8. The RTOS performs the following actions for a given deposition program.

*Task 1*. The application can start by receiving a write request from the TCP/IP server (main PLC), in which case a task called *Load Action Data* retrieves the program data for the actuators, stored temporally into a Modbus table, using a tailored lightweight TCP/IP stack running on the ARM processor.

*Task 2*. The deposition program can then move to nonvolatile memory (i.e., SD or flash) through an *Actions Retrieval* task. Once the program has been stored, the system waits for an initialization signal from the main PLC to start execution.

The system can operate in two modes: in normal and in force/debug modes. The latter case is used for writing commands directly to a Modbus table, interacting directly with an external Human Machine Interface that enables testing arbitrary patterns monitoring any problems with the system. In the former case, the RTOS allocates portions of the program onto SDRAM memory using DMA for faster processing and uses the position and speed information from the system to execute these commands generated by CADFiber.

*Task 3*. In normal operating mode, once the program execution is triggered by the main PLC, the commands are retrieved from the SDRAM memory (running the Actions Processing) software task in the ARM processor, which basically places the initial memory address for a given command and amount of actions to be performed. This information is then sent as a burst to custom hardware accelerator, labeled *Command Accelerator Module* (CAM) in Figure 8.

The CAM module is a hardware accelerated function wrapped using an AXI Intellectual Property Interface module, as depicted in Figure 9(a). This module resides in the programmable logic section of the device and has been implemented in such a manner that the number of actuators can be automatically selected by the application, enabling seamlessly modifying the external interface without undergoing any hardware modifications in the rest of the system. The module was implemented in VHDL using Xilinx ISE, functionally verified and subsequently wrapped with the AXI IPIF using the IP creation infrastructure provided by Vivado.

This CAM module permits as well the interaction between the *Actions Processing* task at the RTOS level and the controlled plant via the HW/SW interface that translates the logical actions in the control program into electrical signals that drive the external electropneumatic controller that is in charge of actually triggering the pneumatic actuators in the robot's head, as depicted on the left side of Figure 7. The module acts basically as a processor-configurable demultiplexer: depending on the head configuration, the module can control 16, 32, and 64 fibers, and, as such, the CAM controls how the data bursts are mapped from memory to the outputs, using a configuration register, a set of multiplexers, and a state machine that controls the shown datapath.

Nonetheless, the capabilities of the Zynq platform foster backward compatibility through the use of QUAD SPI IPs that can be used with previous versions of the electropneumatic control system. This previous version, based on a microcontroller, used an SPI port and I/O expander to control up to 16 actuators; in the Zynq, several of these ports can be mapped in the PL section and control various sets of fiber bundles, but there is an associated timing penalty, as will be discussed in the next section.

*Task 4*. In order to trigger the actions at the correct time stamps, the processor subsystem has been extended with a *Speed and Position Calculation Module* (SPCM) (shown in Figure 9(b)). This hardware accelerator, also implemented in the PL section of the Zynq, is responsible for calculating the position and speed of the robot using the signals from the quadrature encoder in tandem with the circuitry integrated in the AXI IP shown in the figure.

As with the CAM module, the circuitry was described and functionally validated using Xilinx ISE and then exported to Vivado for creating a customized AXI-based hardware accelerator, which can be accessed by the processor via the HW/SW interface provided by the RTOS. In particular, the RTOS retrieves the next position at which a command is to be executed next and stores it into an internal register of the SPCM IP.

This value is compared with the position calculated by the SPCM and, depending upon the current speed, it can anticipate when the next command should be sent to the CAM IP, which is signaled to the RTOS via a message queue, triggered by an interrupt from the IP and captured by the *Action Processing* task, as depicted in Figure 8.

On the other hand, the *Action Processing* task notifies the *Actions Retrieval* task whether it requires more data, so commands are always available for streaming. The processor can then buffer the data corresponding to the next section of the program onto the CAM internal FIFO, effectively eliminating unnecessary waiting times and thus increasing the attainable deposition throughput. It must be noted that these tasks are run cyclically by the FreeRTOS scheduler, which is generated automatically during the compilation of the operation, making the application easier to maintain and scale.

In the following subsection, we will briefly show how the system has been put together using Vivado and discuss aspects related to hardware resources utilization and the performance of the solution.

