applied sciences

MDPI

Article

Decomposition of a Petri Net-Based Cyber-Physical System toward Implementation as an Integrated System within FPGA

Remigiusz Wiśniewski [1,*], Anikó Costa [2], Marcin Wojnakowski [1] and Maxim Maliński [3]

[1] Institute of Control and Computation Engineering, University of Zielona Góra, Prof. Z. Szafrana 2, 65-516 Zielona Góra, Poland; m.wojnakowski@issi.uz.zgora.pl

[2] NOVA School of Science and Technology, Center of Technology and Systems (UNINOVA-CTS), Associated Lab of Intelligent Systems (LASI), NOVA University Lisbon, 2829-516 Lisbon, Portugal; akc@fct.unl.pt

[3] University of Zielona Góra, Prof. Z. Szafrana 2, 65-516 Zielona Góra, Poland

[*] Correspondence: r.wisniewski@issi.uz.zgora.pl

Featured Application: The proposed technique is strictly oriented on practical application. The decomposed Petri net-based cyber-physical system can be easily implemented within the integrated device such as FPGA. An exemplary decomposition of a real-life CPS—beverage production and distribution machine—is shown in the paper. The system is modelled by a Petri net, decomposed with the proposed technique, and finally implemented within an FPGA device.

Abstract: Decomposition is one of the commonly used techniques applied in the design of Petri net-based cyber-physical systems. Such an operation permits the splitting of the initial system into sequential components that can be further implemented as an integrated or distributed system. This paper is focused on the decomposition of the modelled CPS toward its further implementation as an integrated system, namely an FPGA device. The adequate decomposition method is presented and explained in detail. Moreover, the proposed idea is explained by the real-life example of the beverage production and distribution system. The results of the experiments are presented and discussed.

Keywords: Petri nets; decomposition; cyber-physical system; integrated system; FPGA

check for updates

Citation: Wiśniewski, R.; Costa, A.; Wojnakowski, M.; Maliński, M. Decomposition of a Petri Net-Based Cyber-Physical System toward Implementation as an Integrated System within FPGA. *Appl. Sci.* **2023**, *13*, 7137. https://doi.org/10.3390/app13127137

Academic Editor: Yangquan Chen

Received: 11 May 2023
Revised: 6 June 2023
Accepted: 7 June 2023
Published: 14 June 2023

1. Introduction

Petri nets are a mathematical modelling tool that is used to specify and verify concurrent control systems [1]. They were proposed in the 1960s by Carl Adam Petri in his dissertation [2] and have since observed extensive use in areas such as computer science [3], mechanical engineering [4], and others [5]. Essentially, Petri nets consist of two types of nodes: places and transitions [6]. Places form states of the system, while transitions represent actions that can occur in the system [7]. Places and transitions are connected by arcs, which can show the direction of the flow of tokens representing the behaviour of the specified system [8]. Petri nets are especially suitable for modelling systems with complex interactions between multiple concurrent processes such as popular nowadays cyber-physical systems [9]. By representing these interactions graphically, Petri nets can provide easy insight into the behaviour of the system [3]. Furthermore, Petri nets can be analysed using formal methods to verify properties such as boundedness, safeness, or liveness and help identify potential problems or areas for improvement [10]. Boundedness [11] property shortly ensures a finite number of reachable states in the modelled system, safeness [12] makes binary behaviour of a Petri net for logical control while liveness is key in avoiding deadlocks or unwanted states [13].

A cyber-physical system (CPS) integrates computation with physical processes [14]. Its behaviour is defined by cyber and physical parts of the system [15,16]. The first one (cyber) controls the objects and makes decisions, while the physical part refers to the real world and is disposed to environmental influences. The design techniques of CPS include the joint dynamics of computers, software, networks, as well as physical processes. Such a system is used in a variety of domains, including manufacturing systems, vehicular and transportation systems, medical and health-care systems, smart homes and buildings, social networks and gaming, etc. [17–20]. A Petri net is one of the popular specification approaches of a CPS [12,13,18,21]. Due to its concurrent nature and wide support of verification and validation techniques, a Petri net suits flawlessly the modelling of a CPS [22]. Moreover, the system can be easily and intuitively modelled by a graphical representation, while further designing steps are supported by mathematical techniques [23,24].

One of the crucial steps of the design flow of Petri net-based CPS is the decomposition of the system. In short, this is a technique that is used in order to divide a large system into smaller, sub-systems. In the case of a Petri net, the system is usually split into sequential modules (sequential automata), called state machine components (SMCs). Decomposition is especially useful in the case of distributed CPS. In such systems each module can be implemented in a separate device [25–29]. Furthermore, decomposition plays an important role in the case of integrated systems, implemented within digital devices such as field programmable gate arrays (FPGAs). Recent FPGAs offer a very useful possibility of dynamic partial reconfiguration of the system. This technique permits for the replacement of a portion of the already implemented CPS without stopping the device [7,19,30–32]. This means that the functionality of the selected components of the system is modified while the remaining part of the CPS is still working. It is especially useful in the case of concurrent CPS that executes crucial operations that cannot be interrupted. Moreover, the decomposed modules analysed separately can simplify the analysis of complex cyber-physical systems [33].

There exist several popular decomposition techniques of the Petri net-based systems, each of which has its own advantages and disadvantages. In [34], a method based on shared places and/or shared transitions is presented. The paper shows that place invariants of the global model are the same as those of the decomposed model. However, the resulting components cannot be executed in a distributed way. The linear algebra is also applied in the algorithm shown in [33]. The method searches for the place invariants, which are further transformed into sequential components. The main bottleneck of the presented idea relies on the computational complexity, which is exponential in the general case. On the other hand, the proposed method terminates once the decomposition is found, which extremely reduces the computation process.

In the opposite, a polynomial decomposition algorithm is shown in [35]. The idea is based on the graph theory, involving unique properties of perfect graphs. In particular, the colouring of comparability graphs is applied, which is executed polynomial. Although the method permits for optimal decomposition of the system (into the minimal possible number of components), it cannot be applied to all systems. Moreover, the achieved components do not always reflect the designer's needs. The alternate technique of SMCs computation is shown in [36]. The idea is also based on the graph theory, but it applies an approximate algorithm. Therefore, the result is always achieved, however, it may not be optimal. Furthermore, the technique is not strictly oriented on the decomposition of the system, since it searches for the SMCs covering of the Petri net.

Aybar and Iftar [37] introduced a method based on net structure analysis identifying overlapped sections. Overlapped transitions and places are repeated in all resulting components. Other decomposition ideas use input and output places for communication between the decomposed models [38–40]. In [41], a functional splitting method is proposed, based on the definition of a cutting set. The main concern of this idea is to achieve the same behaviour before and after decomposition. The interface among the decomposed components is composed of transitions, which communicate through output and input events, while the common firing rules are also preserved.

Finally, it is worth mentioning that there exist other decomposition methods not only oriented on the implementation of integrated systems. Such techniques can be applied to other important tasks, e.g., analysis of Petri net properties [42], deadlock detection [43], decomposition into automata [44], or decentralised control of Petri nets [45].

