Production Engineering (2020) 14:365–371
https://doi.org/10.1007/s11740-020-00964-x

PRODUCTION PROCESS

Data acquisition and control at the edge: a hardware/software-reconfigurable approach

F. Streit[1] · S. Wituschek[2] · M. Pschyklenk[3] · A. Becher[1] · M. Lechner[2] · S. Wildermann[1] · I. Pitz[3] · M. Merklein[2] · J. Teich[1]

Received: 1 April 2020 / Accepted: 14 April 2020 / Published online: 3 June 2020

Abstract
Today's manufacturing facilities and processes offer the potential to collect data on an unprecedented scale. However, conventional Programmable Logic Controllers are often proprietary systems with closed-source hardware and software and not designed to also take over the seamless acquisition and processing of enormous amounts of data. Furthermore, their major focus on simple control tasks and a rigid number of static built-in I/O connectors make them not well suited for the big data challenge and an industrial environment that is changing at a high pace. This paper, advocates emerging hardware- and I/O reconfigurable Programmable System-on-Chip (PSoC) solutions based on Field-Programmable Gate Arrays to provide flexible and adaptable capabilities for both data acquisition and control right at the edge. Still, the design and implementation of applications on such heterogeneous PSoC platforms demands a comprehensive expertise in hardware/software co-design. To bridge this gap, a model-based design automation approach is presented to generate automatically optimized HW/SW configurations for a given PSoC. As a case study, a metal forming process is considered and the design automation of an industrial closed-loop control algorithm with the design objectives performance and resource costs is investigated to show the benefits of the approach.

Keywords FPGA · PSoC · Hardware/software co-design · Model-based engineering · Industrial automation · Manufacturing technology · Forming process · Motor control

1 Introduction

The usage of more and more sensors in production systems offers opportunities for novel process optimizations, e.g. through the use of machine learning [1]. But this comes at the cost to process large amounts of data already at its source, which poses major challenges for well-known industrial automation principles such as the often applied *automation pyramid* [2]. With the advent of cloud computing, the complexity of automation systems can be greatly reduced when processing the Manufacturing Execution System (MES) and Enterprise Resource Planning (ERP) layers in the cloud and unifying also the lower levels [field, control, and Supervisory Control And Data Acquisition (SCADA)]. However, traditional automation systems completely differentiate the levels of field (sensors/actuators), control [Programmable Logic Controllers (PLCs)], and SCADA by coupling complex proprietary subsystems for each level. Nevertheless, as the volume of data will continue to grow exponentially, we see these levels vanishing and, more important, the requirement to process the majority of data in real-time and thus close to production machines, the so-called *Edge*.

A major challenge of suitable platforms that might cover the tasks of all three levels arises from the growing number and variety of sensors and related data to process in real-time, which can vary dramatically from application to application. Although emerging smart sensor solutions can compress, filter, or convert signals into desired formats in real-time, they are, however, closed systems with often proprietary and inflexible interfaces. Indeed, incompatible interfaces can cause a lot of additional costs, since they have

✉ F. Streit
franz-josef.streit@fau.de

[1] Department of Computer Science 12, Friedrich-Alexander University Erlangen-Nürnberg (FAU), Erlangen, Germany

[2] Institute of Manufacturing Technology, Friedrich-Alexander University Erlangen-Nürnberg (FAU), Erlangen, Germany

[3] Schaeffler Technologies AG & Co. KG, Herzogenaurach, Germany

to be either purchased or developed separately under strict licensing conditions.

