energies

MDPI

Article

SPOF—Slave Powerlink on FPGA for Smart Sensors and Actuators Interfacing for Industry 4.0 Applications

Giacomo Valente [1,*], Vittoriano Muttillo [1], Mirco Muttillo [2], Gianluca Barile [2], Alfiero Leoni [2], Walter Tiberti [1] and Luigi Pomante [1]

[1] Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica, Università degli Studi dell'Aquila, 67100 L'Aquila, Italy; vittoriano.muttillo@univaq.it (V.M.); walter.tiberti@graduate.univaq.it (W.T.); luigi.pomante@univaq.it (L.P.)

[2] Dipartimento di Ingegneria Industriale e dell'Informazione e di Economia, Università degli Studi dell'Aquila, 67100 L'Aquila, Italy; mirco.muttillo@graduate.univaq.it (M.M.); gianluca.barile@graduate.univaq.it (G.B.); alfiero.leoni@graduate.univaq.it (A.L.)

* Correspondence: giacomo.valente@univaq.it; Tel.: +39-0862-434701

Received: 20 March 2019; Accepted: 23 April 2019; Published: 29 April 2019

Abstract: We here present a new PLC-POWERLINK industrial solution for Industry 4.0 applications. The proposed solution provides the capability to separate the sensing functionality from the PLC-side, in demand for the reconfigurable FPGA implementation. In particular, we here provide a framework that supports the interfacing between the POWERLINK protocol and commonly used standards, such as I2C, SPI, and UART. This has been obtained by using a framework built around a soft IP-core Application Processor, which manages the interfacing with several POWERLINK slaves, able to support the data exchange with the POWERLINK Communication Processor. A practical application example and related implementation details are presented in the paper.

Keywords: POWERLINK; PLC; multi-core; FPGA; sensor interfacing

1. Introduction

Presently, industrial automation plays a very important role and it is constantly evolving: Industry 4.0 is a demonstration of that. Industry 4.0 (I4.0) is a phenomenon that aims to achieve a higher level of operational efficiency and productivity, as well as a higher level of automation, driven by both an application pull and a technology push [1]: the former refers to the opportunities for differentiation and competitive advantages, driven by automated, flexible and agile production processes that manufacturers can adopt [2,3]; on the other hand, technology push refers to the number of converging technological developments, which include wide-spread Internet connectivity, increasing miniaturization, cost-effectiveness, and capability of hardware such as sensors and actuators [4]. The technology push has also invested in industrial control systems that, due to the increasing complexity of plants to control, and the consequent need for smarter control techniques [5,6], have been moved from centralized solutions toward distributed feedback-based ones [7], requiring the introduction of modern technologies (namely, a technology push). In fact, typical industrial control systems, composed of Programmable Logic Controllers (PLCs), interfacing with sensors and actuators [8–10] and communicating by means of industrial fieldbuses [11], have to deal with even more complex physical plants to control, requiring more functionality, reactivity and adaptivity close to sensors (called edge) to enable innovative control techniques (such as the ones based on data [6]). Moreover, in addition to above-indicated requirements, adaptivity and reactivity, necessary to respond to variations of

phenomena in complex physical scenarios, bring with them the necessity for smart sensing solutions, in order to observe specific triggers that drive adaptation itself.

However, the adoption of new technologies can be costly and risky for small and medium companies. For instance, hardware and software development for distributed industrial control systems are often locked to a closed proprietary flow that requires compatible platforms and, consequently, involves the replacement of the current motors and mechanical systems to update a control system or its elements.

Therefore, there is a growing necessity for methodologies and related tools to support upgrading of the old equipment (i.e., to enforce reusability), while enhancing them with new features that allow providing functionality at the edge (i.e., to enforce flexibility). This represents our research problem. In addition, a solution to this problem needs to provide adaptation to different communication protocols, since distributed industrial control systems use different communication protocols that cannot be directly mixed, in order to maintain the integrity of the industrial field system.

In the literature, the answers to this problem can be found in two main areas: (i) software-based solutions, with and without protocol redundancy implementation, and reconfigurable-based solutions. Software-based solutions [12,13] satisfy the required flexibility, but they can present issues when their correctness needs to be verified. In these solutions, the coexistence of multiple communication protocols is either provided in software or with a protocol redundancy (i.e., implementation of a variety of communication protocol stacks to make the device compatible with multiple protocols): the former can be unable to meet the real-time needs required by high-speed industrial protocols, while the latter drives to an increase of the costs and occupied area when the number of the protocols grows. For these reasons, in recent years, with the development of programmable devices, reconfigurable hardware technology has been considered to solve the problem of smart interfacing [14–16]. The use of this technology is proposed in [14] to develop a node of interface with sensors and actuators in which two different protocols can be dynamically configured. In [16], an FPGA-based solution is adopted for the implementation of interfaces in a control system applied on *Urban Lighting Network*. Also, the work in [17] implements the control scheme and communication protocol on FPGA. We believe that this paved the way, i.e., the adoption of reconfigurable-based solutions (that, without loss of generality, we consider as FPGAs) to implement distributed control systems, gives the correct line of solving to the problem stated above. In fact, this makes possible to enforce reusability, by adopting FPGA to interface to different protocols; it allows enforcing flexibility since new functionality can be added exploiting both new accelerators and new software. Also, the use of FPGAs permits the interface to different fieldbuses, by exploiting their reconfiguration, and, finally, these platforms would give the possibility to implement the smart sensing solutions above indicated.

However, there are no works that offer a framework that supports (i) on upgrading of old types of equipment, interfacing with a fieldbus but paving the way to interface with others, (ii) on adding new functionality at the edge and (iii) on implementing smart sensing solutions.

Our contributions in this paper focus on the development of a framework, named *Slave Powerlink on FPGA* (SPOF) that targets industrial control systems interconnected with POWERLINK fieldbus and that supports on the three points above indicated. Our framework acts on FPGA-based platforms. We focus our attention on POWERLINK, since it is widely adopted in the industry [13] and it offers strong portability [12].

