A portable and unified approach to control system simulation

Silvano Casoria, Jean Mahseredjian, Raymond Roussel, Julien Beaudry, Gilbert Sybille
IREQ / Hydro-Québec
1800 Lionel-Boulet
Varennes, Québec, Canada
J3X 1S1

Abstract – This paper demonstrates how control system diagrams assembled in Simulink can be extracted and embedded in an external application. The presented external application is EMTP, but the proposed method allows connecting to other transient analysis applications. In addition to achieving portability and unification, the new approach benefits from Simulink's advantages in modeling and simulating dynamical systems. The main test case is the CIGRE-HVDC benchmark.

Keywords: Control systems, HVDC, EMTP, Simulink

I. INTRODUCTION

The most common approach for system control studies is based on large scale applications, such as the EMTP, EMTDC, HYPERSIM, PSB, SABER and SPICE. These are specialized simulation packages with built-in methods for the simulation of control systems. EMTP [1], for example, uses an internal module, named TACS (Transient Analysis of Control Systems), where control systems are represented using generic blocks and solved separately from the network equations. There is no standardized format for exchanging control system diagrams between established software environments. Different block selections can allow simulating the same control diagram; a one-to-one block match between software is not necessarily available. Moreover, in some cases, the simulation of controls might require insertion of extra blocks related to underlying simulator limitations or methods.

EMTP, as other transient analysis packages, has closed-code architecture, but provides built-in methods for communicating with external functions. This is mainly established for advanced user-defined modeling. An external function can be created directly using a programming language or it can be automatically generated from an external application.

Several general purpose mathematical modeling applications are now providing advanced environments for solving complex network problems. MATLAB [2] uses a specialized Toolbox, named Simulink [3], for simulating control systems. Simulink is capable of simulating dynamical systems. It supports linear and nonlinear systems modeled in continuous (variable time-step) time, sampled (discrete) time and a combination of both. Different parts of a control system can be also sampled at different rates. Simulink has a powerful graphical user-interface with a large library of blocks. Since Simulink is built on top of MATLAB, it has access to a wide range of MATLAB based tools. Such tools allow to conceive and optimize Simulink based systems. A typical example is the `rltool` [4] function. It is available through MATLAB's Control System Toolbox [4]. It allows designing a SISO compensator using Root-Locus techniques.

Interfacing Simulink with the EMTP provides several advantages for control system simulation. If the interfacing method can be made sufficiently general and if control system diagrams created in Simulink can be completely extracted, then portability to other applications is achieved. Since MATLAB and Simulink are widely used, creating and simulating control systems through Simulink provides unification.

One other very significant aspect is testing and prototyping. Control systems created in Simulink can be tested in relation with the actual network, using the Power System Blockset (PSB)[5], which is another MATLAB based application. Connecting Simulink to an external specialized application, brings in the large scale simulation capabilities while maintaining advanced expressive power and allowing an efficient design cycle.

This paper presents and tests a DLL (Dynamic Link Library) based interface with Simulink for control system simulation. This method is different from the one reported in [6], it achieves portability, maximizes computational speed and consequently demonstrates new capabilities for solving larger systems. The selected main test case is the CIGRE-HVDC [7] benchmark.

Although this paper is based on EMTP, the demonstrated method is applicable to other simulation packages with compiled user-defined code embedding options.

II. INTERFACING WITH SIMULINK

Simulink is based on a graphical user interface (GUI) where control system diagrams are assembled using a built-in library of control blocks. A control system can be made stand-alone by defining all input sources and connecting all outputs. It can also have disconnected inputs and outputs using Simulink's `Inport` and `Outport` blocks. Such a "disconnected" example (model `trigger.mdl`) is shown Figure 1. It is designed to simulate the current control system for a dc converter. This diagram is ready for connecting to the network simulation on the EMTP side. The inputs (`Inport` blocks) are used to measure ac system voltages and the dc current. The outputs `F1Y` to `F6Y` (`Outport` blocks) are used to send firing signals on EMTP based thyristors. One extra output, named `TALP` is used to send back the firing angle, for visualization purposes only. Simulink allows using multilevel subsystem blocks. Double-clicking on the `firing_signals` block opens the subsystem window shown in Figure 2.

