IEEE Power and Energy Technology Systems Journal

Received 6 October 2018; accepted 12 November 2018. Date of publication 19 November 2018; date of current version 25 March 2019.

Digital Object Identifier 10.1109/JPETS.2018.2881483

Parallel High-Fidelity Electromagnetic Transient Simulation of Large-Scale Multi-Terminal DC Grids

NING LIN (Student Member, IEEE), AND VENKATA DINAVAHI (Senior Member, IEEE)

Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 2V4, Canada

CORRESPONDING AUTHOR: N. LIN (ning3@ualberta.ca)

This work was supported by the Natural Science and Engineering Research Council of Canada (NSERC).

ABSTRACT Electromagnetic transient (EMT) simulation of power electronics conducted on the CPU slows down as the system scales up. Thus, the massively parallelism of the graphics processing unit (GPU) is utilized to expedite the simulation of the multi-terminal DC (MTDC) grid, where detailed models of the semiconductor switches are adopted to provide comprehensive device-level information. As the large number of nodes leads to an inefficient solution of the DC grid, three levels of circuit partitioning are applied, i.e., the transmission line-based natural separation of converter stations, splitting of the apparatus inside the station, and the coupled voltage-current sources for fine-grained partitioning. Components of similar attributes are written as one CUDA C function and computed in massive parallelism by means of single-instruction multi-threading. The GPU's potential as a new EMT simulation platform for the analysis of large-scale MTDC grids is demonstrated by a remarkable speedup of up to 270 times for the Greater CIGRÉ DC grid with time-steps of 50 ns and 1 μs for device-level and system-level simulation over the CPU implementation. Finally, the accuracy of GPU simulation is validated by the commercial tools SaberRD and PSCAD/EMTDC.

INDEX TERMS CIGRÉ B4 DC grid, graphics processing unit (GPU), high-voltage direct current (HVDC), large-scale systems, modular multilevel converter (MMC), multi-terminal DC (MTDC), multi-threading, parallel processing.

I. INTRODUCTION

The high-voltage direct current (HVDC) system based on the modular multi-level converter (MMC) has gained tremendous attention, and it can be expanded into a multi-terminal DC (MTDC) system by connecting several stations [1]. Prior to the on-site commissioning tasks, the control and protection strategies need to be designed. The electromagnetic transient (EMT) simulation provides such a platform and is thus essential to the development of MTDC grids [2], [3].

Currently, the central processing unit (CPU) prevails in off-line EMT-type solvers, such as SaberRD®, PSpice®, and PSCAD/EMTDC® where time-domain simulations are conducted [4]–[6]. The single-core CPU becomes extremely inefficient in computing a complex system such as the MMC, forcing the investigation of new methods such as model-order reduction with the subsequent loss of device or equipment model details. Under circumstances that many repetitive components exist, multi-threading programming methods supported by multi-core CPU (MCPU) are able to improve the simulation speed. The drawback is that the parallelism is not sufficiently high, neither is the speedup over single-core CPU satisfactory when a lot of irregularities that exempt themselves from being computed concurrently exist.

Therefore, new methods enabling faster simulation of an MMC or even the MTDC system are imperative. One common approach is the development of simplified MMC models by omitting some switching details, such as the averaged value model, detailed equivalent model and their variants [7]–[9]. Nevertheless, the fully-detailed model resembles a real system is always preferred, particularly when switching transients are required for power loss calculation and the subsequent converter design evaluation where the idealized models of the insulated gate bipolar transistor (IGBT) are no longer applicable. The method that partitions a circuit into multiple parts was also explored [10] to raise the simulation speed. A common deficiency of preceding methods is, with an expansion of the DC grid, the simulation efficiency declines dramatically due to a largely sequential processing manner on the CPU.

The necessity of EMT computation of large-scale DC grids on the graphics processing unit (GPU) is thus manifested. Hitherto, only a few cases are available on AC systems [11]–[13]. While they exhibit highly complex behavior such as frequency-dependency and nonlinearities in equipment such as transmission lines and cables, rotating machines, and transformers, their computational burden pales in comparison to large-scale DC grids with multiple power converters with device-level models wherein the circuit complexity can quickly become untenable due to the large number of discrete nonlinear switching devices, and the high number of voltage levels for MMCs used in practical DC grids. Thus, GPU for power converter simulation emerges [14], [15], albeit the scale is still far from MTDC simulation for both device-level and system-level performance preview.

In this paper, an accurate electromagnetic-thermal simulation of MTDC grid is proposed on the massively parallel architecture of the GPU; significant efforts are made to improve the off-line simulation efficiency from three aspects, i.e., the processor, the modeling approach, and the computational algorithm. The irregularity of the MTDC grid poses a great challenge to GPU simulation, i.e., the small number of many components restricts massively parallel execution, which is the factor that GPU relies on to derive speed advantage over CPU. For example, taking the 11 HVDC converters in the CIGRÉ B4 DC grid as 11 threads falls short of massive parallelism. Thus, fine-grained circuit partitioning is applied to the MMC and the hybrid HVDC breaker (HHB) where a large number of identical circuit units exist to conform to the single-instruction-multiple-thread (SIMT) architecture of the GPU. Meanwhile, the dynamic parallelism [16] is featured to cater to the hierarchical MTDC system.

The paper is organized as follows: Section II provides a brief overview of the GPU. Section III elaborates the fine-grained partitioning of the MTDC system. In Section IV, detailed modeling of fundamental MTDC system components is demonstrated, and Section V presents the design of GPU kernels. Section VI provides simulation results and comparison regarding accuracy and computational efficiency from a variety of DC systems, and the conclusions are drawn in Section VII.

II. OVERVIEW OF GPU ARCHITECTURE

The NVIDIA® GeForce GTX 1080 (Pascal architecture) [17] consists of 4 graphics processing clusters, each of which is composed of 5 streaming multiprocessors (SM^G). As shown in Fig. 1(a), inside a SM^G there is 96KB shared memory, and up to 128 CUDA cores. The SM^G which has a high parallelism schedules warps of 32 threads to CUDA cores. Thus, with 20 highly parallel multiprocessors, the GPU is equipped with a total number of 2560 CUDA cores. Fig. 1(b) describes the GPU's computational architecture. The main function running on the host CPU invokes a compute grid ($grid^G$) on the GPU for the global function $Kernel_x$ which is defined as a kernel coded by CUDA C. Data from the CPU and the kernel is stored in the global memory for easy access. By definition, a $grid^G$ contains multiple compute blocks and each block consists of a number of threads, meaning circuits with the same attribute can be coded as one kernel implemented in the SIMT manner. During the computation process, supported by GPUs with compute capability $sm\_3.5$ and thereafter, an arbitrary thread can launch its own child $grid^G$ from the device where a new kernel $Kernel_y$ operates, and if necessary, it can launch the grandchild $grid^G$ [16]. The synchronization function is used to ensure that the child $grid^G$ will return to its upper level only when all of its threads complete their tasks, and so does the original $grid^G$ for $Kernel_x$. Afterwards, the GPU hands over the process control to CPU.

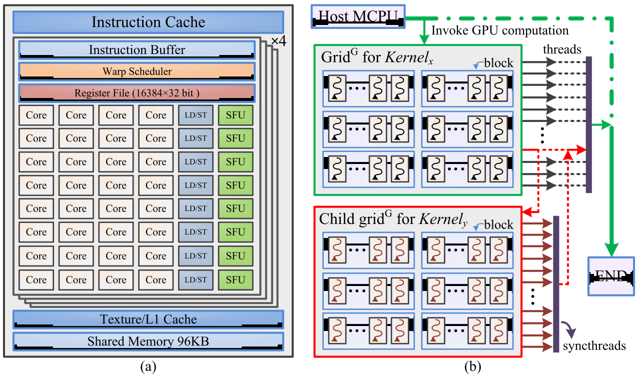


FIGURE 1. GPU architecture: (a) NVIDIA® Pascal streaming multiprocessor diagram, (b) dynamic parallelism.

In this work, the GPU CUDA C codes are run on NVIDIA® GeForce GTX 1080 and the CPU OpenMP® C program on a workstation with 64-bit operating system, 24 2.00-GHz Intel® Xeon® E5-2620 CPUs, and 16.0 GB of RAM.

III. FINE-GRAINED PARTITIONING OF MTDC GRID

In Fig. 2, the CIGRÉ B4 DC grid test system [1] containing 3 DC subsystems (DCS) is used as the testbench for GPU-based EMT simulation. Its details are given in the Appendix, and the converter stations are numbered in the figure. The positive power flow direction is defined, along with some annotated lines in DCS2.