In more detail, the main contributions of the work are reported in the following:

- we propose an innovative framework (SPOF) that allows upgrading old equipment of industrial control systems based on Programmable Logic Control (PLC) devices, connected to sensors and actuators throughout the POWERLINK fieldbus. In particular, the framework allows strongly customizing the final hardware platforms that implement the control, in order to tailor it for the application field, throughout the use of Intellectual Property cores (IP-cores), specifically provided by the framework, to be implemented on Field Programmable Gate Arrays (FPGAs).

- We enrich the SPOF framework with the possibility to introduce a non-intrusive run-time monitoring system within the final hardware platform.
- We validate the proposed framework on a real industrial control system provided by B&R ACOPOS [18].

The paper is organized as follows. In Section 2, a background related to industrial control systems and POWERLINK protocol is reported, together with a detailed state-of-the-art analysis related to frameworks that allow the adoption of POWERLINK on industrial control systems. Then, the design methodology that stands behind the proposed framework is presented in Section 3, while in Section 4 the SPOF framework and the description of the main blocks and protocols involved in the system are introduced. Section 5 presents a validation of SPOF in a real industrial control system and, finally, Section 6 reports some conclusive considerations and presents future works.

2. Background and State of Art

In this section, a background related to industrial control systems is first presented; then, a state-of-the-art analysis related to solutions that allow the adoption of POWERLINK on industrial control systems is given.

2.1. Background

A typical distributed industrial control system platform can be seen as Programmable Logic Controllers (PLCs) interfacing with sensors and actuators, communicating by means of industrial fieldbuses [11]. A PLC is composed of a Central Processing Unit (CPU) that performs the main controller functions, a power supply and some I/O modules used to control actuators and to collect data from sensors [19–22]. On the other hand, examples of fieldbuses are PROFIBUS®, DeviceNet™, CANopen®, Modbus®, EtherCAT® and POWERLINK.

The POWERLINK communication standard is often chosen as the fieldbus, since it is completely open-source and it offers excellent time determinism. Products with POWERLINK interface are provided by many of the biggest industry manufacturers (e.g., B&R ACOPOS, Baldor MicroFlex e100, and Parker Aries EPL) [13]. Devices that combine processors functionality with the POWERLINK open media access control (MAC) layer are available on the market [12]. As an integral part of the Ethernet standard IEEE 802.3, the Ethernet POWERLINK layer (EPL) protocol can be implemented with any Ethernet chip available on the market and, in addition to traditional Application Specific Integrated Circuits (ASIC)-based approaches, also FPGA solutions are available. In an EPL network, up to 240 slave controlled nodes (CNs) can be managed by a master node (MN) in a topology of network that can be implemented in line, star or tree type configurations [13].

The communication is structured in cycles and two temporal phases can be distinguished within each cycle. The first is a *Time Division Multiple Access* (TDMA) phase, divided into several time slots. The time slots are equal to the number of CNs, and only one CN can transmit in each slot. The phase begins with a "*start of cycle*" (SoC) frame, broadcasted by the MN to allow the synchronization of all CNs. Then, each CN responds with "*poll response*" (PRs) frames to "*poll request*" (PRq) frames transmitted by the MN. This is a deterministic communication phase that, without collisions, ends with a final broadcast PRs frame sent by MN after all CNs have been serviced. The second phase is an *asynchronous* communication phase, starts with a "*start of asynchronous*" (SoA) frame, transmitted by MN to all the nodes. With this message, a single node is enabled to send an "*asynchronous send*" frame (ASnd) with proper data. An *idle phase* is interposed between the ASnd and a new SoC in order to fill the remaining cycle time. Figure 1 shows the timing diagram of an EPL communication cycle. The EPL messages contain the message type (SoC, PRq, PRs, SoA, or ASnd), the destination and source node addresses and the payload.

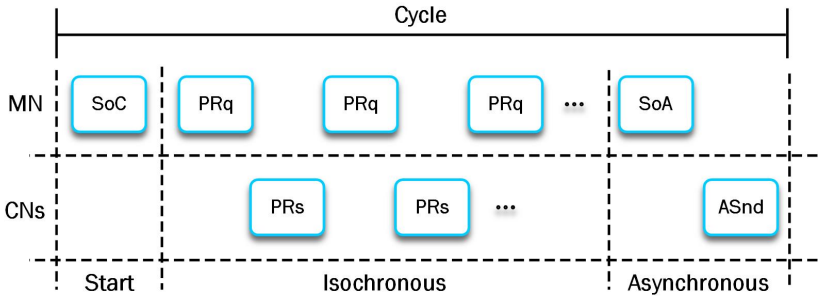

Figure 1. Timing diagram of an EPL communication cycle.

2.2. State of Art

The POWERLINK protocol is widely adopted in industry automation and, at the same time, it is still a research topic [14,23,24]. It is used in compact system solutions for parallel robot control [25], as an Energy Efficient Ethernet standard on Industrial Ethernet systems [26] and also as the backbone in systems that incorporate seamless connection capabilities of smart wireless devices [27]. Furthermore, in [15] the TDMA mechanism of the POWERLINK protocol has been extended to manage a wireless interface.

In [17], the POWERLINK Protocol is used to develop a control system of distributed energy resources composed by photovoltaic panels, windmills with induction generator, supercapacitors, Absorbent Glass Mat (AGM) battery, controllable load, power converters and electric power system. These components can be connected to make a micro-grid system controlled by an energy management system based on DSP, ARM Cortex A9 processors and FPGA hardware. EPL is used to build a communication channel that allows a reduction in adverse effects caused by sudden changes in the power generated from renewable energy sources.

The EPL protocol is also used in computerized numerical control (CNC) systems to interconnect the machine servo drives with real-time controls executed by processors with a real-time operating system (RTOS). In [13], an open PC-based CNC system with EPL communication protocol is presented, in order to provide a low-cost interfacing based on a general-purpose PC with free open-source RTOS (Linux RTAI) and CNC control software (EMC2). This allows the interfacing, without dedicated hardware, to various commercially available servo drives.