To summarise the discussion above, it can be noticed that the decomposition of the Petri net-based systems is an important step in the designing flow of CPS. However, there are several important issues related to this process, including obtaining adequate components (that is, expected by the designer), computational time (due to the exponential computational complexity of several algorithms), as well as proper synchronisation of the decomposed components. In this paper, we propose a decomposition method oriented on the integrated system. Moreover, the presented technique is illustrated by a real-life example of the beverage production machine implemented within an FPGA device. Let us underline that the proposed solution significantly differs from previous techniques presented by authors in [29,33]. Contrary to the already published ideas, the proposed technique is oriented toward the rapid decomposition process. Therefore, the achieved number of components are examined at each step of the algorithm. Moreover, the designer is able to select the required components manually, which makes the whole process much more flexible and comfortable. Note that this paper is purely focused on the decomposition aspects, while the remaining aspects of the prototyping flow (such as analysis and verification of the CPS, as well as modelling of the decomposed components) can be found in other authors' works including [7,11–13,21,22,32,33,35].

The main contributions of this work are summarised as follows:

- Proposition of a novel decomposition technique of Petri net-based cyber-physical systems oriented on the further realisation within integrated systems;
- Illustration of the proposed idea by a real-life cyber-physical system of beverage production and distribution system;
- Experimental validation of the proposed method by a physical implementation of the integrated system within an FPGA.

2. Definitions and Notations

This section presents definitions and notations used in the paper [1–4,6–8,10,46].

Definition 1. *A Petri net N is a 4-tuple: $N = (P, T, F, M_0)$, where:*

- $P = \{p_1, p_2, \ldots, p_n\}$ *is a finite set of places,*
- $T = \{t_1, t_2, \ldots, t_m\}$ *is a finite set of transitions,*
- $F \subseteq (P \times T) \cup (T \times P)$ *is a finite set of arcs,*
- $M_0 : P \to \mathbb{N}$ *is an initial marking (state).*

Definition 2. *A marking (state) M of a Petri net $N = (P, T, F, M_0)$ is defined as a subset of its places: $M \subset P$. A place belonging to a marking is called a marked place. A marking can be changed by transition firings. Firing of a transition t changes marking by removing a token from each input places of t, and adds a token to each output place of t.*

Definition 3. *Incidence matrix of a Petri net $N = (P, T, F, M_0)$ with n places and m transitions is a matrix $A_{|T| \times |P|}$ of integers, given by:*

$$a_{ij} = \begin{cases} -1, & (p_j, t_i) \in F \\ 1, & (t_i, p_j) \in F \\ 0, & \text{otherwise} \end{cases}. \tag{1}$$

A cell a_{ij} of matrix A refers to transition t_i and place p_j. The columns of the matrix correspond to places, while the rows refer to transitions of a Petri net.

Definition 4. *A place invariant (p-invariant) of a Petri net $N = (P, T, F, M_0)$ is a n-vector $\vec{x}$ of integers such that $A\vec{x} = 0$. The set of places corresponding to nonzero entries in a p-invariant $\vec{x} \geq 0$*

is a support of the p-invariant. A net is covered by p-invariants if every place $p \in P$ belongs to at least one support of a p-invariant.

Definition 5. *A state machine is a Petri net for which every transition has exactly one input place and exactly one output place, i.e.,* $\forall t \in T : |\bullet t| = |t\bullet| = 1$.

Definition 6. *A state machine component (SMC) of a Petri net* $N = (P, T, F, M_0)$ *is a subnet* $S = (P', T', F', M'_0)$ *generated by* $P' \subseteq P$ *such that:*

- *S is a state machine;*
- $P' \in P$, $T' \in T$, $F' \in F \cap (P' \times T') \cup (T' \times P')$;
- *S is strongly connected and has exactly one token in initial marking.*

Definition 7. *A state machine decomposition (S-decomposition) of a Petri net* $N = (P, T, F, M_0)$ *is a set* $\mathcal{S} = \{S_1, \ldots, S_n\}$ *such that each component* $S_j \in \mathcal{S}$ *is an SMC and each place* $p_i \in P$ *belongs to exactly one component* $S_j \in \mathcal{S}$ *. If place* $p_i \in P$ *exists in more than one* $S_j \in \mathcal{S}$ *, it is replaced in all remaining components (except one) by a non-operational place (NOP).*

3. The Proposed Decomposition Method

This section introduces a decomposition technique of a Petri net-based CPS oriented toward realization as an integrated system, e.g., An FPGA device. The proposed decomposition technique is shown in Algorithm 1. The method involves the linear algebra technique. The sequential components are obtained by computation of the place invariants with further manual selection of required SMCs.

Algorithm 1 Decomposition of a Petri net-based integrated system

```
Data: Incidence matrix $A_{|T|\times|P|}$ of Petri net $N = (P, T, F, M_0)$
Result: S-decomposition $\mathcal{S}$ of $N = (P, T, F, M_0)$
1. Initialisation:
    (a) form a unit matrix $Q = [D|A]$ and initialise $D$ as an identity matrix;
    (b) initialise: $\mathcal{S} = \varnothing$;
2. For each column $t$ of matrix $A$:
    (a) find row pairs that annul the $t$-th column of $A$ and append it to matrix $Q$;
    (b) delete rows of $Q$ whose intersection with the $t$-th column is not equal to 0;
    (c) reduce the redundant rows of $Q$ (rows that binary cover to the other ones).;
3. For each row $r$ of $A$ whose all elements are equal to 0:
    (a) obtain support $I_r$ of place invariant from matrix $D$;
    (b) construct a subnet $S_r$ of $N$ according to Definition 5. If $S_t$ forms a proper SMC, add
        $S_r$ to the Sdecomposition-: $\mathcal{S} = \mathcal{S} \cup S_r$;
    (c) remove $r$ from Q.
4. Examine whether $\mathcal{S}$ fulfils the designer needs:
    (a) if $\mathcal{S}$ fulfils the needs:
        • manually select the most suitable SMCs, according to the requirements (the
          redundant components are removed from $\mathcal{S}$);
        • search for places that occur in more than one SMC and manually replace them
          by NOPs according to Def. 7. (subsequent replaced places can be joined).
    (b) otherwise:
        • if $t < |T|$ go to the step 2;
        • else return $\mathcal{S} = \varnothing$ with information that the system cannot be decomposed.
5. Synchronisation: for each transition $t \in T$ shared by two or more components $\mathcal{S}' \in \mathcal{S}$:
    (a) add a synchronisation signal $z_{ti}$ to input place of $t$ in each component $S_i \in \mathcal{S}'$;
    (b) assign a logical conjunction of all synchronisation signals $z_{ti}$ to $t$.
6. Return the set of decomposed and synchronised components $\mathcal{S}$.
```

It should be noted that the general idea was initially shown in our previous papers [29,33]. However, the technique proposed in this paper is significantly modified in order to adjust the obtained components to the designer's requirements. In particular, the components are obtained subsequently and examined at each step of the algorithm, whether they fulfil the designer requirements. Therefore, there is no need to apply an additional selection process. The main advantage of such a proposition is the obtained set of components, which is strictly adjusted to the designer's needs.