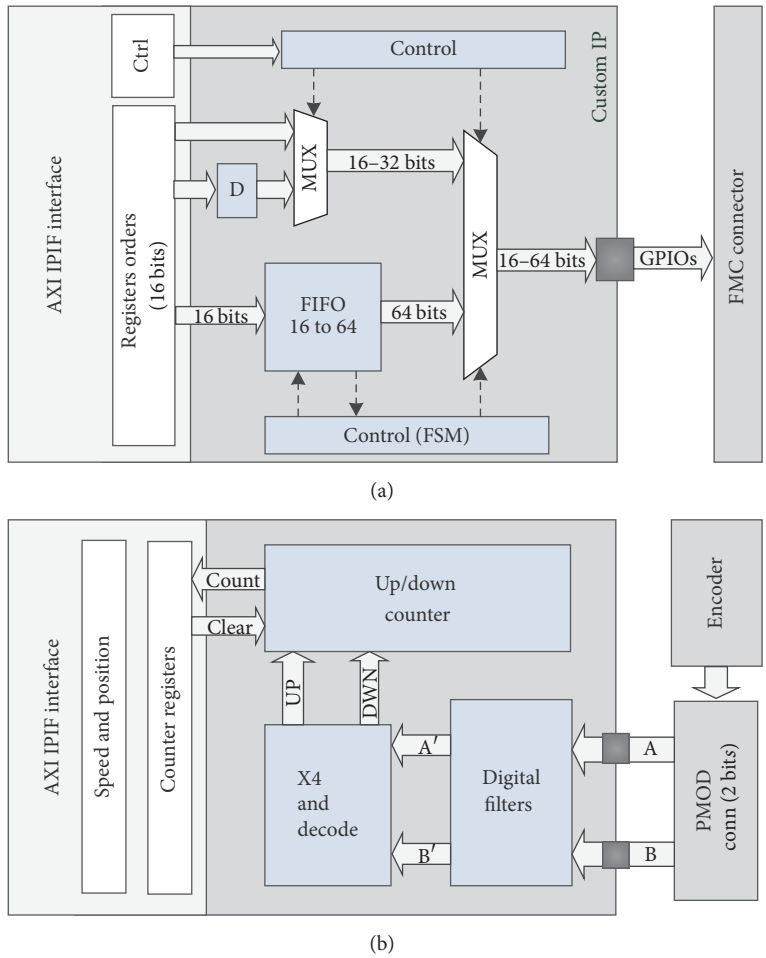



Figure 9: Hardware accelerators for the (a) Command Accelerator Module and the (b) Speed and Position Calculation Module.

*5.4. Implementation Results in the Zynq-7000 Platform.* As described in the previous section, the CAM and SPCM HDL descriptions have been integrated as AXI-based hardware accelerators so they can be integrated into the proposed SoC platform, using Vivado IP integrator. Furthermore, the IPIF interface enables the application engineer to interact with the underlying hardware modules via the HW/SW interfaces via a simplified API and to exploit their functionalities by encapsulating them as FreeRTOS tasks. The SoC platform introduced in Section 5.2 (and depicted in Figure 7) was thus created using Vivado, targeting the ZedBoard (which integrates the Zynq-7010 EPP, with a bus clock of 100 MHz), as depicted in Figure 10.

The synthesis results showing the resource utilization for each of the modules are summarized in Table 1. The table compares the resource utilization of the CAM and SPCM modules with that of a single instance of the QUAD SPI, the means of communication of the MCU-based implementation.

On the other hand, the overall resource utilization of the modules in the PL region of the Zynq device is summarized in Table 2. The resource utilization, including other modules for putting the SoC together, accounts for 7.4% (LUT) and 4.9% (FF). No BRAM or DSP blocks have been used, leaving ample resources for implementing other modules or control algorithms, as shown in Figure 7.

Table 1: Hardware resource utilization of each of the modules.

| Module | LUT | FF | BRAM/DSP |
|---|---|---|---|
| QUAD SPI | 339 | 539 | 0 |
| CAM | 104 | 220 | 0 |
| SPCM | 93 | 186 | 0 |

Table 2: Overall resource utilization for the modules in the PL section.

| Resource type | Usage | Total available resources | Utilization (%) |
|---|---|---|---|
| LUT | 1297 | 17,600 | 7.40% |
| FF | 1716 | 35,200 | 4.90% |

The platform description was then exported to SDK, where the application was put together using the FreeRTOS Zynq port and associated files. The design was then programmed onto the device to carry out the tests, first in the mechatronic testbed to be presented in Section 6.1 and,