In [12], an Open Platform for Abstraction of Real-Time Communication (OPARTC) is proposed for control applications in Networked Medical Systems (NMS), based on POWERLINK protocol. In this field, a "*Real-time Communication Abstraction Provider*" (RT-CAP) has been defined as a central component of the OPARTC for the static and dynamic network configuration and for the synchronization of the real-time nodes. A development board, processor-based, has been used as "*Real-time Communication Abstraction Bridge*" (RT-CAB) unit to implement slave functionality and interfacing the master unit (RT-CAP) with sensors and actuators (interfaces such as UART, SPI, I2C, etc.).

As stated in the introduction, due to the various application requirements, the final shape of a complete distributed industrial control system is a combination of equipment that provides a hybrid system. These devices use different communication protocols that cannot be directly mixed, and software, protocol redundancy, and reconfigurable-based solutions have been adopted to manage these communication requirements, together with proper behaviour in real time systems. In particular, reconfigurable-based solutions have been taken into account in recent years, thanks to the development of programmable devices [14–16]. The use of this technology is proposed in [14] to develop a node of interface with sensors and actuators in which two different Ethernet protocols (EtherCAT and POWERLINK) can be dynamically configured. In [16], an FPGA-based solution is adopted for the implementation of both MNs and CNs for the implementation of a control system for a LED Urban Lighting Network based on the POWERLINK protocol.

As shown by the state of the art, the POWERLINK protocol is adopted in many application scenarios, such as robot control, smart grids, industrial automation, and different works have been done to support its implementation. In industrial control systems, the MNs may be implemented through PLCs, but also solutions based on PCs and microprocessors are available. Also, for the slave functionality of the controlled nodes, the technologies for the implementation may be different (microprocessors, DSPs, FPGAs). The SPOF framework supports the implementation of the slave functionality of a CN with an FPGA-based solution. Although this system can be managed by any kind of MN, the focus is on PLC-based control systems. As in [12], with SPOF the obtained slave POWERLINK is based on a separation between the control functionality, performed by the PLC, and the task of I/O management. The latter is needed to be the slave node, together with the actuation of external communications to sensors and actuators (through interfaces such as UART, SPI and I2C) and with a partial pre-processing.

By means of a slave implemented using SPOF framework, briefly indicated as SPOF-slave in the following, the PLC can transmit and receive, with a single POWERLINK link, information with several external devices, as shown in Figure 2. This enforces re-usability, since we can interface, using a single platform, with multiple external devices: as counter-example, in [12] one slave unit is allocated for each external device. Compared to other solutions that implement the POWERLINK slave on FPGA (e.g., [16]), the SPOF system defines a general purpose interface for the interconnection with any device with a standard communication protocol (and, as an additional element, also special purpose interfaces may be added). With the proposed elements, SPOF allows upgrading old equipment, to interface with POWERLINK, to add new functionality at the edge and to implement smart sensing solutions. Thanks to the fact that we target FPGAs, the number of interfaces (e.g., UART ports, SPI slaves, I2C slaves) can be selected, with reference to the available FPGA area and pinout, according to system specifications (this would not be possible if general purpose SoC solutions were used).

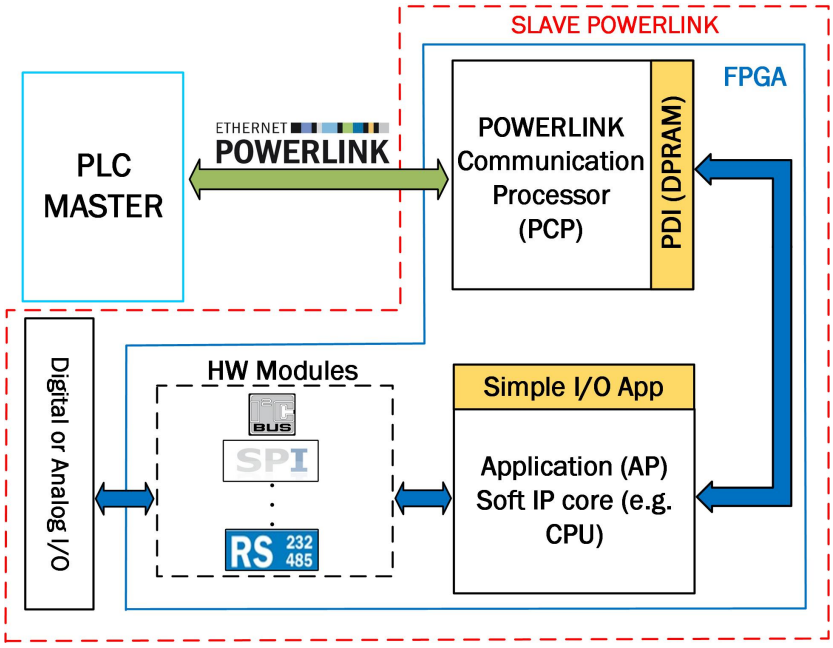

Figure 2. SPOF HW/SW Framework Architecture. POWERLINK slaves can be implemented both as stacks on the application processor or as a dedicated hardware communication block: this is schematically indicated with the POWERLINK Communication Processor (PCP).

3. The Design Methodology

We propose a framework that targets FPGA platforms. Our solution deals with the fact that from an industrial point of view, two characteristics are required: intrinsically flexible implementations

and preference for chips able to work on several designs, i.e., to enforce reusability. This lets manufacturers amortize development cost over a large number of units.

The qualities that, in our opinion, have to be satisfied by tools (such as SPOF) and platforms to be considered as a valid support for upgrading old equipment of industrial control systems are the following:

- **Support to integration**, i.e., to support on the integration of sensors and actuators with the equipment already present on-site, for maximum reuse purpose.
- **Support to edge computing**, i.e., to offer the possibility to implement smart functionality at the edge by means of new modules, in order to perform some kind of pre-processing, to enforce security and safety of the new control system and to act smarter for a specific process.
- **Support to monitoring**, i.e., to offer the possibility to implement smart sensing solutions, in order to observe specific triggers that drive reconfiguration and adaptation.