The proposed algorithm consists of six main stages. As the input data, an incidence matrix of a Petri net is read. The main operations are executed on the unit matrix Q that unifies the resulting matrix D (initially set as an identity matrix), and incidence matrix A of a Petri net. The first three steps of the method are aimed at searching for subsequent sequential components in the system. The presented algebraic operations transform the unity matrix by annulling the subsequent transitions (columns). After each such operation, matrix A is examined in accordance with the rows that contain entirely zeros. The existence of such values means that place invariants are found (whose supports can be read from D). Moreover, step 3b verifies whether the obtained invariants form proper SMCs and includes them in the temporal set of decomposed components $\mathcal{S}$. At the fourth stage the designer manually examines whether obtained SMCs fulfil the assumed requirements. If the result satisfies the needs, the redundant components are removed from the set $\mathcal{S}$. Otherwise, the procedure is repeated, and subsequent invariants are searched (from step 2). There is also a possibility that the system cannot be decomposed to the state machine components. If such a situation holds (that is, all transitions are examined and components included in set $\mathcal{S}$ do not cover all Petri net places), adequate information is returned to the designer. Let us note that such a situation is rare, and in most cases, indicates errors or mistakes in the model. Finally, the fifth step performs synchronisation of the decomposed components. It is assumed that all SMCs work in the same time domain (the same clock signal is shared among components, with the same frequency), thus such a process is much easier in comparison to the distributed systems (where particular components strictly depend on the destination device and its time domain). In the proposed algorithm the synchronisation is simply executed by applying additional signals to the transitions and places preceding them. Eventually, the decomposed and synchronised system is returned as an output within step 6.

Moving on to the computational complexity of the proposed method, it should be noted that it is exponential in the general case. It is possible that the number of invariants grows exponentially at each operation (step 2), hence the method may not find the result in the assumed time. On the other hand, the improvements applied to the presented solution highly reduce such a risk. In particular, invariants are examined at each operation (step 4), thus the computation process can be safely terminated.

4. Case study Example—Beverage Production and Distribution Machine

The proposed decomposition method is illustrated by a real-life example of a cyber-physical system. Firstly, the description of the Petri net model is presented. Next, the proposed decomposition technique is applied. Finally, the decomposed system is implemented within the FPGA device.

4.1. Description of the System

Figure 1 presents a beverage production and distribution machine. To explain the decomposition process, we a modified version of a beverage production and distribution machine (previously shown in [35]) is used. The presented system works as follows. The machine starts in an idle state and pressing a button on the operator console (input signal x1) initiates the production process. Two valves (output signal y1 and y2) open, and two containers (1st container and 2nd container) are filled with liquid ingredients. Sensors x2 and x3 signal when the upper limit of each container has been reached, respectively.

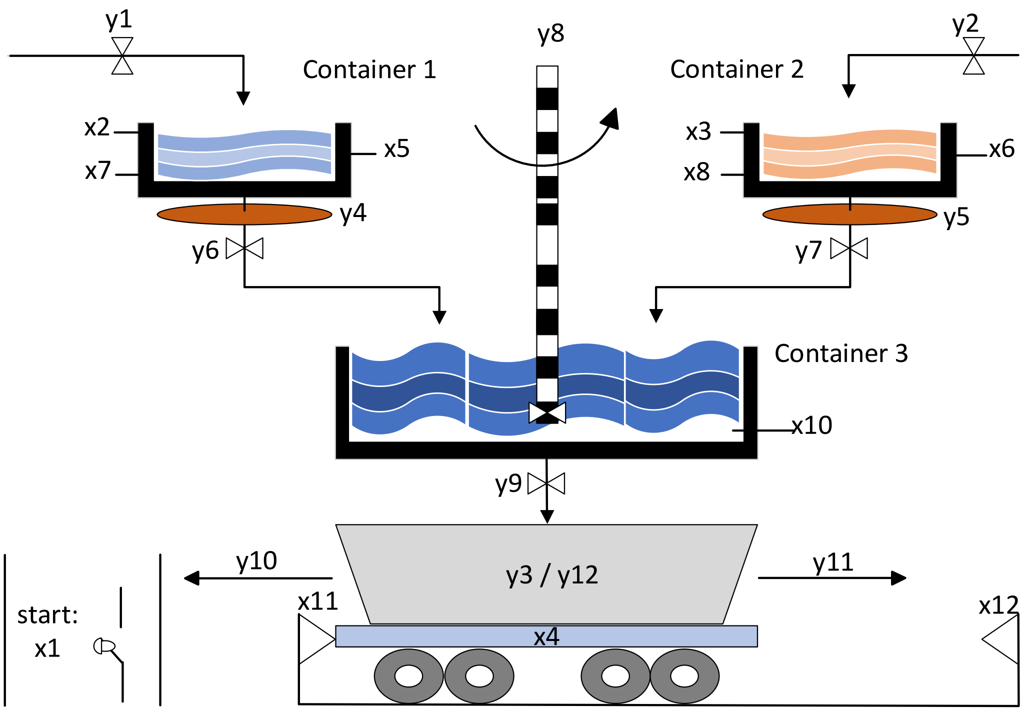


Figure 1. Beverage production and distribution machine.

At that point, the liquid in two containers is warmed up (y4 and y5, respectively) and a cup is being loaded on the trolley (y3). The charged cart (x4) moves to the left edge (y10) until it reaches the leftmost position (x11), and the liquid in containers is warmed until it reaches the required temperature (x5 and x6). The output valves of the two containers are opened (y6 and y7), and the ready ingredients from both containers are mixed (y8) in the main (third) container. When both upper containers (the first and second one) are empty (x7 and x8), the beverage is ready for distribution (x9). Then, the output valve of the main container opens (y9), and the liquid is poured into the cup on the trolley. Once the main container is empty (x10) and thus the cup is full, the trolley moves to the right (y11), and after it reaches the appropriate position (x12), the cup is removed from the trolley, and the system returns to its initial state.

Figure 2 shows a Petri net model of the discussed CPS. The Petri net-based specification contains twenty places and sixteen transitions. Moreover, there are twelve input signals associated with sensors (for instance liquid level sensor) and twelve output signals assigned to physical parts of the beverage machine. The association between transitions and input signals is presented in Table 1, while Table 2 exposes the relations between places and output signals. For example, sensor x1 (pressing of the start button) is assigned to the transition t1, while filling up of the first container is signalized by output y1 (associated with place p2).

Table 1. Association between transitions and inputs.