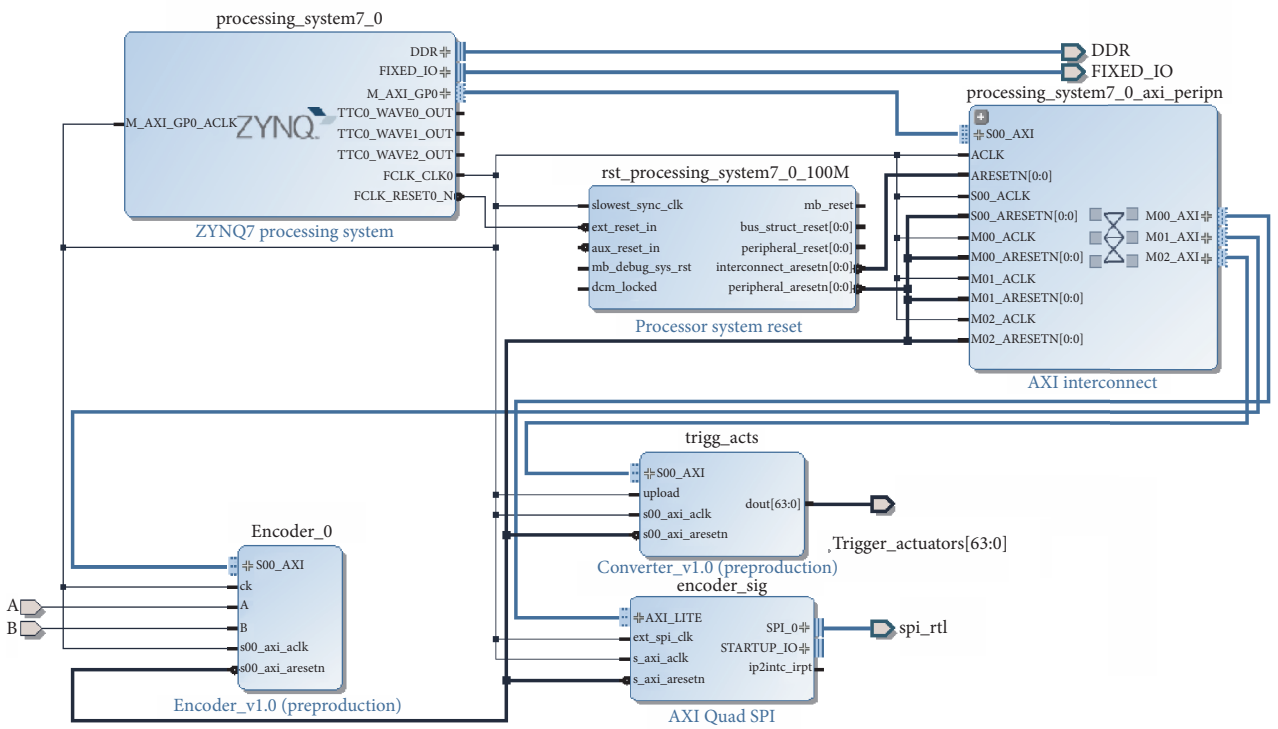



Figure 10: Implementation of the proposed architecture using Vivado.

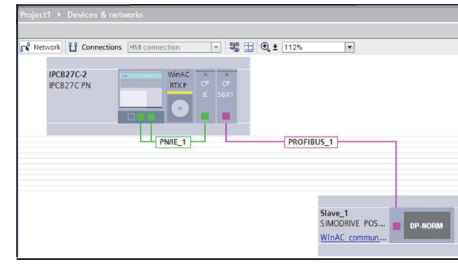
once validated, in the actual AFP robot in the ComposiTIC facilities.

## 6. Experimental Results and Validation of the Solution

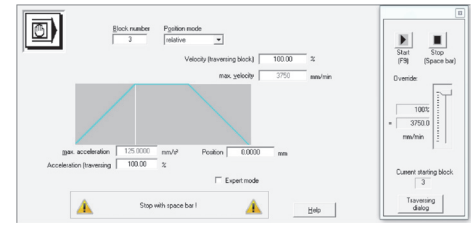
In this section, we briefly discuss how the proposed platform has been initially tested in order to experimentally validate the correctness of the software application and to assess the performance gains that were expected using FPGAs in this particular context, deploying initially a custom mechatronic testbed to validate the solution in a safe setting. Subsequently, we delve into experimental tests carried in the real robot and how the proposed SoC FPGA-based solutions have enabled us to speed up the fiber deposition process in the AFP robot while increasing the precision in the overall process.

*6.1. A Mechatronic Platform for Fast-Prototyping Purposes.* The main rationale for using a mechatronic testbed was to emulate the fiber deposition process (which is essentially performed in a single axis) by moving a deposition head with the aid of one-axis actuator. This one-axis mechatronic system is driven by SIMODRIVE POSMO A motor (from Siemens), which is controlled by SIMATIC Box PC (IPC 827C) running a real-time operating system (WinAC RTX) and enables the control of the motor via Profibus DP.

The control architecture was created using the Total Integrated Architecture (TIA) software module by Siemens, as depicted in Figure 11(a). This software enables creating the control interface between the SIMATIC Box PC and




(a)



(b)

Figure 11: Architecture interface of the Siements Sysmatic motor controller.

the motor and setting configuration parameters such as the motor rotational speed, max acceleration, and the desired position, as depicted in Figure 11(b). This architecture enabled us to control the speed and acceleration of the overall system and to test response time of the proposed solution.

