Application note

Circuit Decoupling for Dummies

Version: 1.0.1

Last update: 2018-06-15

Author: Christian Dufour

Opal-RT Technologies

Contents

Document revision history

v 1.0	First version	May 30, 2018
V1.0.1	Added distribution example	June 15, 2018

1. Executive summary

This document objective is to explain the various methods used to decouple circuits into smaller parts at the equation level in order to increase the total computational speed.

Some decoupling methods do not introduce any error such as: Distributed Parameter Line models (CP, FD, Wideband) and SSN and are favored for accuracy reasons.

Delay decoupling at DC-bus capacitors is very effective and do not introduce noticeable errors. This method should be used whenever possible.

Stubline are to be used for transformer secondary leakage inductance substitution and other cases with somehow large inductance. Stubline substitution should always be validated by simulation.

Finally, SSN is a good method to decouple circuits, like distribution systems with short line. SSN is however limited in this manner because it uses internal decoupling at the algorithm level (it parallelize the computation of SSN groups within the algorithm). This is approximation-free but less effective than others methods that decouple completely solver tasks.

2. Decoupling accuracy (A) and effectiveness (E)

Many methods exists to decouple power system and power electronic circuits at the equation level:

1- Distributed Parameter Lines (i.e. using Bergeron propagation method). This includes FD and Wideband lines.

2- DC-bus capacitor decoupling with delay

3- Stubline substitution

4- SSN

5- Miscellaneous delay insertion methods.

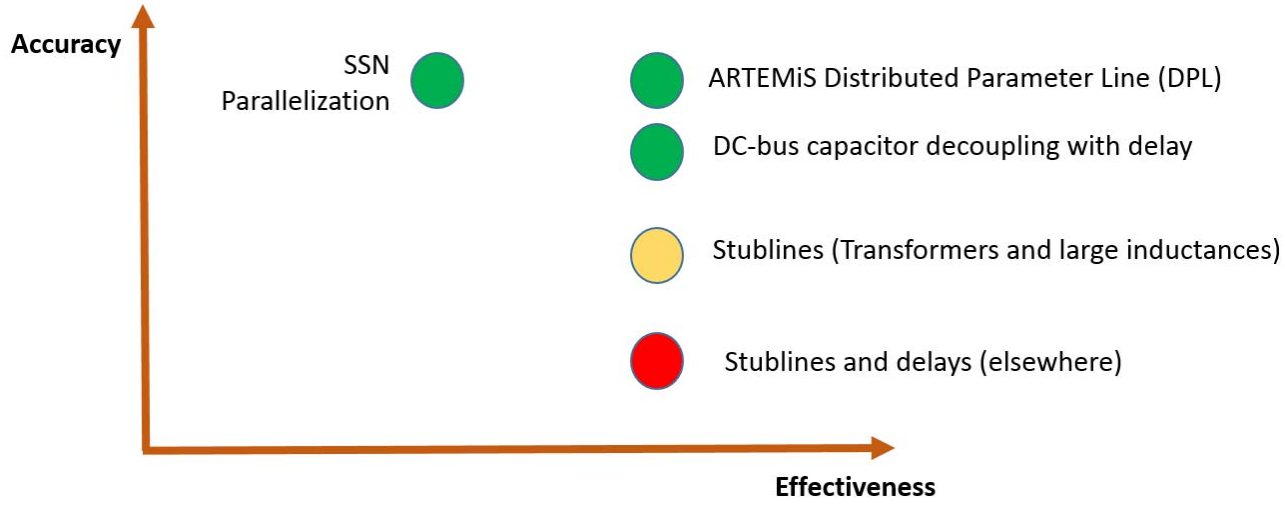

Figure 1: Relative accuracy and effectiveness of common decoupling methods

Figure 1 summarize the relative accuracy and effectiveness of these methods. We describe them next with A: accuracy and E: effectiveness tags.

2.1 Distributed Parameter line models: A++ E++

Distributed parameters line models are the most effective way to decouple network equations because they are not an approximation and they create 2 independent computational tasks at each end of the line.

