I. Troch, F. Breitenecker, eds. ISBN 978-3-901608-35-3

Simulation of Faults in Electric Drive Systems with Modelica®

Dietmar Winkler, Clemens Gühmann
Technische Universität Berlin, Germany

Corresponding author: D. Winkler, Technische Universität Berlin, Chair of Electronic Measurement and Diagnostic Technology, 10587 Berlin, Germany; Dietmar.Winkler@tu-berlin.de

Abstract. For simulating electric drive systems the `freeFOClib` (short for "free Field-Oriented Control library") is being developed. This Modelica[1] library can be seen as an extension to the *Modelica Standard Library* (MSL). The library can be used to build a field-oriented control system for existing machine models from the *Modelica Standard Library*, investigate the impact of electric faults (e.g., battery faults, inverter faults, machine faults) on a electric drive system, and run simulations to estimate the fuel consumption of hybrid electric vehicles. The paper will give some information of the content and structure of the library. Measurements from a drive test-bench are retrieved to verify the model parameters and the simulation results, especially the fault simulation results.

Keywords: Fault simulation, Electric faults, Motor drives, Modelica, Free libraries.

1 Introduction

The simulation of systems is a very useful method to investigate scenarios and procedures which will give prior unknown experimentation results. Thus we can test if a certain experiment will damage the test equipment or even worse might prove to have dangerous impacts on test-engineers.

Especially fault scenarios are the ideal application field for simulation runs up front. We like to know hat happens if certain devices fail and perhaps derive security measures which will protect our applications if a certain fault occurs.

In automotive applications there are a huge number of electric motors involved. Some of them bigger some smaller. Some with a more severe impact on the safety of the car and especially its passengers. Taking into account that electric motors are now also used for traction (hybrid or pure electric vehicles), braking (brake-by-wire), and steering (steer-by-wire) a lot of (expensive) tests have to be carried out.

As noted before, simulation can help here but the models must represent signals from different physical domains (e.g., electrical, mechanical, thermodynamical). The modelling language Modelica was especially developed to simplify the simulation in different physical domains within one simulation model. The multidomain capability allows us to build simulation models of hybrid electric vehicles easier than with other simulation tools. Furthermore it allows us to concentrate on the physics of a model rather than building models which represent mathematical equations which in turn then represent the actual physical behaviour [1, 2].

This paper will introduce some of the physical equations to model electrical induction machines using a equivalent circuit diagram. We show how these equations can be transformed into Modelica syntax. A short overview of the to be published library `freeFOClib` is given and some example fault simulations are shown.

2 Mathematical fundamentals of electrical induction machines

The classical approach to describe an electrical machine is by the use of the so called *equivalent phase circuit diagram*. In the following the machine equations shall be explained taking an asynchronous induction machine as an example. There are however much similarities to the synchronous machine which shall not be investigated further in this short paper (see for example [3]).

2.1 Physical equations

Figure 1 represents the equivalent circuit for one phase of the electrical machine. It consists of resistances representing the ohmic losses in the field windings (R_s and R_r) and the stray inductance ($L_{s\sigma}$ and $L_{r\sigma}$) in the stator and the rotor. The rotor and stator are coupled via the mutual inductance (L_m).

[1] Modelica® is a registered trademark of the Modelica Association which develops this free modelling language → www.modelica.org

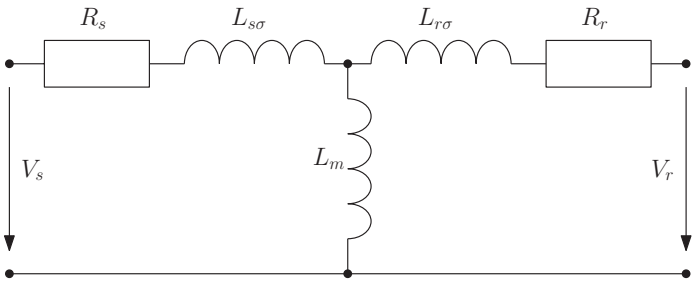

Figure 1: Equivalent circuit diagram of an electrical induction machine

The standard machine equations for the rotor and stator voltages can be written as:

