A fast Selection Algorithm Based on Binary Numbers for Capacitor Voltage Balance in Modular Multilevel Converter

Tao Wang, Hua Lin, Zhe Wang, Yajun Ma, Xingwei Wang
State Key Laboratory of Advanced Electromagnetic Engineering and Technology
Huazhong University of Science and Technology, Wuhan, China
M201671360@hust.edu.cn

***Abstract*—Considering the capacitor voltage balance (CVB) in modular multilevel converter (MMC), classical complete sorting algorithms, especially used in MMC with a large number of submodules (SMs), usually result in too much computational load and resource consumption. A fast selection algorithm based on binary numbers is proposed to achieve CVB in MMC. The algorithm needs not fully sorting all the numbers, and the complexity of it rapidly decreases by using a binary division mechanism. These properties make the proposed algorithm fast and resource-saving, and easily being implemented in many types of microcontrollers. Taking into account the needs of engineering applications, some techniques used to improve algorithm speed and reduce Filed Programmable Gate Array (FPGA) resource consumption are discussed when implementing algorithms in FPGA. The required resources and execution time of the proposed algorithm is evaluated, and a comparison between the proposed algorithm and some other sorting algorithms is used to verify the conclusions.**

Keywords—modular multilevel converter; capacitor voltage balancing; sorting algorithm; field programmable gate array;

I. INTRODUCTION

Modular Multilevel Convertor (MMC) has been received an increasingly concern in recent years. MMC has many advantages compared with the conventional voltage source convertors, such as scalability, high modularity, low switching losses, low Total Harmonic Distortion (THD), high redundancy and high reliability. Thanks to those advantages, MMC has become a promising solution in High Voltage Direct Current (HVDC) [1]-[4].

Despite of the advantages above, one of the most critical problems in MMC is the capacitor voltage imbalance. By using Sorting Algorithms (SAs), an effective Capacitor Voltage Balance (CVB) method is proposed in [5]-[7]. The approach usually adopted in Near Level Control (NLC), and doesn't need capacitor voltage control loop. The generated pulses are reassigned to SMs according to the sorting results of the capacitor voltage and the arm current direction. This approach is effective in CVB, does not require tuning control parameters and is decoupled from other control loops, especially used in MMC with a large number of SMs. Therefore, many kinds of SAs optimized for a much more effective implementation in MMC are presented in literature. Two parallel Sorting Networks (SNs) and a Capacitor Voltage Mapping Strategy (CVMS) are introduced in [8] and [9] respectively, which is suitable to implement in FPGA. However, with a significantly increased number of SMs in HVDC applications, these SNs will consume too much resources and are unable to be used actually. Based on a special hardware design, "the Tortoise and the Hare" sorting method shows fast response in [10], but its application is limited. A dynamic grouping sorting method is proposed in [11], which can reduce the computational burden and the switching frequency at the same time, and the performance of the algorithm is good from the simulation results. Several SM selection methods have been proposed in the literature with different features [12]–[16]. Each particular method has its own advantages, can either reduce the voltage ripple on the capacitors of the modules, reduce the switching frequency, increase the converter efficiency. However, these methods primarily aim to solve other issues and computational problem still exists in the applications of a large number of SMs such as HVDC.

Completely sorting all the capacitor voltage data makes the classic sorting algorithms execute very slowly, but it is not required in the applications of MMC. We only need to select a certain number of SMs with the maximum or minimum voltage, the order between the selected data and the order between the remaining data are useless. In many implementations, unnecessary steps have been taken to get unnecessary results. For this reason, a new Capacitor Voltages Dividing and Selecting Algorithm (CVDSA) based on the binary numbers is proposed in this paper. Compared with other sorting algorithms, this algorithm consumes less resources, executes fast, and the result is absolutely accurate. As a result, CVDSA can be easily implemented in low-cost microprocessors, such as, FPGA and Digital Signal Processors (DSPs) devices.

The structure of the article is as follows. An overview on MMC and the basic principles of the NLC are given in section II. In section III, the basic principles of CVDSA are discussed in detail. For a good practical implementation, some of the skills on the FPGA realization is talked in section IV. The required resources and the execution time of CVDSA are estimated in section V, and an implementation in FPGA is introduced to verify efficiency and correctness of CVDSA as well. Finally, a comparison among several different sorting algorithms is drawn to verify the conclusions in section VI.