A. FREQUENCY DEPENDENT LINE MODEL

The transmission line provides a natural circuit partition due to the time delay induced by traveling waves. The frequency dependent line model (FDLM) is able to describe all underground cables and overhead line geometries accurately in the phase domain [18]. Its Norton equivalent circuit is shown in Fig. 3(a), where the history current is expressed by

$$I_{his(k/m)}(t+1) = \mathbf{Y_c} * v_{(k/m)}(t+1) - 2\mathbf{H} * I_{(m/k)r}(t-\tau), \tag{1}$$

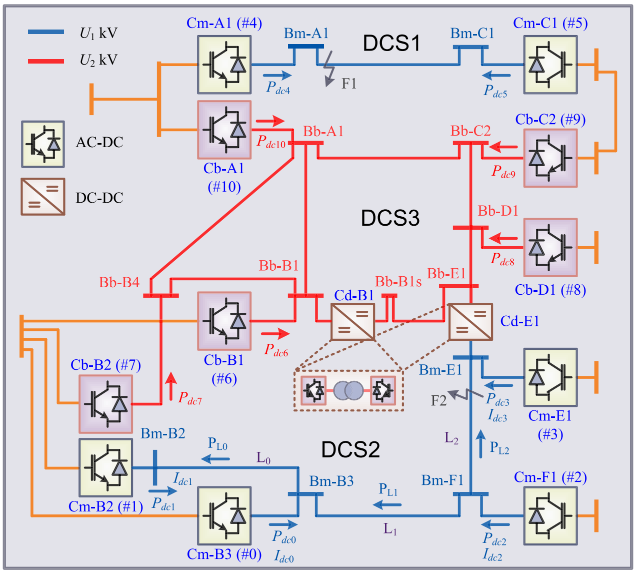


FIGURE 2. **CIGRÉ B4 DC grid test system.**

where $*$ symbolizes convolution, $I_{(m/k)r}$ is the reflected current, and $\mathbf{Y_c}$ and $\mathbf{H}$ denote the characteristic admittance matrix and the propagation matrix, respectively, and the traveling wave is manifested by the travel time τ.

With I_{his} and the admittance G known, the terminal voltage $v_{(k/m)}$ can be obtained by solving the circuit where the FDLM locates. Then, the terminal current is calculated by

$$i_{(k/m)}(t) = G \cdot v_{(k/m)}(t) - i_{his(k/m)}(t), \tag{2}$$

where $i_{(k/m)}$ is used to calculate the incident current, by which the history currents can be updated as

$$I_{(k/m)i}(t+1) = \mathbf{H} * I_{(m/k)r}(t-\tau). \tag{3}$$

To facilitate circuit computation, the Norton equivalent circuit of FDLM is converted to its Thévenin counterpart, as shown in Fig. 3(b). Therefore, voltages V_{hisk} and V_{hism} participate in computation, whereas the update of FDLM's parameters is undertaken with currents I_{hisk} and I_{hism}.

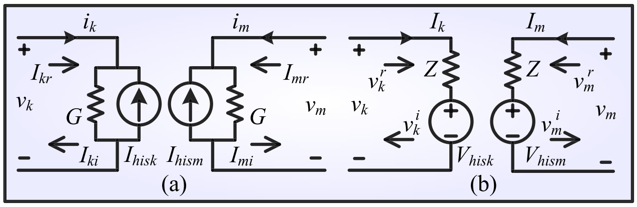


FIGURE 3. **Transmission line model: (a) Norton equivalent circuit, and (b) the Thévenin equivalent circuit.**

B. MTDC MULTI-LEVEL PARTITIONING SCHEME

The irregular CIGRÉ DC grid is partitioned by 3 levels to accommodate massive parallelism. The first level is the natural separation of converter stations by DC transmission lines.

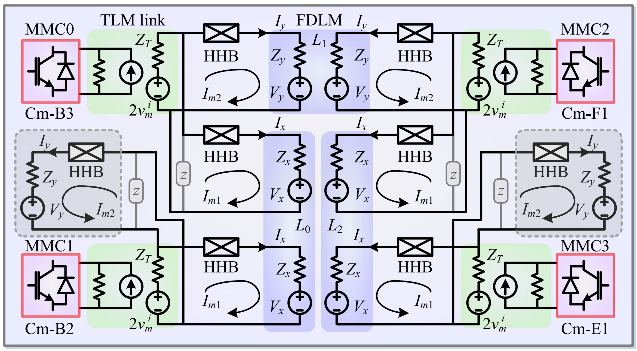


FIGURE 4. **Circuit partitioning of HVDC stations in the DC subsystem DCS2 by transmission line models.**

In Fig. 4, the 4-terminal DCS2 is shown as an example, where FDLMs are discretized to separate all the stations. Despite that, a station still corresponds to a huge admittance matrix since both MMC and the HHB contain hundreds of nodes. Thus, facilitated by the MMC DC capacitor which can be deemed as a transmission line model link (TLM-*link*) [19], the second level of circuit partitioning is applied.

Some virtual branches are introduced in DC yards to unify the configuration for SIMT execution. Therefore, identical names are assigned to components at the same position, which enables programming the DC yard as one GPU kernel. However, distinct circuit topologies lead to different computation algorithms. For example, the DC yards of Cm-B3 and Cm-F1 can be uniformly written as:

$$\mathbf{I_m} = \begin{bmatrix} \sum_{i=1}^{2} Z_{HHB_i} + Z_x + Z_y & -Z_{HHB_2} - Z_y \\ -Z_{HHB_2} - Z_y & Z_T + Z_{HHB_2} + Z_y \end{bmatrix}^{-1} \times \begin{bmatrix} V_y + \sum_{i=1}^{2} (-1)^i V_{HHB_i} - V_x \\ 2v_m^i - V_{HHB_2} - V_y \end{bmatrix}, \tag{4}$$

where Z_{HHB} and V_{HHB} forms the companion model of an HHB, $Z_{x/y}$ and $V_{x/y}$ is the FDLM, and v_m^i constitutes TLM link model of a DC capacitor along with its impedance Z_T.

On the contrary, in single-branch DC yards, the mesh current can be obtained instantly as:

$$I_{m1} = \frac{2v_m^i - V_{HHB_1} - V_x}{Z_T + Z_{HHB_1} + Z_x}, \tag{5}$$

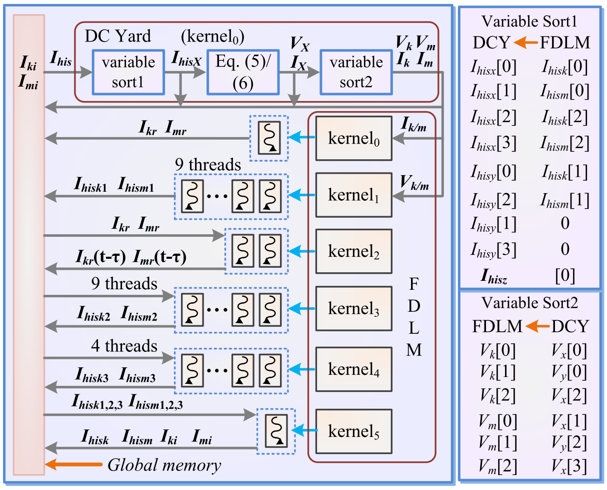
since it is obvious that $I_{m2} = 0$.

In the overall CIGRÉ B4 DC grid, one station may connect up to three other stations. To enable all DC yards to be computed by one kernel, rather than by multiple kernels with lower parallelism, the standard DC yard should have 3 branches. Thus, the virtual line currents in double-branch DC yard and single-branch DC yard are $I_z = 0$ and $I_y = I_z = 0$, respectively, whereas the actual branch current can be determined from mesh currents calculated by either (4) or (5).

Comparing the FDLM voltage variable names in Fig. 3(b) and Fig. 4, the names in the latter figure should be sorted. Throughout coupled DC yards, the one with smaller station number is defined as terminal k. Thus, variables belonging to the station $V_{x/y/z}$ and $I_{x/y/z}$ can be mapped to those of FDLM, i.e., $v_{k/m}$ and $I_{his(k/m)}$. Taking DCS2 for example, between $MMC0$ and $MMC2$ is the DC line L_1; thus, $V_{y0} = V_{hisk1}$, $V_{y2} = V_{hism1}$, where the number in the subscripts on the left and right side denotes MMC number and line number, respectively. Similarly, for the other two lines, the relationship between station variables and line variables are $V_{x0} = V_{hisk0}$, $V_{x1} = V_{hism0}$; $V_{x2} = V_{hisk2}$, $V_{x3} = V_{hism2}$, while the variables in virtual branches do not have effective assignments. On the other hand, updating FDLM's information requires its terminal voltage and current, which obey the same rule. The former is calculated by

$$v_{k,m} = I_{x/y} \cdot Z_{x/y} + I_{his(x/y)} \cdot Z_{x/y}, \tag{6}$$

and the latter $i_{k,m}$ is chosen from either I_x or I_y.