These line model are based on wave propagation and have embedded delays inside their equations. They include the Bergeron line with losses (also called DPL or CP-line), the modal-domain frequency dependant line model (also called FD-line or Marti-line) and the Wideband line/cable model (also called Universal Line model). All these models are available in ARTEMiS.

The main limitation of these models with regards to real-time is that they require a propagation delay larger than the simulation time step. For a line with propagation speed of 300000km/s, this corresponds approximately to 3.3 µs per km. For a cable with propagation speed of 200000 km/s, this corresponds approximately to 5 µs per km (i.e. a 10 km Wideband cable model having its fastest mode at this speed could run in theory below 50µs of simulation time step in SSN)

Note that in Hypersim, Wideband line implementation requires twice this delay because the propagation task ahs been coded to run on a different core.

2.2 DC bus capacitor decoupling with delay: A++ E++

Most DC bus capacitor have the design objective to maintain constant voltage during device operation. This is the same condition that is required to obtain an accurate simulation using a delayed state-variable.

The trick here is to recognize these DC bus capacitors in circuit. They are found at the input stage of inverters typically or elsewhere in DC grids.

Whenever such a DC bus capacitor is present in a circuit, it should be prioritized when searching for decoupling points in a model.

The ARTEMiS library contains a set of block to conveniently decouple DC bus capacitors in artemis/Tools /Decoupling blocks/chopper

2.3 Stubline: A+ E++

Stubline have the same working equations than CP-line model. The only difference is that the stubline internal capacitance is adjusted to obtain an exact one-time-step delay of propagation, thus providing a decoupling similar to CD-lines. The larger the inductance specified, the smaller the internal capacitance. This equivalent capacitance is visible on the stubline block and is equal to $C = T_s^2/L$ with the time step $T_s$and inductance L.

Stubline are used in substitution to existing inductances in most cases. The most common usage is to substitute transformer leakage inductance by stublines having the same inductance; this application works well most of the time. Stublines can also be used in substitution of large smoothing reactors for example.

The problem of stublines is that they add some equivalent capacitance to circuit when used in replacement of inductances. This is typically visible by the oscillation that stubline will add to some simulations. Users are advised to always compare the stubline substitution method with a reference model.

2.4 SSN: A++ E+

SSN implements some parallelization within its algorithm, this is sometimes called in-step parallelization. This parallelization is less effective than the one of DPL, delays and stubline because the latter creates independent tasks while SSN uses internal threads to compute in parallel part of its algorithm, the group calculations. But the complete circuit simulated by SSN is still using one main task or S-function as all groups are linked with a common admittance matrix solution.

2.5 Other decoupling methods ☹

It is not recommended to add spurious delays or stubline in a model without some serious expertise and/or experience.

Stublines should not be substituting real transmission line because parameters will be different. In addition, the 3-phase stubline model is really 3 single phase stublines without mutual coupling.

2.6 Mixing decoupling methods ☺

It makes perfect sense to use several decoupling methods for a model. A DPL or delayed DC-bus capacitor connected between 2 SSN models will use all the advantages of all the methods: 2 smaller main SSN tasks instead of a bigger one, each of these SSN task using many cores to compute its group equations. DC-bus decoupling should always be preferred to SSN when the choice is possible.

3. Decoupling effectiveness in real-time and offline

These decoupling and parallelization methods will have an impact in both real-time and offline simulations.

3.1 Stubline, delays and DPLs

In general, breaking the complete model in smaller parts produces a gain of speed in most cases. Let's consider state-space based models in the ABCD formulations. If we have a systems with 5 coupled states, it will take 25 multiply and add operation to compute $A*x$ (x is the state vector). If we manage to break this system into smaller parts, the number of mult-add operation decreases.

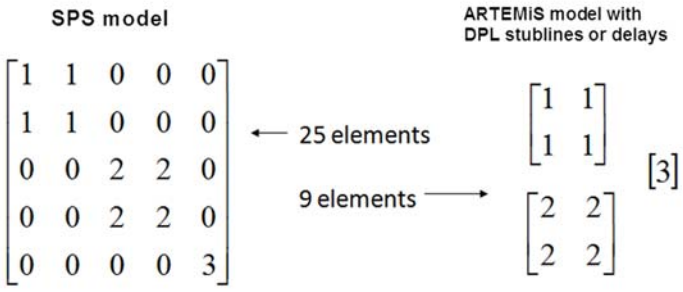

Figure 2: Delay, stubline or DPL decoupling effect on the A matrix of a model

If we take the example a system with 5 states that we decouple into 3 systems of {2,2,1} states (shown above), we go from 25 mult-add to 9 in offline mode.

