Letters

Computer-Aided Identification of Equivalent Power Electronics Converters

Guipeng Chen, *Member, IEEE*, Liping Mo, Yuwei Liu, Xinlin Qing, and Yihua Hu, *Senior Member, IEEE*

***Abstract*—Equivalent power electronics converters (PECs), which have same performance characteristics but dissimilar configurations, are easily mistaken as different converters to be repeatedly studied in the practice, resulting in extra workload. Therefore, it is essential to implement the equivalence identification to avoid the undesired repeated research. In order to accomplish this goal, a computer-aided solution is proposed in this letter, aiming to quickly and precisely identify the equivalent PECs. First, two identification rules are figured out, with which different PECs can be systematically judged to be equivalent or not. Then, software Altium Designer is used to automatically acquire the information of PECs, including the components and the connecting relationships. After that, software MATLAB is utilized to further complete the data process, circuit loop search, and final judgment. Compared with the conventional manual identification method, which is totally dependent on researcher's effort, the proposed computer-aided scheme is more convenient, accurate, and practical, which is beneficial for the topology research in both academic and industrial fields.**

***Index Terms*—Computer aided, equivalent power electronics converter (PEC), graph theory, MATLAB.**

I. Introduction

FOR engineering applications with various system conditions and requirements, different power electronics converters (PECs) are preferred to obtain excellent performances. Hence, numerous researches have been devoted to the topology derivation of new PECs, aiming to provide more favorable topologies for the practical applications [1]–[9]. With the increasing study on the topology derivation, equivalent PECs, which have dissimilar configurations but same performance characteristics with the existing ones, will be inevitably proposed. Take the two dual-input single-output (DISO) dc–dc converters in Fig. 1 as an example. The converter in Fig. 1(a) has been proposed in [1], whereas the another one in Fig. 1(b) can be easily derived from the single-input dual-output buck/buck converter in [2] by replacing one output by one input. They seem to be different, but are actually equivalent. The converter in Fig. 1(b) will turn to the one in Fig. 1(a), by moving the position of diode D_2 from top to bottom and swapping the positions of voltage source V_2 and switch S_2. Particularly, this phenomenon will become more common when the topology derivation method is systematic that multiple topologies are simultaneously derived and several of them may be equivalent. For example, 22 viable three-port topologies were obtained in [3] with the programmable topology derivation method and 12 of them are equivalent with the other 10. Because the performance characteristics of equivalent PECs including the voltage gain, efficiency, and power density are the same, repeated study on them is unnecessary and should be avoided in the practice, otherwise extra workload would be undesired incurred. Therefore, it is very essential to effectively identify the equivalent PECs.

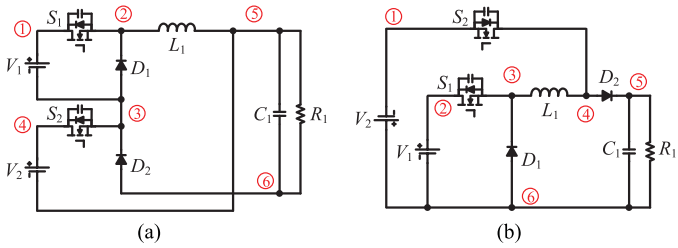

Fig. 1. Two equivalent DISO dc–dc converters with different connections. (a) First. (b) Second.

The equivalence identification of PECs can be realized by manually modifying the position of components and then judging whether they are equivalent or not. Nevertheless, the manual method is time consuming and, more importantly, it is easy to mistakenly regard two equivalent PECs as different, e.g., the two equivalent converters in Fig. 1 are easily considered to be different at first glance. This problem is getting more severe with the increasing complexity of topology configurations. In a word, the manual method is not convenient and precise enough. Nowadays, mathematical and computer-aided tools have been used in more and more PEC researches to accelerate the topology derivation, e.g., the graph theory in [11] and the programmable solution in [3]. In this letter, a rigorous computer-aided method based on the graph theory is also proposed to substitute the conventional manual method, aiming to quickly and precisely accomplish the identification of equivalent PECs. Moreover, thanks to the

Manuscript received January 30, 2019; revised March 17, 2019; accepted April 1, 2019. Date of publication April 8, 2019; date of current version June 28, 2019. This work was supported in part by the National Key R&D Program of China (2017YFE0112400), in part by the National Postdoctoral Program for Innovative Talents (BX201700144), and in part by the China Postdoctoral Science Foundation (2017M622074). *(Corresponding author: Xinlin Qing.)*

G. Chen, L. Mo, Y. Liu, and X. Qing are with the School of Aerospace Engineering, Xiamen University, Xiamen 361005, China (e-mail: cgp2017@xmu.edu.cn; 980900094@qq.com; 745948914@qq.com; xinlinqing@xmu.edu.cn).

