Overview of the CASTLE circuit simulator

Raymond J. Allen
Plasma Physics Division
Naval Research Laboratory
Washington, DC, USA
raymond.allen@nrl.navy.mil

David Hinshelwood
Plasma Physics Division
Naval Research Laboratory
Washington, DC, USA
david.hinshelwood@nrl.navy.mil

***Abstract*—CASTLE (Circuit Analysis and Simulation with Transmission-Line Emphasis) is a PC based code for simulating nonlinear analog circuits, designed primarily for Pulsed Power. CASTLE is an MNA (Modified Nodal Analysis) code that does transient analysis in a way that is very similar to SPICE and its derivatives. One key difference is that CASTLE has a fixed time step while SPICE has a variable time step. There are two main reasons for the fixed time step. One is exact reproduction of the output of our transmission line code, Bertha. This is very important of benchmarking with and transfer of specialized physics based models of Pulsed Power diodes. The second reason is to avoid the too-small-time-step failures that SPICE often has with Pulsed Power circuits. The CASTLE simulator has the ability to speed simulation using a unique matrix reduction technique. CASTLE uses the self-isolating property of transmission lines, in a similar way as transmission-line codes, to turn one large matrix into several smaller ones. This can lead to dramatic speed advantages for very large circuits, compared to SPICE. CASTLE has recently added an MPI (Message-Passing Interface) mode to further speed simulation. Also, there is now a dll version (for both Windows and Linux) of CASTLE that allows it to be run from the command line or other programs such as Python or C++. CASTLE includes a schematic entry GUI with sufficient quality for presentation or publication. This generates a SPICE compatible netlist that is used by the simulator.**

Keywords—CASTLE, Bertha, MPI, MNA, nodal analysis, transmission-line code, netlist

I. Introduction

CASTLE (Circuit Analysis and Simulation with Transmission-Line Emphasis) is an analog circuit schematic entry and transient analysis simulation code designed for Pulsed Power applications. At its core, it is an MNA (modified nodal analysis) code, similar to SPICE (Simulation Program with Integrated Circuit Emphasis) [1] and its modern derivatives, such as the popular LTSpice [2]. CASTLE was created and is maintained by the Pulsed Power Physics Branch (Code 6770) of the Plasma Physics Division at the Naval Research Laboratory. The Windows GUI (Graphical User Interface) version of CASTLE comprises a schematic entry tool, a netlist editor, and a trace plotter. There is now a new library version of CASTLE that allows netlists to be simulated in a variety of ways on both Windows and Linux. The latest versions also have a useful help file and manual.

CASTLE has several features that make it easier to simulate pulsed power circuits, compared to other codes. Perhaps the main difference when compared to SPICE is a fixed time step. The fixed time step in CASTLE has several advantages. First, it avoids the “time step too small” error that often occurs when simulating pulsed power circuits with a SPICE derivative. This error often occurs when there are sudden changes in voltage or current in a circuit. As this is usually a goal in pulsed power circuits, it is often a problem that CASTLE completely avoids. However, one does have to consider the choice of time step value in CASTLE. Usually somewhere in the range of 0.1 to 0.01 ns is a good choice for typical pulsed power circuits.

CASTLE is also largely free of convergence failures, another common problem when simulating Pulsed Power circuits with SPICE derivatives. This error occurs when repeated iterations of a time step do not result in a voltage stabilizing to within a tolerance. There is only one circuit element in CASTLE, the semiconductor diode, whose presence could allow this kind of failure to occur. In our testing, this kind of problem is extremely rare, especially when the reverse breakdown voltage value is appropriately chosen.

Another feature in CASTLE is special consideration for transmission-line (TL) elements. Pulsed Power circuits often contain a string of several TL elements. SPICE and derivatives require an explicit ground connection at each TL connection whereas TL codes do not. CASTLE borrows this feature from TL codes and automatically assigns grounds for simple TL connections. This simplifies schematic entry and avoids a common error and improves the visual appearance. CASTLE also features a TL capacitor and a TL inductor element. These are one (inductor) or two (capacitor) time step elements that replicate the way inductors and capacitors are implemented in TL codes. These elements allow perfect replication of TL code models, allowing one to more easily move a TL code model to CASTLE and verify exact replication. CASTLE also contains special TL based elements such as transmission line transformers. When simulated, these are automatically broken down into single time step elements of varying impedance and internally simulated in the same way that TL codes do for improved efficiency.

One of the most important features of CASTLE is the collection of custom elements with physics based models, such as railgun, pinched-beam diode, rod-pinch diode, etc. The bulk

Work supported by the Defense Threat Reduction Agency (DTRA).

of these models were ported from the Code 6770 TL code, Bertha [3]. Again, the use of a fixed time step allowed for exact replication of the TL code output for validation of the implementation. Most of these models have a custom dialog in the GUI, for setting parameters. Adding new custom elements to CASTLE is relatively easy, compared to other codes.