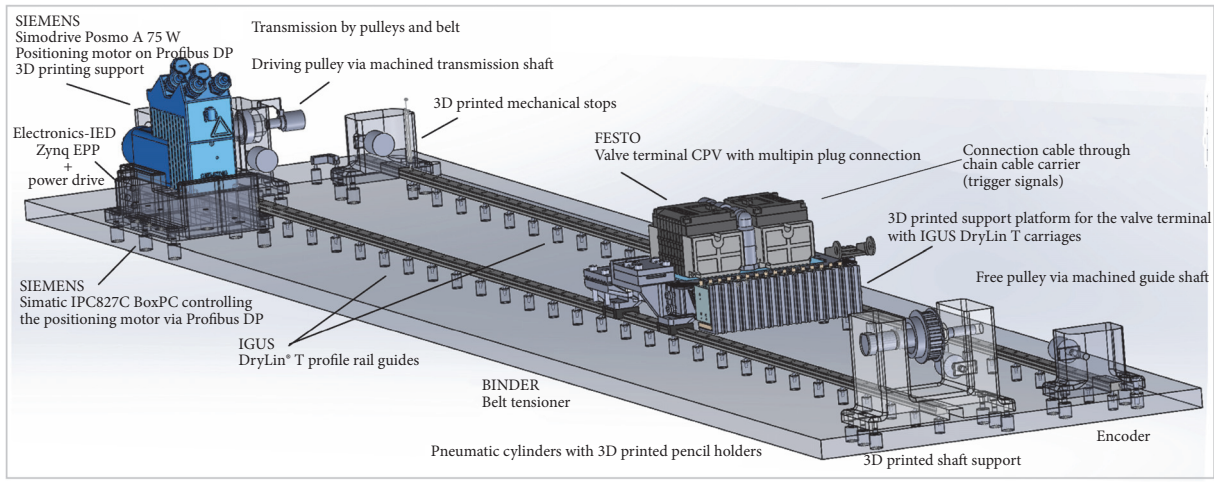



Figure 12: Reconfigurable mechatronic testbed platform and one axis actuator for fast-prototyping and experimental purposes.

The mechatronic testbed or fast-prototyping platform is depicted in Figure 12. It integrates a Siemens SIMODRIVE POSMO A [47] motor (controlled using the WinAC RTX OS running in SIMATIC Box PC, model IPC 827C), the Zynq-based intelligent electronic device controller, and, finally, the FESTO electropneumatic head, with several internal mechanical actuators that emulate the behavior in the actual robot head.

The mechatronic system has been conceived as a one-axis deposition system, as we are solely interested in the response time of the actuators in the robot head as it moves along a deposition trajectory. The FESTO air distribution system, driven electronically by the proposed IED, is fixed to a base plate attached to a rail system on the bottom and attached to a motor and a shaft support in the other end which enables free movement in a single axis, using a BINDER belt tensiometer.

It must be noted that the deposition head deployed in the testbed mechatronic system is not necessarily identical to the one in the actual AFP robot, since at this stage we were not interested in implementing the behavior of each actuator independently but more in the overall response time. Indeed, the main rationale for the FPGA-based implementation presented in this paper is to test the response time of the actuators in the AFP deposition head (see Figure 5(b)). Nonetheless, the response time for actuators responsible for the fiber cutting is the most sensible and we concentrated our efforts on characterizing and testing the limits of these in order to push the limits of the original platform.

Therefore, for the experiments performed with the aid of the mechatronic testbed, the programs generated using CAD-Fiber were preprocessed in order to include only commands associated with the fiber rolling and cutting actuators, which were then sent to the embedded IED, either as complete programs or applied directly using the HMI client. As mentioned previously, a TCP/IP server has been implemented using a lwIP stack running in the Zynq EPP 7000, which has been conceived in such a way that the actuators in the dummy robot head can be controlled using the Command Accelerator Module in the programmable logic section of the SoC FPGA device.

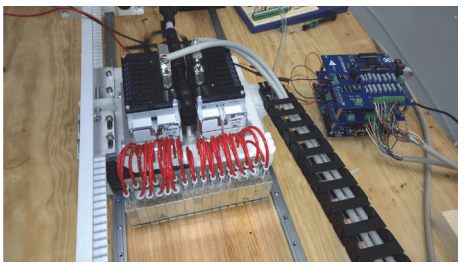


Figure 13: IED control system for the head (pneumatic valve actuator terminal).

For this purpose, a ZedBoard platform integrating a Zynq EPP 7000 device has been extended with a custom board to interface the IED with external signals, as depicted in Figure 13. For instance, the FMC connector in the board has been used to connect the processor subsystem in the device, via the customized hardware accelerator in the programmable logic section of the Zynq, with the power electronics in charge of driving an arrangement of Siemens electropneumatic valves. The latter subcomponent acts as the bridge between the electronic control systems and the actual mechanical action and represents thus the most important component of the system.

Furthermore, other components and sensors have been integrated in the mechatronic platform and interfaced with the IED. For instance, a quadrature pulse decoder has been integrated in the same axes of the Siemens SIMODRIVE POSMO A motor (through a mechanical coupling and the belt tensiometer) to provide information about the position of dummy deposition head, which is fed to the Speed and Position Calculation module in the programmable section of the SoC FPGA through an external port.