As a remedy, this paper proposes and advocates the use of Field-Programmable Gate Array (FPGA)-based platforms to achieve the required flexibility and scalability by exploiting the advantage of hardware reconfiguration and acceleration. From the analysis of the *similarity* of many data acquisition, monitoring, and control tasks of PLC systems available today, it can be concluded that a high flexibility is needed to be able to sense not only different types of digital and analog signals, but also different numbers of inputs and outputs using the same platform (*I/O reconfigurability*). Moreover, in order to be able to cover a multitude of different communication protocols at the hardware interfaces, the *hardware reconfigurability* of FPGAs is exploited. In hardware, time-critical signals can be directly sampled, filtered, fused, or pre-processed, e.g. converted from the time into the frequency domain, in parallel with other processing tasks without causing any additional Central Processing Unit (CPU) load. In order to combine the advantages of PLCs with the high performance offered by an FPGA, hybrid systems combining reconfigurable logic together with microprocessors, memory blocks, and peripherals on a single chip exist.

As a first contribution, this paper presents such a PSoC platform, containing in addition to an FPGA two CPUs that can be used to process even higher level tasks in software such as process visualization, which plays a decisive role in the evaluation of the suitability for use in industrial environments. As will be demonstrated, the PSoC guarantees reliable in-situ signal processing even for applications with changing data loads in time-critical open- as well as closed-loop control applications.

As a second contribution, a design methodology to partition and automatically generate hardware/software configurations from a given control application to a given PSoC platform is presented. The model-based design approach starts with the description of a behavioral model defined in MATLAB/Simulink. Based on this description, the framework optimizes and generates from a block diagram code for the target PSoC architecture. As a result, synthesizable function blocks of control tasks can be implemented in hardware on the FPGA, while high-level tasks are compiled and deployed in software on the processor system.

Third, a real-world case study on position control of a metal forming process is presented, demonstrating the advantages of the PSoC architecture and automatic design flow. We conclude that hardware/software reconfigurable PSoC architectures may provide the required computational power and the needed I/O reconfigurability at rather low cost for the processing of huge amounts of sensor data right at the edge, thus avoiding the congestion of networks to servers and clouds.

The remaining of the paper is organized as follows: Sect. 2 proposes the concept of PSoC-based PLCs, while Sect. 3 details the tool flow for automatic hardware/software generation. A demonstrator to analyze the requirements of forming processes is presented in Sect. 4. Finally, experimental results of the control of such a metal forming process is demonstrated in Sect. 5, followed by a conclusion and outlook on future directions in Sect. 6.

2 PLCs based on hardware and software reconfigurable PSoCs

In recent years, various platforms have been proposed and evaluated for future automation tasks [3]. Starting with software-programmable General-Purpose Processors (GPPs) [4] over hardware-reconfigurable FPGAs [5] to optimized physical hardware in the form of Application-Specific Integrated Circuits (ASICs) [6]. Traditional PLCs used for industrial control cannot cope with additional tasks such as data acquisition and processing of several digital and analog inputs and sensors in real-time for which typically highly specialized Digital Signal Processor (DSP) and analysis units are used.

However, flexibility is of great importance for modern production plants as operating times of up to twenty years are often standard. Moreover, each individual production machine imposes different requirements not only on the number but also on the types of in- and outputs. Thus, the reconfigurability of also the I/O hardware interfaces and support of common field-bus protocols is a must. An ideal system must therefore—apart from low cost requirements—be capable to offer this high flexibility in combination with sufficient computing power to enable the simultaneous processing of tasks at the field, control, and SCADA level close to the edge. Here, FPGA-based PSoCs and thus hardware reconfigurable platforms seem to be a promising solution. Such architectures could make a new era of automation systems possible, since they provide the required computing power to allow, for instance, even 2D image sensor [7] or acoustic signal processing [8] in real-time right at the data source.