We designed SPOF with the aim of guaranteeing these fulfillments. SPOF is developed with reference to the design flow reported in Figure 3, a simplification of the V-model [28], where starting from requirements, three main steps are done to design a system that satisfies them (high-level and low-level design and implementation). Finally, a validation is done to check if the implemented system satisfies initial requirements. In particular, in the SPOF framework, the final system resulting from the design flow aims to satisfy the typical requirements that exist in an upgrade of equipment in industrial control systems, i.e., (a) to interface old platforms with a different system bus, (b) to interface to one or multiple industrial fieldbuses and (c) to perform some pre-processing on data, such as application of data fusion algorithms. This translates into two main classes of requirements managed by the SPOF framework:

- *Functional Requirements*, where users indicate the required functionality for the SPOF-slave. These requirements cover the (a) and (c) points above indicated, while the point (b) is related only to the interfacing with POWERLINK bus;
- *Monitorability Requirements*, where users indicate the required monitoring actions on the SPOF-slave. We provide these additional requirements to support on the implementation of smart sensing solutions.

For instance, a functional requirement can be the implementation of a POWERLINK slave able to interface with one UART and one I2C sensor. Together with this, the monitorability requirement can be the "monitoring of signatures that have to be transmitted, along the system bus, every T seconds". In the next section, it will be shown how SPOF allows designing a system that satisfies these kind of requirements.

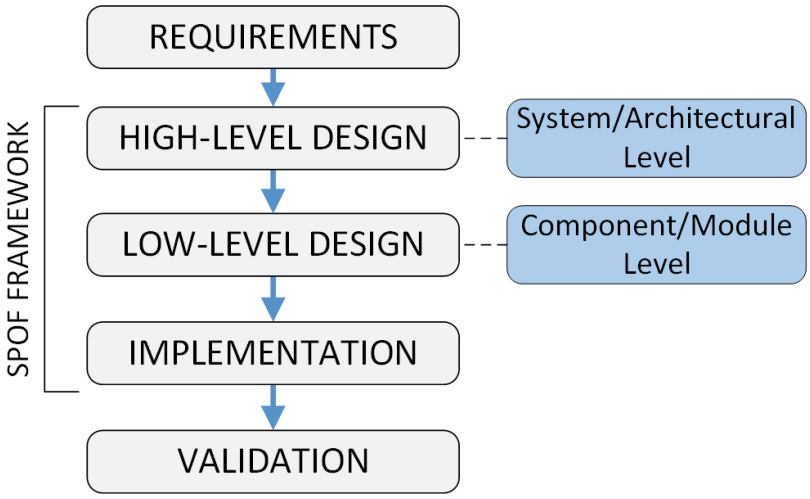

Figure 3. Reference Design Flow.

4. The Proposed Framework

In this section, the proposed framework is described in detail. The SPOF framework, in the first version, targets users with expert knowledge of FPGA, since its entry point is an FPGA implementation tool, such as Intel Quartus II. Users provide requirements in natural language and, thanks to their knowledge, they directly use the SPOF framework on Intel Quartus II Integrated Development Environment (IDE) to implement the SPOF-slave. User requirements are one-to-one linked with solutions provided by SPOF, so the only required knowledge is related to the use of IDE that, in general, is attributed to expert FPGA designers.

In the current SPOF implementation, we provide the framework with the capability to satisfy the following functional requirements:

- a configurable number of slave POWERLINK nodes;
- a configurable number of SPI-slave interfaces;
- a configurable number of I2C-slave interfaces;
- a configurable number of UART-slave interfaces

For the monitorability requirements, we provide the framework with the capability to satisfy the following ones:

- counting the number of user-defined events;
- measuring the time between two user-defined events

Considering the example reported in the previous section, users can satisfy functional requirements by dragging one UART-slave IP-core and one I2C-slave IP-core, both provided by SPOF framework, and they can satisfy the monitorability requirement by dragging one event monitor and one-time monitor.

The following subsections describe the different parts of the SPOF framework.

4.1. Main Core and System Bus

The SPOF-slave obtained by using the SPOF framework is built around a soft IP-core *Application Processor* (AP) that manages the interfacing with several POWERLINK slaves, able to support the data exchange with a POWERLINK *Communication Processor* (PCP, [29]), together with the interfacing with a number IP cores acting as controllers for the management of the serial external buses (UART, I2C, SPI). The selected AP is the Nios II, as shown in Figure 4.

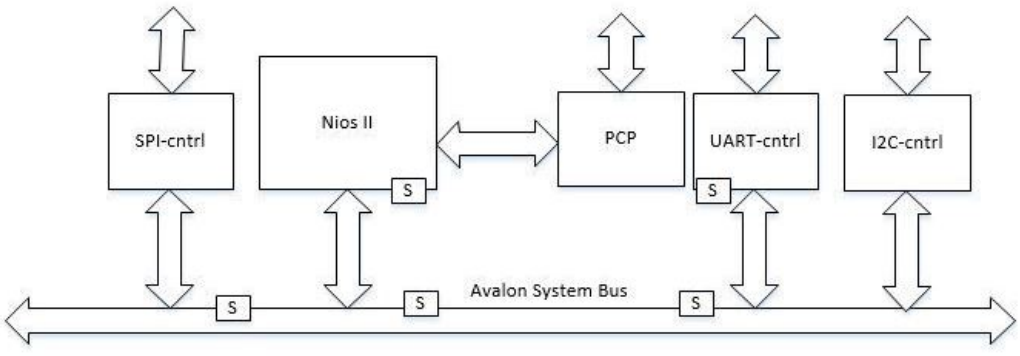

Figure 4. Hardware platform of a typical SPOF-slave: Nios II represents the application processor, PCP represents the POWERLINK communication processor, UART, SPI and I2C-cntrl represent controllers for interfacing with different buses, while S represents sniffers, part of the monitoring system.