It must be emphasized that the use of the Zynq EPP has enabled us to seamlessly move from an initial standalone

Table 3: Performance comparisons between the different solutions.

| Metrics/solution | Cutting speed | Cutting precision | Response time |
|---|---|---|---|
| BoxPC-based solution | 400 mm/s | 1 mm | 2.5 ms/cycle |
| MCU-based solution | 1000 mm/s | 0.1 mm | 250 us/cycle |
| Proposed solution | 1500 mm/s | 0.05 mm | 50 us/cycle |
| Gain in performance | *3.75x* | *20x* | *50x* |

prototype to a fully functional proof-of-concept mechatronic system and, finally, to its deployment in the actual AFP robot. Preliminary results obtained using the testbed as well as actual tests in the robot will be detailed in the next section.

*6.2. Experimental Results and Performed Tests.* For testing the increased processing capabilities of the proposed architecture, tailored test patterns were generated using CADFiber and tested using three different solutions for benchmarking purposes: the original BoxPC-based solution, a platform based on a Renesas microcontroller running FreeRTOS, and, finally, the SoC FPGA-based Zynq implementation, as depicted in Table 1.

Several design patterns, customly designed to trigger the cutting sequences in the deposition program were first generated and tested in the mechatronic subsystem presented in Section 6.1. These patterns needed to be tested over a distance of one meter, given the geometric limitations of the one-axis actuator; the required acceleration and nominal speed of the system were remotely controlled via Profibus DP using the SimoCom A software (as depicted in Figure 11).

As thoroughly discussed in the article, the main goal of the MCU and SoC FPGA-based implementation was to speed up the command processing and triggering process in order to attain a higher response time in the overall deposition process.

Additionally, the control system needs to constantly calculate the current speed and position of the head in order to anticipate any upcoming commands (i.e., triggers) accordingly. The original PLC-based implementation relied on a feedback loop, which introduced a significant delay and hindered the entire deposition process; this issue has been overcome in the two subsequent solutions by integrating this computation directly in the embedded system.

Several tests were carried out to compare the performance of the various solutions outlined above, which are summarized in Table 3 and discussed as follows. As stated previously, the maximum deposition speed attainable by the PLC-based solution was 400 mm/s, which leads to a response time of 2.5 ms per cycle. The MCU-based IED solution fares much better in this regard, attaining a response time of 250 us per cycle (10x), which enabled us to depose complex patterns at over 1000 mm/s.

Nonetheless, the response time is not the only limiting factor. Despite of the gain in response time obtained by using the MCU-based solution, the number of actuators that can be triggered at once was rather limited, imposing a constraint in the fabrication time, as shown in Figure 14.

Due to confidentiality issues, we concentrate here only on relatively simple test pieces, as depicted in Figures 14(a) and 14(b), respectively. As the results in the analysis of the figure show, the increased deposition afforded by the SoC FPGA solution (1500 mm/s, with a 50x response time over the PLC-based solution), in tandem with a larger number of available triggers, has enabled us to significantly reduce the fabrication throughput.

The results in Figure 14(c) show a reduction in the fabrication process of 12%, 15%, and 55%, respectively, over the original implementation. Nonetheless, it must be noted that this reduction is greatly dependent on the geometry of the piece and on the frequency and number of rolling and cutting commands and thus requires further investigation.

## 7. Conclusions

The Industrial Internet of Things (IIoT), the idea that all systems should be connected on a global scale in order to share information, is quickly becoming a reality. Today, a growing number of companies, especially in the industrial equipment markets, are taking IIoT one step further by creating complex systems that integrate sensors, processing capabilities, and adaptable communications protocols to form intelligent factories, smart energy grids, and even smart cities.

In this paper, we have presented the implementation of a SoC FPGA-based intelligent electronic device, which has been seamlessly integrated into a previously existing infrastructure for an advanced fiber placement system. In this sense, the implementation proposed here can be subscribed to the smart factories paradigm, since the overall platform is in fact a distributed control system, which relies on complex industrial communication network to properly operate. Furthermore, some of the most demanding aspects of the original application have been migrated to a SoC FPGA to add a higher degree of intelligence and flexibility in the control of the deposition subsystem, which can accommodate future developments as well.

The very specific requirements of the application, which demanded not only very low response times but also flexibility in terms of reconfiguration of the deposition head and the control hardware and software, made a strong case for the use of FPGAs. The application necessitated a real-time and low-latency Ethernet communication, remote configuration and storage of the deposition programs, and the availability and customization of a large number of I/Os. Moreover, it also had to be backward compatible and to accommodate future developments (i.e., new industrial protocols for any-to-any connectivity approaches and also more intelligence and/or on-board processing on the edge).

Indeed, in order to maximize profitability, factories seek more flexibility in their layouts, more information about the process and manufactured products, more intelligence in the processing of this data, and an effective integration of the human experience/interaction (HMIs). However, as new technologies are introduced into the factory sector, those creating them need to overcome several constraints. The first and the most important is that production cannot stop.