This work was supported by the National Natural Science Foundation of China under Grant 51741703.

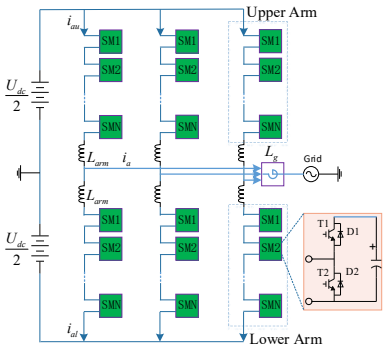


Fig. 1. Schematic presentation of a grid connected MMC.

II. Basic Operation of MMC

A schematic representation of a three-phase grid connected MMC is shown in Fig. 1. It consists of an upper arm and a lower arm per phase-leg, where each arm composed by N series connected SMs and an arm inductor Larm [17]. The half-bridge structure is the most commonly used SM structure because of its simple structure, in which, the floating capacitor is inserted or bypassed by controlling the two switches to generate different voltage levels at the AC output port [3].

Basically, there are two popular modulation strategies with different CVB approaches for MMC. The one called Carrier Phase-Shifted PWM (CPS-PWM) is an individual modulation method, and the another called NLC is a global arm modulation method. More specifically, the former assigns each SM with an individual PWM pulse. The CVB of each arm is achieved by adding a closed-loop capacitor voltage control for each SM. Since the capacitor voltage of each SM is different during the MMC operation, the reference of PWM is adjusted to generate individual pulses. The main drawback of this approach is the tuning of controller parameters, which makes the control system complex. The response characteristics of the system vary depending on the controller parameters, and an inappropriate controller parameter may cause the whole system to be unstable. However, the latter takes each arm as a control unit, and the number of inserted SMs in each arm is decided by the top closed-loop controller and the NLC modulation strategy. When it comes to each arm, the insertion or bypass of the SMs in the arm are determined by the capacitor voltage sorting result and the arm current direction. In this case, the CVB in each arm can be achieved always, no close control loop is needed, and the best decision, inserts the SMs with the maximum or the minimum capacitor voltage, also makes the method robust. The NLC modulation strategy has another advantage that the switch frequency can be decreased dramatically under some optimized sorting and selecting strategies [11] [12] [14-16], However, the SMs using CPS-PWM have the unchanged switch frequency. According to the NLC strategy, the MMC control schema is depicted in Fig. 2. The top output power controller calculates the basic voltage reference for each phase, then an extra control component generated by the circulating current controller and a DC bias voltage are added to this reference. The insertion indices n is then calculated using the NLC strategy [18]. The insertion indices for the upper and lower arm n^*_{ku} and n^*_{kl}, respectively, are send to the CVB control module along with the

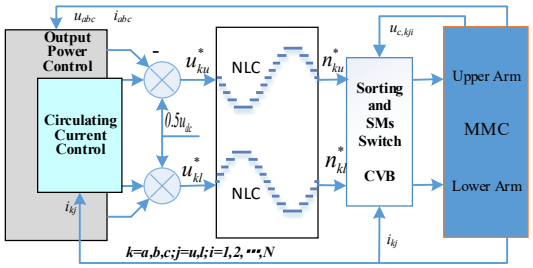


Fig. 2. A control diagram of MMC using NLC.