Y. Hu is with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3BX, U.K. (e-mail: y.hu35@liverpool.ac.uk).

Color versions of one or more of the figures in this letter are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TPEL.2019.2909544

computerization, the proposed method can retain its favorable advantages even when the topology configurations of PECs are complicated.

II. Computer-Aided Identification of Equivalent PECs

In this section, the proposed computer-aided method to identify equivalent PECs is introduced in detail. First, two rules are figured out, with which different PECs can be systematically judged to be equivalent or not. Then, two commonly used software, Altium Designer and MATLAB, are utilized to automatically complete the judgment.

A. Identification Rules

In essence, the topology configuration of PECs is determined by two parts, components and connections. PECs having different components are certainly different and, thus, the first rule of equivalent PECs is that they must have same components, i.e., the number of different types of components such as switch, diode, inductor, and capacitor must be totally the same. In addition, for PECs with same components but different connections, they are equivalent if their connecting relationships turn to be the same after moving some components to their other series-connected positions. However, this criterion is not easy to implement in the practice, and it is needed to be transformed into a realizable rule. Actually, changing the series-connected positions of components has no influence on the composition of circuit loop. Assume that the number of loops in a PEC is N_l, and the current of each loop is denoted as $\{i_1(t), i_2(t), \ldots, i_{Nl}(t)\}$. Then, the current $i_{c\_j}(t)$ $(j = 1, 2, \ldots, N_c)$ of each component is equal to $\sum_{p=1}^{N_l} k_{j\_p} i_p(t)$. k_{j_p} is equal to 1 or -1 if the jth component is in pth loop, and is equal to 0 if not. Likewise, the voltage relationships among different components $v_{c\_j}(t)$ $(j = 1, 2, \ldots, N_c)$ are also obtained as $\sum_{p=1}^{N_l} m_{j\_p} v_{c\_j}(t) = 0$, where m_{j_p} is also equal to 1 or -1 if the jth component is in pth loop, and is equal to 0 if not. When the circuit loop is one-to-one match in two PECs, their variables $i_p(t)$, k_{j_p}, and m_{j_p} are the same, and the current/voltage relationships among all components are totally the same in two PECs, i.e., they are equivalent. Hence, judging whether the circuit loops of PECs are one-to-one match can be set as the second rule of identification. As a summary, the two rules are listed in the following.

1) Rule 1: The number of different components is the same.
2) Rule 2: Circuit loops of PECs are one-to-one match.

With the proposed two rules, equivalent PECs can be identified by both manual effort and computer program in theory. However, it is difficult to quickly and precisely find all circuit loops in Rule 2 with manual effort in the practice, especially when the PECs are complicated. Therefore, this letter also aims to develop a computer-aided solution to accurately and conveniently complete the identification.

B. Computer-Aided Solution

The proposed computer-aided solution is illustrated in Fig. 2. Software Altium Designer is first employed to acquire the circuit information with its Netlist function, and then a MATLAB code is developed to automatically complete the data process,

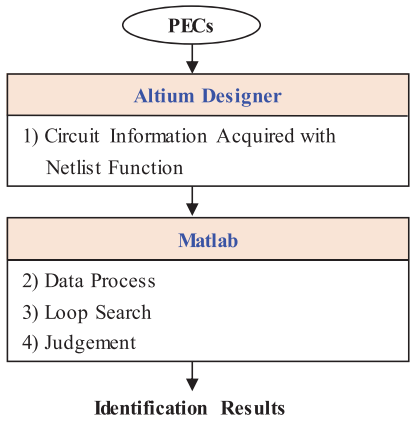

Fig. 2. Proposed computer-aided identification of equivalent PECs.