Figure 1 outlines the concept of a PSoC-based system, where a dual-core processor can be used to support high-level tasks such as process visualization, while the tightly coupled reconfigurable hardware area of the FPGA can be programmed to provide signal sampling, fusion, data conversion, and other computationally-intensive tasks. Based on pre-fabricated reconfigurable cells called Configurable Logic Blocks (CLBs) and reconfigurable connections within the FPGA, digital hardware modules may be designed for different acquisition functions and stored in a module library, similar to a library of software functions. Different to software, though, each hardware module computes a function

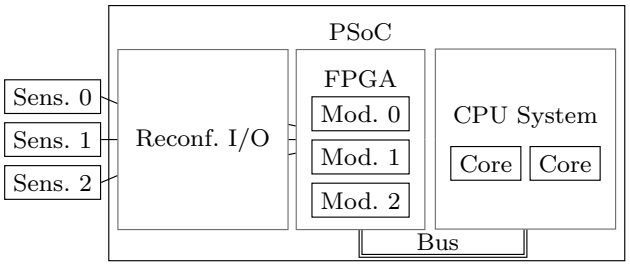

Fig. 1 Schematic illustration of a PSoC with reconfigurable I/Os, programmable hardware FPGA, and programmable multi-core processor CPU system. Multiple sensor inputs can be processed in parallel by instantiating dedicated processing modules on the FPGA

not only internally in parallel, but also the execution of the modules occurs in parallel. For example, in Fig. 1, the three modules Mod. 0, Mod. 1, and Mod. 2 could be instantiated to simultaneously process the three attached sensors Sens. 0, Sens. 1, and Sens. 2. Here, the number as well as the computational complexity of the modules determines the amount of CLBs required to map these functionality to hardware, which is usually specified as resource requirements in the form of Flip-Flops (FFs), Lookup Tables (LUTs), and DSP slices.

Also the types and number of signals connected to an FPGA may be reconfigured for individual needs: I/O pins of the FPGA can be reprogrammed equally as inputs, outputs or even bidirectionally. This I/O reconfigurability makes it possible to implement on-demand a wide range of common industrial bus interfaces such as for instance Controller Area Network (CAN) [9], Modbus [10], and EtherCAT [11] on the same pins. In this way, application-specific interface modules and sensors can be connected or exchanged as required. Therefore it can be argued that PSoCs based on FPGAs can meet the complex requirements of upcoming industrial automation systems in providing an optimal ground for edge-processing of as well field-bus, control, and data acquisition tasks on the same low-cost platform.

3 MoDesA: model-based design automation for PSoCs

Trends in industrial applications such as condition monitoring [12] and predictive maintenance [13] have the potential to be a turning point in the world of industrial services. Nevertheless, the design of such applications to run on a given PSoC platform must be fully automated in order to be accepted. Moreover, it is necessary to provide design tools that allow developers to specify applications for PSoCs at a very high level of abstraction. In the design flow for PSoCs called Model-Based Design Automation (MoDesA) and published in [14] and [15], the user starts from a behavioral model defined in MATLAB/Simulink. Here, the system is modeled by a set of blocks connected to exchange signals using an interactive graphical environment. Extensive analysis and simulation capabilities then enable a first validation of the system functionality. The goal of design automation is partitioning the blocks into hardware and software, and then to generate and load synthesized hardware modules as well as executable software directly to the platform.

To exemplify, Fig. 2 illustrates the steps required to partition such a system model in Simulink to generate hybrid hardware/software implementations. In MoDesA, the designer only has to tag each of the functional Simulink blocks as either hardware or software. Adjacent blocks with