Variable Sort1

DCY ←	FDLM
I_{hisx}[0]	I_{hisk}[0]
I_{hisx}[1]	I_{hism}[0]
I_{hisx}[2]	I_{hisk}[2]
I_{hisx}[3]	I_{hism}[2]
I_{hisy}[0]	I_{hisk}[1]
I_{hisy}[2]	I_{hism}[1]
I_{hisy}[1]	0
I_{hisy}[3]	0
I_{hisz}	[0]

Variable Sort2

FDLM ←	DCY
V_k[0]	V_x[0]
V_k[1]	V_y[0]
V_k[2]	V_x[2]
V_m[0]	V_x[1]
V_m[1]	V_y[2]
V_m[2]	V_x[3]

FIGURE 5. **DC yard kernel structure and variable sort algorithm.**

The GPU kernel for the DC yard, which also includes the FDLM, is designed in Fig. 5. The relationship between DC yard and FDLM variables is realized by CUDA C in the form of device function so it can be accessed directly by kernels. After the variable sort process as described above, (1)-(3) can be processed in the FDLM kernel without distinguishing the terminal, meaning that the parallelism is heightened. The introduction of redundant branches enables all DC yards to have the same inputs and outputs, facilitating concurrent computation using SIMT on the GPU.

IV. FUNDAMENTAL COMPONENTS MODELING

A. IGBT ELECTRO-THERMAL TRANSIENT CURVE-FITTING MODEL

In system-level simulation, the IGBT is idealized by the two-state switch model (TSSM) with fixed on- and off-state resistances. On the other hand, complex models involving device physics are too burdensome to compute and often cause numerical divergence. Thus, the transient curve-fitting model (TCFM) becomes the best choice, which provides device-level information while also ensures computational efficiency.

The proposed datasheet-driven TCFM contains two parts: the static characteristics and the switching transients. The former is realized by a current-dependent resistor, based on the static V-I curves available in the manufacturer's datasheet. By piecewise linearizing the terminal voltage V_{CE} as a function of collector current I_C, it can be expressed by

$$r_s(I_c, V_g, T_{vj}) = \frac{V_{CE}}{I_C} \\ = a_1(I_c, V_g, T_{vj}) + a_2(I_c, V_g, T_{vj}) \cdot I_C^{-1}, \tag{7}$$

where a_1 and a_2 are coefficients dependent on factors such as the gate voltage V_g, and junction temperature T_{vj}.

Meanwhile, switching transients, including turn-on and turn-off processes, are modeled separately. The rise time t_r and fall time t_f reflecting these two stages are sensitive to factors such as T_{vj}, R_g, and I_C. Therefore, according to Stone-Weierstrass Theorem, the relationship can be described by the following polynomial function

$$t_{r,f}(s_1, s_2, s_3) = k_0 \cdot \prod_{i=1}^{2} s_i + \sum_{i,j=1,2,3}^{i \neq j} k_i s_i s_j + + \sum_{i=1}^{3} b_i s_i + b_0, \tag{8}$$

where the variable s_i represents one of the 3 factors, and k_i, b_i are constants.

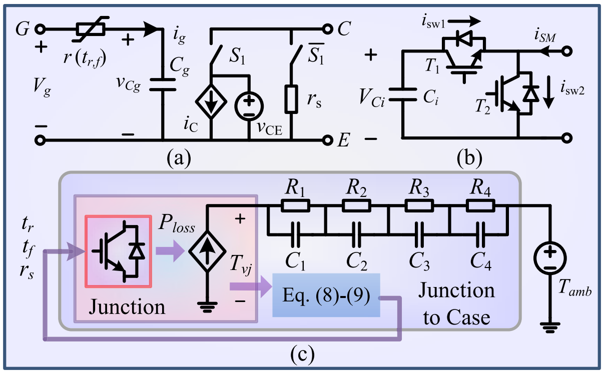


FIGURE 6. **IGBT curve-fitting model: (a) Equivalent circuit, (b) IGBT in MMC submodule, and (c) electro-thermal network.**

The turn-on and turn-off waveforms of 5SNA 2000K450300 StakPak IGBT Module [20] are obtained from a bridge-structure test circuit providing an identical electromagnetic environment to that of an MMC. A controlled current source i_C produces the transient currents, as given in Fig. 6(a). Take the turn-on current in Fig. 7(a) for example, when the IGBT is ordered to turn on, $V_g = 1$, so the capacitor C_g is charged with a time constant of $\tau = r \cdot C_g$. Stipulating that it takes the collector current i_C a time of τ to reach its maximum, and combine with the fact that i_C is virtually a

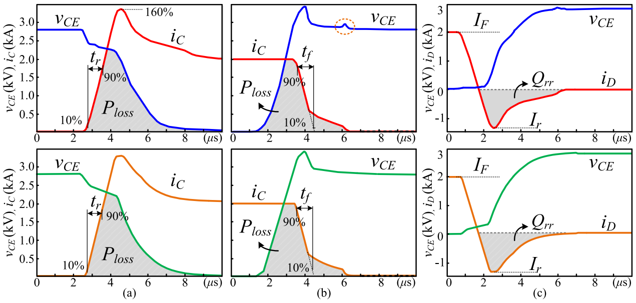


FIGURE 7. **5SNA 2000K450300 StakPak IGBT Module experimental (top) and simulated (bottom) waveforms: (a) IGBT turn-on, (b) IGBT turn-off, and (c) diode reverse recovery.**

straight line, r can be calculated by

$$\frac{(90\% - 10\%)I_C}{t_r} = \frac{(160\% - 0\%)I_C}{r \cdot C_g}, \tag{9}$$

where C_g is set as 1nF. Afterwards, i_C rises much slower and eventually begins to drop in a largely exponential trend, while v_{Cg} keeps rising. Depending on the requirement on accuracy, multiple exponential sections are used. Thus, the controlled current source can be generally expressed as

$$i_C(t) = \begin{cases} 0, & (v_{Cg} = 1), \\ i_C(t - \Delta t) + k_1 \cdot \Delta t/\tau, & (v_{Cg} \leq 1 - e^{\frac{-2t_r}{\tau}}), \\ (i_C(t - \Delta t) - 1) \cdot e^{\frac{-t}{k_2}} + I_C, & (v_{Cg} > 1 - e^{\frac{-2t_r}{\tau}}), \end{cases} \tag{10}$$

where k_1 and k_2 denote the rise and fall rates of the current, and I_C is the steady-state current which is added to ensure the current will not fall to 0. After the IGBT enters steady-state, v_{Cg} is set to 1. Meanwhile, the switch S_1 opens and the IGBT static part takes over. Similarly, when the turn-off process begins, $V_g = 0$ and v_{Cg} gradually falls from 1V. The transient voltage curves can also be obtained in the same manner that uses v_{Cg} as the control signal.

As Fig. 6(b) shows, since the two switches T_1 and T_2 in a submodule are complementary, and the simulated IGBT waveforms already considers the reverse recovery phenomenon of the other diode, only the active IGBT is calculated, while the terminal voltage and current of the opposite diode can be calculated based on nodal voltage and current. The results are given in Fig. 7 where the simulated curves demonstrate a good agreement to experimental waveforms.

Then, the instantaneous power consumption by an IGBT can be calculated, and sent to the electro-thermal network, as shown in Fig. 6(c). The IGBT loss P_{loss} is diffused through its case, and the thermal impedance equalizes to an RC network. The terminal voltage of P_{loss} is deemed as the junction temperature T_{vj}, calculated by

$$T_{vj} = \sum_{i=1}^{4}((P_{loss} + \frac{2v_{Ci}^i}{Z_{Ci}}) \cdot \frac{R_i Z_{Ci}}{R_i + Z_{Ci}}) + T_{amb}, \tag{11}$$

where the ambient temperature $T_{amb} = 25°\text{C}$, Z_{Ci} and v_{Ci}^i constitutes the EMT model of a capacitor. Then, T_{vj} is fed back to update IGBT parameters by (7) and (8).