Transition	Input	Condition
t1	x1	Pressed start button
t2	x2	First container filled
t3	x3	Second container filled
t4	x4	Cup is put on trolley
t5	x5	First ingredients ready
t6	x6	Second ingredients ready
t7	x11	Trolley at left edge
t8	-	-
t9	x7	First container empty

Table 1. *Cont.*

Transition	Input	Condition
t10	x8	Second container empty
t11	x9	Mixing ended
t12, t13	-	-
t14	x10	Main container empty
t15	x12	Trolley at right edge
t16	!x4	Cup is removed from trolley

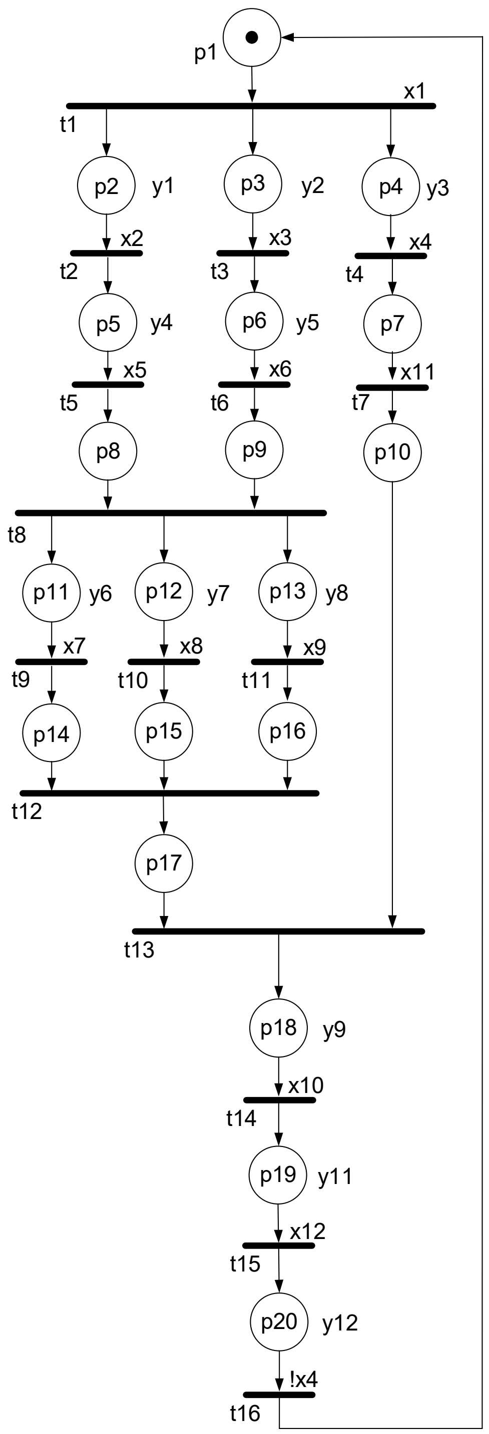


Figure 2. A Petri net-based model of the beverage distribution and production machine.

Table 2. Association between places and outputs.

Place	Output	Task
p1	-	-
p2	y1	Fill up first container
p3	y2	Fill up second container
p4	y3	Load cup
p5	y4	Warm up 1st container
p6	y5	Warm up 2nd container
p7	y10	Move trolley to left
p8, ... , p10	-	-
p11	y6	Open first valve
p12	y7	Open second valve
p13	y8	Mix ingredients
p14, ... , p17	-	-
p18	y9	Open main valve
p19	y11	Move trolley to right
p20	y12	Unload cup

4.2. Decomposition of the System

Let us now decompose the above system according to the second decomposition technique. This time the CPS is oriented into the implementation within integrated device. Let us apply Algorithm 1 in order to decompose the system. Firstly, an incidence matrix A_1 of a Petri net N_1 is read. The matrix consists of $|T| = 16$ rows and $|P| = 20$ columns, as shown in Figure 3.

$$A_1 = \begin{bmatrix}
-1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

Figure 3. Incidence matrix A_1 of Petri net N_1 that describes beverage machine.

At the first step of the algorithm a unit matrix Q given as $[D|A_1]$ is formed, while D is equal to the identity matrix. During the subsequent stages, algorithm searches for the place invariants that form proper SMCs. Such operations are executed within steps 2–4, until set $\mathcal{S}$ (of all found SMCs) cover all places of the Petri net. For the given example, this condition is fulfilled at the 15th iteration (examination of 15th transition). The following place invariants are obtained:

$I_1 = [10010010010000000111]$, with support: $\{p_1,\ p_4,\ p_7,\ p_{10},\ p_{18},\ p_{19},\ p_{20}\}$,
$I_2 = [11001001001001001111]$, with support: $\{p_1,\ p_2,\ p_5,\ p_8,\ p_{11},\ p_{14},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$,
$I_3 = [10100100101001001111]$, with support: $\{p_1,\ p_3,\ p_6,\ p_9,\ p_{11},\ p_{14},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$,
$I_4 = [11001001000100101111]$, with support: $\{p_1,\ p_2,\ p_5,\ p_8,\ p_{12},\ p_{15},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$,
$I_5 = [10100100100100101111]$, with support: $\{p_1,\ p_3,\ p_6,\ p_9,\ p_{12},\ p_{15},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$,
$I_6 = [11001001000010011111]$, with support: $\{p_1,\ p_2,\ p_5,\ p_8,\ p_{13},\ p_{16},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$,
$I_7 = [10100100100010011111]$, with support: $\{p_1,\ p_3,\ p_6,\ p_9,\ p_{13},\ p_{16},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$.

All the above invariants form proper SMCs, thus set $\mathcal{S}$ consists of seven components $\mathcal{S} = \{S_1, \ldots, S_7\}$, as follows:

$S_1 = \{p_1,\ p_4,\ p_7,\ p_{10},\ p_{18},\ p_{19},\ p_{20}\}$,
$S_2 = \{p_1,\ p_2,\ p_5,\ p_8,\ p_{11},\ p_{14},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$,
$S_3 = \{p_1,\ p_3,\ p_6,\ p_9,\ p_{11},\ p_{14},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$,
$S_4 = \{p_1,\ p_2,\ p_5,\ p_8,\ p_{12},\ p_{15},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$,
$S_5 = \{p_1,\ p_3,\ p_6,\ p_9,\ p_{12},\ p_{15},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$,
$S_6 = \{p_1,\ p_2,\ p_5,\ p_8,\ p_{13},\ p_{16},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$,
$S_7 = \{p_1,\ p_3,\ p_6,\ p_9,\ p_{13},\ p_{16},\ p_{17},\ p_{18},\ p_{19},\ p_{20}\}$.

Since the above SMCs cover all places of the Petri net, the designer manually selects the most suitable components and checks, whether obtained solution fulfils the needs. In the presented example, four SMCs are selected as a final result: $\{S_1, S_2, S_5, S_6\}$, while the redundant three $\{S_3, S_4, S_7\}$, are reduced from the final solution. At the subsequent step, all places that occur in more than one SMC are manually replaced NOPs. This operation simply substitutes places that belong to multiple components by no-operational ones in order to avoid doubled actions performed by the decomposed controller. In the discussed example, the following result is obtained after this operation:

$S_1 = \{p_1,\ p_4,\ p_7,\ p_{10},\ NOP_1,\ p_{19},\ p_{20}\}$,
$S_2 = \{\mathrm{NOP}_2,\ \mathrm{p}_2, \mathrm{p}_5, \mathrm{p}_8, \mathrm{p}_{11}, \mathrm{p}_{14}, \mathrm{NOP}_3\}$
$S_5 = \{\mathrm{NOP}_4,\ p_3,\ p_6,\ p_9,\ p_{12},\ \mathrm{p}_{15}, \mathrm{NOP}_5\}$,
$S_6 = \{\mathrm{NOP}_6, \mathrm{p}_{13}, \mathrm{p}_{16}, \mathrm{p}_{17}, \mathrm{p}_{18}, \mathrm{NOP}_7\}$.