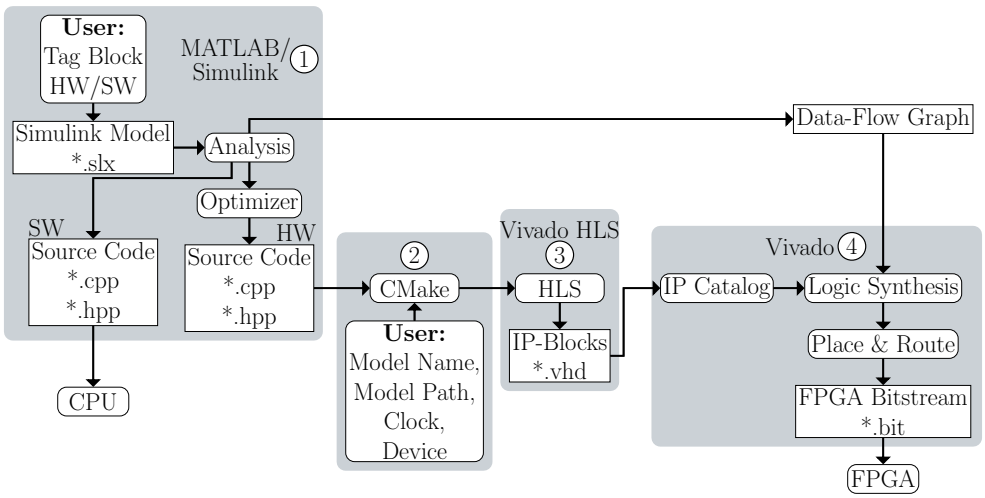

Fig. 2 MoDesA: model-based design flow for industrial automation systems: starting from a MATLAB/Simulink model to generate a hardware/software co-designed implementation for FPGA-based PSoCs [14]

identical tagging form *islands* representing hardware or software domains, whereas edges crossing these domains indicate the boundaries of these islands. In short, with just one binary tag ($\{hw, sw\}$), a system engineer can decide which blocks should be implemented in hardware and which in software.

To link two different domains, adapters are required to enable an inter-domain communication. For being able to automatically generate the communication interfaces between the blocks within one island, a data-flow graph describing the data dependencies between these blocks is created. Data source blocks, which are often used as virtual sensor inputs for accurate testing and simulation, as well as data sink blocks for debugging and process visualization are not considered for the final code generation. This offers the possibility to pass complete test environments to the MoDesA code generator and makes a seamless integration with physical data sources possible.

The whole tool flow in Fig. 2 is described next. In Step 1, the individual Simulink blocks of the application tagged as either *hw* or *sw* are analyzed before the MATLAB Target Language Compiler (TLC) is used for custom code generation. Here, software e.g. domain-specific C/C++ files for each island of connected *sw*-tagged blocks is generated. These source files are compiled to the CPUs of the PSoC platform for sequential execution. For those islands to be implemented in hardware on the FPGA, High-Level Synthese (HLS)-conform C/C++ code is generated and then passed to the HLS step. Different to the software, each block within a hardware island is interconnected by streaming interfaces to enable their pipelined execution.

The platform-independent programming tool CMake[1] is applied in Step 2 to guide the HLS- and FPGA vendor synthesis tools Vivado. From there on, CMake creates all the files needed to apply the automatic hardware synthesis for the desired FPGA platform.

The hardware design Steps 3–4 in Fig. 2 are described in the following. First, MoDesA transforms the hardware-optimized C/C++ code into an Intellectual Property Block (IPB) implementation consisting of several hardware description files via HLS. Furthermore, Vivado HLS immediately provides initial estimates of essential design objectives such as expected resource requirements, latency and the maximum achievable clock rate of the created system. Thus, the impact in terms of performance or FPGA utilization of a possible hardware implementation can be quickly assessed when the model needs to be adapted.

In the last step, MoDesA applies the Xilinx IPB integrator to load the generated hardware blocks as hardware description files from the previously defined IPB catalog into the Xilinx synthesis tool Vivado. The initial data-flow graph is used here to connect the instantiated hardware blocks from the IPB catalog according to their data dependencies to create a netlist of the instantiated modules. Then, logic synthesis including place and route of the hardware design is performed, and finally a bitstream generated to be loaded into the FPGA.

4 Analysis of I/O and data acquisition requirements of forming processes

Forming technology is characterized by an excellent process and resource efficiency and is ideal for the production of large quantities that require high and consistent quality. However, future forming techniques demand new methods and technologies to enable the development towards visions such as zero-failure production and complete process transparency. For example, a comprehensive data acquisition and processing would make it possible to detect and react to anomalies that occur during forming process even before damage occurs [16]. In this way, in-situ control through constant calibration of machine parameters would on the one hand significantly reduce the rejection of components and on the other hand optimize process efficiency [17]. Unfortunately, proprietary and inflexible PLC solutions make it difficult to process the necessary data directly at the machine.