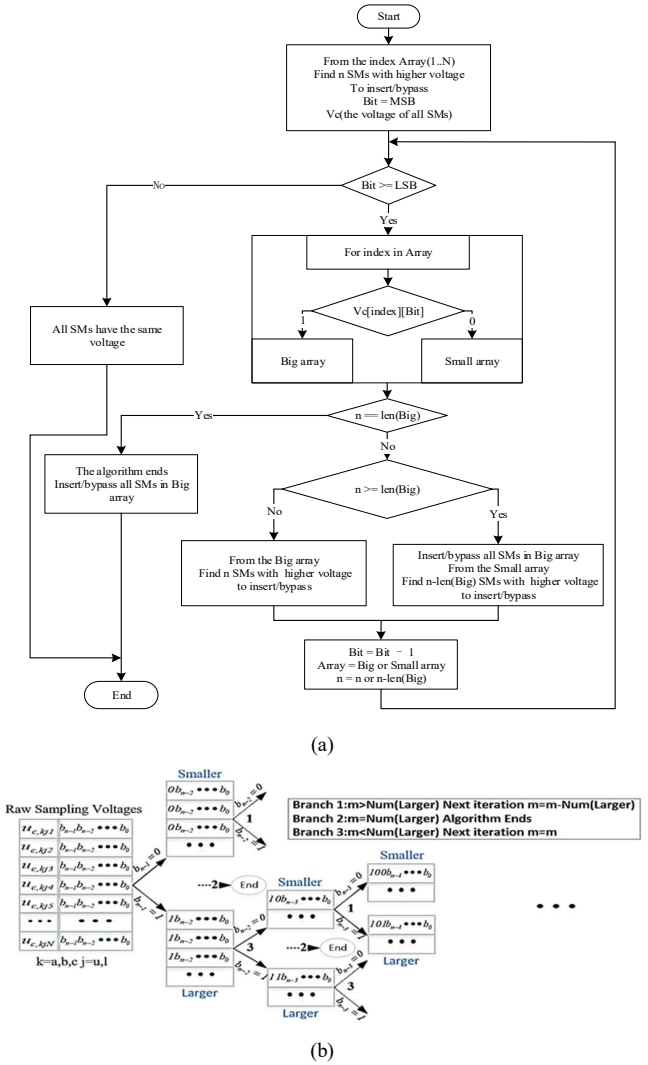
sampling capacitor voltages $u_{c,kji}$ and the measured arm current i_{kj}. The CVB controller consists of the sorting method and the SM switching pulses generator module, which selects the proper SMs according the sorting result and reassigns the gate signals to driver circuits. The sorting method is a key component in CVB since it occupies almost all the computational load, especially, in MMC with a large number of SMs where the heavy computational load cannot be completed in a control period. In this paper, we focus on reducing the computational complexity of the sorting method and achieving a resource-saving and fast implementation of the proposed algorithm in FPGA from the perspective of engineering applications.

III. Basic Principles of the Proposed Algorithm

The two basic principles of CVDSA, reduces the computational load, are listed below.

1. A group of binary numbers can be divided into two groups according to the value of a specific bit with different weights, one group is smaller in numerical value, and another is much larger.
2. The nonconforming data should be discarded from the sorting algorithm as soon as possible to reduce the computational load.

Based on the above two principles, considering the positive arm current, m SMs with the lowest capacitor voltages should be selected from the N SMs in one arm according to the sampled capacitor voltages. The process of CVDSA is described as follow, one bit is considered in each iteration, and from the most significant bit (MSB) to the least significant bit (LSB) of the sampled capacitor voltage binary numbers, firstly, use the principle 1 to divide the sampled data, secondly, according to the principle 2, if the number of binary numbers in the larger number group, expressed as $Num(Larger)$, is greater than m, then we find the m SMs in the larger number group and discard all the items in the smaller number group; if equal, then the question is solved immediately; and if less than, then we find the rest of $m - Num(Larger)$ SMs in the smaller number group, the SMs corresponding to the items in the larger number group are selected and all the items in the larger number group are discarded from the sorting algorithm as well. The size of CVDSA in each iteration rapidly decreases through the division mechanism. A schematic description of CVDSA is given in Fig. 3, where each iteration is the same but in a much smaller size. As a result, the CVDSA proposed in this paper is faster than the classical sorting algorithms.



(a)

(b)

Fig. 3. The description of the CVDSA. (a) the flow chart of the CVDSA. (b) The Schematic diagram of the proposed algorithm.

In order to describe the proposed algorithm more clearly, a simple example is given in Fig. 4. In this example, there are 8 SMs in each arm, the bit length of capacitor voltages is 4, and supposed that 5 SMs with the highest voltage will be inserted in. The two tables in Fig. 4(a) show the sampling result and the complete sorting result respectively, and the whole workflow of CVDSA is presented in Fig. 4(b). Four qualified SMs are selected in the first iteration, and the last one in third iteration. Compared with Fig. 4(a), we can see that the selected SMs are indeed the five SMs with the maximum voltage and much fewer operation steps are required.