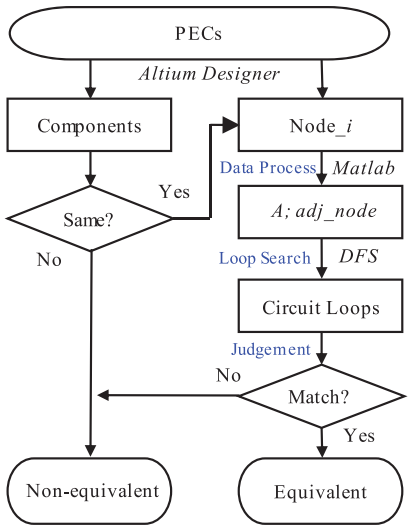

Fig. 3. Detailed process of proposed computer-aided identification.

circuit loop search, and final judgment. The aforementioned four aspects will be detailed in the following.

1) Circuit information acquired with Netlist function: By inputting the PEC into software Altium Designer, all components and their connecting relationships with nodes will be automatically listed by applying "Design » Netlist for Document » Protel," as shown in Fig. 3. If the components of different PECs are not totally the same, Rule 1 is obviously not satisfied and, thus, PECs are not equivalent. Otherwise, the acquired connecting components of each node Node_i = {a_i1, a_i2, …} (i = 1, 2,…, N) will be employed to further check whether Rule 2 is satisfied or not. According to the format in software Altium Designer, the element in each Node_i (i = 1, 2,…, N) contains a component symbol, a number symbol, and a port symbol. For example, the node ⑤ in Fig. 1(a) is connected with four components $\{C_1\ L_1\ R_1\ V_2\}$, and Node_5 = {C1-1 L1-2 R1-1 V2-2} is obtained. Port symbols "−1" and "−2," respectively, represent the two ports of components. These two symbols do not have any influence on the components without direction characteristics, such as resistor, capacitor, and inductor, whereas they represent

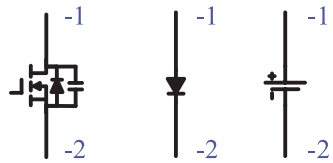

Fig. 4. Ports' denotation of components with direction characteristics.

the direction for other components including switch, diode, and voltage source, whose ports are denoted in Fig. 4. It is noteworthy that other software besides the Altium Designer can also be employed as long as they could achieve circuit information of PECs automatically. Software Altium Designer is chosen in this letter because it is easy to use and has been widely used by scholars and engineers.

2) Data process: With the acquired Node_i ($i = 1, 2, \ldots, N$) of a PEC, the connecting matrix A representing the connection of components among different nodes, will be further obtained in MATLAB after data process, as shown in Fig. 3. This step is a mimic of building the adjacency matrix in graph theory. At default, A is set as a $N \times N$ matrix with all zeros. If the pth element in Node_j and qth element in another Node_k satisfy $a\_jp = a\_kq \pm 1$, then there is a component connecting between node j and k, and $A(j, k) = A(j, k) +$ Component (a_jp), $A(k, j) = A(k, j) +$ Component (a_kq) are obtained. The function "Component (a_im)" is designed that it is equal to the component symbol and the number symbol of element a_im when it is nondirectional, e.g., Component (R1-1) = Component (R1-2) = R1. While it is equal to a combination of the component symbol, the number symbol and "pn" (or "np") when the component is directional and includes port symbol "-1" (or "-2"), e.g., Component (V1-1) = V1pn and Component (V1-2) = V1np. Moreover, based on the connecting matrix A, the adjacent nodes array $adj\_node(i)$ that lists all adjacent nodes connected with the node i is also calculated in MATLAB. It can be easily known that $adj\_node(i)$ includes node j if $A(i, j) \neq 0$.

3) Loop search: With the connecting matrix A and adjacent nodes array $adj\_node(i)$, all N nodes of the PEC will be one-by-one set as the root node to search its corresponding circuit loops based on the depth-first-search (DFS) algorithm in MATLAB. The DFS algorithm is very classic and well known to be used to traverse or search graph structures [12]. With simple modification, it is evolved to be able to find all circuit loops by one-by-one setting node i ($i = 1, 2, \ldots,$ or N) as the root node and exploring as far as possible along the adjacent nodes before backtracking. The condition of backtracking is that all the adjacent nodes in $adj\_node(i)$ of the current node have already been searched or the current node is the root node. If the backtracking condition is finding the root node and the number of searched nodes is larger than two, the circuit loop will be saved by multiplying the corresponding elements in the matrix A (e.g., the circuit loop ①②③① in Fig. 1(a) is saved as S1pn × D1np × V1np) and, then, the algorithm will backtrack. Otherwise, it will directly backtrack to the node that has nonsearched adjacent nodes, and then continue searching. It is noteworthy that once all the loop search of node i is finished, the connection matrix A needs to be updated by setting the values of ith row and column to zero before the search of next node $i + 1$, so that the repeated search of circuit loops containing node i will be avoided.