II. Speed Advantage For Large Circuits

Most of our Pulsed Power circuits are relatively small and simulation speed is nearly instantaneous on modern computers. However, we have encountered the need to simulate very large systems, and CASTLE can offer a significant advantage. In one case, we found that converting a circuit from LTSpice to CASTLE resulted in a simulation speed increase by a factor greater than 150. There are two reasons for this dramatic speed advantage; a unique matrix subdivision algorithm, and the addition of a MPI (Message-Passing Interface) mode. This is despite the CASTLE matrix solver not yet being optimized for speed.

A. Automatic Matrix Subdivision

Automatic Matrix Subdivision is a unique feature in CASTLE that leverages the self-isolating property of transmission lines. The bulk of the computational effort in MNA codes lies in inverting the sparse conductance matrix. The size or order of the matrix, N, is equal to the number of nodes plus the number of voltage sources in the circuit. The number of calculations required to invert the matrix scales with N^X, where x ranges from ~2 (for an optimized sparse matrix solver) to 3 (for a general purpose matrix solver). What CASTLE can do is divide one large matrix into several smaller matrices and hence create the dramatic speed increase for very large circuits.

The way CASTLE models TL elements is equivalent to the "characteristics" method [4] used by SPICE, as shown graphically in Fig. 1. One variation is that CASTLE uses current sources rather than voltage sources to the same effect. The important feature of this model that CASTLE leverages is the fact that the two ends of the TL element are not electrically connected.

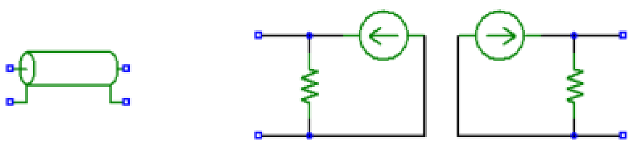

Fig. 1. The schematic model of a TL (left) along with its solver representation (right)

At the start of simulation, CASTLE inspects the netlist to find any sections of the circuit that are isolated by TLs. CASTLE then creates an array of smaller matrices from these sections and solves them during simulation rather than using one large matrix. An example of circuit segment that can be simulated independently with its own smaller matrix is shown in Fig. 3. CASTLE can take the two resistors and the two ends of the TL elements and treat them as an independent circuit for simulation. In some cases, one may want to add TL elements to a circuit in order to realize this speed increase. In most cases with Pulsed Power, there are actual TLs in the physical circuit that can be added to the model for this purpose.

Work supported by the Defense Threat Reduction Agency (DTRA).

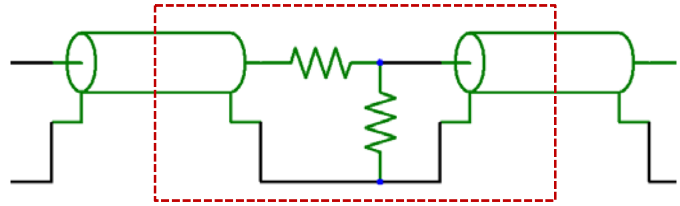

Fig. 2. An example of a circuit section (enclosed by dashed box), isolated by TLs, that CASTLE can automatically simulate with its own, smaller matrix for improved speed.

B. MPI Mode

In the past, the speed of CASTLE was limited by its singly threaded matrix solver. This has now been overcome with the use of an MPI model. New circuit elements, junctions and blocks, have been added for this purpose. These elements allow one to divide a large circuit into sections and have each section run in its own thread. This can improve speed nearly linearly with the number of threads used, limited by the number of cores of the computer's processor. One possible improvement that can be made in the future for CASTLE is to implement a multi-threaded sparse matrix solver, such as the one used by LTSpice.

III. The Schematic Entry GUI

The CASTLE schematic entry GUI has many of the features found in more developed codes, such as LTSpice. Features include the graphical placing of elements and wiring, an optional grid, zooming, text labels, parameter evaluation, sub-circuits, copy and paste, and several sheet sizes. For an example, a complete circuit model for the Gamble II generator is shown in Fig. 3. Historically, this model was created from the original Bertha text file and verified to give the same results. Later, modifications were made to include a new Marx and a constant, 3 Ω, output line.

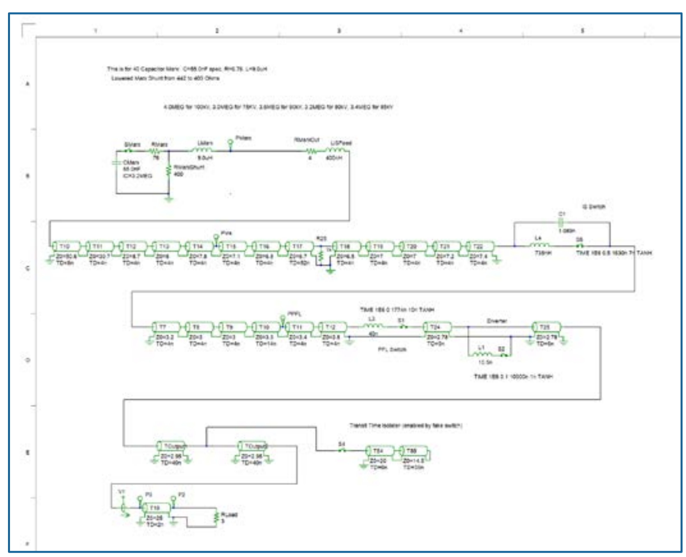

Fig. 3. A screen shot of Gamble II circuit model in the CASTLE schematic entry GUI.