SM Index	Voltage
0	1001
1	0101
2	0110
3	1010
4	1100
5	1101
6	0011
7	0001

Sort

SM Index	Voltage
7	0001
6	0011
1	0101
2	0110
0	1001
3	1010
4	1100
5	1101

(a)

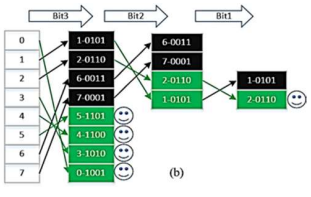


Fig. 4. The diagram of the simple example. (a) the raw sampling data and sorting result; (b) The workflow of CVDSA in the example.

Memory usage and execution time are usually considered to evaluate the performance of an algorithm [19]. For memory usage, the algorithm needs one array to store the raw sampling data of capacitor voltages, which takes up nN bits, and two identical arrays stored indexes of the SMs in each arm are used to operate CVDSA, which consumes $2Nceiling(Log_2N)$ bits. Compared with the classic selection sorting algorithm, only one more index array is used, and the total needed memory (M_{CVDSA}) in bits is expressed as (1).

$$M_{CVDSA} = N(2ceiling(Log_2N) + n) \quad (1)$$

Regarding the execution time, the maximum number of total iteration steps is limited to n, where n is the bit length of the sampled capacitor voltages, and this limits the time complexity of CVDSA to $O(N)$, where the big O notation describes the limiting behaviors of a function. In the worst case, all the SMs have the same voltages, the algorithm requires a total of nN steps, which means a linear relationship is established between the execution time and the number of SMs.

IV. SOME USEFUL TECHNIQUES TO IMPLEMENTATE CVDSA IN FPGA

In general, FPGA is suitable for implementing algorithms with the parallel structure. However, that is not always true, because we can easily implement a complex soft processor in the FPGA. Such as the Nios® II processor, it is the widely used soft processor in the FPGA industry. In other words, FPGA is also suitable for implementing the serial structure algorithms, if handled properly, achieving serial structure algorithms in FPGA have more advantages over the microcontroller or DSP implementations. For example, in section III, writing a SM index from one index array to another index array is taken as one logical step, however, it consists of three steps (read-read-write) in total, read the SM index from the first index array, read the capacitor voltage from the capacitor voltage array according to the read index, and write the SM index to the second index array according to the specific bit of the read capacitor voltage. It can be done in one clock period using the pipeline and delay-chain technology in FPGA, however, the same result is difficult to achieve in the microprocessor or DSP, because their functions are fixed and cannot be optimized for our specific programs. So, we can implement algorithms more efficiently due to the flexibility of FPGA, and there are basically two considerations when implementing more efficient algorithms in FPGA.

A. Using the embedded memory units of FPGA

For a successful routing in the target FPGA, adequate resource margins must be ensured. This allows commercial synthesis software, such as Quartus® II, to make better layout and optimization for our programs easily, and much more resource margins are needed for a faster implementation. As a result, saving resource is always taken into account when implementing algorithms in FPGA. For that reason, we should always store the algorithm data using the embedded memory units of FPGA, rather than registers in the Logic Cells (LCs). There is only one register in each LC usually. Almost all of the algorithms need to deal with large amounts of data. In this way, we can always save resources in FPGA. With this in mind, four arrays discussed in the section III are implemented using the

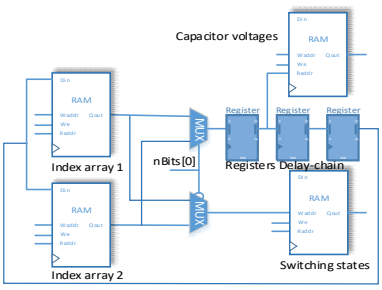


Fig. 5. the storage structure of CVDSA using 4 RAMs.