$$\vec{v}_s = R_s\vec{i}_s + L_{s\sigma}\frac{\mathrm{d}\vec{i}_s}{\mathrm{d}t} + \frac{\mathrm{d}\vec{\Psi}_{ms}}{\mathrm{d}t} \tag{1}$$

$$\vec{v}_r = R_r\vec{i}_r + L_{r\sigma}\frac{\mathrm{d}\vec{i}_r}{\mathrm{d}t} + \frac{\mathrm{d}\vec{\Psi}_{mr}}{\mathrm{d}t} \tag{2}$$

where Ψ_{mx} are the flux vectors and defined as

$$\vec{\Psi}_{ms} = \mathbf{L_{ss}} \cdot \vec{i_s} + \mathbf{L_{sr}}(\gamma) \cdot \vec{i_r} \tag{3}$$

$$\vec{\Psi}_{mr} = \mathbf{L_{rs}}(\gamma) \cdot \vec{i_s} + \mathbf{L_{rr}} \cdot \vec{i_r} \tag{4}$$

In the equations (3) and (4) you can see the four inductance matrices. Each of these have the dimension $(m \times m)$ where m stands for the number of phases (three in most cases). $\mathbf{L_{ss}}$ and $\mathbf{L_{rr}}$ represent the self-inductance matrices for the stator and the rotor, respectively.

$$\mathbf{L_{ss}} = \begin{bmatrix} L_m & L_{sm} & L_{sm} \\ L_{sm} & L_m & L_{sm} \\ L_{sm} & L_{sm} & L_m \end{bmatrix} \tag{5}$$

and

$$\mathbf{L_{rr}} = \begin{bmatrix} L_m & L_{rm} & L_{rm} \\ L_{rm} & L_m & L_{rm} \\ L_{rm} & L_{rm} & L_m \end{bmatrix} \tag{6}$$

If the the machine is symmetrically built then the matrices can be further simplified by providing the equations:

$$L_{sm} = L_{rm} = L_m \cdot \cos\left(\frac{2}{m}\pi\right) \tag{7}$$

This also means that stator and rotor self-inductance would be identically.

We still got the two matrices $\mathbf{L_{sr}}(\gamma)$ and $\mathbf{L_{rs}}(\gamma)$ left. These represent the mutual coupling between the stator and the rotor side and vice versa. Unfortunately the coupling effect depends on the rotor position γ. For a three phase machine $(m = 3)$ this gives:

$$\begin{aligned} \mathbf{L_{sr}}(\gamma) &= [\mathbf{L_{rs}}(\gamma)]^T \dots \\ &= \begin{bmatrix} \cos(\gamma) & \cos(\gamma + \frac{2}{3}\pi) & \cos(\gamma - \frac{2}{3}\pi) \\ \cos(\gamma - \frac{2}{3}\pi) & \cos(\gamma) & \cos(\gamma + \frac{2}{3}\pi) \\ \cos(\gamma + \frac{2}{3}\pi) & \cos(\gamma - \frac{2}{3}\pi) & \cos(\gamma) \end{bmatrix} \end{aligned} \tag{8}$$

2.2 Implementation in Modelica

With the equations for the machine model derived all what is left is the translation into the modelling language. Implementing the equations (5), (6), and (8) in Modelica is straight forward:

```
// stator self inductance:
Lss = fill(fill(Lsm, 3), 3)
  - diagonal(fill(Lsm - Lm, 3));
// rotor self inductance:
Lrr = Lss;
// mutual inductance s->r:
Lsr = Lm .*
  {{cos(gamma),cos(gamma - 2/3*pi),
    cos(gamma + 2/3*pi)},
   {cos(gamma + 2/3*pi),cos(gamma),
    cos(gamma - 2/3*pi)},
   {cos(gamma - 2/3*pi),
    cos(gamma + 2/3*pi),cos(gamma)}};
// mutual inductance r->s:
Lrs = transpose(Lsr);
```

Now the flux equations (3), (4) and the voltage equations (1), (2) need to be transformed into Modelica code:

```
// mutual fluxes:
psi_ms = Lss*i_s + Lsr*i_r;
psi_mr = Lrs*i_s + Lrr*i_r;
// mutual voltages:
v_ms = der(psi_ms);
v_mr = der(psi_mr);
```

As you can see in Modelica you can simply take the physical equations and write them down in Modelica syntax. No further reordering or manipulation is required. This makes modelling much easier and the modeller can grasp the idea behind a certain model much quicker when looking at the code.