In real-time, assuming perfect coding and neglecting communication delays and overheads, we obtain even greater gain. In the above example, the 3 subsystems computes in parallel and the worst of them (=2 states) can be taken to compute the parallel computational gain; here, 4 mult-add.

Considering perfectly equal separation of an ABCD system into n parts, the best offline computation gain is equal to n. In real-time, parallel computation of these n parts on m processors/cores can lead to a maximum gain of $n*m$.

This gain is typically less than in theory because of intercore commutation latency and the difficulty to split a model into equal part.

This gain can sometimes be higher than in theory because of processors memory cache effect. Indeed, smaller tasks have a tendency to be executed in core cache memory, which is much faster than main memory.

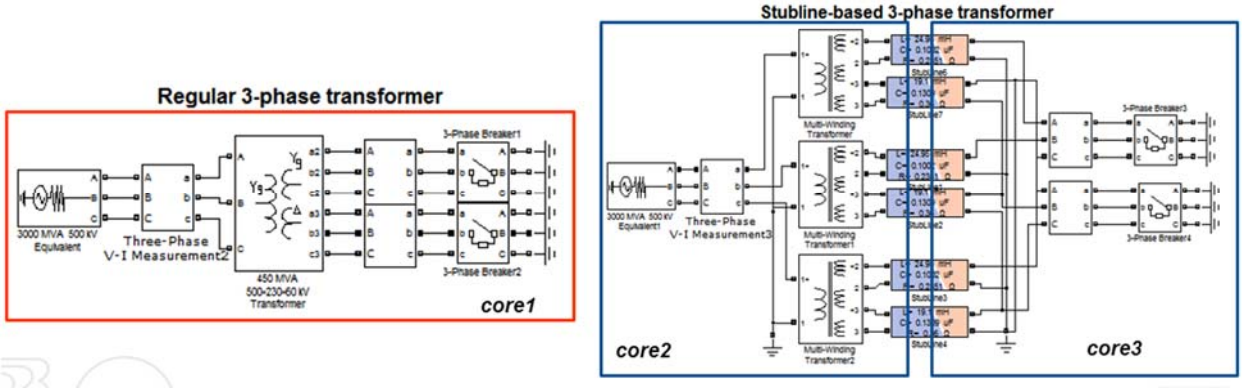

Figure 3: Stubline substitution method for a large grid 3-phase transformer

In the circuit of Figure 3, usage of stublines at the transformer secondary windings help splits the full system of equation into 2 parts; these 2 smaller parts can after be simulated in parallel. Assuming balanced split of equations and perfect hardware (free of communication overhead), the possible speed gain is 2 in offline simulation and 4 in real-time. This model is available in the on-line demo section of ARTEMiS.

3.2 SSN

SSN also provide speed gains by virtue of breaking equations in smaller parts and parallel computation.

Let's take a simple case, below, with x states, y inputs and z switches.

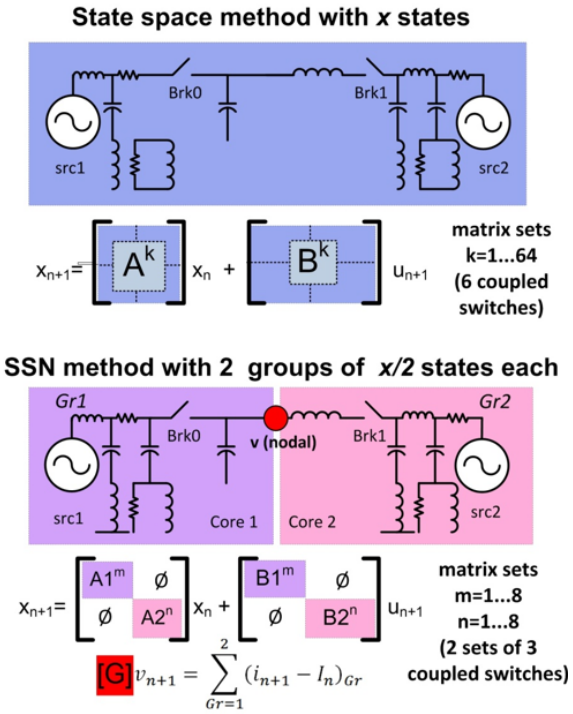

Figure 4: State-space vs. SSN computational costs compared.