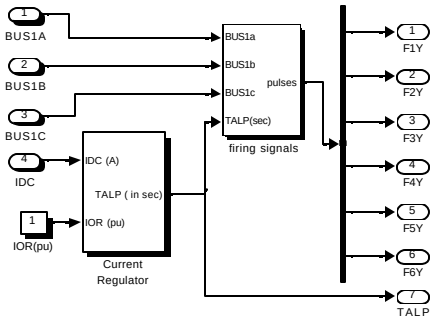

Figure 1 A Simulink diagram (`trigger.mdl`) with `Inports` and `Outports`

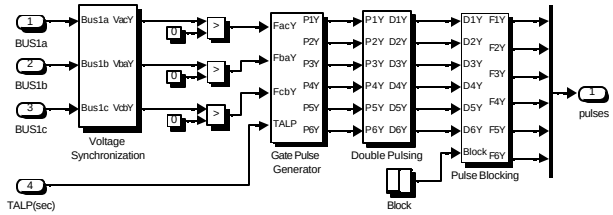

Figure 2 The `firing_signals` block of Figure 1

EMTP has black-box type devices available in TACS for calling external DLL based functions. These devices can have an arbitrary number of inputs and an arbitrary number of outputs. A DLL is a Dynamic Link Library. It can be called from the host program without recompiling and relinking with that program. All is needed is to provide the name of the DLL and its location to the host program. To create a DLL from a Simulink diagram, such as the one shown in Figure 1, it is necessary to utilize the Real Time Workshop (RTW). RTW is an add-on to Simulink. It allows generating C code from a Simulink model. It can be used for non-real-time applications. One more component required to perform the automation presented in this paper is the Target Language Compiler (TLC).

TLC is an integral part of RTW. It allows to customize the generated C code for a given application. TLC is used to transform an intermediate form of a Simulink block diagram into target specific code. The compiler generates its code based on target files. To be more specific, for the model `trigger.mdl` shown in Figure 1, the tab `RTW` in the dialog box `Simulation Parameters`, is shown in Figure 3. When this tab is opened the first time, the user must provide or modify the fields `System target file` and `Template makefile`. When the Build button is clicked, the following actions are taken by the RTW: compiles the diagram to produce `trigger.rtw`; invokes TLC which compiles the TLC program `EMTP-Simulink.tlc` and operates on `trigger.rtw` to produce C code; creates a makefile named `trigger.mk` from the template makefile `EMTP-Simulink-PC.tmf`; produces the DLL code `trigger.dll`. The `trigger.rtw` file is a hierarchical database whose contents provide a description of the individual blocks within the Simulink model. The template makefile is system dependent. In this example it is designed for the Microsoft Visual C/C++ compiler. A different template makefile is used for the Unix version. All `tlc` and `tmf` files are preprogrammed and available to the EMTP user on UNIX and PC platforms.

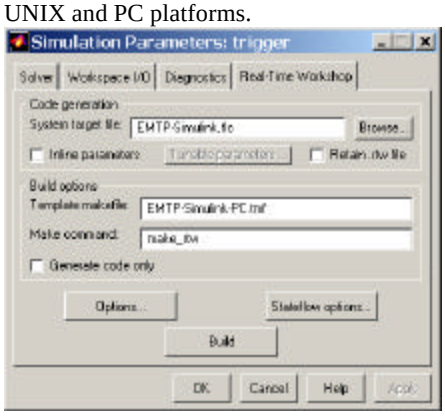

Figure 3 RTW options

Once the DLL is created, it is completely isolated from the MATLAB environment and can be used without the presence of MATLAB on a computer.

Before clicking on the `Build` button, the user must select solver options under the `Solver` tab. The current release of RTW does not support variable step solvers, only fixed step solvers can be selected. In addition to the discrete solver method, a drop-down menu offers some other ODE solvers. Most of those solvers are inappropriate or inefficient for most practical cases, that is why it is recommended to convert the simulated system into a discrete time frame version. The `c2d` (from continuous to discrete) [4] function can be used to make the conversions. One more requirement is to select the fixed time-step value; it must be a multiple or equal to the one on the EMTP side.

The EMTP user identifies `trigger.dll` in the EMTP black-box device data forms and interconnects all necessary inputs and outputs. The rest is automatic.