3 Purpose of the library

The Modelica language is specified in the so called Modelica Specifications [4] and comes with the free *Modelica Standard Library* (MSL) [5] which contains a huge collection of models for different physical domains (e.g., electrical, mechanical, thermodynamical).

For the simulation of machines the Modelica Standard Library contains a sub-library called: `Modelica.Electrical.Machines` [6]. This library contains basic three-phase models of asynchronous and synchronous machines as well as DC machine models. To control these machines the modeller still has to provide his own controller models.

So in order to simulate more complex electric drive applications a new "free Field-Oriented Control Library" (`freeFOClib`) is being developed [7].

4 Library structure

The `freeFOClib` should allow the user to model and simulate all aspects of the an electric drive. A standard electric drive normally consist of components like power sources, power electronics, controllers, electric machines, and interfaces. The communication of the blocks can be done either via the classic approach by use of input and output connectors or by the use of so-called bus signals. The bus structure orients itself on the new bus structure of the Vehicle Interfaces Library [8]. This, for example, should allow easier simulation of power-train simulations of hybrid electric vehicles.

Figure 2 shows a graphical representation of the uppermost hierarchy level of the library.

The library consists of:

- `UsersGuide`: Every Modelica library should contain this. It gives the user information on how to use the library as well as some information about the release history and participating developers.
- `Examples`: To get the user going some example simulation models are included. There are sub-packages of examples for the different parts (e.g., complete drive systems, batteries, inverters, machines)
- `Batteries`: This is a sub-package that contains different battery models (currently NiMH, Li-Ion).

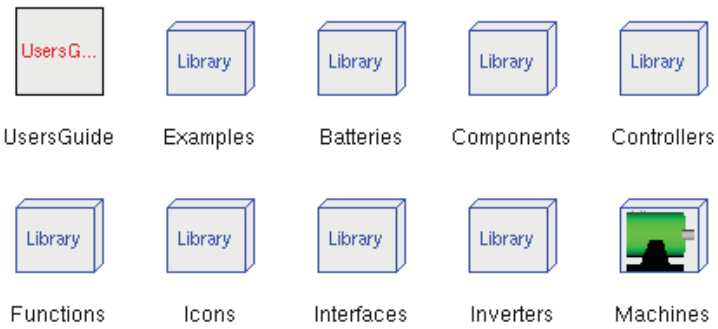

Figure 2: Top-level packages of the `freeFOClib`

- `Components`: Models which are library-wide used and therefore do not fit exclusively into any of the other sub-packages are placed here.
- `Controllers`: In here controllers for the control of electric drives together with the necessary flux models are placed.
- `Functions`: Custom functions which are used by the `freeFOClib`.
- `Icons`: Special icons for the models.
- `Interfaces`: The Interfaces sub-package includes different interface models. Mostly they are made of a partial type so that one can simply extend from the interface model which fits the application most.
- `Inverters`: These models are used to transform the control signals into electrical signals which can then be applied to the machine models.
- `Machines`: This library contains models of synchronous and asynchronous machines of different types.

In addition to the different controller types for the field-oriented control as well as the battery models (important for automotive applications) also new machine models for fault-simulations have been developed. These are using the m-phase presentation where the faults can be introduced directly into the components without the need of a $d-q-0$-transformation.

5 Library applications

The development of the `freeFOClib` was started with specific purposes in mind:

- field-oriented control of induction machines
- fault-simulations to investigate electrical and mechanical impacts of machine faults
- state of charge estimations for batteries in HEV applications
- investigate adaptive controller algorithms for electric machines

In automotive applications, for example, often the term drivability of a car is used. With drivability the car manufacturers often relate to the overall operating qualities of the power train. This could include things like idle mode characteristics, throttle response, and acceleration capability. In a hybrid electrical vehicle for example we got an electric motor acting directly or indirectly on the drive train.