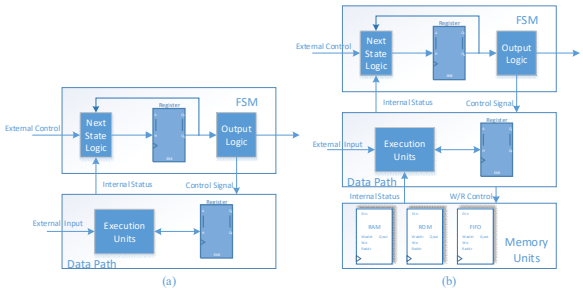


Fig. 6. Improve FSMD. (a) A simple schematic of FSMD; (b) a schematic of the three-tier FSMD by splitting Data-Path into two parts.

embedded memory units as four RAMs, and the physical connections among them is depicted as Fig. 5. A registers delay-chain is used to achieve the three steps (read-read-write) in one clock period. The read and write control of the four connected RAMs is the remaining part of CVDSA, which just consumes a small amount of the registers in LCs.

B. *Using a clear-thinking three-tier FSMD*

A FSMD is a universal specification model that can represent all hardware designs. The FSMD model was first proposed by Gajski in [20]. In general terms, the FSMD uses a FSM to process the algorithm data. The classical schematic of the FSMD is given in Fig. 6(a), and it is a two-tier structure. However, in the design process, the data path is still too complicated to design the algorithm clearly. In practical applications, we divide the data path into two parts to form a three-tier FSMD, as shown in Fig. 6(b). In the design of an algorithm, we first determine a reasonable storage structure of the algorithm, just as shown in Fig. 5, and the registers in the data path are used to control the Read/Write behaviors of these memories, which basically includes Read/write addresses control and Read/Write enable signals control. The FSM in the top layer controls the Data-Path layer, and is separated from the Memory Units layer. Under this design framework, the design process is clear and simple, the realization of efficient algorithms is easy, and it is beneficial to the commercial synthesis software for a further optimization of algorithms.

V. Evaluation and Verification of the Proposed Algorithm

Some examples are given in this section to verify the effectiveness of the two techniques proposed in section IV and the high efficiency of the proposed CVDSA.

A. *Verify the effectiveness of proposed techniques*

A design example using the above design skills is implemented in a low-cost Altera® Cyclone IV FPGA (EP4CE10E22C8N), where the number of SMs in one arm is 256, and the bit length of capacitor voltage is 12. The compilation report in Quartus® II is given in Table I, as we can see, the resource consumption is limited at a low level, and the clock frequency is high because the memory block performance specifications given in the datasheet of Altera® Cyclone IV FPGA is 238MHz. As the number of SMs increases, only the memory unit consumption will increase and the Logic Cell consumption remains unchanged basically. From the results of this example, we can easily implement CVDSAs of the total six arms in parallel in a low-cost FPGA and there are still adequate resources remains for other functions of the control system, such as data sampling, pulses allocation and communications. By the way, the CVDSA can also be used for fixed-point and floating-point numbers in the same way.

Table. I. The compilation report of the given example

Logic Cells(LCs)	LUT-Only LCs	146	260/10320 (2.52%)
	LUT/Register LCs	107	
	Register-Only LCs	7	
Memory Bits		7424/423936 (1.75%)	
Clock Frequency (the worst case)		202MHz	

B. *Verify the high efficiency of CVDSA*

Table II shows the full comparison among the selection sorting algorithm, the proposed CVDSA, and the CVMS in [9], where the t_{clk} is the period of the operation clock. Compared with the selection algorithm, the CVDSA is close in the memory consumption, but takes a big advantage in the execution time. Compared with the CVMS, the CVDSA consumes much less memory because of the M may be 64 to ensure the results are accurate enough. Although the CVDSA requires more execution steps, the clock frequency can be four times higher than the CVMS since the CVDSA is simple in FPGA implementation. According to an example given in [9], with the same number of SMs, the execution time of CVDSA is much less.

The whole control system using NLC approach combined with SA strategy is depicted in Fig. 7. Due to the complexity of the MMC control system, FPGA is often used as a phase controller to assist the top DSP controller. Since the raw sampling data are directly acquired by FPGAs, implementing SA in FPGAs is more efficient and flexible, and greatly reduces the data exchange between the two controllers.

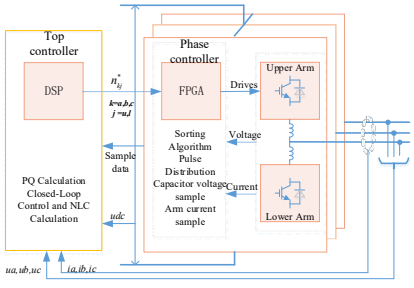


Fig. 7. The whole MMC control system using NLC and SA

Table. II. A full comparison among selection sort algorithm, the proposed CVDSA and the CVMS in [9].