B. METAL-OXIDE VARISTOR

The metal-oxide varistor (MOV) is a nonlinear component whose V-I characteristic takes the form of

$$v_M = k_M(\frac{i_M}{I_{ref}})^{\alpha_M^{-1}} \cdot V_{ref}, \tag{12}$$

where k_M and α_M are constants, V_{ref} is the protection voltage, and I_{ref} is the corresponding current. Then, the discrete-time companion model can be derived by

$$G_M = \frac{\partial i_M}{\partial v_M} = \frac{\alpha_M I_{ref}}{k_M V_{ref}} \cdot (\frac{v_M}{k_M V_{ref}})^{\alpha_M - 1}, \tag{13}$$

$$I_{Meq}(t + \Delta t) = i_M(t) - G_M v_M(t). \tag{14}$$

As can be seen, the MOV's conductance is still reliant on its terminal voltage v_M that is the outcome of EMT calculation, meaning an appropriate G_M may not be immediately found. Therefore, the Newton-Raphson method should be adopted for numerical convergence. In addition, (12) is divided into 10 piecewise linear segments to expedite the simulation.

C. THREE-PHASE TRANSFORMER

For a n-winding transformer, its basic V-I characteristic is represented by the following differential equation [21]:

$$\mathbf{v_T} = \mathbf{i_T R} + \mathbf{L}\frac{d}{dt}\mathbf{i_T}, \tag{15}$$

where $\mathbf{v_T}$ and $\mathbf{i_T}$ are both n-D vectors of the terminal voltages and currents, $\mathbf{R}$ is a diagonal matrix of winding resistances, and $\mathbf{L}$ contains self- and mutual inductances of the windings.

Discretization of the above equation, using Trapezoidal rule, leads to

$$\mathbf{i_T}(t + \Delta t) = \mathbf{G_T v_T}(t + \Delta t) + \mathbf{I_{his}}(t), \tag{16}$$

where the admittance matrix and the history current are

$$\mathbf{G_T} = [\mathbf{I} + \frac{\mathbf{L}^{-1}\mathbf{R}}{2}\Delta t]^{-1} \cdot \frac{\mathbf{L}^{-1}}{2}\Delta t, \tag{17}$$

$$\mathbf{I_{his}}(t) = 2\mathbf{G_T}(\mathbf{I} - \mathbf{R G_T})\mathbf{v_T}(t) + (\mathbf{I} - 2\mathbf{R G_T})\mathbf{I_{his}}(t - \Delta t). \tag{18}$$

In the DC grid, the admittance matrix of any system containing the 3-phase transformer has a minimum of 6×6 elements. The Gaussian Elimination is employed for the solution of the following universal matrix equation of an N-node system containing the transformer

$$\begin{aligned} &[\mathbf{v_T}|v_{ext7}, \ldots, v_{extN}]^T \\ &\quad = (\begin{bmatrix} \mathbf{G_T} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}_{N \times N} + diag[G_{ext1}, \ldots, G_{extN}])^{-1} \\ &\qquad \cdot ([\mathbf{I_{his}}|\mathbf{0}]^T + [J_{ext1}, \ldots, J_{extN}]), \end{aligned} \tag{19}$$

where the 3-phase transformer corresponds to the first 6 nodes, and elements with subscript ext are contributed by its surrounding components. The external conductance results in a diagonal matrix, which is added to the transformer admittance matrix. Similarly, the current vector from the outer system is also combined with that of the transformer.

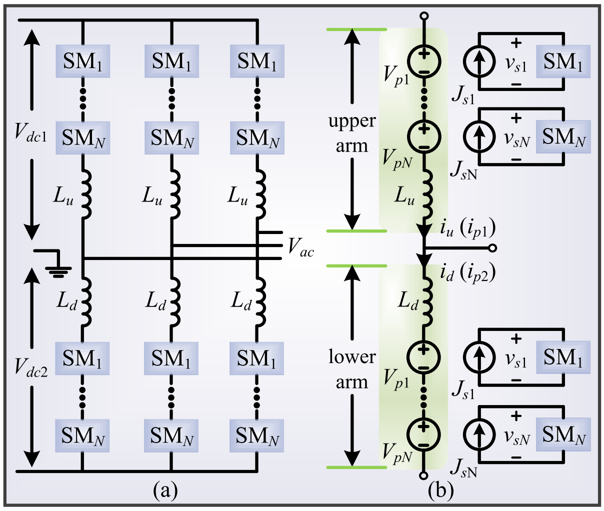


FIGURE 8. (a) Three-phase MMC topology, and (b) computational structure for one phase with *V-I* coupling.

V. MTDC GRID MASSIVE PARALLEL COMPUTATION

A. HVDC CONVERTER MODELING

Fig. 8(a) is an $(N+1)$-level, 3-phase MMC which can operate as a terminal of the HVDC grid. To withstand high voltages, there should be a sufficient number of submodules – usually in the range of dozens to a few hundred. To avoid computing a huge matrix equation, simplification of the converter is necessary. The fact that the arm current is alternating at a frequency much lower than that of the digital simulation rate indicates that the third-level partitioning method splitting all the submodules from the arms by a pair of coupled voltage-current sources as in Fig. 8(b) will not affect the accuracy. Then, the partitioned MMC corresponding to a collection of small matrices can be processed more efficiently, particularly when the processor supports parallel computation.

Fig. 9(a) gives a transformer-connected MMC EMT model. The Norton equivalent circuit of the MMC arm is

$$G_p = (Z_{Lu.d} + r_{arm})^{-1}, \tag{20}$$

$$J_p = G_p \cdot (\sum_{i=1}^{N} V_{pi} + 2v^i_{Lu,d}), \tag{21}$$

where r_{arm} is the arm inductor's parasitic resistance. Then, the MMC is solved by (19) in the EMT program.

In Fig. 9(b), the MMC controller is given. Depending on the actual demand, the outer-loop controller which is based on *d-q* frame compares various feedbacks, such as DC and AC voltages, active and reactive powers, with their references. Then, the Inverse Park's Transformation restores signals to three phases, and the inner-loop controller employing phase-shift control (PSC) [22] regulates individual phases.

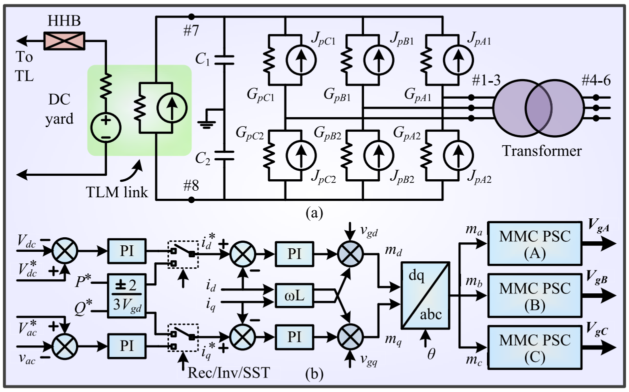


FIGURE 9. (a) EMT model of an 3-phase MMC main circuit, (b) a general controller scheme for various control targets.

B. MMC GPU COMPUTATIONAL KERNEL

As can be seen, extensive symmetries exist in the MMC. The three phases have an identical topology and so do the three inner-loop controllers, which contain three parts: generation of carriers, averaging control, and balancing control (BC), as shown in Fig. 10(a). For CPU simulation, identical algebraic operations are conducted repeatedly due to the sequential implementation manner, e.g., the definition of N carriers, and IGBT gate voltage V_g generation for $2N$ SMs. In contrast, the PSC computational structure in GPU sees a dramatic change as generally the controller can be implemented in parallel. Thus, the PSC is composed of several kernels, and signals transmitted between them are stored in the global memory. The SM DC voltages are summed up by multiple threads, as indicated in Fig. 10(b). For one phase, the CPU needs to conduct the summation $(2N-1)$ times, while in GPU, it is

$$N_{sum} = \log_2(2^M), 2^{M-1} < 2N \leq 2^M. \tag{22}$$

Thus, for an arbitrary MMC, if $2N$ is less than 2^M, the extra numbers are compensated by zeros so that in the addition process defined as in *Kernel*$_0$, an even number of variables is always reserved during the summation.

The bulk of averaging control is realized by *Kernel*$_1$, and its output is sent to *Kernel*$_2$ where the balancing control is carried out in the SIMT manner. Meanwhile, repeated definition of the carriers as in the CPU is avoided; instead, they are defined only once in each thread. Finally, *Kernel*$_3$ receives the output array $\mathbf{V_g}$ and the submodule calculation begins with each thread corresponding to one SM. The output array $\mathbf{v_C}$ is sent to the global memory so that *Kernel*$_0$ can assess it.