The `freeFOClib` contains an example model which allows for simulation of three different types of faults. These could be faults of the battery, faults in the inverter, and faults in the machine. In Figure 3 an example of an electric drive system is depicted.

5.1 Inverter and battery faults

To simulate faults in the battery and/or in the inverter, the example model in Figure 3 contains a fuse component `fuse_DC`. This model disconnects the DC current when a surge current is detected that violates the maximum rating of the battery. The surge-proof fuse is not triggered right away but after a first order delay time which can be parametrised. Whenever the fuse is triggered the inverter gets a signal which will switch of the firing signals for the inverter bridge in turn.

Another kind of battery fault would be a short circuit of the supply side of the inverter. This is accomplished by a simple switch that is triggered by a Boolean signal and which connects both support voltage connectors of the inverter.

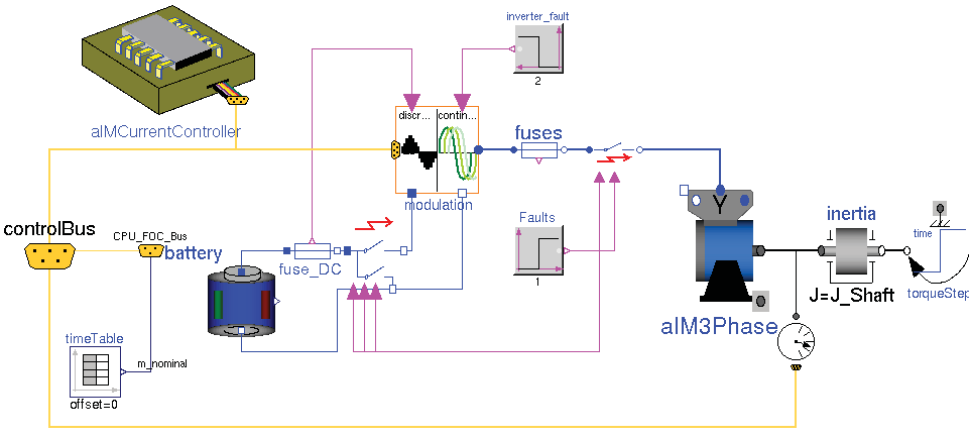

Figure 3: Example model of an electric drive system.

And at last a fault of firing signals can be applied directly via a signal `inverter_fault`. The combination of the different switches gives the ability to build even more fault scenarios.

5.2 Machine faults

In the electric machine models of our library the following fault scenarios can be simulated:

- open-circuit of a stator phase (e.g., a connecting cable is broken)
- short-circuit of one or more phase windings (e.g., insulation failure because of thermal stress within the stator or rotor)
- short-circuit phase to ground (e.g., insulation failure because of mechanical damage)

Each of these faults will have some influence of the torque produced by the electric drive.

5.2.1 Open-circuit fault

Here an example simulation of a synchronous machine having a open-circuit fault. This could occur if a connecting cable comes loose. In Figure 4 you can see the three phase currents and the mechanical torque depicted over time just before and after the connection of one phase was opened at the time of two seconds.