The applied NOPs strictly refer to the functionality of the system. For example, p_{18} in the first component is replaced by NOP_1, since operation assigned to this place is executed within S_6. Similarly, p_1 occurs only within S_1, while in all remaining components it is replaced by NOPs (NOP_2 in S_2, NOP_4 in S_5, and NOP_6 in S_6), etc. Note that enumeration of NOPs does not influence on the final result (it is just an indexation), thus designers can do it according to their needs. In the presented example, three NOPs are initially marked: NOP_2, NOP_4, and NOP_6 since they refer to the initially marked place p_1. Furthermore, there are NOPs that replace more than one place. For example, NOP_3 replaces four subsequent places: $p_{17}, p_{18}, p_{19}, p_{20}$. Such a joining of NOPs is not necessary (the system can be decomposed by replacing a place per one NOP) but may reduce the utilised area of the FPGA.

Figure 4 illustrates the decomposed Petri net. It should be underlined that obtained four components strictly refer to the functionality of the CPS. The first SMC is in charge of managing the trolley (with all associated conditions and actions). Components S_2 and S_5 control containers (first and second, respectively). Finally, actions assigned to S_6 are related to the third container.

4.3. Experimental Validation of the Proposed Method

This section presents the experimental validation of the proposed decomposition technique. The obtained model of the beverage production and distribution system was implemented in the real FPGA device. In particular, Nexys 4 DDR prototyping board with the FPGA device xc7a100tcsg324-2 from Artix 7 family was used to perform hardware validation. Firstly, the decomposed modules were described in the Verilog hardware language. The applied encoding style is based on the rules shown in other authors' works, e.g., [7,33]. In particular, each of decomposed SMCs is described as a sequential module. An exemplary description of S_1 is shown in Listing 1. There are three main sections within this module. The first one triggers switching states in the system. It is described by an *always* block, and it is activated by a positive clock signal (common oscillator for all components) or asynchronous reset. The second section prepares adequate switching values for the next state. It is also described by an *always* block; however, the sensitivity list includes input signals (and the value of the current state). Finally, the last block associates outputs with particular states by continuous assignments.

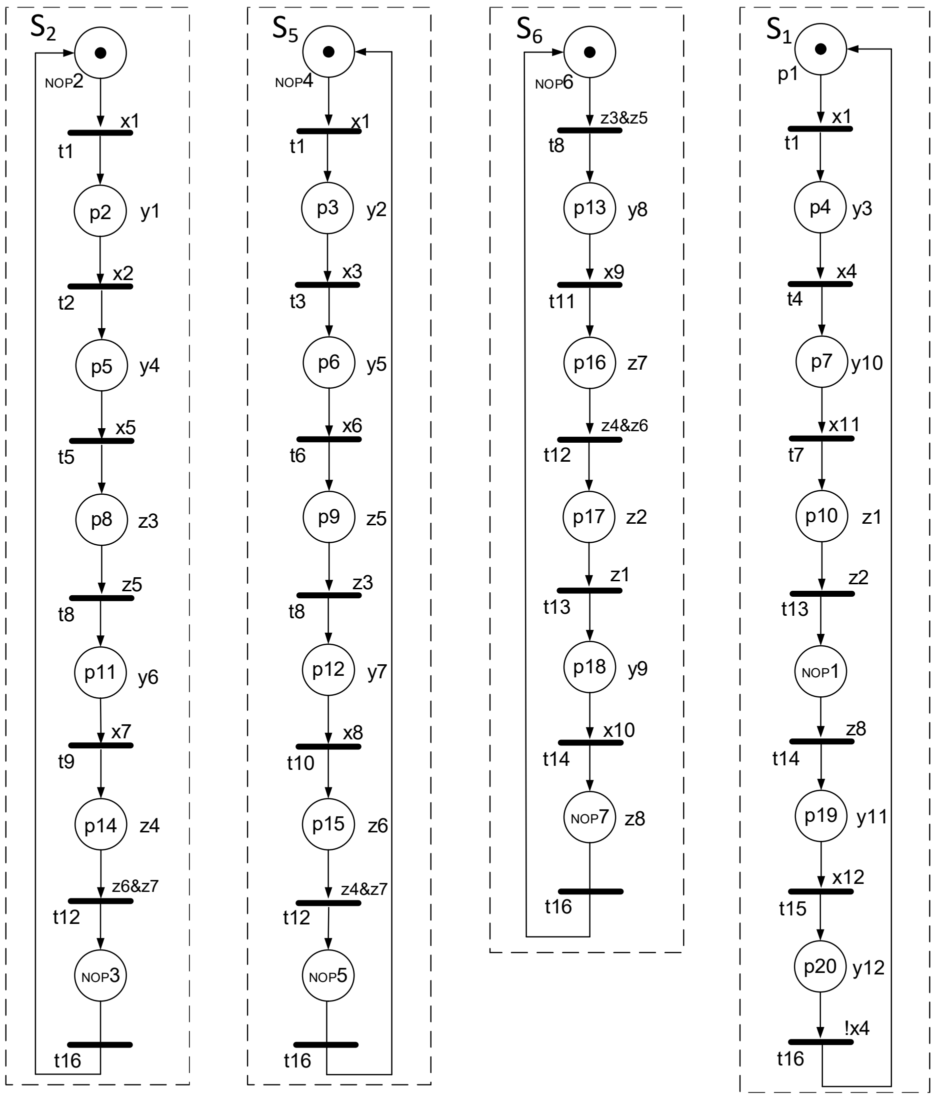


Figure 4. The decomposed CPS of production and distribution beverage machine.

All the SMCs are controlled by the *top-level* module. This module simply includes an instance of each component. Listing 2 shows the Verilog code for the presented beverage production and distribution machine. Note that the *top-level* module contains twelve input and twelve output signals that are directly used in the modelled CPS. The internal signals (vector *z*) are applied to synchronise the decomposed components. Additional *clk* and *reset* signals are used for synchronisation (clock) and restarting the system, respectively.