Peripherals and AP communicate by means of an Intel *Avalon* bus [29]. The bus is a crucial component of the system since it adapts to satisfy the users' functional requirements. Avalon peripherals can be memories and processors, as well as traditional peripheral components, such as

a UART, timers and bus bridges, but even any custom Avalon peripheral. The Avalon bus module is automatically generated by Intel Quartus II tool, avoiding manually connection between bus and peripherals. It automatically generates the arbitration logic and the necessary resizing that allows easily supporting peripherals of different widths. In Figure 5 [29], a detailed view is shown, by means of a block diagram of an Avalon-based system, where the main core is connected to the left side, while slave interfaces are connected to the right.

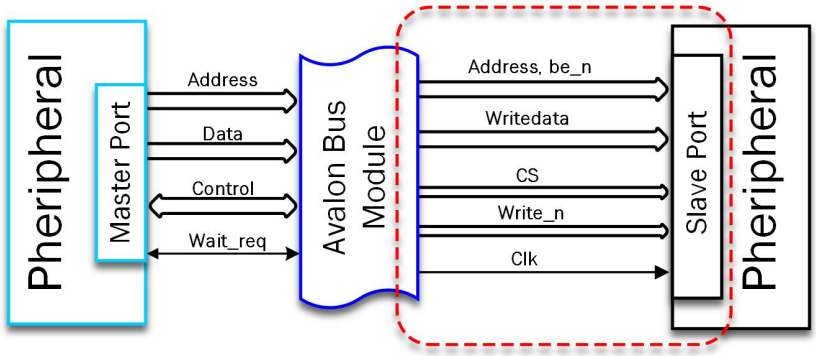

Figure 5. Avalon-based system.

4.2. Monitoring System

To satisfy monitorability requirements, a run-time monitoring system is provided by the SPOF framework. The monitoring system is composed of several sniffers, distributed into the final architecture, as shown in Figure 4. To avoid as much as possible introducing overhead in the application execution, the proposed monitoring system is completely hardware, i.e., it does not have an impact on application execution time. This is important for systems with real-time requirements. The complete monitoring system can be implemented using a library of elements, called AIPHS ([30,31]). In particular, AIPHS is composed of hardware monitoring units, called sniffers (S), connected to a Global Monitor unit (GM): GM interfaces with sniffers in order to initialize and collect their information. APIs to filter, to organize and to use monitoring information are also provided. The final monitoring system can also send results in a specified format by writing them on a different destination (either external communication device or memory), depending on what is required by users. In Figure 6, it is shown how the monitoring system is implemented. Starting from some monitorability requirements, expressed by means of metrics, sniffers are built in order to satisfy them. Sniffers are composed of three parts: Target Bus Adapter, GM Interface and Nucleus. The Nucleus acts on interconnection independent signals, performing the measurements required. On the other hand, the Target Bus Adapter takes interconnection dependent signals and brings them to Nucleus. Finally, GM Interface is a block that communicates with a global monitor. In the case of SPOF, the Global monitor is represented by the Nios II processor. The metrics can be of different types: in SPOF, accepted metrics are (i) memory exploitation metrics, (ii) HW communication metrics (iii) overall execution time metrics and (iv) code coverage metrics [30]. With these metrics, the sniffer generation is automatic. In addition, users can design their own sniffers, customizing the monitoring system. AIPHS works basing on a library concept: elements to compose sniffers are included in three libraries, namely LIB_ADAPT for Target Bus Adapter, LIB_GM for GM Interface and LIB_NUCLEUS for Nucleus. Event Monitor and Time Monitor cited in the previous sub-section are part of LIB_NUCLEUS.

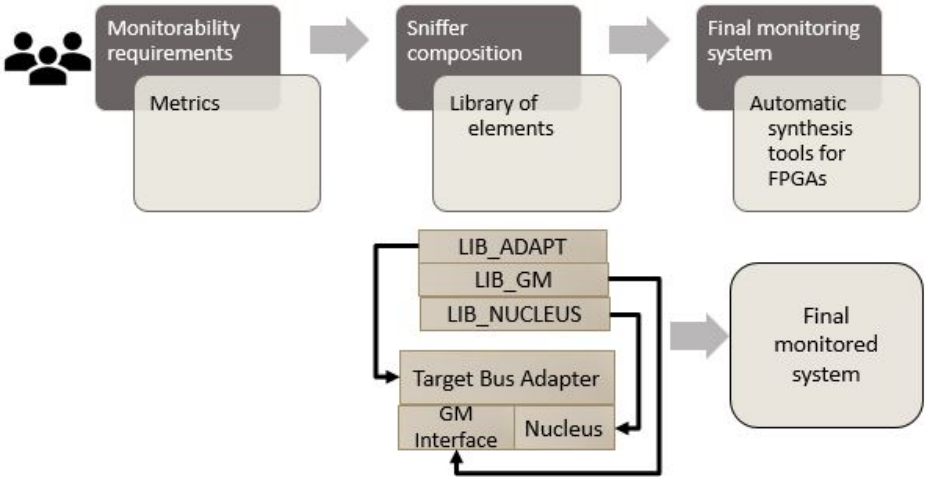

Figure 6. Monitoring system generation using AIPHS. Starting from some monitorability requirements, sniffers are composed taking elements from three libraries. Finally, the monitoring system is implemented as part of the whole system on FPGA.

5. Validation Activities

Validation activities for the SPOF framework have been considered with the goal of evaluating:

- the capability to satisfy the three characteristics that we identified for new tools and platforms for developing future industrial control systems, i.e., support to integration, support to monitoring and support to edge-computing (Section 3);
- the impact of SPOF in terms of logic occupied area;
- the impact on the development flow time to port the old equipment functions on FPGA.