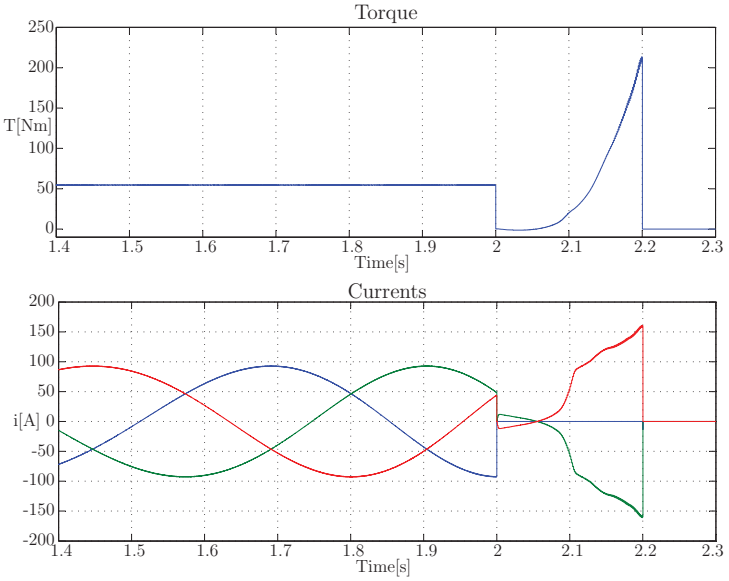

Figure 4: Fault scenario: open-circuit of one stator phase

As you can see, the torque drops instantaneously and but the controller try to keep the torque output until at some point the fuses for the remaining phase currents get triggered.

5.2.2 Short-circuit within a phase winding

In Figure 5 a fault of the insulation between the phase windings of a stator coil is modelled. Such a fault can be caused by, for example, over-temperature or overload which in turn leads to an overheated stator winding. This behaviour is modelled by reducing the inductance value abruptly by 20 percent. In Figure 5 you can see the three phase currents and the mechanical torque over time just before and after the connection of one phase was opened at the time of $T = 2\,sec$.

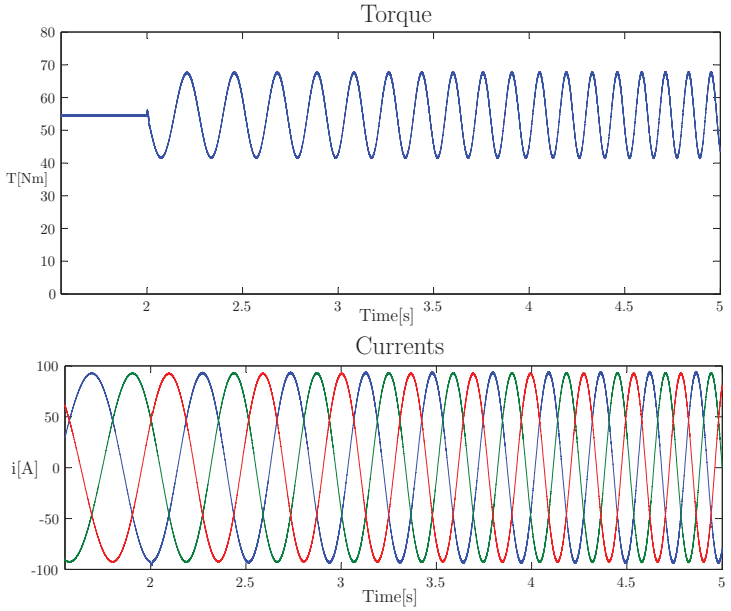

Figure 5: Fault scenario: short-circuit of phase windings

At first sight the electrical impact seems not to be very drastic. However since the field-oriented control now calculates the wrong control values the torque starts to oscillate quite considerable. If this electric drive is applied in a hybrid electric vehicle, for example, this could lead to reduction of drive comfort. But not only this, depending on the mechanical system such oscillation could become unstable and cause major damage.

5.3 Field-oriented control loop

Having a variety of different flux-models available allows the modeller to investigate different kind of control strategies for electric drives. So for example by varying some of the controllers parameters (e.g., the machine rotor resistance) one can test how robust the drive control behaves at a certain rotor speed when using different flux-models for the estimation of the flux position.

6 Electric drive test-bench

On important aspect of every simulation is the verification of the model parameters and the simulation behaviour in general. For this a test bench consisting of a asynchronous induction machine (the test object) as drive and a DC-machine as load coupled together via a torque-meter. The torque-meter an analogue voltage signal for the load torque. This signal is then read in via high-speed analogue-digital converter card. For the speed signal an incremental resolver is used which is attached to the machine shaft. The TTL signal of this speed sensor is then used by the electric inverter, a *Siemens SINAMICS G120* for the field-oriented control algorithms.

Especially validating the fault simulation behaviour is quite a challenge. No one would like to destroy the test equipment whilst doing the actually test run. The validation of test models is therefore done in two steps

6.1 Validation of machine parameters

This measurements can be undertaken without any harm to the test equipment. Standard methods are used to evaluate machine parameters such as rotor inertia (J), stator and rotor resistances (R_s and R_r), stator and rotor stray inductance ($L_{\sigma s}$ and $L_{\sigma r}$) and the mutual inductance (L_m), to name only a few.