4) Judgment: After loop search of a PEC, all circuit loops will be added together, which is denoted as loop_sum. Furthermore, for each loop_sum, the number symbols of same type components would be exchanged to avoid its influence, and then an array Sum_Loop consisted of different loop_sum with all possible number symbols would be finally obtained for a PEC. Finally, PECs can be simply identified as equivalent or not by checking whether their Sum_Loop are equal or not.

TABLE I
COMPONENTS AND THEIR CONNECTING RELATIONSHIPS WITH NODES IN THE NETLIST OF ALTIUM DESIGNER

Fig. 1(a)	Components=[C1, D1, D2, L1, R1, S1, S2, V1, V2]
	Node_1= [S1-1 V1-1]
	Node_2= [D1-2 L1-1 S1-2]
	Node_3= [D1-1 D2-2 S2-2 V1-2]
	Node_4= [S2-1 V2-1]
	Node_5= [C1-1 L1-2 R1-1 V2-2]
	Node_6= [C1-2 D2-1 R1-2]
Fig. 1(b)	Components=[C1, D1, D2, L1, R1, S1, S2, V1, V2]
	Node_1= [S2-2 V2-2]
	Node_2= [S1-1 V1-1]
	Node_3= [D1-2 L1-1 S1-2]
	Node_4= [D2-1 L1-2 S2-1]
	Node_5= [C1-1 D2-2 R1-1]
	Node_6= [C1-2 D1-1 R1-2 V1-2 V2-1]

III. EXAMPLE CASE STUDY

Following the proposed computer-aided solution in the previous section, the identification of two PECs in Fig. 1(a) and (b) is taken as an example to be detailed introduced in this section. First, their components and the connecting components of each node Node_i ($i = 1, 2, \ldots, 6$) obtained in the software Altium Designer are summarized in Table I. According to Table I, the components are totally the same in these two PECs and thus Rule 1 is satisfied. Then, with the connecting information Node_i, the connection matrix of Fig. 1(a) and (b) is automatically obtained in the following equations with MATLAB code:

$$A_1 = \begin{array}{c} \\ ① \\ ② \\ ③ \\ ④ \\ ⑤ \\ ⑥ \end{array} \begin{array}{c} \begin{array}{cccccc} ① & ② & ③ & ④ & ⑤ & ⑥ \end{array} \\ \begin{bmatrix} 0 & \text{S1pn} & \text{V1pn} & 0 & 0 & 0 \\ \text{S1np} & 0 & \text{D1np} & 0 & \text{L1} & 0 \\ \text{V1np} & \text{D1pn} & 0 & \text{S2np} & 0 & \text{D2np} \\ 0 & 0 & \text{S2pn} & 0 & \text{V2pn} & 0 \\ 0 & \text{L1} & 0 & \text{V2np} & 0 & \text{C1+R1} \\ 0 & 0 & \text{D2pn} & 0 & \text{C1+R1} & 0 \end{bmatrix} \end{array}. \quad (1)$$

TABLE II
ADJACENT NODES ARRAY *adj_node* OF FIG. 1(A) AND (B) OBTAINED WITH MATLAB CODE

	Fig. 1(a)	Fig. 1(b)
adj_node(1)	{2, 3}	{4, 6}
adj_node(2)	{1, 3, 5}	{3, 6}
adj_node(3)	{1, 2, 4, 6}	{2, 4, 6}
adj_node(4)	{3, 5}	{1, 3, 5}
adj_node(5)	{2, 4, 6}	{4, 6},
adj_node(6)	{3, 5}	{1, 2, 3, 5}