5.1. Test Scenario

A PLC-POWERLINK master node has been connected to other POWERLINK slave nodes in an industrial control system that delivers electricity, as shown in Figure 7. It is worth noting that only a part of the control system has been considered for our test scenario, which is the part where the PLC is connected to three actuators using POWERLINK: this part represented the equipment already presented on-site.

An FPGA, the Intel Cyclone IV EP4CE115F29C7, has been introduced in the control system hardware platform, in order to evaluate implementations provided by the SPOF framework when interfacing to new modules. The Nios II configuration is the Nios II/e. For this test, new modules are represented by different sensors, connected using SPI, I2C and RS232 interfaces: they provide, respectively, ten points of temperature monitoring, twelve accelerators for seismic monitoring and one Bluetooth connection to exchange data with other remote components, as shown in Figure 7. Temperature sensors were already presented in the old equipment, while accelerators and Bluetooth are completely newly added. Sensors have to collect data to communicate them to master node through POWERLINK.

The PLC was provided by B&R and programmed using SIMULINK and B&R automation studio, while Intel FPGA was programmed using Intel Quartus II. In the B&R solution, the POWERLINK Slave can be used in three different configurations: (i) *POWERLINK slave with Direct I/O* that consists of a POWERLINK Communication Processor (PCP) that can be directly connected to up to 32 I/O lines, (ii) *POWERLINK slave with external host processor* that consists of an application processor on a dedicated chip and (iii) *POWERLINK slave with internal host processor*, where the application processor is a Soft IP-Core within the same FPGA as the PCP. SPOF adopts the third configuration, as previously shown in Figure 4.

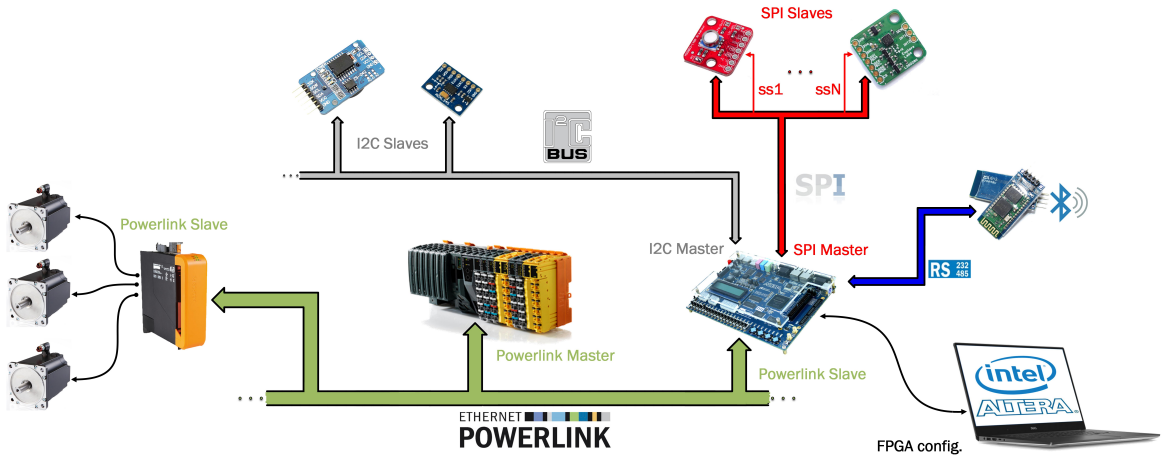

Figure 7. SPOF Use Case Scenario.

5.2. Support to Integration

In the first test, we validated the support to integration offered by SPOF. Starting from a control system with PLC acting as POWERLINK master node, with some actuators connected as slave nodes, we used SPOF to implement interfaces for the new sensors above indicated: we started with one temperature sensor, one accelerator and the Bluetooth sensor. Then, we added all other sensors just by reconfiguring different time the POWERLINK slave on FPGA, without the need to reprogram the PLC. In general, we highlighted the possibility to introduce new HW elements without re-programming the PLC side. The final HW platform contains one Nios II/e processor, one SPI controller, one UART controller and one I2C controller, together with a monitoring system, as explained in the next sub-section.

5.3. Support to Monitoring

In the second test, we validated the support for monitoring offered by SPOF. We implemented two types of sensing solutions directly on the FPGA by using the SPOF framework:

- *Fault Recovery Triggering*: in case of hardware fault of SPI controller, the system had to switch, in real-time, to a new SPI controller already implemented as redundancy on FPGA. The monitoring system has been implemented using a time monitor from the SPOF monitoring library, and it was able to guarantee a recovery action of 50 clock cycles by switching to the new SPI controller.
- *Run-time Verification of Custom Behaviours*: a monitoring system able to verify the correctness of working of slave POWERLINK node has been developed. It monitors the causality of the phases of the POWERLINK protocol and it stores traces in case of a fault. The goal is to allow an offline debug. The monitoring system has been implemented using an event monitor from the SPOF monitoring library, together with a Direct Memory Access (DMA) to flush out traces.

With this test, we highlighted the possibility to satisfy monitorability requirements.

5.4. Support to Edge Computing

In the third test, we evaluated the possibility to introduce a custom computation, close to sensors, to enhance the real-time response to critical conditions. This is possible only with hard real-time guarantees. We collected sensor data in a buffer, we synchronized the communication between PLC and AP/PCP device sending data with respect to priority of different data values (temperature, positioning data, humidity, pressure, etc.) and we locally managed part of the control that was implemented by PLC. With the third test, we highlighted the possibility to offload part of the work done by the PLC to edge computing platforms, thanks to the FPGA presence that allows the introduction of bounded time response circuits.

5.5. Area Occupation

The area occupation of the SPOF-slave implemented for our test-bed is reported in Table 1. It can be noticed that the occupied area is a tenth of the total FPGA area, but the number of pins used is almost half of available ones. In general, pins represent the limit to the number of sensors that can be interfaced. On the other hand, on the FPGA there is enough space to implement algorithms to do pre-processing on data, such as performing some of the PLC functions. This enforces the support for edge-computing provided by SPOF.