For calling a DLL from the EMTP FORTRAN code, it is required to use a generic and fixed call statement:

```
CALL DLLModel(dllname,in,out,Dt,step)
```

`DLLmodel` is a fixed C function. It receives the name of the DLL (`dllname`) to load, a vector of input values `in`, the integration time-step `Dt`, the step number `step` and returns a vector of output values `out`. Any number of Simulink models with any number of input and output arguments can be called by simply clicking on the `Build` button of each model and naming it on the EMTP side.

III. EXAMPLE 1

This example is based on a HVDC system model [8] (see Figure 9-7, test case 9-1) originally created with the EMTP. The control system is the one shown in Figure 1. EMTP simulates only the network part. The 6-pulse rectifier bridge is connected to a Yg-Y transformer (grounded on the primary side). The 3-phase ac system is composed of a 3-phase Thevenin equivalent connected to ac filters. The voltage level is 225.5kV on the ac line side and 205.45kV on the valve side. The dc system voltage is 250kV. It is desired to control a current of 1.6kA. Only the rectifier is modeled using thyristors, the inverter side is just

a constant dc source. The smoothing reactor is 350mH. This system has an effective short circuit ratio (ESCR) of about 7. Inside the voltage synchronization block of Figure 2 is an emulation of equidistant firing control scheme. More data on this test case can be found in [8].

The dc current waveform is shown in Figure 4. Three waveforms are superposed in this figure. Waveform 1 is the EMTP-only simulation with a time-step of $50\mu s$. Waveform 2 is the same simulation with the control system modeled entirely in Simulink. Due to the usage of continuous time blocks, the selected solver is ODE4 [3].

The reason waveforms 1 and 2 are not in close agreement at startup (see also zoomed window), is because the Simulink based control avoids an internal numerical delay and is able to startup one time-step earlier. A third waveform (3) with the same data case as for waveform 1, but with a reduced time-step, is used to get a closer match. Full inverter side voltage is applied at $5.5ms$, this is followed by the closing of the current control integrator switch at $25ms$. A dc fault is applied at $50ms$ and the blocking of firing pulses at $100ms$ concludes the last simulated event.

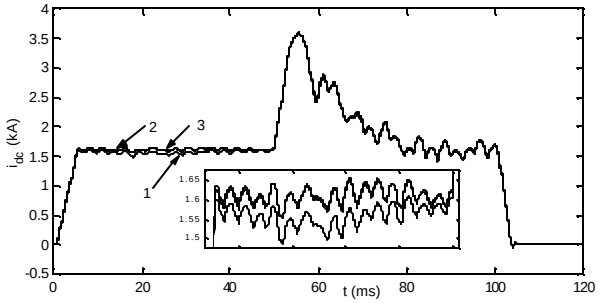

Figure 4 DC current for Example 1, EMTP and EMTP -Simulink simulations

IV. EXAMPLE 2

This example illustrates the modeling of a more complex HVDC transmission link [7]. Perturbations are applied in order to examine control system performance. The complexity of this HVDC model enables, among other things, to provide a reference for testing digital simulators.

The HVDC transmission network is identical to the one detailed in [7] (see system A1, Figure AI-1).

Two test cases are created. Test Case 2.1 (TC2.1) models the complete control system in Simulink while EMTP simulates only the network part. Test Case 2.2 (TC2.2) assembles the entire system in EMTP and simulates the control part with EMTP's TACS module. Identical simulation results are expected.

TC2.1 uses two separate DLLs created from Simulink, one for the rectifier side and for the inverter side. The rectifier control model is shown in Figure 5. Another model with similar functionalities is available for the inverter side.

The control system uses a Discrete 12-pulse HVDC control block available in the controls library of the PSB [9]. It implements commonly used control schemes and related sophistication in HVDC controls. This block can operate as a rectifier or as an inverter. It is a masked block with its dialog box shown in Figure 6. There are no continuous states in this controller. All Laplace domain blocks are discretized using the `c2d` [4] function, the Tustin method and a sample time of $25\mu s$. Tustin discretization is identical to trapezoidal integration.