Adding a single SSN node in the middle of the circuit has many effects:

1- The number of multiply-add for the state vector update goes from x^2 to $2*(x/2)^2=x^2/2$.

2- The number of multiply-add for the input vector update goes from $x*y$ to $2*(x*y/2)=x*y/2$

3- The number of switch in each group is $z/2$ instead of z for the complete circuit. This is important in real-time applications has circuits permutations are pre-computed. Dividing the switches in different groups can avoid memory overflow in this case if x,y and z are large. Memory requirement for full permutation pre-calculation is $(x^2+x*y)*2^z$ while it is equal to $(x^2+x*y)*2^{z/2}$. With large z, the impact on memory usage is very important.

4- A nodal admittance matrix problem is created in SSN. In the case above with a single node, the solution is trivial. More generally, solving linear system of equation of the YV=I requires LU method which takes $n^3/3$ operations typically (for LDL^T, $2n^3/3$ for standard LU) for large n, n being the number of SSN nodes. Larger n will slow down calculation speed in cubic way therefore.

Therefore, under the condition that the nodal admittance matrix is kept relatively small, perfect split of equation (equal SSN groups) and ideal computational hardware (no communication latency between cores), the SSN offline gain of the circuit above is 2 in offline mode and 4 in real-time.

In reality, several factors will decrease these gains:

a- The nodal admittance matrix LU or LDL^T solution time is not negligible in most cases. Above 25 nodes typically, this can become dominant. This greatly depends on the circuit however because of the numerous optimization of the SSN factorisation algorithm.

b- The SSN parallelisation methods uses threads for group calculation, all linked to a main SSN process (instead of independent processes in stubline/delays/DPL parallelization). This makes a more intensive use of communication links between parallel cores in modern CPUs. These threads also share some common memory in hardware and cache memory trashing phenomenon can occur sometimes.

3.2.1 Notes on SSN

We also found experimentally that in the current implementation of SSN and RT-LAB (as of 2018), parallelization gain tends to saturate above 5 cores per SSN task.

The LU/LDL^T factorization part of the SSN algorithm is not parallelized in the current implementation.

Finally, parallel SSN computation is available only for real-time simulation on RT-LAB, on RT-Linux running targets. Other decoupling gains are available on all platforms.

3.3 Analysis and optimization of decoupling

ARTEMiS provides some feedback on decoupling and parallelization effectiveness during offline simulation. It is aimed at hinting the user about good real-time modeling.

Main indicators are: total memory usage and number of operations. These indicators can be viewed in the Diagnostics pane of the Simulink model. A listing is showed here for a model comprised of a single SSN model with 3 groups.

…

(1) ARTEMIS-SSN: approx. memory required: 0.48334 Mb (including nodal matrix)
SSN group info
Group 1 : 2 states, 4 inputs, 5 outputs, 0 switches.
Group 2 : 6 states, 10 inputs, 17 outputs, 6 switches.
Group 3 : 6 states, 13 inputs, 17 outputs, 6 switches.
SSN nodal matrix is of rank 4 (0 % of zeros) (0 prefactorized col/rows)
(2) ARTEMIS-SSN: LU/LDLT factorization/repeat solution operation count: 18
(3) ARTEMIS-SSN: State-space operation count: 993

…

Item (1) show the estimated memory requirements on the real-time target. Smaller memory span means faster calculation in cache memory. The key to keep this number low is by avoiding groups with large number of switch. Here we have 2 groups with 6 switches (-> 2^6=64 permutation, fine). Number of switch per group should be kept ideally below 9-10 (depending on the number of state, input and outputs). SSN throws an error if it detects more than 15 switches (2^{15}=32768 permutations) inside a group.