In order to investigate the requirements for both data acquisition and process control of real-world forming processes, a demonstrator was constructed at the Institute of Manufacturing Technology. The demonstrator is a metal forming press equipped with a software and hardware reprogrammable PSoC, which allows to connect and process several sensors and actuators. This makes it for the first time possible to investigate those parameters of a forming process which are responsible for component defects or which enable an improvement of the process quality.

The essential parts together with the mechanical construction of the demonstrator are illustrated in Fig. 3. As can be seen, the basic construction of the demonstrator is based on the design of conventional presses with the main components consisting of drive, frame and guide. The frame is constructed as an O-frame in a 1-stand design and forwards the forces arising during the forming process. Linear guides are mounted on the load frame to transmit the relative movements of the press. However, the heart of the system is a servo press, which is characterized by precise and reproducible kinematics through an electrical drive. This electrical drive is based on an electronically commutated three-phase Brushless DC (BLDC) motor. To determine speed and position of the rotor, the motor is equipped with Hall sensors. The precise movement of the plunger towards the press bed is computed by the PSoC based on the Hall sensor outputs. In addition to the Hall sensors, two limit switches are installed on the press

[1] https://cmake.org/.

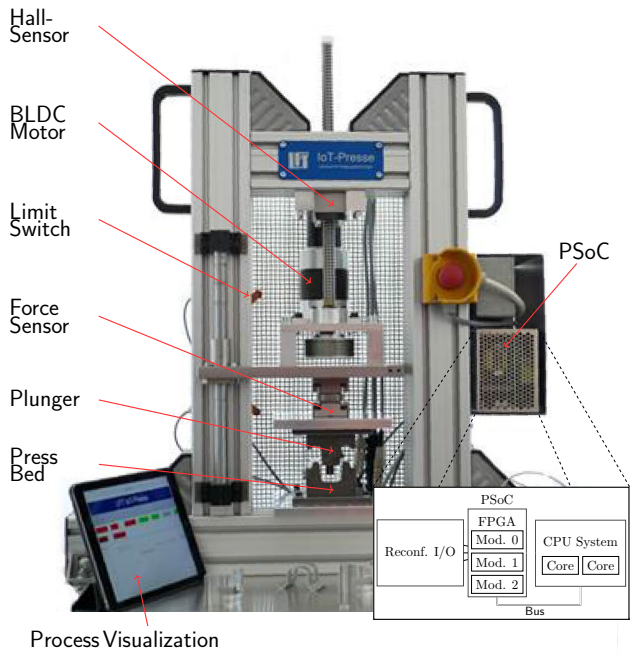

Fig. 3 Metal sheet forming press demonstrator constructed by the Institute of Manufacturing Engineering Erlangen. The press is equipped among Hall and force sensors with a PSoC as edge processing platform

to enable the calibration of two absolute plunger positions. Based on the digital outputs of the limit switches, the PSoC can calculate the range between the upper and lower limits for plunger travel. Obviously, the limit switches fulfil also an additional safety function. If the plunger exceeds the limit switch, the triggered signals must be immediately and directly evaluated. In the following implementation, this time-critical function is implemented in the programmable hardware of the PSoC. This guarantees a short reaction time of only a few clock cycles and thus prevents possible damage. As an additional sensor input, a force transducer is installed in the force flow of the plunger to continuously measure precise forces of the forming process in the z- and x- and y-direction. Here, the force transducer makes use of the piezoelectric principle to convert the force in every Cartesian direction into an anlog voltage signal. This analog signal must be sampled and preprocessed on the PSoC device before being combined with the position data to provide instant real-time information about the force-displacement ratio. Furthermore, the plate below the force sensor allows various tools to be mounted on the plunger. In the same way, the press bed allows the mounting of a wide variety of forming tools.