The occupied area of the monitoring system adds 2% to the Total Area, mainly due to the DMA circuit. It is an area overhead that can be admitted since it removes the execution time overhead on the monitored application.

Table 1. Area Occupation.

Logic Elements	SPOF Area	Total Area	Percentage	Monitor Area
Logic elements	12,099	114,480	11%	1%
Combinational functions	10,268	114,480	9%	2%
Dedicated logic registers	7746	114,480	7%	
Registers	7881	-	-	-
Pins	240	529	45%	-
Memory bits	265,952	3,981,312	7%	2%
Embedded Multiplier 9-bit elements	4	532	<1%	-
PLLs	1	4	25%	-

6. Conclusions

In this work, a framework, called SPOF, to support on upgrading of old industrial control systems equipment, in order to adopt them in new ones, has been presented. It targets PLC-POWERLINK industrial solutions and it is based on the possibility to separate the sensing and pre-processing functionality from the PLC-side, demanding the former to an FPGA implementation. The methodology that stands behind the framework, and the framework itself, have been described. Then, it has been shown how the framework provides support on (i) integration, (ii) monitoring and (iii) edge-computing and, in the final validation activity, it has been verified that the three points above-indicated are satisfied. Several improvements can be introduced in the current solution. On the software side, the usage of a properly operating system can improve the flexibility of the solution while keeping the functionality verified (NIOS II supported, for example, by *uCLinux* [32] and *FreeRTOS* [33]). On the hardware side, more sensor interfaces can be added, together with new IP-cores that support on execution of PLC applications on edge.

Author Contributions: Data curation, M.M. and A.L.; Formal analysis, V.M. and W.T.; Funding acquisition, L.P.; Investigation, G.B.; Methodology, V.M., W.T. and L.P.; Software, G.V., M.M. and A.L.; Validation, G.B.; Writing—review & editing, G.V.

Funding: This research has not been funded.

Acknowledgments: Authors would like to thank Dynacom srl and Smartcolor srl for supporting the project. Authors would like also to thank Vincenzo Sulli for the critical review process.

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

References

1. Hunt, R. *Design for a Circular Economy in Industry 4.0*; Charter, M., Ed.; Routledge: Abingdon, NY, USA, 2018.
2. Lasi, H.; Fettke, P.; Kemper, H.G.; Feld, T.; Hoffmann, M. Industry 4.0. *Bus. Inf. Syst. Eng.* **2014**, *6*, 239–242, doi:10.1007/s12599-014-0334-4. [CrossRef]

3. Nicola, A.D.; Mascio, T.D.; Lezoche, M.; Tagliano, F. Semantic Lifting of Business Process Models. In Proceedings of the 2008 12th Enterprise Distributed Object Computing Conference Workshops, Munich, Germany, 15–18 September 2008; pp. 120–126, doi:10.1109/EDOCW.2008.55. [CrossRef]
4. Kagermann, H. Change Through Digitization—Value Creation in the Age of Industry 4.0. In *Management of Permanent Change*; Springer Fachmedien Wiesbaden: Wiesbaden, Germany, 2015; pp. 23–45, doi:10.1007/978-3-658-05014-6_2.
5. Wang, D.; Guan, X.; Liu, T.; Gu, Y.; Shen, C.; Xu, Z. Extended Distributed State Estimation: A Detection Method against Tolerable False Data Injection Attacks in Smart Grids. *Energies* **2014**, *7*, 1517–1538, doi:10.3390/en7031517. [CrossRef]
6. Smarra, F.; Jain, A.; de Rubeis, T.; Ambrosini, D.; D'Innocenzo, A.; Mangharam, R. Data-driven model predictive control using random forests for building energy optimization and climate control. *Appl. Energy* **2018**, *226*, 1252–1272, doi:10.1016/j.apenergy.2018.02.126. [CrossRef]
7. Lun, Y.Z.; D'Innocenzo, A.; Smarra, F.; Malavolta, I.; Benedetto, M.D.D. State of the art of cyber-physical systems security: An automatic control perspective. *J. Syst. Softw.* **2019**, *149*, 174–216, doi:10.1016/j.jss.2018.12.006. [CrossRef]
8. Depari, A.; Sisinni, E.; Flammini, A.; Ferri, G.; Stornelli, V.; Barile, G.; Parente, F.R. Autobalancing Analog Front End for Full-Range Differential Capacitive Sensing. *IEEE Trans. Instrum. Meas.* **2018**, *67*, 885–893. [CrossRef]
9. Barile, G.; Ferri, G.; Parente, F.R.; Stornelli, V.; Sisinni, E.; Depari, A.; Flammini, A. A CMOS full-range linear integrated interface for differential capacitive sensor readout. *Sens. Actuators A Phys.* **2018**, *281*, 130–140. [CrossRef]
10. Ferri, G.; Stornelli, V.; Parente, F.R.; Barile, G. Full range analog wheatstone bridge-based automatic circuit for differential capacitance sensor evaluation. *Int. J. Circuit Theory Appl.* **2017**, *45*, 2149–2156. [CrossRef]
11. Sauter, T. The Three Generations of Field-Level Networks—Evolution and Compatibility Issues. *IEEE Trans. Ind. Electron.* **2010**, *57*, 3585–3595, doi:10.1109/TIE.2010.2062473. [CrossRef]
12. Hashemi Farzaneh, M.; Knoll, A.; Pfeiffer, J. OPART: Towards an Open Platform for Abstraction of real-time communication in cross-domain applications. In Proceedings of the 2015 6th International Conference on Automation, Robotics and Applications (ICARA), Queenstown, New Zealand, 17–19 February 2015; pp. 246–250, doi:10.1109/ICARA.2015.7081154. [CrossRef]
13. Erwinski, K.; Paprocki, M.; Grzesiak, L.M.; Karwowski, K.; Wawrzak, A. Application of Ethernet Powerlink for Communication in a Linux RTAI Open CNC system. *IEEE Trans. Ind. Electron.* **2013**, *60*, 628–636, doi:10.1109/TIE.2012.2206348. [CrossRef]
14. Fei, H.; Yixin, Z.; Wei, H. Design and research of dynamic partial reconfiguration for industrial Ethernet. In Proceedings of the 2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), Xi'an, China, 3–5 October 2016; pp. 1146–1152, doi:10.1109/IMCEC.2016.7867391. [CrossRef]
15. Zhang, K.; Hu, L.; Zuo, P.; Wu, X.; Miao, K. Wireless extension mechanism and logic design for FPGA-based Ethernet Powerlink node. In Proceedings of the 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), Okayama, Japan, 26–29 June 2016; pp. 1–7, doi:10.1109/ICIS.2016.7550815. [CrossRef]
16. Min, L.; Shengli, L.; Rongrong, W. Realization of LED Urban Lighting Network Based on POWERLINK Industrial Ethernet. In Proceedings of the 2014 7th International Conference on Intelligent Computation Technology and Automation, Changsha, China, 25–26 October 2014; pp. 445–448, doi:10.1109/ICICTA.2014.113. [CrossRef]
17. Wlas, M.; Gackowski, M.; Kolbusz, W. The Ethernet POWERLINK Protocol for smart grids elements integration. In Proceedings of the 2011 IEEE International Symposium on Industrial Electronics, Gdansk, Poland, 27–30 June 2011; pp. 2070–2075, doi:10.1109/ISIE.2011.5984479. [CrossRef]
18. ACOPOS. 2018. Available online: https://www.br-automation.com/en-gb/products/safety-technology/acopos/ (accessed on 16 March 2019).
19. Gabriele, T.; Pantoli, L.; Stornelli, V.; Chiulli, D.; Muttillo, M. Smart power management system for home appliances and wellness based on wireless sensors network and mobile technology. In Proceedings of the 2015 18th AISEM Annual Conference, AISEM 2015, Trento, Italy, 3–5 February 2015.