As an actual MMC has 3 phases, the above parallelism can be enhanced since all 6 arms share the same configuration, and so do the $6N$ submodules. With regard to the controller, *Kernel*$_0$ and *Kernel*$_1$ of PSC have 3 threads, while *Kernel*$_2$ will be launched as a compute grid of $6N$ threads on the GPU. Therefore, a new HVDC converter kernel which contains the PSC kernel (*Kernel*$_0$-*Kernel*$_2$), SM kernel, and the MMC main circuit kernel based on (19) is constructed using the

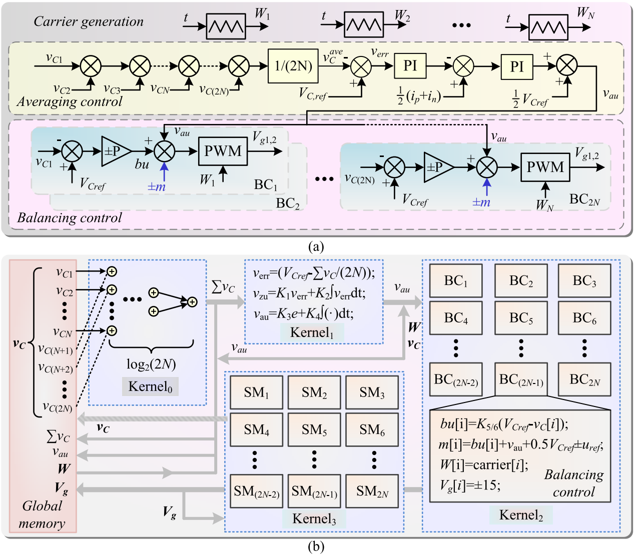


FIGURE 10. **MMC inner loop control for single-phase: (a) Phase-shift control in CPU, (b) single-phase MMC kernel.**

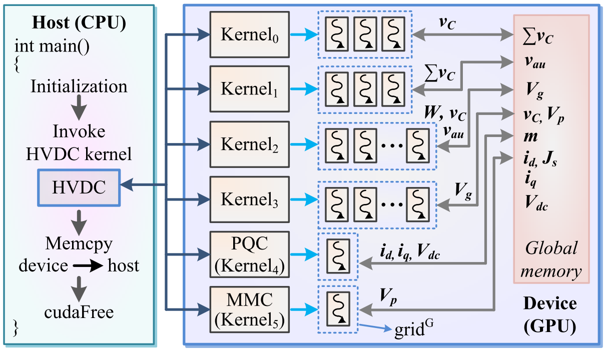


FIGURE 11. **Hierarchical dynamic parallelism implementation of a 3-phase HVDC converter.**

dynamic parallelism feature, as Fig. 11 shows. The number of threads in each compute grid is exactly the same as that of an actual circuit part. For example, there is one outer-loop controller and one MMC main circuit, and accordingly, kernel *PQC* and *MMC* both invoke only thread, whereas *Kernel*$_3$ for SM launches a grid of 6N threads. In the host, after initialization, the HVDC converter kernel is launched by CPU, which then invokes the 6 kernels with various threads.

C. HYBRID CIRCUIT BREAKER MODEL

Fig. 12(a) describes a typical hybrid HVDC breaker comprising of the ultrafast disconnector (UFD), the load commutation switch (LCS), the main breaker (MB), the MOV, the current limiting inductor L, and the residual current breaker (RCB). The full model containing as many components as a real HHB, rather than the scaled-down model with a significant simplification, is preferred in EMT simulation [23], [24]. The subsequent issue is an enormous admittance matrix that is too burdensome to solve. The V-I coupling is applied in Fig. 12(b), and it is reasonable to do so because the inductor helps to avoid an abrupt current change. The creation of a large number of HHB units caters to the massive CUDA cores.

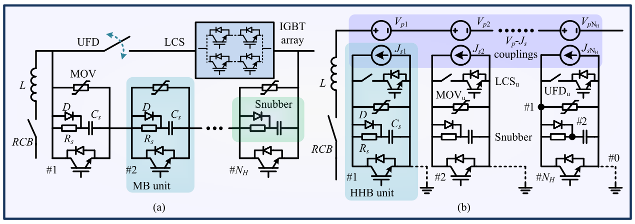


FIGURE 12. **HHB EMT models: (a) the conventional full model, (b) partitioned full model with reduced order.**

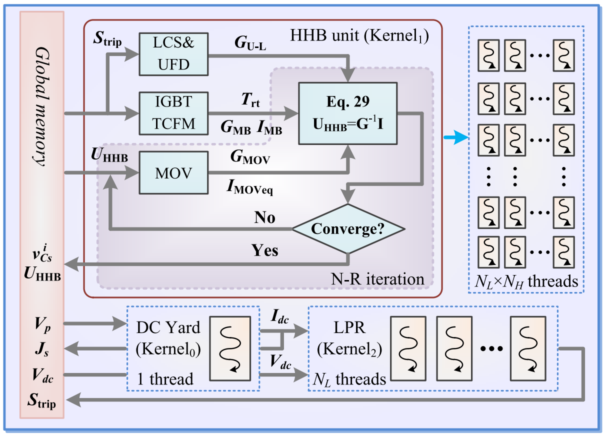


FIGURE 13. **HHB unit kernel design and its EMT calculation manner in conjunction with DC yard and LPR.**

After partitioning, the HHB leaves the RCB, the current limiting inductor, and a series of voltage sources, denoted by V_p, in the DC yard. They are represented by a combination of impedance Z_{HHB} and voltage V_{HHB}, given as

$$HHB = Z_{HHB}J_s + V_{HHB} = (RCB + Z_L)J_s + 2v_L^i + \sum_{i=1}^{N_H} V_{pi}, \tag{23}$$

where Z_L and v_L^i constitute the inductor TLM-stub model [10]. Meanwhile, nonlinearities are confined solely to the HHB units, so the simulation efficiency is improved by avoiding repeated calculations of the originally extremely large circuit in order to derive a convergent result. Instead, the matrix equation applying Newton-Raphson method is merely 2 dimensional, and depending on the status of main breaker IGBTs, it has two forms, which can be expressed by

$$\mathbf{U_{HHB}} = \begin{bmatrix} R_{sD}^{-1} + T_{rt}\cdot G_{MB} & -R_{sD}^{-1}, \\ +G_{MOV} + G_{(U-L)}, & \\ -R_{sD}^{-1}, & R_{sD}^{-1} + G_{C_s} \end{bmatrix}^{-1} \times \begin{bmatrix} J_s - I_{MOVeq} - (1-T_{rt})\cdot I_{MB} \\ 2v_{C_s}^i \cdot G_{C_s} \end{bmatrix}, \tag{24}$$

where T_{rt} is a binary value indicating the steady state and

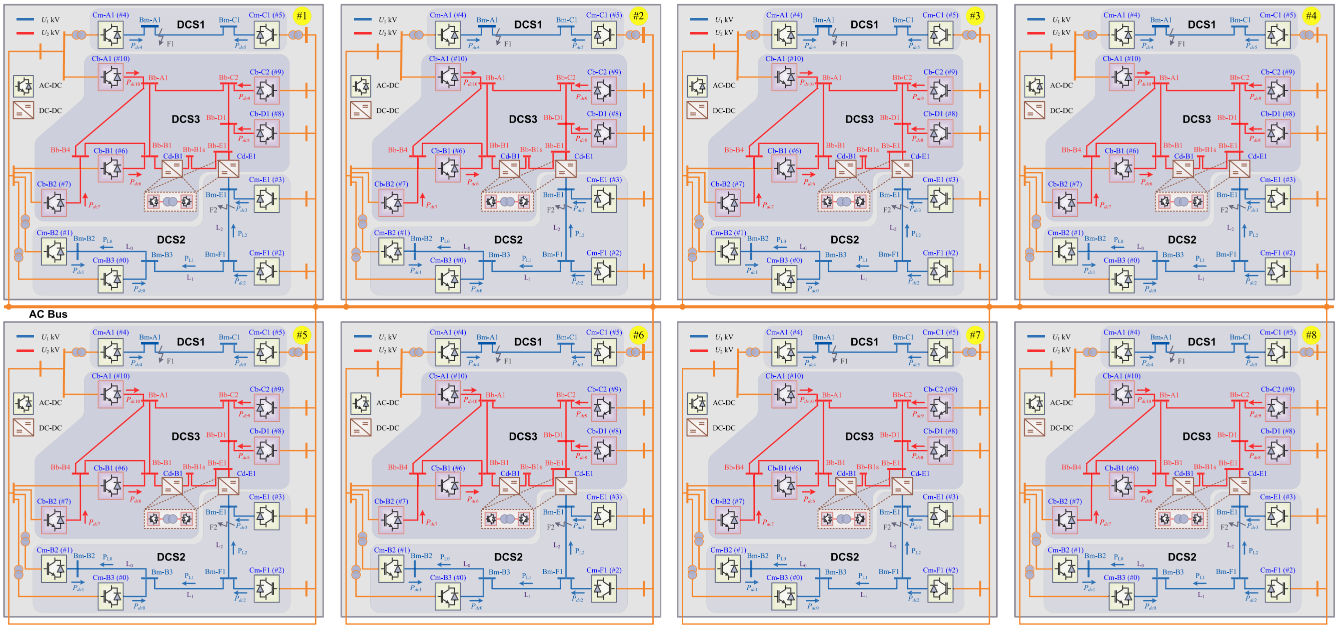


FIGURE 14. **Greater CIGRÉ DC Grid consisting of multiple CIGRÉ DC B4 systems.**

switching state by 0 and 1, respectively, and R_{sD} is the equivalent resistance of the parallel R_s and D in the snubber.