		Selection Sort	The proposed CVDSA	The CVMS in [9]
Time	1) store data	Nt_{clk}	Nt_{clk}	$3Nt_{clk}$
	2) sort data	$0.5N(N+1)t_{clk}$	nNt_{clk}	
	3) output result	Nt_{clk}	Nt_{clk}	Nt_{clk}
Memory		$N(ceiling(Log_2N)+n)$	$N(2ceiling(Log_2N)+n)$	$N(Mceiling(Log_2N)+n)$
Accurate result		Yes	Yes	No

Under this control system, the selection sorting algorithm and the proposed CVDSA with different numbers of SMs are implemented in a low-cost Altera® Cyclone IV FPGA. The two algorithms end with generating a pulse, and the time between two adjacent pulses is the actual measured execution time, which is depicted as Fig. 8. In the implementations, the algorithms operate in the worst case with a 200 MHz clock frequency, the bit length of sampling capacitor voltages is 12, and the numbers of SMs are 128, 256, 512, 1024 respectively. Fig. 9 shows the measured results. Clearly, a linear fitted curve indicates that the time complexity of CVDSA is $O(N)$. However, the execution time of the selection SA exponentially increases in function to the number of SMs, which highlights the improvement obtained by using the proposed CVDSA. Moreover, the proposed algorithm, even applied in MMC with 1024 SMs, can finish in $70\mu S$, but the execution time of the selection SA is $2657.25\mu S$ in the same case. As a result, the proposed algorithm completely meets the requirement of the actual MMC control system, even in MMC with a large number of SMs.

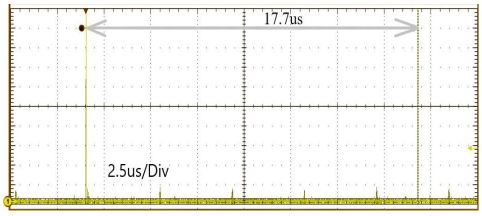


Fig. 8. The waveform of measuring execution time of SAs.

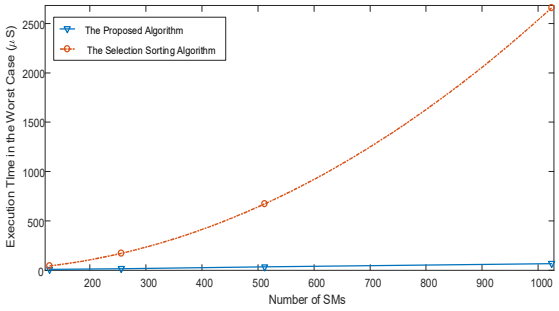


Fig. 9. Measured execution time and fitted curves of the proposed CVDSA and the selection sorting algorithm under different numbers of SMs.

CONCLUSIONS

In this paper, a fast selection algorithm based on binary numbers for CVB is proposed in MMC applications. The calculation load of the proposed algorithm decreases rapidly through a binary division mechanism, and no need for completely sorting all the data reduces the time complexity of the proposed algorithm to $O(N)$. These features make the proposed algorithm very useful and easy to be implemented in MMC with a large number of SMs. Some useful techniques are also discussed to reduce resource consumption and improve algorithm efficiency when implementing algorithms in FPGA. Of course, an example implemented in the Altera® Cyclone IV FPGA using those discussed skills is given to prove the high efficiency of the proposed algorithm, and the execution time of the algorithm in the worst case is much lower than the sampling period usually used in MMC applications. What's more, the result of the algorithm is absolutely accurate, same with the results of the classic sorting algorithm. In conclusion, the proposed resource-saving algorithm is high efficient as well, and it greatly achieves a compromise between the resource usage and the execution time.

REFERENCES

[1] M. Saeedifard, and R. Iravani, "Dynamic performance of a modular multilevel back-to-back HVDC system," IEEE Trans. Power Del., vol. 25, no. 4, pp. 2903–2912, Oct. 2010.

[2] U. N. Gnanarathna, A. M. Gole, and R. P. Jayasinghe, "Efficient modeling of modular multilevel HVDC converters (MMC) on electromagnetic transient simulation programs," IEEE Trans. Power Del., vol. 26, no. 1, pp. 316–324, Jan. 2011.