20. de Rubeis, T.; Muttillo, M.; Pantoli, L.; Nardi, I.; Leone, I.; Stornelli, V.; Ambrosini, D. A first approach to universal daylight and occupancy control system for any lamps: Simulated case in an academic classroom. *Energy Build.* **2017**, *152*, 24–39. [CrossRef]
21. Pantoli, L.; Muttillo, M.; Ferri, G.; Stornelli, V.; Alaggio, R.; Vettori, D.; Chinzari, L.; Chinzari, F. *Electronic System for Structural and Environmental Building Monitoring*; Lecture Notes in Electrical Engineering; Springer: Berlin/Heidelberg, Germany, 2019; Volume 539, pp. 481–488.
22. Barile, G.; Leoni, A.; Pantoli, L.; Stornelli, V. Real-Time Autonomous System for Structural and Environmental Monitoring of Dynamic Events. *Electronics* **2018**, *7*, 420, doi:10.3390/electronics7120420. [CrossRef]
23. Pang, H.; Li, J.; Ruan, Y.; Huang, Y.; Shi, J.; Qin, S. Formalization and Verification of the Powerlink Protocol Using CSP. In Proceedings of the 2016 23rd Asia-Pacific Software Engineering Conference (APSEC), Hamilton, New Zealand, 6–9 December 2016; pp. 321–328, doi:10.1109/APSEC.2016.052. [CrossRef]
24. Knezic, M.; Dokic, B.; Ivanovic, Z. Theoretical and Experimental Evaluation of Ethernet Powerlink PollResponse Chaining Mechanism. *IEEE Trans. Ind. Inform.* **2017**, *13*, 923–933, doi:10.1109/TII.2016.2634554. [CrossRef]
25. Vaida, C.; Pisla, D.; Covaciu, F.; Gherman, B.; Pisla, A.; Plitea, N. Development of a control system for a HEXA parallel robot. In Proceedings of the 2016 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR), Cluj-Napoca, Romania, 19–21 May 2016; pp. 1–6, doi:10.1109/AQTR.2016.7501318. [CrossRef]
26. Maestro, J.A.; Reviriego, P. Energy Efficiency in Industrial Ethernet: The Case of Powerlink. *IEEE Trans. Ind. Electron.* **2010**, *57*, 2896–2903, doi:10.1109/TIE.2009.2036640. [CrossRef]
27. Li, Y.C.; Hong, S.H.; Huang, X.; Chen, G.; Liang, X. Implementation of a Powerlink-WirelessHART gateway for industrial automation. In Proceedings of the 2016 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Chiang Mai, Thailand, 28 June–1 July 2016; pp. 1–6, doi:10.1109/ECTICon.2016.7560909. [CrossRef]
28. US Dept of Transportation, Federal Highway Administration. Systems Engineering Guidebook for ITS. 2018. Available online: https://www.fhwa.dot.gov/cadiv/segb/index.cfm (accessed on 16 March 2019).
29. Intel Altera Website. Available online: https://www.intel.com/content/www/us/en/products/programmable.html (accessed on 27 April 2019).
30. Valente, G.; Muttillo, V.; Pomante, L.; Federici, F.; Faccio, M.; Moro, A.; Ferri, S.; Tieri, C. A Flexible Profiling Sub-System for Reconfigurable Logic Architectures. In Proceedings of the 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP), Heraklion Crete, Greece, 17–19 February 2016; pp. 373–376, doi:10.1109/PDP.2016.86. [CrossRef]
31. Moro, A.; Federici, F.; Valente, G.; Pomante, L.; Faccio, M.; Muttillo, V. Hardware performance sniffers for embedded systems profiling. In Proceedings of the 12th International Workshop on Intelligent Solutions in Embedded Systems (WISES), Ancona, Italy, 29–30 October 2015, pp. 29–34.
32. uCLinux Homepage. 2018. Available online: http://www.uclinux.org/ (accessed on 16 March 2019).
33. FreeRTOS Homepage. 2018. Available online: http://www.freertos.org/ (accessed on 16 March 2019).