Listing 1. Description of exemplary *S-component* (*S1*) (actions performed by a trolley).

```
// Description of S1
module s1 (
     output y3, y10, y11, y12,
     output z1,
     input x1, x4, x11, x12,
     input z2, z8,
     input clk, reset
);
     reg [2:0] state, next;
     parameter P1 = 3'b000, P4 = 3'b001, P7 = 3'b011, P10 = 3'b010,
                     NOP1 = 3'b110, P19 = 3'b111, P20 = 3'b101;
     always @(posedge clk or posedge reset) begin
         if (reset) state <= P1;
         else state <= next;
     end
     always @ (state or x1 or x4 or x11 or x12 or z2 or z8) begin
         case (state)
             P1:  next <= (x1) ?  P4 :  P1;
             P4:  next <= (x4) ?  P7 :  P4;
             P7:  next <= (x11) ?  P10 :  P7;
             P10:  next <= (z2) ?  NOP1 :  P10;
             NOP1:  next <= (z8) ?  P19 :  NOP1;
             P19:  next <= (x12) ?  P20 :  P19;
             P20:  next <= (!x4) ?  P1 :  P20;
             default:  next <= P1;
         endcase
     end
     assign y3 = (state == P4) ?  1'b1 :  1'b0;
     assign y10 = (state == P7) ?  1'b1 :  1'b0;
     assign y11 = (state == P19) ?  1'b1 :  1'b0;
     assign y12 = (state == P20) ?  1'b1 :  1'b0;
     assign z1 = (state == P10) ?  1'b1 :  1'b0;
endmodule
```

Listing 2. Description of the *top-level* module.

```
// Description of top-level
module bevarege_decomposed (
    output [1:12] y,
    input [1:12] x,
    input clk, reset
  );
  wire z[1:8];
  s1 trolley      (y[3],y[10],y[11],y[12],z[1],z[8],x[1],x[4],
                   x[11],x[12],z[2],z[8],clk,reset);
  s2 container1   (y[1],y[4],y[6],z[3],z[4],x[1],x[2],x[5],x[7],
                   z[5],z[6],z[7],clk,reset);
  s5 container2   (y[2],y[5],y[7],z[5],z[6],x[1],x[3],x[6],x[8],
                   z[3],z[4],z[7],clk,reset);
  s6 container3   (y[8],y[9],z[2],z[7],z[8],x[9],x[10],z[1],
                   z[3],z[4],z[5],z[6],clk,reset);
endmodule
```

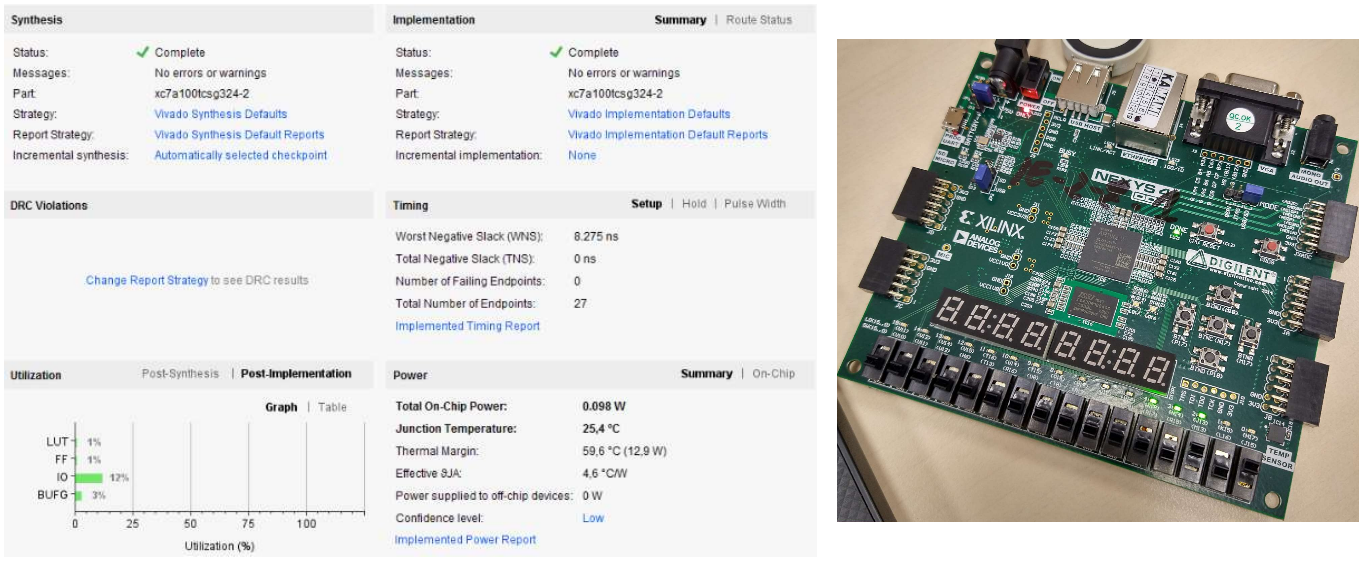
The modelled system was logically synthesised and implemented with Xilinx Vivado 2022.1.2 software. Table 3 presents the utilisation of an FPGA. The decomposed CPS consumes just a fraction of the device's resources. It requires 25 LUTs blocks (<1%) and

27 flip-flops (<1%). Moreover, the system utilises 12% of IO blocks (they are used for validation of the CPS).

Table 3. Utilization of FPGA device resources.

Device Resources	Utilization
LUTs blocks	25 of 63,400 (<1%)
Flip-flops	27 of 126,800 (<1%)
IO blocks	26 of 210 (12%)

Finally, the functionality of the system was tested within the physical FPGA. The validation was performed with the use of switches and LEDs available on the Nexys 4 DDR prototyping board (Figure 5 right). In particular, input signals were assigned to switches, while output signals were connected with diodes to visualise the operation of the real-life system in an accessible way.



Synthesis

Status: ✓ Complete
Messages: No errors or warnings
Part: xc7a100tcsg324-2
Strategy: Vivado Synthesis Defaults
Report Strategy: Vivado Synthesis Default Reports
Incremental synthesis: Automatically selected checkpoint

DRC Violations

Change Report Strategy to see DRC results

Utilization Post-Synthesis | **Post-Implementation**

Graph | Table

Implementation **Summary** | Route Status

Status: ✓ Complete
Messages: No errors or warnings
Part: xc7a100tcsg324-2
Strategy: Vivado Implementation Defaults
Report Strategy: Vivado Implementation Default Reports
Incremental implementation: None

Timing **Setup** | Hold | Pulse Width

Worst Negative Slack (WNS): 8.275 ns
Total Negative Slack (TNS): 0 ns
Number of Failing Endpoints: 0
Total Number of Endpoints: 27
Implemented Timing Report

Power **Summary** | On-Chip

Total On-Chip Power: **0.098 W**
Junction Temperature: **25,4 °C**
Thermal Margin: 59,6 °C (12,9 W)
Effective ϑJA: 4,6 °C/W
Power supplied to off-chip devices: 0 W
Confidence level: Low
Implemented Power Report

Figure 5. Summary of synthesis and implementation of the CPS (**left**), and the utilised Nexys 4 DDR board (**right**).

5. Conclusions

A novel decomposition method of Petri net-based CPS aimed at the integrated systems was proposed in the paper. The presented technique is based on linear algebra and involves place invariants computation in order to obtain sequential components (SMCs). Contrary to the existing decomposition methods, the proposed algorithm is strictly aimed at the designer's needs. Therefore, the selection of particular components is executed manually. The proposed idea was validated experimentally, with the use of integrated device. In particular, a beverage production and distribution cyber-physical system was modelled by a Petri net, decomposed with the presented concept, and finally implemented within an FPGA device.