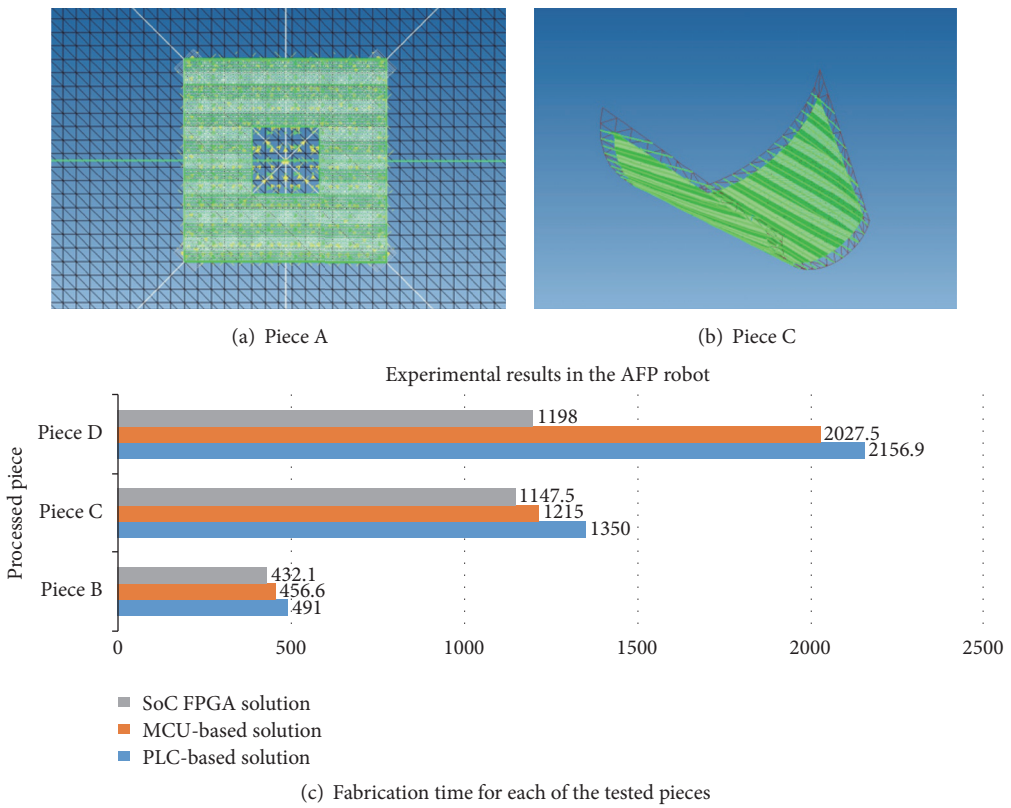



FIGURE 14: ((a) and (b)) Two examples of pieces used to test the capabilities of the proposed solution. (c) Deposition times for three pieces using the SoC FPGA-based solution, the MCU-based solution, and the original PLC-based solution.

New technologies must be compatible with old systems and interoperability among vendors should be facilitated. This has been achieved in our implementation by deploying reconfigurable devices, which have been demonstrated to be an exceptional rapid prototyping tool over the years, as well as a means to close the gap between hardware and software development, promoting as well important aspects as customization and upgradeability in the field, potentially reducing the costs associated with production downtimes.

Furthermore, modern industrial solutions should provide the means for taking the next step in automation, leading to more autonomous or decentralized analytics. In this sense, recent strides in reconfigurable devices (in particular, the introduction of Extended Processing Platforms such as the Zynq EPP 7000m) are making this convergence more likely, and the incorporation of such devices in the automation domain seems quite logical to us. In this sense, we envision the incorporation of prognosis approaches in the control loop in order to determine, predict, and prevent possible wear-out in the actuators, which would severely affect the performance of the fiber deposition head.

In the particular context of the application presented here, the use of SoC FPGAs enabled not only improving over the original implementation of the system in terms of performance but also migrating the two previous solutions in a seamless manner, while respecting the constraints cited above. In order to make this passage more straightforward, a SoC FPGA was chosen (the Zynq EPP 7000) in order to port the FreeRTOS implementation previously developed while gaining in customization capabilities through programmable logic and the extended number of I/Os.

As thoroughly discussed in this article, the SoC FPGA-based implementation of the fiber deposition control system introduced significant speed-up over the original PLC-based and MCU-based solutions, this done by overcoming the communication bottleneck of the former solution, while increasing the number of actuators that could be controlled by the latter. Furthermore, the hardware accelerators in the reconfigurable logic section of the device and the reduced latency in the communication gained through the increased integration have improved the real-time performance of the application. We have performed several experimental tests, first in a mechatronic testbed and subsequently in the actual robot, with various synthetic programs and later with actual pieces, showing a significant improvement in the attainable precision at higher speeds and, thus, improved throughput in the fabrication process.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

The authors wish to acknowledge Coriolis Composites for the support during the duration of this project. In the same vein,

they express their most sincere gratitude to the ComposiTIC research center for the support and access to its facilities. They also acknowledge Xilinx and Digilent for their generous donation of the ZedBoard platforms used in this study.