The other indicators (2) and (3) shows the number of operation for the state-space and nodal admittance parts. These number are affected by the SSN separation mainly. Bigger groups produces more states-space operations. Smaller groups made by using more SSN nodes will increase the LU factorization operation count. Since the nodal solution has a number of operation proportional to n^3 (n number of nodes), it can grow very fast in systems with large number of SSN nodes.

Note that this listing is printed for all decoupled sub-networks of the model so it is important to take into account all the listings when optimizing a model globally.

4. Examples

4.1 Complex train drive system with AC/DC feed.

The following model is a complex drive model with AC/DC feeder and several inverters and loads, including several motors, here PMSM.

The first thing to notice in this model are the numerous DC bus capacitors. They are used to create decoupled drive partitions in RED. There are 2 remaining sub-circuits in the model, in BLUE and GREEN; they are to be simulated by 2 SSN tasks, one with 12 SSN nodes and the other with only 1.

RT-LAB separation: all sub-circuits can be simulated in dedicated core in RT-LAB. In addition, the SSN ones can used several core each to solve their equations.

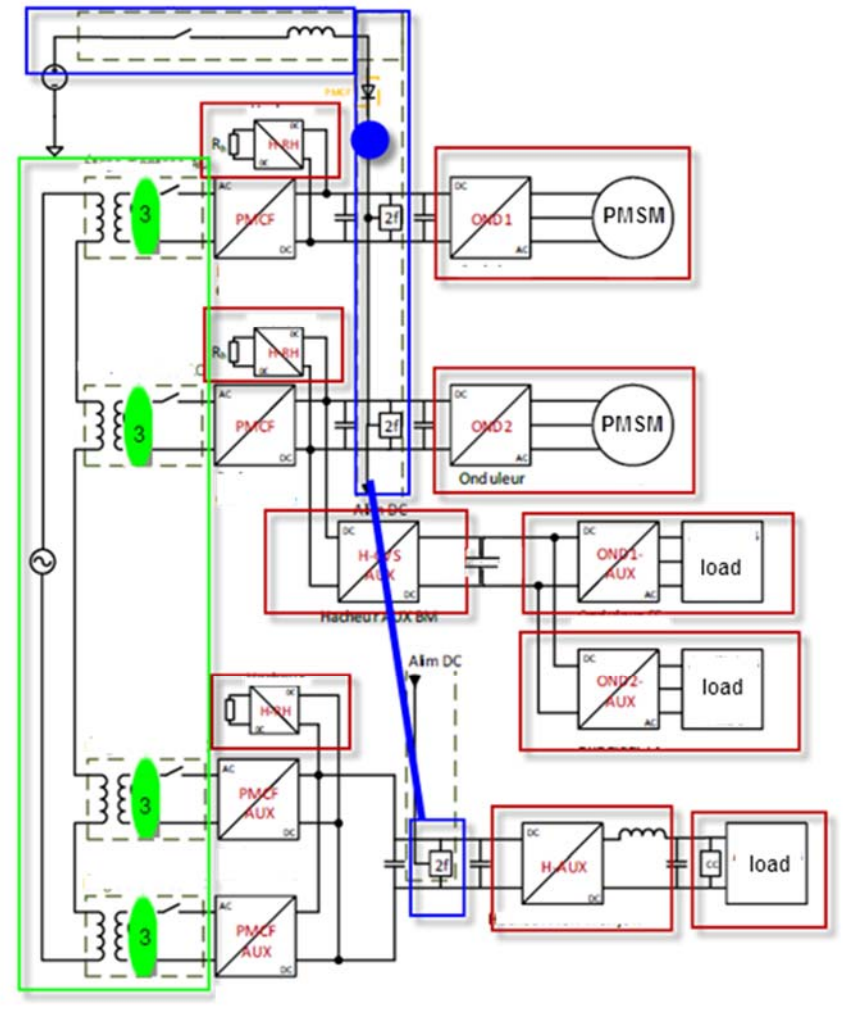

Figure 5: A complex drive system and decoupling into small simulation tasks.