It could be shown that the PSoC can implement in parallel the acquisition of input data, as well as perform closed-loop control for not only the output voltage of the motor and thus its speed, but also the forming process as a whole. Finally, apart from real-time signal acquisition and control, the PSoC is able to additionally deliver the SCADA-type functionality of a web server for immediate process visualization, where users can connect with their mobile end device to display the recorded process data directly.

5 Case study

To demonstrate the capabilities of an FPGA-based PSoC solution for machine control, the design flow MoDesA introduced in Sect. 3 is applied to the motor control application of the sheet metal press presented in Sect. 4.

In the first step, a behavioral model of the system is created in MATLAB/Simulink. In this environment, the control task of the sheet metal forming including physical parameters like motor inductance and inertia can then be simulated. Figure 4 illustrates the signals that need to be exchanged between the motor drive as part of the press (left) including analog and

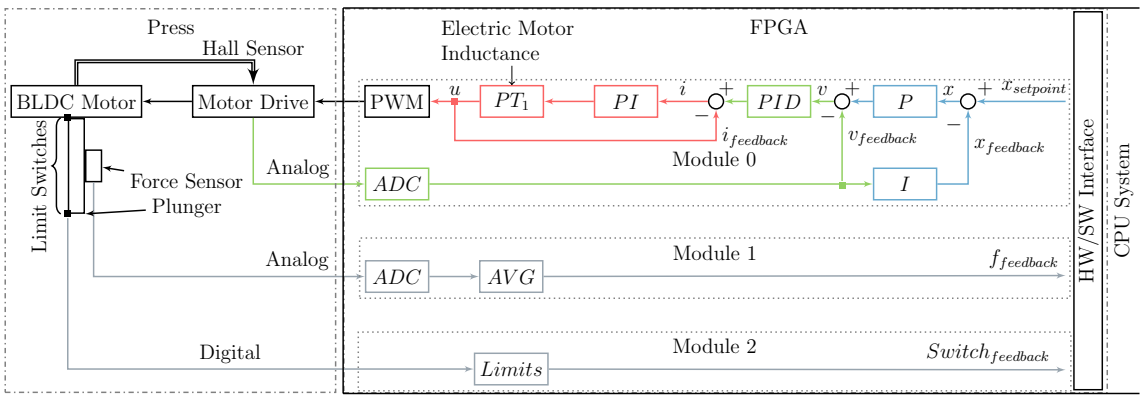

Fig. 4 Partitioning of the position control function for the BLDC motor of the metal forming press with 3 hardware modules identified for position control (Module 0), force measurements (Module 1), and crash control (Module 2)

Table 1 Resource requirements and latency per algorithmic block of the sheet metal forming application generated by our methodology on a Xilinx Zynq PSoC

Module	Block	Num.	FFs	LUTs	DSPs	Latency
0	P controller	abs.	103	59	0	2
		(%)	0.1	0.1	0.0	–
	I controller	abs.	106	48	0	3
		(%)	0.1	0.1	0.0	–
	PI controller	abs.	145	93	0	3
		(%)	0.2	0.2	0.0	–
	PID controller	abs.	189	125	0	3
		(%)	0.2	0.2	0.0	–
	PWM	abs.	27	75	1	–
		(%)	0.1	0.2	0.5	–
1	AVG	abs.	108	78	0	4
		(%)	0.1	0.2	0.0	–
2	Limits	abs.	69	39	0	2
		(%)	0.1	0.1	0.0	–
	HW/SW Interface	abs.	5871	4826	0	–
		(%)	5.5	9.1	0.0	–
Total		abs.	6618	5343	1	11
		(%)	6.2	10.0	0.5	–