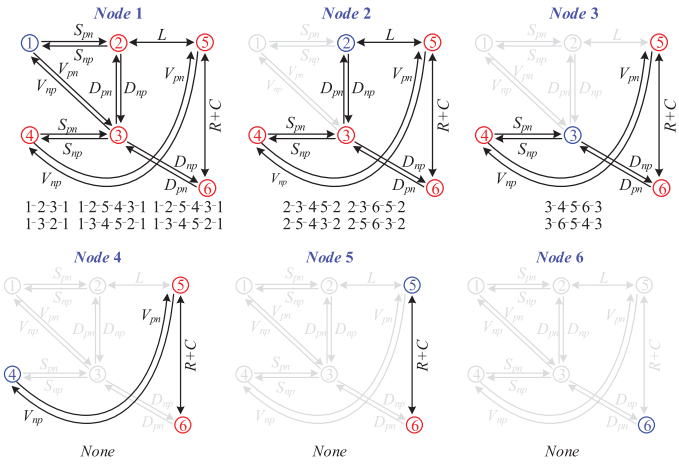

Fig. 5. Updated connection of the PEC in Fig. 1(a) before the search of each node.

$$A_2 = \begin{array}{c} \\ ① \\ ② \\ ③ \\ ④ \\ ⑤ \\ ⑥ \end{array} \begin{array}{c} \begin{array}{cccccc} ① & ② & ③ & ④ & ⑤ & ⑥ \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & \text{S2np} & 0 & \text{V2np} \\ 0 & 0 & \text{S1pn} & 0 & 0 & \text{V1pn} \\ 0 & \text{S1np} & 0 & \text{L1} & 0 & \text{D1np} \\ \text{S2pn} & 0 & \text{L1} & 0 & \text{D2pn} & 0 \\ 0 & 0 & 0 & \text{D2np} & 0 & \text{C1+R1} \\ \text{V2pn} & \text{V1np} & \text{D1pn} & 0 & \text{C1+R1} & 0 \end{bmatrix} \end{array}. \quad (2)$$

Besides, the array *adj_node* of Fig. 1(a) and (b) are also correspondingly achieved in Table II.

Then, the loop search of PECs will also be implemented in MATLAB. Taking the PEC in Fig. 1(a) as an example, the connection of searching each node is illustrated in Fig. 5. When first node ① is set as the root node, the connection is the same as the matrix A_1 in (1). Before the search of node ②, the node ① and its connecting components will be deleted, which is also similarly implemented before the search of the remaining nodes ③, ④, ⑤, and ⑥. Besides, the detailed circuit loop search of node ① in Fig. 1(a) is also shown in Fig. 6. Node ① is set as the root node. Then, according to *adj_node* (1) = {2, 3}, node ② is searched in the first procedure, and in the second procedure, Node ① is found according to *adj_node* (2) = {1, 3, 5}. Because node ① is the root node and the number of searched nodes is

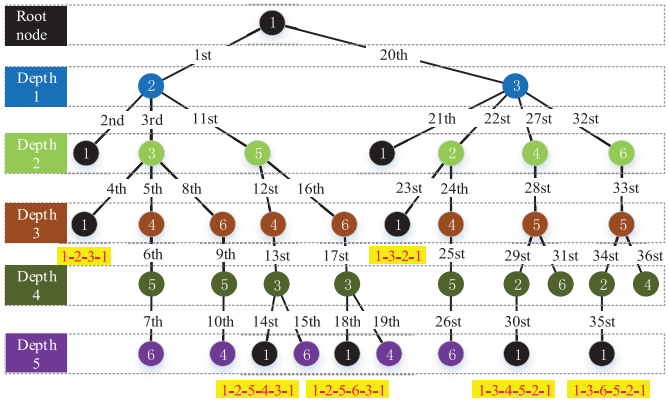

Fig. 6. Circuit loop search of node ① in Fig. 1(a) based on DFS.