4.2 Distribution power system

Typical distribution systems are composed of very short lines and therefore only SSN decoupling should be used [2][3].

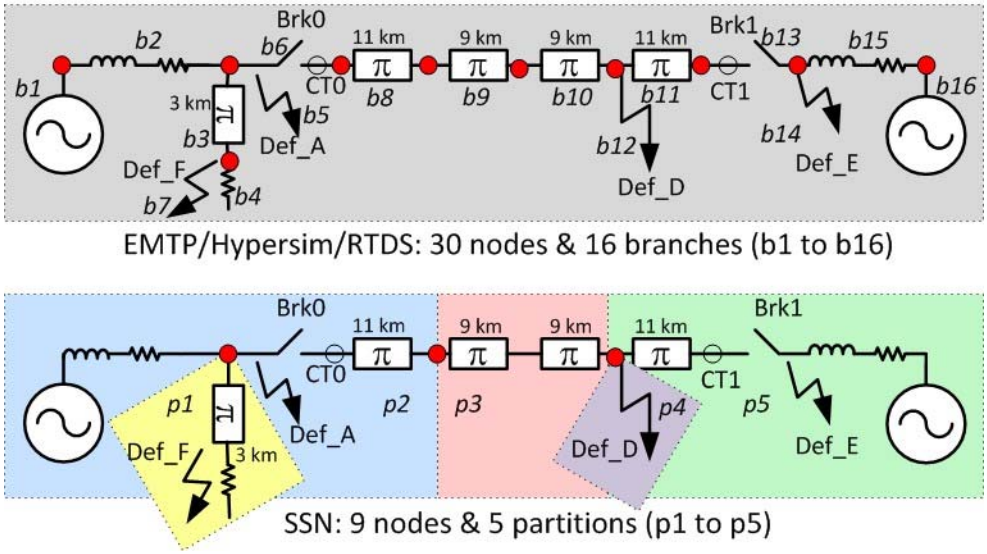

Figure 6: Small distribution system

Figure 6 shows such a small distribution system decoupled into 5 SSN groups using 9 SSN nodes. (It also compares the node usage in standard nodal solvers on the top of the image).

Using stublines for this model in place of short lines would result in very bad accuracy.

4.2.1 Short line modeling: pi-line vs. coupled inductance.

The model of Figure 6 is available in 2 versions actually in the ARTEMiS on-line demos: one with pi-lines and one with coupled inductors instead of pi-line. (**ssn_distributiongrid_A1.mdl ssn_distributiongrid_A2.mdl**) The second one is equivalent to neglecting the very small capacitance of short distribution lines.

The use of coupled inductances will increase the simulation speed and is recommended in study case involving very small capacitance (ex: <1e-11F in total) or in voltage regulation studies. In many cases derived from transient stability software, the capacitance data is not available anyway.

5. Notes

5.1 CPU vs. FPGA

It is to be noted that in FPGA models, the effect of adding delays is a model is much less important because of the typical time steps below 1 µs. Adding a delay at 1µs has much less impact than at 20µs (typical CPU time step). Even for stublines, the computed internal capacitor (and therefore introduced error) will by much smaller for the same inductance at such 20X lower time step.

SSN still does not exist on FPGA. The main complexity here is with making LU factorization on FPGA.

6. References

[1] C. Dufour, J. Mahseredjian , J. Bélanger, “A Combined State-Space Nodal Method for the Simulation of Power System Transients”, IEEE Transactions on Power Delivery, Vol. 26, no. 2, April 2011 (ISSN 0885-8977), pp. 928-935

[2] C. Dufour, J. Bélanger, "On the Use of Real-Time Simulation Technology in Smart Grid Research and Development", IEEE Transactions on Industry Applications, Volume 50, Issue 6, Nov/Dec 2014. Print ISSN: 0093-9994, Online ISSN: 1939-9367, Digital Object Identifier: 10.1109/TIA.2014.2315507

[3] C. Dufour, G. Sapienza, “Testing 750 node distribution grids and devices using optimized parallel delay-free real-time solvers and modern grid protocols” 2015 International Symposium on Smart Electric Distribution Systems and Technologies (EDST), Vienna, Austria, Sept. 8-11, 2015