[3] S. Debnath, J. Qin, B. Bahrani, M. Saeedifard, and P. Barbosa, "Operation, control, applications of the modular multilevel converter: A review," IEEE Trans. Power Electron., vol. 30, no. 1, pp. 37–53, Jan. 2015.

[4] J. Pou, S. Ceballos, G. Konstantinou, V. Agelidis, R. Picas, and J. Zaragoza, "Circulating current injection methods based on instantaneous information for the modular multilevel converter," IEEE Trans. Ind. Electron., vol. 62, no. 2, pp. 777–788, Feb. 2015.

[5] M. Glinka and R. Marquardt, "A new ac/ac multilevel converter family," IEEE Trans. Ind. Electron., vol. 52, no. 3, pp. 662–669, Jun. 2005.

[6] F. Deng and Z. Chen, "A control method for voltage balancing in modular multilevel converters," IEEE Trans. Power Electron., vol. 29, no. 1, pp. 66–76, Jan. 2014.

[7] K. Wang, Y. Li, and Z. Zheng, "Voltage balancing control and experiments of a novel modular multilevel converter," in Proc. IEEE ECCE, 2010, pp. 3691–3696.

[8] Mattia Ricco, L. Mathe, and R. Teodorescu. "FPGA-based implementation of sorting networks in MMC applications." European Conference on Power Electronics and Applications 2016:1-10.

[9] Mattia Ricco, Laszlo Mathe, and Remus Teodorescu. "New MMC capacitor voltage balancing using sorting-less strategy in nearest level control." 2016 IEEE Energy Conversion Congress and Exposition (ECCE) IEEE, 2016:1-8.

[10] D. Siemaszko, “Fast sorting method for balancing capacitor voltages in modular multilevel converters,” IEEE Trans. Power Electron., vol. 30, no. 1, pp. 463–470, Jan. 2015.

[11] Fei, C., Yang, X., Jia, H., Yu, L., Cao, F., & Shukai, X. U. (2015). A capacitor voltage balance control algorithm for modular multilevel converter suitable to field programmable gate array. Power System Technology.

[12] Antonioferreira, Abel, and Oriol Gomisbellmunt. "Comparison of cell selection methods for modular multilevel converters." international conference on environment and electrical engineering (2016): 1-8.

[13] Q. Tu, Z. Xu, and L. Xu, “Reduced Switching-frequency modulation and circulating current suppression for modular multilevel converters,” IEEE Tran. on Power Delivery, vol. 26, no. 3, pp. 2009–2017, 2011.

[14] A. Hassanpoor, A. Roostaei, S. Norrga, and M. Lindgren, “Optimization Based Cell Selection Method for Grid-Connected Modular Multilevel Converters,” IEEE Transactions on Power Electronics, vol. 31, no. 4, pp. 2780–2790, apr 2016.

[15] A. Hassanpoor, A. Nami, and S. Norrga, “Tolerance Band Adaptation Method for Dynamic Operation of Grid-Connected Modular Multilevel Converters,” IEEE Transactions on Power Electronics, vol. 8993, no. 99, pp. 1–10, 2016.

[16] A. Hassanpoor and K. Ilves, “Tolerance-band modulation methods for modular multilevel converters,” Power Electronics, IEEE Transactions on, vol. 30, no. 1, pp. 311–326, 2015.

[17] M. A. Perez, S. Bernet, J. Rodriguez, S. Kouro, and R. Lizana, “Circuit Topologies , Modeling , Control Schemes , and Applications of Modular Multilevel Converters,” IEEE Trans. Power Electron., vol. 30, no.1, pp. 4–17, 2015.

[18] P. M. Meshram and V. B. Borghate, “A Simplified Nearest Level Control (NLC) Voltage Balancing Method for Modular Multilevel Converter (MMC),” IEEE Trans. Power Electron., vol. 30, no. 1, pp. 450–462, 2015.

[19] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, Introduction to algorithm, MIT Press, Cambridge, 2009.

[20] D. Gajski and L. Ramachandran, “Introduction to high-level synthesis,” IEEE transactions on Design and Test of Computers, pp. 44–54, 1994.