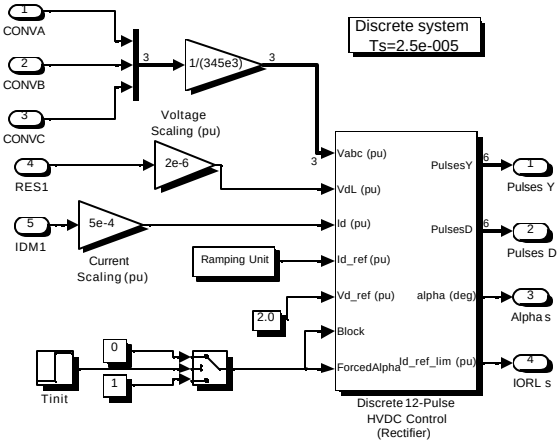

Figure 5 Rectifier control model `rectifier_ctrl.mdl`

Figure 6 Dialog box for the 12-pulse control block of Figure 5 acting as a rectifier

An unmasked view of the 12-pulse control block is shown in Figure 7. The Functions block subsystem is shown Figure 8. These figures allow appreciating GUI capabilities available in Simulink. Block masking is a

powerful feature for encapsulating controllers and providing quick access to modifiable parameters.

A. Inputs and outputs

Referring to Figure 5, input 1 (`Vabc`) to the 12-pulse control block is a vectorized signal of the three line-to-ground voltages measured at the primary of the converter transformer for the 345kV network. These three voltages are used to synchronize the pulse generation on the line voltages. Inputs 2 and 3 are the dc line voltage (`VdL`) and current (`Id`). These inputs are filtered before processing by the regulators. A first order filter is used on `Id` input and a second order filter is used on `VdL` input. Filter parameters are entered in the dialog box of Figure 6.