IV. CASTLE NETLISTS

For simulation, CASTLE generates a netlist from the schematic. The netlist can be edited in the CASTLE GUI after generation. One could also create the netlist with an external text editor and then opened by CASTLE, if the graphical schematic is not needed. The netlist is generally very similar to that of SPICE and may even be compatible in simple cases that do not include CASTLE specific features. It is this netlist version of circuits that is used by the .dll and .so versions of CASTLE, described later.

V. CASTLE OUTPUTS AND TRACE PLOTTER

After simulation, all voltages and currents in the model are available to plot within the GUI or to save as text files for use by an external data plotting or analysis program. The plotting capability of CASTLE is presently quite rudimentary. However, CASTLE also has the ability to directly export traces to our groups data capture and analysis program, Stella, which allows one to easily manipulate traces and can produce presentation quality plots. The schematic can include certain control commands, such as ".plot" that enable automatic data plotting and/or export.

VI. SIMULTANEOS COUPLING TO OTHER CODES

CASTLE can be connected to other types of codes for simultaneous simulations. Coupling is enabled by an alternate, granular interface that allows passing of waves back and forth along stub transmission lines. CASTLE can be called by any code that supports transmission-line boundaries. CASTLE can call any code that provides this interface.

For example, CASTLE has been coupled to the PIC (particle-in-cell) code, Chicago. This enables one to more easily drive simulations with more complex circuit inputs. This is aided by both codes using a fixed time step. CASTLE is also integrated into Bertha and vice-versa. So, CASTLE circuits can have Bertha sub-circuits and Bertha can use CASTLE sub-circuits. This allows one to achieve the raw speed of a pure-TL code while modeling key components with the greater accuracy of a circuit code. It would be straightforward to integrate CASTLE into any other TL codes.

VII. NEW WAYS TO RUN CASTLE

Newer versions of CASTLE can run simulations in new ways. As of CASTLE version 4, using the Windows GUI was the only way to run simulations. With CASTLE version 5, a new castle.dll file was made available. The .dll file allows one to run CASTLE netlists from Python, Matlab, and regular C++ codes. This has been used for automated circuit parameter optimization. CASTLE simulations can also be run from the Windows command line, allowing batch processing. Stella now has integrated CASTLE support so that CASTLE netlists can be run from Stella macros.

The very latest, CASTLE version 6, includes a new Linux .so file version that can allow running of simulations on Linux machines. This will allow running of CASTLE simulations on HPC. When combined with MPI, should result in extremely fast simulations of large circuits

VIII. SUMMARY

CASTLE is an MNA circuit simulator, similar to SPICE, but with a fixed time step that avoids simulations errors that are common with Pulsed Power circuits and also allows direct validation with transmission-line circuit codes. CASTLE includes a unique matrix subdivision algorithm and also an MPI mode that allows very large simulations to run > 150X faster than LTSpice, for example. CASTLE includes a GUI for schematic entry with publication quality views. CASTLE has several unique circuit elements for Pulsed Power physics based diode models and also some special transmission line elements. We were able to convert the Gamble II circuit model directly from TL code to CASTLE, verify it gave identical results, and now more easily update it. CASTLE can plot output traces directly, or one can save to Stella or text file. The latest versions of CASTLE include a .dll file that allows CASTLE simulations to be run from Python, Matlab, command line, or just about any program allowing for automated parameter optimization. CASTLE is relatively easily coupled to other codes that employ a fixed time step, such as the PIC code, Chicago. The very latest version of CASTLE includes a .so file that allows simulations to be run on Linux and potentially on HPC (High Performance Computing) platforms.

REFERENCES

[1] SPI Nagel, L., & Pederson, D. (1973). SPICE (Simulation Program with Integrated Circuit Emphasis). UC Berkeley.CE

[2] One Technology Way, [online] Available: http://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html.

[3] Hinshelwood, D D. 1983. "BERTHA: a versatile transmission line and circuit code". United States.

[4] F. H. Branin, "Transient analysis of lossless transmission lines," in Proceedings of the IEEE, vol. 55, no. 11, pp. 2012-2013, Nov. 1967, doi: 10.1109/PROC.1967.6033.

[5] Stella, an NRL data capture and analysis code. Contact: david.hinshelwood@nrl.navy.mil

[6] Chicago is being developed by Voss Scientific with partial support from the Defense Advanced Research Projects Agency under Contract No. W31P4Q-15-C-0055 and the Air Force Office of Scientific Research under Contract No. FA9550-14-C-0034.