## References

[1] V. Vyatkin, "Intelligent mechatronic components: Control system engineering using an open distributed architecture," in *Proceedings of the 2003 IEEE Conference on Emerging Technologies and Factory Automation, ETFA 2003*, pp. 277–284, prt, September 2003.

[2] G. Morel, P. Valckenaers, J.-M. Faure, C. E. Pereira, and C. Diedrich, "Manufacturing plant control challenges and issues," *Control Engineering Practice*, vol. 15, no. 11, pp. 1321–1331, 2007, Special Issue on Manufacturing Plant Control: Challenges and Issues INCOM 2004 11th IFAC INCOM'04 Symposium on Information Control Problems in Manufacturing.

[3] K. Thramboulidis, "Challenges in the development of mechatronic systems: The Mechatronic Component," in *Proceedings of the 13th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2008*, pp. 624–631, deu, September 2008.

[4] M. N. Rooker, G. Ebenhofer, and T. Strasser, "Reconfigurable control in distributed automation systems," in *Proceedings of the 2009 ASME/IFToMM International Conference on Reconfigurable Mechanisms and Robots, ReMAR 2009*, pp. 705–714, gbr, June 2009.

[5] C. Paiz and M. Porrmann, "The utilization of reconfigurable hardware to implement digital controllers: a review," in *Proceedings of the 2007 IEEE International Symposium on Industrial Electronics, ISIE 2007*, pp. 2380–2385, June 2007.

[6] C. E. Pereira and L. Carro, "Distributed real-time embedded systems: Recent advances, future trends and their impact on manufacturing plant control," *Annual Reviews in Control*, vol. 31, no. 1, pp. 81–92, 2007.

[7] S. Z. Ahmed, G. Sassatelli, L. Torres, and L. Rougé, "Survey of new trends in industry for programmable hardware: FPGAs, MPPAs, MPSoCs, structured ASICs, eFPGAs and new wave of innovation in FPGAs," in *Proceedings of the 20th International Conference on Field Programmable Logic and Applications, FPL 2010*, pp. 291–297, September 2010.

[8] C.-H. Yang and V. Vyatkin, "Design and validation of distributed control with decentralized intelligence in process industries: a survey," in *Proceedings of the IEEE INDIN 2008: 6th IEEE International Conference on Industrial Informatics*, pp. 1395–1400, kor, July 2008.

[9] A. Zoitl, T. Strasser, K. Hall, R. Staron, C. Sunder, and B. Favre-Bulle, "The past, present, and future of iec 61499," in *Holonic and Multi-Agent Systems for Manufacturing*, vol. 4659 of *Lecture Notes in Computer Science*, pp. 1–14, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[10] E. Monmasson, L. Idkhajine, and M. W. Naouar, "FPGA-based controllers," *IEEE Industrial Electronics Magazine*, vol. 5, no. 1, pp. 14–26, 2011.

[11] A. de la Piedra, A. Braeken, and A. Touhafi, "Sensor systems based on FPGAs and their applications: a survey," *Sensors*, vol. 12, no. 9, pp. 12235–12264, 2012.

[12] L. Gomes, E. Monmasson, M. Cirstea, and J. J. Rodriguez-Andina, "Industrial electronic control: FPGAs and embedded systems solutions," in *Proceedings of the 39th Annual Conference of the IEEE Industrial Electronics Society, IECON 2013*, pp. 60–65, aut, November 2013.

[13] A. Chang, "Innovative platform-based design for the industrial internet of things," *Xilinx Xcell Journal*, vol. 92, pp. 32–37, 2015.

[14] A. Astarloa, "Intelligent gateways make a factory smarter," *Xilinx Xcell Journal*, vol. 94, pp. 14–21, 2016.

[15] A. Khamis, D. Zydek, G. Borowik, and D. S. Naidu, "Control System Design Based on Modern Embedded Systems," in *Computer Aided Systems Theory - EUROCAST 2013*, R. Moreno-Díaz, F. Pichler, and A. Quesada-Arencibia, Eds., vol. 8112 of *Lecture Notes in Computer Science*, pp. 491–498, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, ISBN 978-3-642-53861-2.

[16] S. Ichikawa, M. Akinaka, H. Hata, R. Ikeda, and H. Yamamoto, "An FPGA implementation of hard-wired sequence control system based on PLC software," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 6, no. 4, pp. 367–375, 2011.

[17] A. Lüder, L. H. M. Foehr, T. Holm, T. Wagner, and J.-J. Zaddach, "Manufacturing system engineering with mechatronical units," in *Proceedings of the 15th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2010*, esp, September 2010.

[18] IEC, *Internaional Standard IEC 6113-3: Part 3: Programming Languages*, IEC, 1993.

[19] V. Vyatkin, "IEC 61499 as enabler of distributed and intelligent automation: State-of-the-art review," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 4, pp. 768–781, 2011.

[20] IEC, *Internaional Standard IEC 61499-1: Function Blocks - Part 1: Architecture*, IEC, 2005.