Besides the benefits mentioned above, there are also limitations to the proposed technique. First of all, obtaining the sequential components can be exponential in the worst case. This means, that in the case of extremely complex systems, decomposition may take a long time. On the other hand, introduced manual selection may essentially speed-up this process, since the designer is able to interrupt the process once the obtained components cover all places of the modelled CPS. Furthermore, the proposed method is strictly oriented on integrated systems. Nevertheless, since the decomposed components form sequential automata, they can be applied to distributed systems. However, in such a case, more advanced synchronisation mechanisms ought to be provided, since components may work in different time-domains. Therefore, this is an excellent starting point for future works.

In particular, plans for the future include the development of the decomposition algorithm oriented on the distributed systems. This means that the CPS can be implemented within several devices, e.g., a set of connected microprocessors such as Arduinos, PLCs, etc. Besides splitting the system into sequential components, such a method ought to include proper synchronisation between devices in order to assure proper functionality of the whole CPS.

Author Contributions: Conceptualization, R.W. and A.C.; Methodology, R.W., A.C. and M.W.; Software, R.W.; Validation, R.W. and M.W.; Formal analysis, R.W. and A.C.; Investigation, R.W., A.C. and M.W.; Resources, R.W., A.C., M.W. and M.M.; Data curation, R.W.; Writing—original draft, R.W. and M.W.; Writing—review & editing, R.W., A.C. and M.W.; Visualization, R.W. and M.M.; Supervision, R.W. and A.C.; Project administration, R.W.; Funding acquisition, R.W. and A.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by NATIONAL SCIENCE CENTRE, POLAND, grant number 2019/35/B/ST6/01683. This research was funded (in part) by the Portuguese FCT program, Center of Technology and Systems (CTS) UIDB/00066/2020/UIDP/00066/2020. The APC was funded by A. Costa and R. Wisniewski.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Girault, C.; Valk, R. *Petri Nets for Systems Engineering: A Guide to Modeling, Verification, and Applications*; Springer: Berlin/Heidelberg, Germany, 2003; ISBN 978-3-540-41217-5.
2. Petri, C.A. *Kommunikation mit Automaten*; Mathematisches Institut der Universität Bonn: Bonn, Germany, 1962.
3. van der Aalst, W.M.P. Workflow Verification: Finding Control-Flow Errors Using Petri-Net-Based Techniques. In *Business Process Management: Models, Techniques, and Empirical Studies*; van der Aalst, W., Desel, J., Oberweis, A., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2000; pp. 161–183. ISBN 978-3-540-45594-3.
4. Li, Z.; Zhou, M. Elementary Siphons of Petri Nets and Their Application to Deadlock Prevention in Flexible Manufacturing Systems. *IEEE Trans. Syst. Man Cybern.-Part Syst. Hum.* **2004**, *34*, 38–51. [CrossRef]
5. Bakhtari, A.R.; Kumar, V.; Waris, M.M.; Sanin, C.; Szczerbicki, E. Industry 4.0 Implementation Challenges in Manufacturing Industries: An Interpretive Structural Modelling Approach. *Procedia Comput. Sci.* **2020**, *176*, 2384–2393. [CrossRef]
6. Reisig, W. (Ed.) Nets Consisting of Places and Transistions. In *Petri Nets: An Introduction*; EATCS Monographs on Theoretical Computer Science; Springer: Berlin/Heidelberg, Germany, 1985; pp. 62–76. ISBN 978-3-642-69968-9.
7. Wiśniewski, R. *Prototyping of Concurrent Control Systems Implemented in FPGA Devices*; Advances in Industrial Control; Springer: Berlin/Heidelberg, Germany, 2017; ISBN 978-3-319-45810-6.
8. David, R.; Alla, H. *Petri Nets and Grafcet: Tools for Modelling Discrete Event Systems*; Prentice-Hall, Inc.: Upper Saddle River, NJ, USA, 1992; ISBN 978-0-13-327537-7.
9. Lee, E.A.; Seshia, S.A. *Introduction to Embedded Systems: A Cyber-Physical Systems Approach*, 2nd ed.; The MIT Press: Cambridge, MA, USA, 2016; ISBN 978-0-262-53381-2.
10. Murata, T. Petri Nets: Properties, Analysis and Applications. *Proc. IEEE* **1989**, *77*, 541–580. [CrossRef]
11. Wojnakowski, M.; Wiśniewski, R. Verification of the Boundedness Property in a Petri Net-Based Specification of the Control Part of Cyber-Physical Systems. In *Technological Innovation for Applied AI Systems*; Camarinha-Matos, L.M., Ferreira, P., Brito, G., Eds.; Springer: Cham, Switzerland, 2021; pp. 83–91.
12. Wojnakowski, M.; Wiśniewski, R.; Bazydło, G.; Popławski, M. Analysis of Safeness in a Petri Net-Based Specification of the Control Part of Cyber-Physical Systems. *Appl. Math. Comput. Sci.* **2021**, *31*, 647–657. [CrossRef]
13. Wojnakowski, M.; Popławski, M.; Wiśniewski, R.; Bazydło, G. Hippo-CPS: Verification of Boundedness, Safeness and Liveness of Petri Net-Based Cyber-Physical Systems. In *Technological Innovation for Digitalization and Virtualization*; Camarinha-Matos, L.M., Ed.; Springer: Cham, Switzerland, 2022; pp. 74–82.
14. Lee, E.A. Cyber Physical Systems: Design Challenges. In Proceedings of the 2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC), Orlando, FL, USA, 5–7 May 2008; pp. 363–369.
15. Lee, J.; Bagheri, B.; Kao, H.-A. A Cyber-Physical Systems Architecture for Industry 4.0-Based Manufacturing Systems. *Manuf. Lett.* **2015**, *3*, 18–23. [CrossRef]