When the HHB is initiated following the detection of line fault, the instantaneous current during the breaking period is

$$i_{dc}(t) = \frac{V_{dc}}{R_p} \cdot (1 - e^{-\frac{R_p}{L_p}t}) + \frac{P_{dc}(0)}{V_{dc}} e^{-\frac{R_p}{L_p}t}, \qquad (25)$$

where R_p and L_p denote the equivalent resistance and inductance of the power flow path, and $P_{dc}(0)$ is the power before fault. At the end of the breaking period Δt_1^f, the fault current reaches its peak, and the fault clearance period lasting Δt_2^f takes over. Thus, the number of HHB units N_H could be calculated – suppose a $\delta\%$ margin is reserved to ensure safe operation – by the following equation

$$N_H \cdot V_{CES}(1 - \delta\%) = V_{dc} + L_p \frac{i_{dc}(\Delta t_1^f)}{\Delta t_2^f}, \qquad (26)$$

where V_{CES} is IGBT's maximum collector-emitter voltage.

D. HHB GPU COMPUTATIONAL KERNEL

The GPU computational structure for HHB contains two parts: the voltage source side and the HHB units. The former is included in the linear DC yard function, while the latter constitutes an independent kernel with Newton-Raphson iteration, as Fig. 13 shows. The key for GPU to have a speed leverage over CPU toward the same configuration as Fig. 12(b) is SIMT processing of all HHB units. Then, in the HHB unit kernel, the varistor, the LCS-UFD branch, and the IGBT TCFM are realized by CUDA C device functions so that they can be accessed by the kernel directly, and only the MOV function that causes nonlinearity may have to be called multiple times by the iterative Newton-Raphson process in which (24) is computed repeatedly until a precise result is derived. It should be pointed out that the HHB is always applied in conjunction with line protection (LPR), and therefore, the kernel is briefly drawn for illustration of the coordination between HHB kernel and the protection device.

E. CONSTRUCTION OF LARGE-SCALE MTDC GRIDS

A further expansion of the MTDC grid is carried out for simulating the Greater CIGRÉ DC Grid which is composed of several interconnected CIGRÉ DC systems, as shown in Fig. 14. The hierarchy in GPU simulation remains the same regardless of the expansion. Though the MMCs may have a varying voltage level, the SM parallelism maintains since all submodules constitute independent circuits and therefore are designed into one kernel which still computes all threads concurrently when invoked. In the MMC arms, the summation of V_{pi} in (21) no longer takes a unified form but the parallelism still exists with each MMC arm being distinguished by the thread number.

VI. GPU SIMULATION RESULTS AND EFFICIENCY

A. MMC DEVICE-LEVEL RESULTS

The transients of a semiconductor switch are reflected directly by the rise/fall time and ultimately affects the junction temperature. The simulated curves of (8) and experimental results available in the datasheet under different collector currents are shown in Fig. 15(a). Three sections of linear functions are used for the t_f-I_C curve, while the rise time curve requires two sections, and when the y-axis is turned into logarithmic, like in the datasheet, both curves bend. Fig. 15(b)-(d) are device-level results from a 5-level MMC with 16kV DC bus, 2kHz carrier frequency, and 2kA, 60Hz output current. The relationship between IGBT average power loss and its switching frequency is drawn in Fig. 15(b).

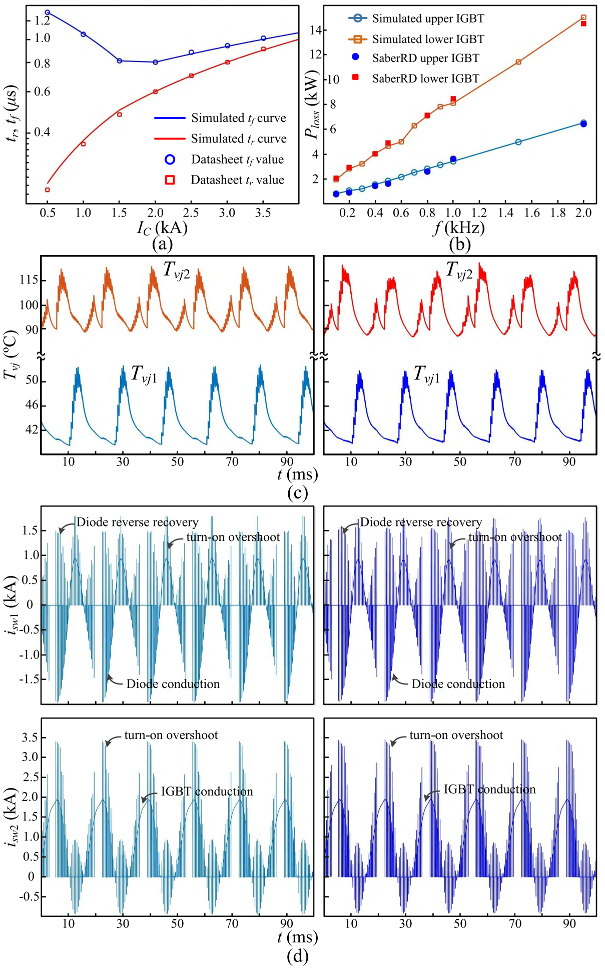


FIGURE 15. IGBT device-level results ((c)-(d): proposed model (left) and SaberRD® (right)). (a) rise and fall times, (b) averaged power loss, (c) IGBT junction temperatures, and (d) current waveforms of upper and lower switches.

The average losses in both switches rise steadily along with the frequency, as the transient power loss becomes increasingly significant. Meanwhile, the loss on the lower switch is more severe, resulting in a higher junction temperature than its counterpart in Fig. 15(c), as the upper diode and lower IGBT are expected to be subjected to a larger current, as shown in Fig. 15(d), which also shows intensive diode reverse recovery and IGBT turn-on overshoot currents that otherwise are not available in the ideal switch model. As an industry standard tool frequently referred to for device-level information for guidance on power converter design, the above device-level results have been validated by SaberRD® simulation.

B. GPU SIMULATION OF MTDC GRID CASES

The DCS2 subsystem in Fig. 2 is taken as a typical MTDC grid example since its scale is very close to existing projects as well as those under research and development.

Installation of HHBs in the MTDC grid enhances its resilience to DC line faults, and the results are provided in Fig. 16. Fig. 16(a) sees minor DC voltage disturbances by the fault F_2 occurring at t=3s due to the swift action of the HHBs, while the DC currents in Fig. 16(b) have

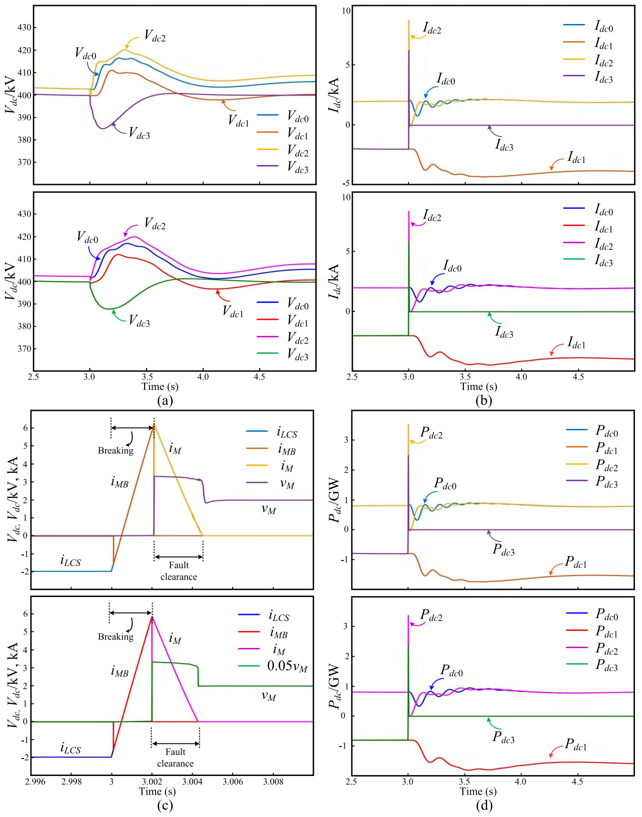


FIGURE 16. DCS2 simulation results from GPU (top) validated by PSCAD/EMTDC® (bottom). (a) DC voltages, (b) DC line currents, (c) actions of Cm-E1 HHB, and (d) station power.