only two, the circuit loop ①-②-① will not be saved and the algorithm will backtrack to node ②. In the next two procedures, the node ③ will be searched, and then, the node ① will be found again according to *adj_node* (3) = {1, 2, 4, 6}. Because the searched loop ①-②-③-① has three nodes, it is saved and, then, the algorithm backtracks to node ③. The following searching procedures are similar, and other five loops ①-②-⑤-④-③-①, ①-②-⑤-⑥-③-①, ①-③-②-①, ①-③-④-⑤-②-①, and ①-③-⑥-⑤-②-① are also found for node ①, as shown in Fig. 6. Likewise, for other nodes ②-⑥, their loops can also be searched, but with the updated connection in Fig. 5. Four loops ②-③-④-⑤-②, ②-③-⑥-⑤-②, ②-⑤-④-③-②, and ②-⑤-⑥-③-② are found for node ②, two loops ③-④-⑤-⑥-③ and ③-⑥-⑤-④-③ are found for node ③, and no loop is found when the rest nodes ④, ⑤, and ⑥ are set as root node. Besides, according to the connection matrix A in (1), these 12 found loops will be, respectively, saved as S1pn × D1np × V1np, S1pn × L1 × V2np × S2pn × V1np, S1pn × L1 × (R1+C1) × D2pn × V1np, V1pn × D1pn × S1np, V1pn × S2np × V2pn × L1 × S1np, V1pn × D2np × (R1 + C1) × L1 × S1np, D1np × S2np × V2pn × L1, D1np × D2np × (R1 + C1) × L1, L1 × V2np × S2pn × D1pn, L1 × (R1 + C1) × D2np × D1np, S2np × V2pn × (C1 + R1) × D2pn, and D2np × (R1 + C1) × V2np × S2pn.

Likewise, the loop search of the PEC in Fig. 1(b) can also be similarly completed after the aforementioned process. It also has 12 loops: S2np × L1 × S1np × V1pn × V2pn, S2np × L1 × D1np × V2pn, S2np × D2pn × (R1 + C1) × V2pn, V2np × V1np × S1pn × L1 × S2pn, V2np × D1pn × L1 × S2pn, V2np × (R1 + C1) × D2np × S2pn, S1pn × L1 × D2pn × (R1 + C1) × V1np, S1pn × D1np × V1np, V1pn × (R1 + C1) × D2np × L1 × S1np, V1pn × D1pn × S1np, L1 × D2pn × (R1 + C1) × D1pn, D1np × (R1 + C1) × D2np × L1}. With the circuit loops of PECs in Fig. 1(a) and (b), their loop_sum will be calculated by adding all the circuit loops and then the Sum_Loop is accordingly obtained through exchanging the number symbol of loop_sum. The Sum_Loop is totally the same in these two PECs. Therefore, Rule 2 is also satisfied, and hence, it can be judged that these two PECs are equivalent. In order to achieve a better understanding, the unidirectional 12 loops of PECs in Fig. 1(a) and (b), which are

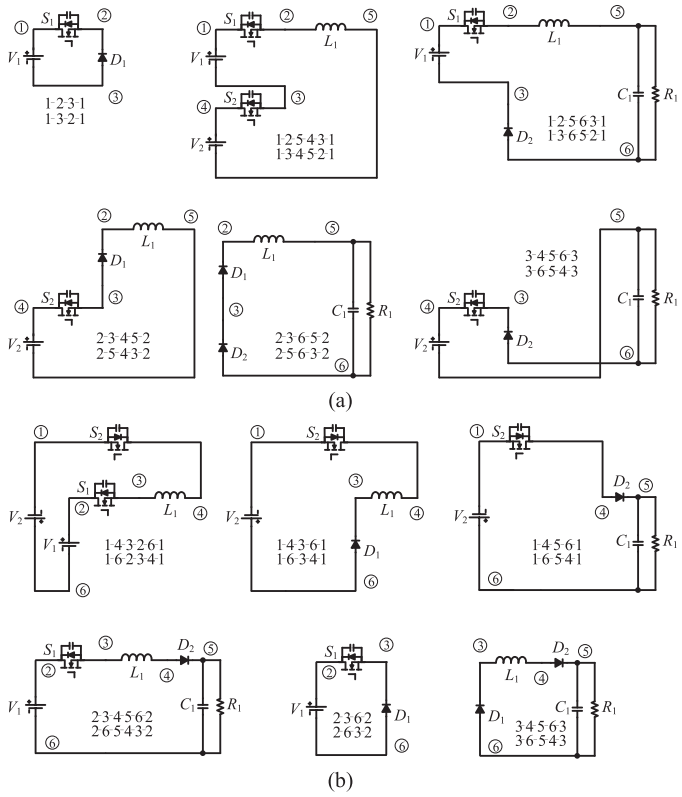

Fig. 7. Six bidirectional circuit loops of two PECs in Fig. 1. (a) First. (b) Second.