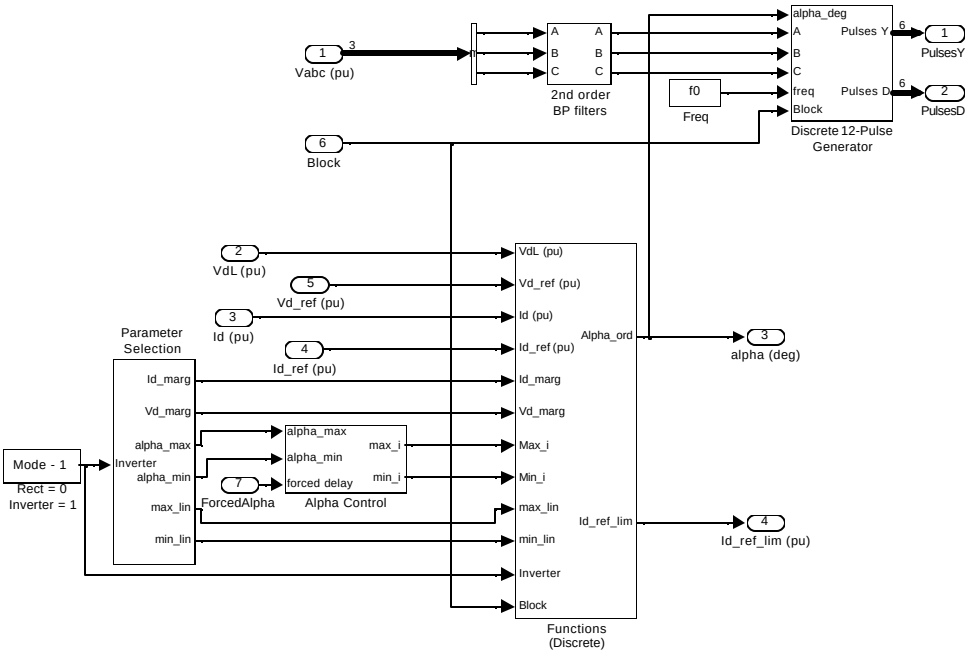

Figure 7 Unmasked view for the 12-pulse control block shown in Figure 5

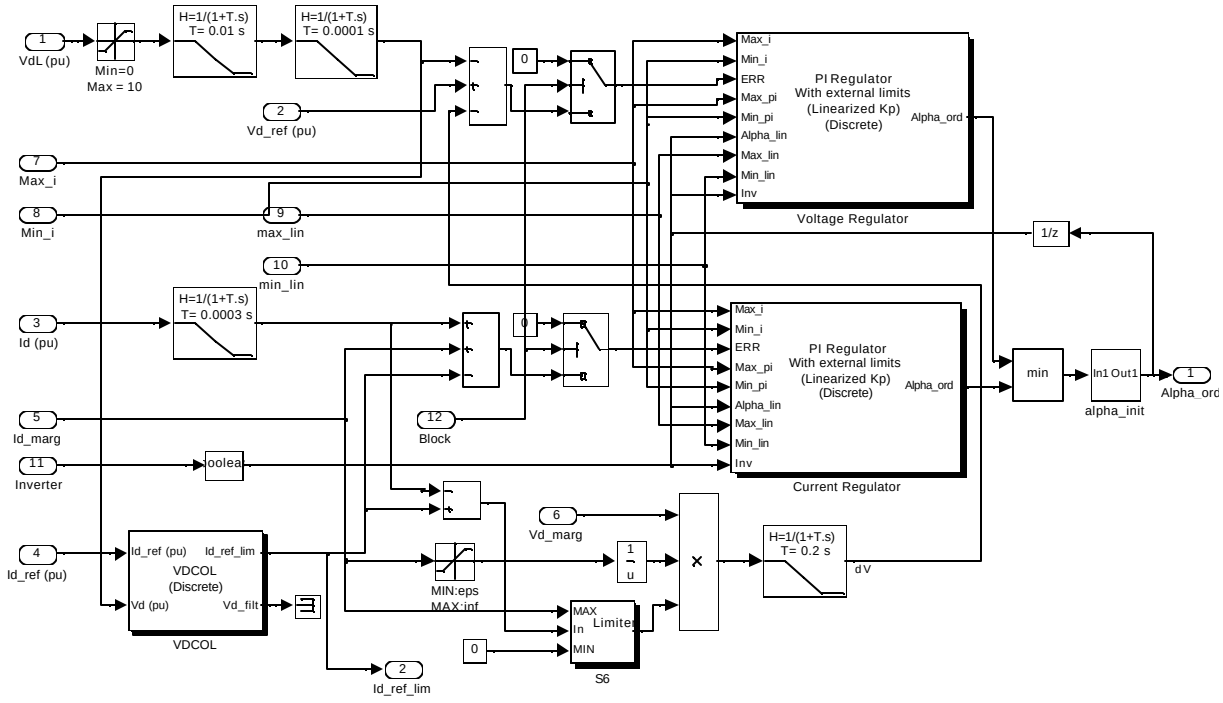

Figure 8 Functions block subsystem shown in Figure 7

Currents and voltages measured in EMTP are per-unitized (1pu current is 2kA and 1pu voltage is 500kV) before they are used in the controllers. Inputs 4 and 5 (`Vd_ref` and `Id_ref`) are the pu dc voltage and current references respectively. In the rectifier of Figure 5, `Vd_ref` is set to a high value to operate in current control mode. Input 6 (`Block`) is a logical signal, it blocks the converter when set to 1. Input 7 (`ForcedAlpha`) is a logical signal used for protection purposes. If this signal is set to 1, the firing angle will be forced to the value defined in the above dialog box.

The first two output blocks (`Pulses_Y` and `Pulses_D`) contain vectorized signals for the six pulses sent to each of the six-pulse converters connected to the Y and Δ windings of the converter transformer. The third output (`Alpha_s`) is the firing delay angle α ordered by the regulator. The fourth output (`Id_ref_lim`) is the actual reference current value, from the VDCOL function explained below. These two are outputs are sent back into EMTP for visualization only.

B. Control functions

Main control functions are:

- Voltage and current regulators of the proportional integral type (PI). The minimum principle is being used.
- Current order limiter depending on the dc voltage (VDCOL).
- Delay angle alpha limiter.
- Current and voltage margins.
- Second-order bandpass filter for ac voltages used for firing pulse synchronization.
- Equiangle (individual phase control scheme) synchronization with an adjustable pulse width and optional double pulsing.

A brief explanation on some of the above control functions is given below.

The 12-pulse control block of Figure 5 implements the steady-state characteristic of Figure 9. In normal operation, the rectifier controls the current and `Id_ref`, whereas the inverter controls the voltage and `Vd_ref`. The `Id_margin` and `Vd_margin` parameters are entered through the inverter dialog box. The system normally operates at point 1 (see Figure 9). However, during a severe contingency producing a voltage drop on the ac network feeding the rectifier, the operating point will move to point 2. The rectifier will be forced to alpha-minimum mode and the inverter will take the current control mode. This model does not implement gamma-control.

Another important control function (see block VDCOL in Figure 8) is implemented to change the reference current according to the value of the dc voltage. VDCOL (Voltage Dependent Current Order Limiter) automatically reduces the `Id_ref` set point when `VdL` decreases, as for example, during a dc line fault or a severe ac fault. Reducing `Id_ref` also reduces the reactive power demand on the ac network, helping to recover from fault. The VDCOL parameters of the 12-Pulse control block dialog box are explained in Figure 10.