16. Hahanov, V. *Cyber Physical Computing for IoT-Driven Services*; Springer: Berlin/Heidelberg, Germany, 2018; ISBN 978-3-319-54824-1.
17. Majdzik, P. A Feasible Schedule for Parallel Assembly Tasks in Flexible Manufacturing Systems. *Int. J. Appl. Math. Comput. Sci.* **2022**, *32*, 51–63. [CrossRef]
18. Patalas-Maliszewska, J.; Posdzich, M.; Skrzypek, K. Modelling Information for the Burnishing Process in a Cyber–Physical Production System. *Int. J. Appl. Math. Comput. Sci.* **2022**, *32*, 345–354. [CrossRef]
19. Bazydło, G. Designing Reconfigurable Cyber-Physical Systems Using Unified Modeling Language. *Energies* **2023**, *16*, 1273. [CrossRef]
20. Dey, N.; Ashour, A.S.; Shi, F.; Fong, S.J.; Tavares, J.M.R.S. Medical Cyber-Physical Systems: A Survey. *J. Med. Syst.* **2018**, *42*, 74. [CrossRef]
21. Wojnakowski, M.; Wiśniewski, R.; Popławski, M.; Bazydło, G. Analysis of Control Part of Cyber-Physical Systems Specified by Interpreted Petri Nets. In Proceedings of the 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Prague, Czech Republic, 9–12 October 2022; pp. 1090–1095.
22. Wiśniewski, R.; Wojnakowski, M.; Li, Z. Design and Verification of Petri-Net-Based Cyber-Physical Systems Oriented toward Implementation in Field-Programmable Gate Arrays—A Case Study Example. *Energies* **2023**, *16*, 67. [CrossRef]
23. Patalas-Maliszewska, J.; Wiśniewski, R.; Topczak, M.; Wojnakowski, M. Design Optimization of the Petri Net-Based Production Process Supported by Additive Manufacturing Technologies. *Bull. Pol. Acad. Sci. Tech. Sci.* **2022**, *70*, e140693.
24. Patalas-Maliszewska, J.; Wiśniewski, R.; Topczak, M.; Wojnakowski, M. Modelling of the Effectiveness of Integrating Additive Manufacturing Technologies into Petri Net-Based Manufacturing Systems. In Proceedings of the 2022 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Padua, Italy, 18–23 July 2022; pp. 1–9.
25. Wisniewski, R.; Bazydło, G.; Gomes, L.; Costa, A.; Wojnakowski, M. Analysis and Design Automation of Cyber-Physical System with Hippo and IOPT-Tools. In Proceedings of the IECON 2019—45th Annual Conference of the IEEE Industrial Electronics Society, Lisbon, Portugal, 14–17 October 2019; Volume 1, pp. 5843–5848.
26. Costa, A.; Gomes, L.; Barros, J.P.; Oliveira, J.; Reis, T. Petri Nets Tools Framework Supporting FPGA-Based Controller Implementations. In Proceedings of the 2008 34th Annual Conference of IEEE Industrial Electronics, Orlando, FL, USA, 10–13 November 2008; pp. 2477–2482.
27. Kubica, M.; Opara, A.; Kania, D. *Technology Mapping for LUT-Based FPGA*; Lecture Notes in Electrical Engineering; Springer: Cham, Switzerland, 2021; Volume 713, ISBN 978-3-030-60487-5.
28. Zaitsev, D. Decomposition-Based Calculation of Petri Net Invariants. In Proceedings of the Workshop on Token Based Computing, Bologna, Italy, 21 May 2004.
29. Wiśniewski, R.; Karatkevich, A.; Stefanowicz, Ł.; Wojnakowski, M. Decomposition of Distributed Edge Systems Based on the Petri Nets and Linear Algebra Technique. *J. Syst. Archit.* **2019**, *96*, 20–31. [CrossRef]
30. Guo, C.; Zhang, Y.; Chen, L.; Zhou, T.; Li, X.; Wang, M.; Wen, Z. A Novel Application of FPGA-Based Partial Dynamic Reconfiguration System with CBSC. In Proceedings of the 2012 VIII Southern Conference on Programmable Logic, Bento Gonçalves, Brazil, 20–23 March 2012; pp. 1–4.
31. Wang, L.; Feng-yan, W. Dynamic Partial Reconfiguration in FPGAs. In Proceedings of the 3rd International Symposium on Intelligent Information Technology Application, Nanchang, China, 21–22 November 2009; Volume 2, pp. 445–448.
32. Wiśniewski, R. Dynamic Partial Reconfiguration of Concurrent Control Systems Specified by Petri Nets and Implemented in Xilinx FPGA Devices. *IEEE Access* **2018**, *6*, 32376–32391. [CrossRef]
33. Wiśniewski, R. Design of Petri Net-Based Cyber-Physical Systems Oriented on the Implementation in Field Programmable Gate Arrays. *Energies* **2021**, *14*, 7054. [CrossRef]
34. Christensen, S.; Petrucci, L. Modular Analysis of Petri Nets. *Comput. J.* **2000**, *43*, 224–242. [CrossRef]
35. Wiśniewski, R.; Karatkevich, A.; Adamski, M.; Costa, A.; Gomes, L. Prototyping of Concurrent Control Systems with Application of Petri Nets and Comparability Graphs. *IEEE Trans. Control Syst. Technol.* **2018**, *26*, 575–586. [CrossRef]
36. Clarke, E.M., Jr.; Grumberg, O.; Peleg, D. *Model Checking*; Cyber Physical Systems Series; MIT Press: Cambridge, MA, USA, 1999; ISBN 978-0-262-03270-4.
37. Aybar, A.; Iftar, A. Overlapping Decompositions and Expansions of Petri Nets. *IEEE Trans. Autom. Control* **2002**, *47*, 511–515. [CrossRef]
38. Zaitsev, D.A. Compositional Analysis of Petri Nets. *Cybern. Syst. Anal.* **2006**, *42*, 126–136. [CrossRef]
39. Bruno, G.; Agarwal, R.; Castella, A.; Pescarmona, M.P. CAB: An Environment for Developing Concurrent Application. In *Application and Theory of Petri Nets 1995*; De Michelis, G., Diaz, M., Eds.; Springer: Berlin/Heidelberg, Germany, 1995; pp. 141–160.
40. Nishi, T.; Maeno, R. Petri Net Modeling and Decomposition Method for Solving Production Scheduling Problems. *J. Adv. Mech. Des. Syst. Manuf.* **2007**, *1*, 262–271. [CrossRef]
41. Costa, A.; Gomes, L. Petri Net Partitioning Using Net Splitting Operation. In Proceedings of the 2009 7th IEEE International Conference on Industrial Informatics, Cardiff, UK, 23–26 June 2009; pp. 204–209.
42. Marussy, K.; Klenik, A.; Molnár, V.; Vörös, A.; Majzik, I.; Telek, M. Efficient Decomposition Algorithm for Stationary Analysis of Complex Stochastic Petri Net Models. In *Application and Theory of Petri Nets and Concurrency*; Kordon, F., Moldt, D., Eds.; Springer: Cham, Switzerland, 2016; pp. 281–300.
43. Zhong, C.; He, W.; Li, Z.; Wu, N.; Qu, T. Deadlock Analysis and Control Using Petri Net Decomposition Techniques. *Inf. Sci.* **2019**, *482*, 440–456. [CrossRef]

44. Bouvier, P.; Garavel, H.; Ponce-de-León, H. Automatic Decomposition of Petri Nets into Automata Networks—A Synthetic Account. In *Application and Theory of Petri Nets and Concurrency*; Janicki, R., Sidorova, N., Chatain, T., Eds.; Springer: Cham, Switzerland, 2020; pp. 3–23.
45. Ye, J.; Zhou, M.; Li, Z.; Al-Ahmari, A. Structural Decomposition and Decentralized Control of Petri Nets. *IEEE Trans. Syst. Man Cybern. Syst.* **2018**, *48*, 1360–1369. [CrossRef]
46. Karatkevich, A. *Dynamic Analysis of Petri Net-Based Discrete Systems*; Lecture Notes in Control and Information Sciences; Springer: Berlin/Heidelberg, Germany, 2007; ISBN 978-3-540-71464-4.