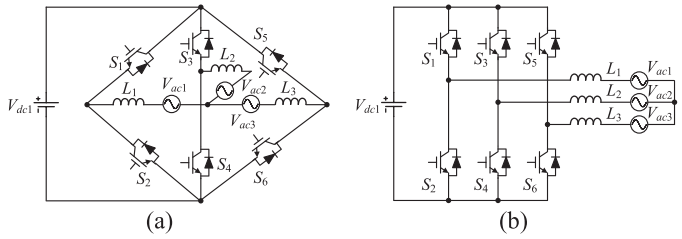

Fig. 8. Two equivalent nonplanar circuits. (a) First. (b) Second.

integrated as 6 bidirectional loops, are also shown in Fig. 7(a) and (b), respectively.

From above, with the proposed computer-aided method, the identification of equivalent PECs in Fig. 1(a) and (b) can be easily and precisely completed. Because most of the identification process was automatically implemented by software Altium Designer and MATLAB, it is also effective for more complicated PECs. For example, the two nonplanar circuits in Fig. 8(a) and (b), which are easily mistaken as different PECs but actually the same, can also be conveniently identified with the proposed method that their components are the same and the 48 loops are one-to-one match.

IV. Conclusion

In summary, this letter figures out two rules and proposes a computer-aided method to identify the equivalent PECs. Unlike the conventional method that is totally dependent on the manual effort, the proposed method completes most identification process with the aid of software Altium Designer and MATLAB, which is therefore easier and more rigorous. The Netlist function of software Altium Designer is utilized to obtain the circuit information of PECs, and then a MATLAB code including data processing and DFS algorithm is developed to automatically judge whether the PECs are equivalent or not. In order to gain a comprehensive understanding, detailed identification of two example equivalent DISO converters is introduced in this letter. Thanks to the computerization, the proposed method is also effective for the identification of more complicated PECs to avoid the undesired repeated research.

References

[1] Y. C. Liu and Y. M. Chen, "A systematic approach to synthesizing multi-input dc-dc converters," *IEEE Trans. Power Electron.*, vol. 24, no. 1, pp. 116–127, Jan. 2009.

[2] W. Ki and D. Ma, "Single-inductor multiple-output switching converters," in *Proc. 32nd IEEE Annu. Power Electron. Spec. Conf.*, 2001, pp. 226–231.

[3] G. Chen, Z. Jin, Y. Liu, Y. Hu, J. Zhang, and X. Qing, "Programmable topology derivation of integrated three-port dc-dc converters with reduced switches for low-cost applications," *IEEE Trans. Ind. Electron.*, to be published.

[4] Y. Li, X. Ruan, D. Yang, F. Liu, and C. K. Tse, "Synthesis of multiple-input dc/dc converters," *IEEE Trans. Power Electron.*, vol. 25, no. 9, pp. 2372–2385, Sep. 2010.

[5] H. Wu, K. Sun, S. Ding, and Y. Xing, "Topology derivation of nonisolated three-port dc–dc converters from DIC and DOC," *IEEE Trans. Power Electron.*, vol. 28, no. 7, pp. 3297–3307, Jul. 2013.

[6] M. R. Mohammadi and H. Farzanehfard, "A new family of zero-voltage-transition nonisolated bidirectional converters with simple auxiliary circuit," *IEEE Trans. Power Electron.*, vol. 63, no. 3, pp. 1519–1527, Mar. 2016.

[7] T. Wu, "Decoding and synthesizing transformerless PWM converters," *IEEE Trans. Power Electron.*, vol. 31, no. 9, pp. 6293–6304, Sep. 2016.

[8] R. Loera-Palomo and J. A. Morales-Saldaña, "Family of quadratic step-up dc-dc converters based on non-cascading structures," *IET Power Electron.*, vol. 8, no. 5, pp. 793–801, May 2015.

[9] J. Zhang, C. Zhao, S. Zhao, and X. Wu, "A family of single-phase hybrid step-down PFC converters," *IEEE Trans. Power Electron.*, vol. 32, no. 7, pp. 5271–5281, Jul. 2017.

[10] J. Kim and G. Moon, "Derivation, analysis, and comparison of nonisolated single-switch high step-up converters with low voltage stress," *IEEE Trans. Power Electron.*, vol. 30, no. 3, pp. 1336–1344, Mar. 2015.

[11] Y. Li, L. Ding, and Y. Li, "Isomorphic relationships between voltage-source and current-source converters," *IEEE Trans. Power Electron*, to be published.

[12] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. Cambridge, MA, USA: MIT Press, 2001.