digital input and output signals and the PSoC (right) that shall perform three different data acquisition and control tasks. In the following, the functionality of the application is described. When a user initiates a forming process, a desired position for the plunger is applied as setpoint $x_{setpoint}$ to a control algorithm, represented as Module 0 in Fig. 4. Shown is the block diagram of a cascaded controller for the calculation of motor current i (red) and speed v (green) as well as plunger position x (blue). The innermost current loop is used to control the torque of the BLDC motor, which influences the motor rotational speed and thus the end position of the plunger. In order to minimize the control error, the motor feedback $v_{feedback}$ is constantly measured using Hall sensors. This analog signal is sampled by an Analog-to-Digital Converter (ADC) before being fed back into the control loop, which is build upon a combination of proportional (P), integral (I), proportional-integral (PI), and proportional-integral-derivative (PID) controllers, while the output voltage of the motor is controlled by a calculated Pulse Width Modulation (PWM) signal. In Module 1, the PSoC simultaneously acquires and calculates the analog force feedback $f_{feedback}$ via a moving average filter (AVG) in parallel to the motor feedback to enable transparent process recordings. The digital inputs of the limit switches are monitored in Module 2 to calculate absolute position values and to prevent the plunger from crashing.

In the second step, the design flow MoDesA is applied in order to generate a suitable hardware/software implementation. Therefore, time-critical components of the presented controller are tagged as hardware blocks for synthesis into hardware modules residing on the FPGA, because of the need for a fast response-time for precise positioning of the metal forming process. Tasks being not time-critical, such as SCADA-typical tasks of handling user inputs from the web interface and data visualization, are mapped to the CPU system for being processed in software (not shown in Fig. 4). Both subsystems are connected by an automatically generated hardware/software interface. For the following evaluation of the generated hybrid hardware/software implementation, a Digilent Zedboard [18] was used as the target PSoC. This low-cost platform carries a Xilinx Zynq xc7z020clg484-1 FPGA device. During the synthesis steps, performance and resource costs were taken into account.

In the following, the synthesis results are discussed. As clock speed, a cycle time of 5 ns was achieved. The amount of FPGA resources required for the individual building blocks of the controller implementation in terms of FFs, LUTs, DSPs is shown in Table 1. Discussion: the numbers indicate that the controller occupies only 10% of the available LUTs and 6% of the available FFs, and thus only a small fraction of the FPGA. Moreover, with a total latency of only 11 clock cycles ($\sim$ 55 ns) for calculating an output value of the motor position, a precise real-time control can be guaranteed.

6 Conclusion and outlook

This paper, advocated the potential and benefits of hardware and software programmable PSoC solutions for combining the processing of tasks on the field, control, and SCADA level right at the edge for forming technology. The diversity encountered in data acquisition and control in today's industrial automation systems cries for more computational

power and more flexibility in types and number of signals to be sampled and processed, some even in real-time. PSoCs can offer viable and affordable solutions by the implementation of data-intensive and time-critical functions directly in FPGA hardware and other high-level functions on CPUs in software. Moreover, in order to support the design automation of PSoCs, MoDesA [14] is presented, a model-based design flow for industrial automation systems that enables the generation of hybrid hardware/software implementations. For showing the capabilities of PSoCs and the related design flow, a real-world forming press demonstrator has been constructed. In a case study, the motor control of a forming process were evaluated in terms of resource utilization and performance. The results show that several sensors can be acquired and processed in parallel, enabling not only real-time control of even advanced control algorithms, but also other signal processing as well as data visualization tasks via a web-based human–machine interface on this low-cost platform. Moreover, it was shown that the automatically generated hardware/software implementation requires only a small fraction of the available FPGA resources.

In future work, it is aimed to study more applications using the proposed approach, which should make it possible to provide better process optimization and minimize component rejections. An example is the compensation of springback directly in the deep drawing process under the influence of varying input parameters in the sheet metal. Here, a direct analysis of force and displacement data at the press can be used to react to fluctuating sheet metal input parameters and compensate the springback. To support fellow researchers, the design flow has been made publicly available at [19].