The `Id_ref` value starts to decrease when `Vd` falls below a threshold value `VdThresh`. `IdMinAbs` is the absolute minimum `Id_ref`. When `Vd` falls below the `VdThresh`, VDCOL reduces instantaneously `Id_ref`. However, when `Vd` recovers, VDCOL limits the `Id_ref` rise time with a time constant defined by `Tup`.

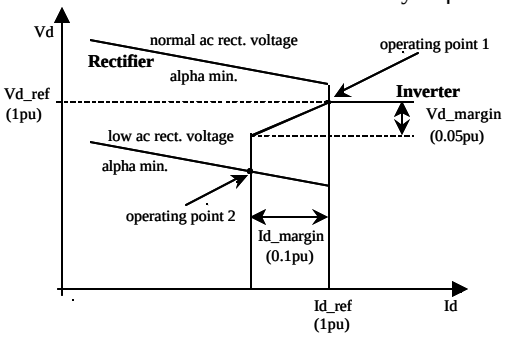

Figure 9 Rectifier and inverter operating characteristic and VDCOL function

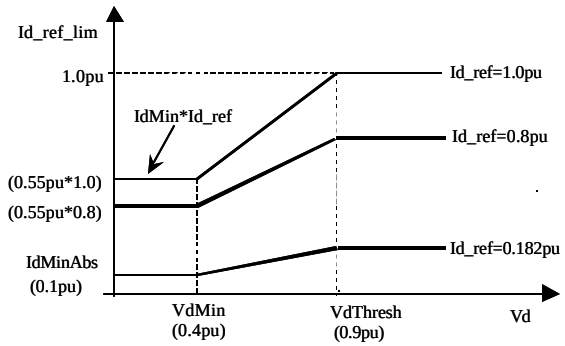

Figure 10 VDCOL characteristic

The rectifier and inverter controls both have a voltage and a current regulator operating in parallel (see Figure 8) and calculating separate firing angles alpha. The effective angle is the minimum of these two angles. Both regulators are of proportional and integral types with gains `Kp` and `Ki`.

Another particularity of the regulator is the linearization of the proportional gain. As `Vd` generated by the rectifier and the inverter is proportional to $\cos\alpha$, the variation in `Vd` due to a change in α is proportional to $\sin\alpha$. With a constant `Kp`, the effective gain would therefore be proportional to $\sin\alpha$. In order to keep a constant proportional gain, independent of α, the gain is linearized by multiplying `Kp` with $1/\sin\alpha$. This linearization is applied for a range of α defined by two limits specified in the dialog box of Figure 6.

C. Simulation results

The control system is programmed to reach steady-state operating conditions at ≈ 1.1s. This is illustrated by the dc current waveform shown in Figure 11. The dc voltage is shown in Figure 12. Both TC2.1 and TC2.2 simulations are superposed on these graphics. They are in very close agreement and computer timings are almost identical. Rectifier and inverter firing angles are shown in Figure 13. They are also confirmed by TC2.2.

Due to a completely discretized model for the control system, the discrete solver option is selected in Simulink's simulation parameters of TC2.1. This is the most efficient and robust approach for simulating large cases. It is also equivalent to the methodology of TC2.2. Both EMTP and Simulink are using a time-step of $25\mu s$.

The reference current follows a ramp from zero to 1pu (2kA) in 1s. The dc current starts to build up at 0.2s, time at which the controller and the pulse generators are deblocked. During the blocked state the error signal to the PI is set to zero. The rectifier controls the current and the inverter controls the voltage. Once steady-state is reached, the firing angles are $13°$ and $141°$ on the rectifier and inverter sides respectively. A 3-phase line to ground fault is applied for 5 cycles at 1.204s at the inverter commutation bus. It is noticed that a commutation failure has occurred during the fault in the inverter. Also, during this period, `alpha_i` reaches its maximum limit of $165°$. The VDCOL function limits the current order to 0.55pu and sets it back gradually to its original value of 1pu, after the threshold value of 0.9pu in the dc voltage has been reached in the recovery period. The system recovers in ≈ 0.7s after fault clearing.