6.2 Validation of fault simulations

With the parameters of the machine evaluated now we use the simulation to investigate the fault behaviour first. This way we can learn what peak values of currents, voltages, torque, and speed will arise. The simulation can already tell us if a certain quantity will exceed the allowed boundaries during a test procedure. We then can adapt the planned test procedure, do another test simulation to avoid a possible violation of the allowed operational region of our test bench system.

Only if all fault simulations show that no violations will occur, we then can proceed to do the actually test run of the faults. Deviation between the measurements and simulation can then be used to further improve the simulation models.

7 Conclusions

In this paper we have presented fault simulations of electric drives. It could be shown how some faults influence the drive's behaviour. Without any adaption of the control algorithms major damage can occur on the drive and the controller itself. With the help of simulation different safe-guard methodologies can be tested and the most appropriate then implemented. Some of the simulation and measurement results were already shown in the conference presentation. These results were used to improve the parametrisation and validation of the physical modelling.

Modelica has been a great help in terms of usability and effort needed for the modelling of the physical behaviour of the whole drive system. The component wise, object-oriented structure makes it possible to reuse the models in different applications with no or only minor modifications.

8 Future work

We are steadily improving the library by adding new drive controllers, new battery models, for example. The developed models of unbalanced electrical machines are going to be a part of the library as well as all components which are necessary for a field-oriented control (hence the name).

The `freeFOClib` will be released as free library which everybody can use and adapt. The idea is to provide a Modelica® library which gives the possibility to model a modern electrical drive system and still leaves the user the freedom to look at the underlying code and perhaps even contribute improvements to it.

The next step is to include a fault simulation in the drive-train of a hybrid electric vehicle. Here work which was already done in a different project, called *Test-Bench of the future* [9], will be integrated. Different scenarios can be tested to investigate the mechanical and electrical impact of electric faults on drive train and the power system, respectively. This gives an idea of what could happen and how severe the impact would be on the drive experience and as a result improve safety functions of control units.

9 References

[1] M. Tiller, *Introduction to Physical Modeling with Modelica*. Kluwer International Series in Engineering & Computer Science, Kluwer Academic Publishers, 2001.

[2] P. A. Fritzson, *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1*. No. 0-471-471631, John Wiley & Sons, Januar 2004.

[3] D. Novotny and T. Lipo, *Vector Control and Dynamics of AC Drives*. Clarendon Press Oxford, 1996.

[4] Modelica Association, *Modelica® – A Unified Object-Oriented Language for Physical Systems Modeling – Language Specification*, version 3.0 ed., 5th September 2007.

[5] Modelica Association, *Modelica® - Free library from the Modelica Association*, version 3.0 ed., 2008.

[6] C. Kral and A. Haumer, "Modelica libraries for dc machines, three phase and polyphase machines," in *Proceedings of the 4th International Modelica Conference* (G. Schmitz, ed.), pp. 549–558, Modelica Association, March 7-8 2005.

[7] D. Winkler, E. Bakhach, F. Döring, and S. Rinderer, "`freeFOClib` - A free Field-Oriented Control library for Modelica." unreleased, see `www.freefoclib.org` for any news.

[8] M. Dempsey, M. Gäfert, P. Harman, C. Kral, M. Otter, and P. Treffinger, "Coordinated Automotive Libraries for Vehicle System Modelling," in *Proceedings of the 5th International Modelica Conference* (D. C. Kral and A. Haumer, eds.), vol. 1, (Vienna), pp. 33–41, Modelica Association, arsenal research, September 2006.

[9] D. Winkler and C. Gühmann, "Hardware-in-the-Loop simulation of a hybrid electric vehicle using Modelica/Dymola," in *The 22nd International Battery, Hybrid and Fuel Cell Electric Vehicle Symposium & Exposition* (E. . Secretariat, ed.), (Yokohama, Japan), pp. 1054–1063, EVS, Japan Automobile Research Institute, 23-28 October 2006.