[21] D. Du, X. Xu, and K. Yamazaki, "A study on the generation of silicon-based hardware Plc by means of the direct conversion of the ladder diagram to circuit design language," *International Journal of Advanced Manufacturing Technology*, vol. 49, no. 5-8, pp. 615–626, 2010.

[22] S. Simard, J. G. Mailloux, and R. Beguenane, "Prototyping advanced control systems on FPGA," *Eurasip Journal on Embedded Systems*, vol. 2009, no. 5, Article ID 897023, pp. 1–5, 2009.

[23] Z. Hajduk, B. Trybus, and J. Sadolewski, "Architecture of FPGA Embedded Multiprocessor Programmable Controller," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 5, pp. 2952–2961, 2015.

[24] J. Álvarez, Ó. López, F. D. Freijedo, and J. Doval-Gandoy, "Digital parameterizable VHDL module for multilevel multiphase space vector PWM," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 9, pp. 3946–3957, 2011.

[25] A. Sathyan, N. Milivojevic, Y.-J. Lee, M. Krishnamurthy, and A. Emadi, "An FPGA-based novel digital PWM control scheme for BLDC motor drives," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 8, pp. 3040–3049, 2009.

[26] C. Buccella, C. Cecati, and H. Latafat, "Digital control of power converters—a survey," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 3, pp. 437–447, 2012.

[27] M. Shahbazi, P. Poure, S. Saadate, and M. R. Zolghadri, "FPGA-based reconfigurable control for fault-tolerant back-to-back converter without redundancy," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 8, pp. 3360–3371, 2013.

[28] X. Lin-Shi, F. Morel, A. M. Llor, B. Allard, and J.-M. Rétif, "Implementation of hybrid control for motor drives," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 4, pp. 1946–1952, 2007.

[29] R. Dubey, P. Agarwal, and M. K. Vasantha, "Programmable logic devices for motion control - a review," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 1, pp. 559–566, 2007.

[30] L. Zhang, P. Slaets, and H. Bruyninckx, "An open embedded hardware and software architecture applied to industrial robot control," in *Proceedings of the 2012 9th IEEE International Conference on Mechatronics and Automation, ICMA 2012*, pp. 1822–1828, chn, August 2012.

[31] T. Sutikno, N. R. N. Idris, A. Jidin, and M. N. Cirstea, "An improved FPGA implementation of direct torque control for induction machines," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 3, pp. 1280–1290, 2013.

[32] M. Bacic, "On hardware-in-the-loop simulation," in *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference, CDC-ECC '05*, pp. 3194–3198, esp, December 2005.

[33] A. Sanchez, A. De Castro, and J. Garrido, "A comparison of simulation and hardware-in-the-loop alternatives for digital control of power converters," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 3, pp. 491–500, 2012.

[34] R. Dubey, *Embedded System Design Using Field Programmable Gate Arrays*, Springer, 1st edition, 2009.

[35] R. Woods, J. McAllister, G. Lightbody, and Y. Yi, *FPGA-based Implementation of Signal Processing Systems*, Wiley, 1st edition, 2008.

[36] I. Kuon and R. Rose, *Quantifying and Exploring the Gap Between FPGAs and ASICs*, Springer, 2010.

[37] D. Groppe, *Robots take on composite layup, machine design*, 2003, http://www.machinedesign.com.

[38] J. Sloan, *Atl and afp: Signs of evolution in machine process control. high-performance composites, composites world*, 2008, http://www.compositesworld.com.

[39] P. Maitani, *Rapid layup: New 3-d preform technology, composites world*, 2012, http://www.compositesworld.com.

[40] G. Dell'Anno, I. Partridge, D. Cartié et al., "Automated manufacture of 3D reinforced aerospace composite structures," *International Journal of Structural Integrity*, vol. 3, no. 1, pp. 22–40, 2012.

[41] Siemens, "Simatic winac rtx the simatic s7 as software controller," http://w3.siemens.com/mcms/programmable-logic-controller/en/software-controller/software-plc-simatic-winac/simatic-winac-rtx/pages/default.aspx, 2016.

[42] Xilinx, "Ds190 - zynq-7000 all programmable soc overview," https://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf.

[43] R. Barry, *Using the FreeRTOS Real Time Kernel*, FreeRTOS Tutorial Books, 2010.

[44] M. Santarini, "Xilinx architects arm-based processor-first, processor-centric device," *Xilinx Xcell Journal*, vol. 71, pp. 6–11, 2010.

[45] Xilinx, *Zynq-7000 all programmable soc technical reference manual*, 2016, https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf.

[46] Crocket and Steward, *The Zynq Book*, The University of Strathclyde, 1st edition, 2015.

[47] Siemens, "Simidrive posmo a -distributed positioning motor on profibus dp - user manual," https://cache.industry.siemens.com/dl/files/007/78797007/att_46305/v1/SIMODRIVE_POSMO_A_BHB_A0813_us.pdf, 2016.