Acknowledgements Open Access funding provided by Projekt DEAL. The work has been supported in part by the Schaeffler Hub for Advanced Research at Friedrich-Alexander University Erlangen-Nürnberg (SHARE at FAU) and the Emerging Fields Initiative (EFI) at Friedrich-Alexander University Erlangen-Nürnberg.

References

1. Monostori L, Markus A, Van Brussel H (1996) Machine learning approaches to manufacturing. CIRP Ann 45:675–712
2. Sauter T (2007) The continuing evolution of integration in manufacturing automation. In: IEEE industrial electronics magazine
3. Malinowski A, Yu H (2011) Comparison of embedded system design for industrial applications. In: IEEE transactions on industrial informatics
4. Mahmud N, Sandström K, Vulgarakis A (2014) Evaluating industrial applicability of virtualization on a distributed multicore platform. In: Proceedings of the 2014 IEEE (ETFA)
5. Kung Y.-S, Tsai M.-H. (2007) FPGA-based speed control IC for PMSM drive with adaptive fuzzy control. In: IEEE transactions on power electronics
6. Jagiella M, Fericean S, Dorneich A (2006) Progress and recent realizations of miniaturized inductive proximity sensors for automation. IEEE Sens J 6:1734–41
7. Streit F-J, Letras M, Schid M, Falk J, Wildermann S, Teich J (2017) High-level synthesis for hardware/software co-design of distributed smart camera systems. In: Proceedings of the 11th international conference on distributed smart cameras
8. Hochradel K, Häcker T, Hohler T, Becher A, Wildermann S, Sutor A (2019) Three-dimensional localization of bats: visual and acoustical. IEEE Sens J 19:5825–33
9. Ziermann T, Wildermann S, Teich J (2009) Can+: a new backward-compatible Controller Area Network (CAN) protocol with up to 16× higher data rates. In: 2009 design, automation and test in Europe conference and exhibition
10. Astarloa A, Moreira N, Bidarte U, Urbina M, Modrono D (2015) FPGA based nodes for sub-microsecond synchronization of cyber-physical production systems on high availability ring networks. In: 2015 international conference on ReConFigurable computing and FPGAs (ReConFig)
11. Orfanus D, Indergaard R, Prytz G, Wien T (2013) EtherCAT-based platform for distributed control in high-performance industrial applications. In: 2013 IEEE 18th conference on emerging technologies factory automation (ETFA)
12. Han Y, Song Y (2003) Condition monitoring techniques for electrical equipment—a literature survey. In: IEEE transactions on power delivery
13. Hashemian HM, Bean WC (2011) State-of-the-art predictive maintenance techniques. In: IEEE transactions on instrumentation and measurement
14. Streit F.-J, Letras M, Wildermann S, Hackenberg B, Falk J, Becher A, Teich J (2018) Model-based design automation of hardware/software co-designs for Xilinx Zynq PSoCs. In: 2018 international conference on ReConFigurable computing and FPGAs (ReConFig)
15. Plagwitz P, Streit F.-J, Becher A, Wildermann S, Teich J (2019) Compiler-based high-level synthesis of application-specific processors on FPGAs. In: 2019 international conference on ReConFigurable computing and FPGAs (ReConFig)
16. Frey P, Lechner M, Bauer T, Shubina T, Yassin A, Wituschek S, Virkus M, Merklein M (2019) Blockchain for forming technology tamper-proof exchange of production data. In: IOP conference series: materials science and engineering, vol 651
17. Hagenah H, Schulte R, Vogel M, Herrmann J, Scharrer H, Lechner M, Merklein M (2019) 4.0 in metal forming questions and challenges. Proc CIRP 79:649–54
18. Xilinx (2016) Zedboard. https://zedboard.org/product/zedboard. Accesssed 25 Mar 2020
19. rc-openlib. https://www.cs12.tf.fau.eu/research/projects/rc-openlib. Accessed 25 Mar 2020