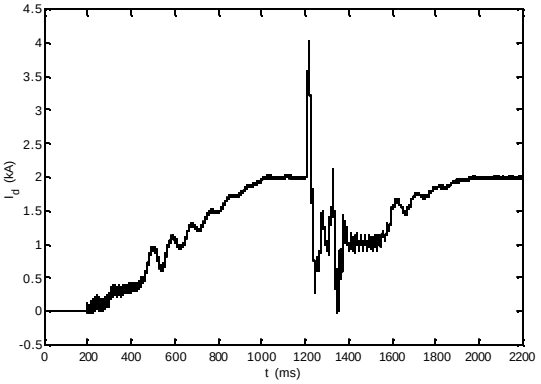

Figure 11 DC current waveforms from test cases TC2.1 and TC2.2

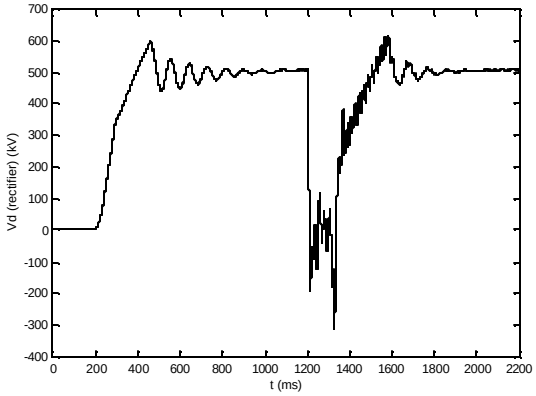

Figure 12 DC voltage waveforms from test cases TC2.1 and TC2.2

Finally, oscillations around 50Hz in dc values are observed during the recovery period. These oscillations are caused by the harmonic interaction occurring between the dc resonance (near fundamental) and the ac resonance (near second harmonic) frequencies. Using a synchronization system of equidistant type, which is less dependent on the commutation voltage zero crossing, may decrease the harmonic interaction effect on the dc.

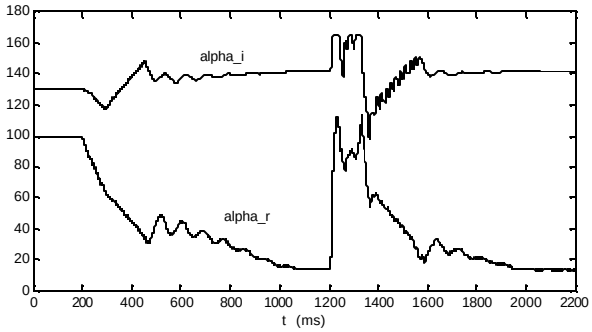

Figure 13 Rectifier and inverter firing angles in degrees for test case TC2.1

V. CONCLUSIONS

This paper has presented a new method for the simulation of control systems in transient analysis of power systems. It is based on creating control system models in Simulink, extracting and connecting to an external application. Extraction is achieved through a Simulink add-on (RTW) allowing converting entire control system models into DLLs.

The new method has been demonstrated using EMTP, but its modular approach makes it portable to other similar applications.

Embedding Simulink in EMTP provides advanced capabilities for control system simulation. The CIGRE-HVDC benchmark has been used to demonstrate such capabilities.

VI. REFERENCES

[1] Electromagnetic Transients Program (EMTP), Development Coordination Group of EMTP.

[2] MATLAB. The MathWorks Inc. Version 5.3.1

[3] Using Simulink. The MathWorks Inc., January 1999, Version 3.

[4] Control System Toolbox, User's Guide. The MathWorks Inc., December 1996.

[5] G. Sybille et al.: Theory and Application of Power System Blockset, a MATLAB/Simulink-Based Simulation Tool for Power Systems, IEEE PES Winter Meeting Conference Proceedings, 2000.

[6] J. Mahseredjian, G. Benmouyal, X. Lombard, M. Zouiti, B. Bressac, L. Gérin-Lajoie: "A link between EMTP and MATLAB for user-defined modeling". IEEE Transactions on Power Delivery. April 1998, Vol 13, Issue 2, pages 667-674

[7] M. Szechtman, T. Wess, C. V. Thio: First Benchmark model for HVDC control studies. Electra No. 135, April 1991.

[8] R. H. Lasseter: EMTP, Workbook IV, TACS. EL-4651, Volume 4, June 1989, EPRI report

[9] Power System Blockset, User's guide, Version 2. The MathWorks Inc., February 2000