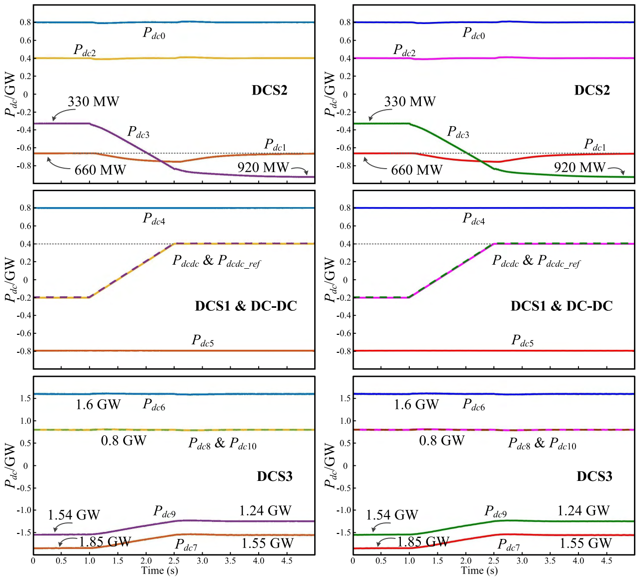


FIGURE 17. CIGRÉ DC grid power reversal simulation by GPU (left) and PSCAD/EMTDC® (right).

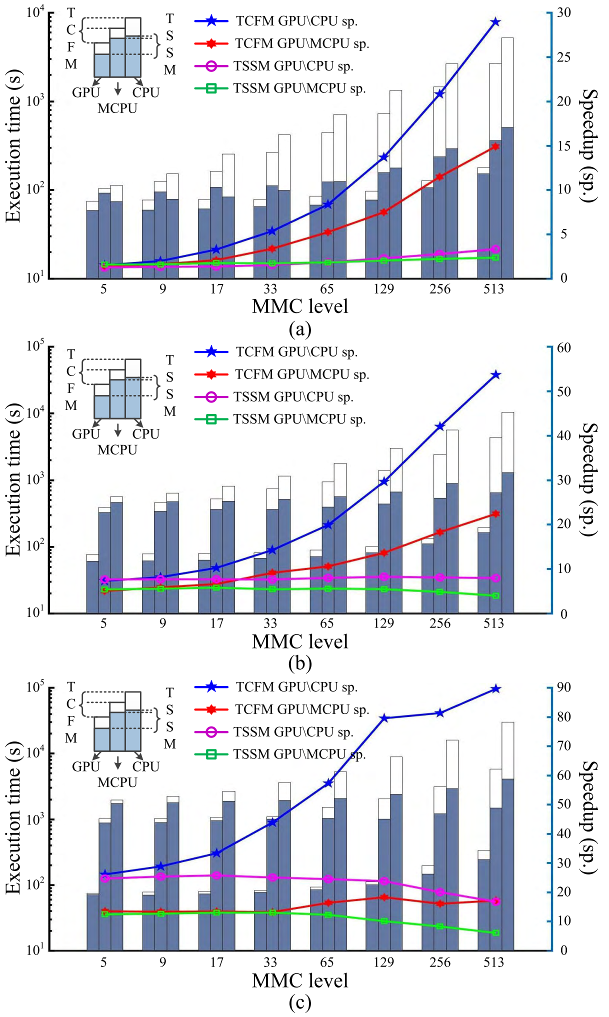


FIGURE 18. GPU performance in simulation of different DC systems with IGBT TSSM and TCFM. (a) HVDC with HHB, (b) DCS-2, and (c) CIGRÉ B4 DC system.

remarkable surges at both MMC2 (Cm-F1) and MMC3 (Cm-E1): I_{dc2} flowing to Cm-E1 keeps increasing before the fault is isolated, and I_{dc3} witnesses a polarity reversion. The function of an HHB is illustrated in Fig. 16(c).When the fault is detected, the LCS turns off and consequently i_{LCS} drops to 0; while the main branch keeps on in the breaking stage for the next 2ms when the current diverts to it, and because of the existence of the current limiting inductor, i_{MB} rises gradually from a negative value to positive. Following the turn-off of MB, the current again is diverted to the MOV where it is quenched in the form of i_M in the fault clearance stage. Fig. 16(d) gives power flow at different positions. Prior to the fault, Cm-B2 and Cm-F1 send approximately 800MW power to Cm-B3 and Cm-E1. After the fault is cleared, the export of Cm-F1 entirely goes to Cm-B3. Thus, in addition to the power from MMC0 (Cm-B3), the remaining inverter receives nearly 1600MW. Details from PSCAD/EMTDC® are also given for validation.

In Fig. 17, simulation of the whole CIGRÉ DC grid is also carried out on the GPU. Power reversal is conducted by ordering the output power of DC-DC converter Cd-E1 to ramp from −200MW to 400MW. Initially, MMC0 and MMC2 as rectifiers in DCS2 release 1.2GW, and MMC1 and MMC3 receives around 660MW and 330MW, respectively. The surplus 200MW is fed to DCS3, as it can be seen that the combined amount of output power from MMC6, MMC8, and MMC10 is 3.2GW, but the inverters MMC7 and MMC9 get approximately 0.2GW more. During power ramp process, as expected, the output power of all rectifier stations remains virtually constant, while only the inverter MMC5 absorbs a fixed 800MW power as DCS1 is relatively isolated from its counterparts. In DCS2, the power MMC3 receives almost triples after the process, while MMC1 is slightly affected during the process, and after that, it restores. Meanwhile, as the power is flowing from DCS3 to DCS2, the power received by MMC7 and MMC9 reduce to around 1.55GW and 1.24GW, showing a deficiency of 400MW compared with that provided by rectifiers in that subsystem. The above statements are validated by PSCAD/EMTDC®.

The GPU's performance in simulating various DC systems with both TSSM and the proposed TCFM are summarized in Fig. 18. All three figures share the trait that it takes a slightly longer time for GPU to compute the TCFM regardless of the MMC level, whilst both the CPU and MCPU frameworks witness a dramatic rise even in the logarithmic axes, which accounts for the fact that device-level semiconductor models are rarely used in CPU-based large-scale system simulation. Fig. 18(a) shows that even for simulating a point-to-point HVDC system with the TSSM, GPU is still advantageous over multi-core CPU, let alone a much larger system with more complex switch models. Therefore, when the TCFM-based DCS2 is the object, as shown in Fig. 18(b), the GPU can attain a speedup of around 22 and 54 respectively over multi-core and single-core CPU. Fig. 18(c) shows a dramatic speedup increase in the CIGRÉ DC simulation by GPU, up to 20 and 90 times faster than multi-core and single-core CPU.

TABLE 1. CPU and GPU execution times of the Greater CIGRÉ DC system (Fig. 14) for 1s simulation.

CIGRÉ	Execution time (s)			Speedup		
NO.	CPU	MCPU	GPU	$\frac{CPU}{MCPU}$	$\frac{CPU}{GPU}$	$\frac{MCPU}{GPU}$
2	14549	1995	162.1	7.3	89.8	12.3
3	22264	2810	179.9	7.9	123.8	15.6
4	30249	3103	195.6	9.7	154.6	15.9
5	36963	4067	200.9	9.1	184.0	20.2
6	44391	4284	208.4	10.4	213.0	20.6
7	52792	4868	210.0	10.8	251.4	23.2
8	60121	5538	221.9	10.9	270.9	25.0

Table 1 shows the leverage that GPU holds in computing the Greater CIGRÉ DC grid is even larger. Take the 101-level

MMC for example, when the number of CIGRÉ DC system rises from 2 to 8, it takes CPU and multi-core CPU 4 times and 2.8 times longer respectively to compute, while it merely increases by 1.4 times in the GPU case. Consequently, GPU simulation is able to seize about 90 to 270 times of speedup over single-core CPU whilst multi-core CPU can only achieve 7 to 11 times speedup. Thus, the adoption of GPU greatly alleviates the computational burden caused by the complexity of the switch model, making the involvement of device-level models in system-level simulation feasible.

VII. CONCLUSION

An efficient methodology for large-scale DC grid simulation using massive parallelism on the GPU is presented in this paper wherein three levels of circuit partitioning are employed to attain fine-grained parallelism. The basic components are designed into CUDA C kernels for SIMT implementation after introducing virtual subsystems to enable components sharing the same property to have the same configuration. The power semiconductor switch is specifically modeled using the prevalent ideal switch model and the transient curve-fitting model to cater for different simulation requirements. Dynamic parallelism appropriately revealed the hierarchy of an MTDC system and therefore is employed. Test results demonstrate that with the same accuracy to existing commercial EMT-type solvers and a dramatic speedup over the CPU-base simulation, the GPU would play a significant role in simulating DC systems of a variety of scales in the future. And since the advantage of data handling capability of GPU becomes overwhelming when more identical components are computed, it is expected to be a new generation of platform for off-line time-domain simulation and particularly, dominant in the area of hybrid system-level and device-level simulation.

APPENDIX

MMC parameters: voltage level 5-513, arm inductance $L_{u,d} = 20$mH, SM capacitance $C_i = 3$mF; AC grid voltage $V_g = 380$kV, AC frequency $f = 60$Hz; DCS1,2 transformer ratio 380/270kV, YY structure; DCS 1,2 rated DC voltage $V_{dc} = \pm 200$kV, DCS 3 rated DC voltage $V_{dc} = \pm 400$kV. Rectifier stations: MMC0 800MW, MMC2 400MW, MMC4 800MW, MMC6 1600MW, MMC8 800MW, MMC10 800MW; Inverter stations: MMC1 ±200kV, MMC3 ±200kV, MMC5 ±200kV, MMC7 ±400kV, MMC9 ±400kV. Transmission line parameters: distance DCS1 $d_1 = 50$km, DCS2 $d_2 = 100$km, DCS3 $d_3 = 100$km; shunt conductance g=10^{-8}mΩ/km, conductor outer radius 10cm, height H=50m, sag 2m, DC resistance $r_{dc} = 0.01\Omega$/km.

REFERENCES

[1] T. K. Vrana, Y. Yang, D. Jovcic, S. Dennetière, J. Jardini, and H. Saad, "The CIGRE B4 DC grid test system," *CIGRE Electra Mag.*, vol. 270, pp. 10–19, Oct. 2013.

[2] P. Wang, N. Deng, and X.-P. Zhang, "Droop control for a multi-line current flow controller in meshed multi-terminal HVDC grid under large DC disturbances," *IEEE Power Energy Technol. Syst. J.*, vol. 5, no. 2, pp. 35–46, Jun. 2018.

[3] D. Jovcic and W. Lin, "Multiport high-power LCL DC hub for use in DC transmission grids," *IEEE Trans. Power Del.*, vol. 29, no. 2, pp. 760–768, Apr. 2014.

[4] S. Debnath and M. Saeedifard, "Simulation-based gradient-descent optimization of modular multilevel converter controller parameters," *IEEE Trans. Ind. Electron.*, vol. 63, no. 1, pp. 102–112, Jan. 2016.

[5] 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, Jan. 2015.

[6] A. A. Gebreel and L. Xu, "DC–AC power conversion based on using modular multilevel converter with arm energy approximation control," *IEEE Power Energy Technol. Syst. J.*, vol. 3, no. 2, pp. 32–42, Jun. 2016.

[7] 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.

[8] D. C. Ludois and G. Venkataramanan, "Simplified terminal behavioral model for a modular multilevel converter," *IEEE Trans. Power Electron.*, vol. 29, no. 4, pp. 1622–1631, Apr. 2014.

[9] J. Wang, R. Burgos, and D. Boroyevich, "Switching-cycle state-space modeling and control of the modular multilevel converter," *IEEE J. Emerg. Sel. Topics Power Electron.*, vol. 2, no. 4, pp. 1159–1170, Dec. 2014.

[10] H. Selhi, C. Christopoulos, A. F. Howe, and S. Y. R. Hui, "The application of transmission-line modelling to the simulation of an induction motor drive," *IEEE Trans. Energy Convers.*, vol. 11, no. 2, pp. 287–297, Jun. 1996.

[11] V. Jalili-Marandi and V. Dinavahi, "SIMD-based large-scale transient stability simulation on the graphics processing unit," *IEEE Trans. Power Syst.*, vol. 25, no. 3, pp. 1589–1599, Aug. 2010.

[12] Z. Zhou and V. Dinavahi, "Parallel massive-thread electromagnetic transient simulation on GPU," *IEEE Trans. Power Del.*, vol. 29, no. 3, pp. 1045–1053, Jun. 2014.

[13] J. K. Debnath, A. M. Gole, and W. K. Fung, "Graphics-processing-unit-based acceleration of electromagnetic transients simulation," *IEEE Trans. Power Del.*, vol. 31, no. 5, pp. 2036–2044, Oct. 2016.

[14] X.-X. Liu, S. X.-D. Tan, H. Wang, and H. Yu, "A GPU-accelerated envelope-following method for switching power converter simulation," in *Proc. Design, Autom. Test Eur. Conf. Exhibit.*, Mar. 2012, pp. 1349–1354.

[15] Z. Zhou and V. Dinavahi, "Fine-grained network decomposition for massively parallel electromagnetic transient simulation of large power systems," *IEEE Power Energy Technol. Syst. J.*, vol. 4, no. 3, pp. 51–64, Sep. 2017.

[16] *CUDA C Programming Guide*, NVIDIA Corp., Santa Clara, CA, USA, Jun. 2016.

[17] *Whitepaper NVIDIA GeForce GTX 1080*, NVIDIA Corp., Santa Clara, CA, USA, 2016.

[18] J. R. Marti, "Accurate modelling of frequency-dependent transmission lines in electromagnetic transient simulations," *IEEE Trans. Power App. Syst.*, vols. PAS-101, no. 1, pp. 147–157, Jan. 1982.

[19] S. Y. R. Hui and K. K. Fung, "Fast decoupled simulation of large power electronic systems using new two-port companion link models," *IEEE Trans. Power Electron.*, vol. 12, no. 3, pp. 462–473, May 1997.

[20] *ABB 5SNA-2000K450300 StakPak IGBT Module, doc. No. 5SYA 1431-01*. Accessed: Mar. 6, 2018. [Online]. Available: http://new.abb.com/semiconductors/stakpak

[21] J. Liu and V. Dinavahi, "A real-time nonlinear hysteretic power transformer transient model on FPGA," *IEEE Trans. Ind. Electron.*, vol. 61, no. 7, pp. 3587–3597, Jul. 2014.

[22] M. Hagiwara and H. Akagi, "Control and experiment of pulsewidth-modulated modular multilevel converters," *IEEE Trans. Power Electron.*, vol. 24, no. 7, pp. 1737–1746, Jul. 2009.

[23] N. Lin and V. Dinavahi, "Detailed device-level electrothermal modeling of the proactive hybrid HVDC breaker for real-time hardware-in-the-loop simulation of DC grids," *IEEE Trans. Power Electron.*, vol. 33, no. 2, pp. 1118–1134, Feb. 2018.

[24] W. Lin, D. Jovcic, S. Nguefeu, and H. Saad, "Modelling of high-power hybrid DC circuit breaker for grid-level studies," *IET Power Electron.*, vol. 9, no. 2, pp. 237–246, 2016.

NING LIN (S'17) received the B.Sc. and M.Sc. degrees in electrical engineering from Zhejiang University, China, in 2008 and 2011, respectively. Later, he was an Electrical Engineer on FACTS and HVDC control and protection. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada. His research interests include massively parallel computation of power electronic systems, real-time simulation, and field-programmable gate arrays.

VENKATA DINAVAHI (S'94–M'00–SM'08) received the B.Eng. degree in electrical engineering from the National Institute of Technology, Nagpur, India, in 1993, the M.Tech. degree in electrical engineering from IIT Kanpur, India, in 1996, and the Ph.D. degree in electrical and computer engineering from the University of Toronto, Toronto, ON, Canada, in 2000. He is currently a Professor with the Department of Electrical and Computer engineering, University of Alberta, Edmonton, AB, Canada. His research interests include real-time simulation of power systems and power electronic systems, electromagnetic transients, device-level modeling, large-scale systems, and parallel and distributed computing.