

LUT University
School of Energy Systems
Degree Programme in Electrical Engineering

Simo Kuusela

ELECTRIC DRIVE SIMULATOR FOR SYSTEM SIMULATIONS

Examiners : Professor Jero Ahola
M.Sc. Jarkko Lalu

ABSTRACT

LUT University
School of Energy Systems
Electrical Engineering

Simo Kuusela

Electric drive simulator for system simulations

2019

Master's Thesis

61 pages, 36 figures, 6 tables

Examiners: Professor Jero Ahola

M.Sc. Jarkko Lalu

Keywords: electric drive, variable frequency drive, induction motor, common DC, DC link, simulation

ABB Virtual Drive is a software tool for simulating electric drive system. In this thesis different use cases and challenges for this software were studied. New and improved electric drive simulator was created and integrated to Virtual Drive as a solution for many of the challenges found. The simulator included simulation of variable frequency drive, induction motor and load. Special attention was given to realizing a common DC system and being able to simulate a multi-drive system in real-time using the new simulator. The simulator was compared to ABBs high fidelity simulator and was found to produce very similar results while being almost hundred times faster to run.

TIIVISTELMÄ

LUT-Yliopisto
School of Energy Systems
Sähkötekniikka

Simo Kuusela

Sähkökäyttösimulaattori systeemisimulointeihin

2019

Diplomityö

61 sivua, 36 kuvaa, 6 taulukkoa

Tarkastajat: Professori Jero Ahola

DI Jarkko Lalu

Hakusanat: sähkökäyttö, taajuusmuuttaja, induktiomoottori, yhteinen välipiiri, simulointi

ABB Virtual Drive on ohjelmistotyökalu sähkökäyttösystemien simulointiin. Työssä tutkittiin erilaisia käyttökohteita ja haasteita kyseiselle ohjelmistolle. Ratkaisuksi moniin löydettyihin haasteisiin Virtual Drive ohjelmistoon luotiin ja integroitiin uusi ja parempi sähkökäyttösimulaattori. Simulaattori sisälsi taajuusmuuttajan, induktiomoottorin ja kuorman simuloinnin. Simulaattorin luomisessa erityishuomiota sai yhteisen välipiirin systeemin toteuttaminen, sekä mahdollisuus simuloida usean taajuusmuuttajan systeemiä reaaliajassa. Simulaattoria verrattiin ABB:n korkean tarkkuuden simulaattoriin ja tuloksista havaittiin uuden simulaattorin tuottavan hyvin samanlaisia tuloksia, vaikka sitä oli lähes sata kertaa nopeampi ajaa.

ACKNOWLEDGEMENTS

First of all, I would like to thank ABB and Jarkko Lalu for providing the topic for this thesis. I would also like to thank Jarkko Lalu for giving lots of ideas and motivation throughout this work. From ABB I would also like to thank Aki Penttinen for his comments and help with the programs and tools that were used in this thesis. From LUT University I would like to thank D.Sc. Markku Niemelä and D.Sc. Tuomo Lindh for providing helpful comments and guidance whenever I needed it. Additional thanks to Tuomo Lindh for providing the opportunity to work under his supervision and giving encouragement throughout this thesis work.

Simo Kuusela

November 2019

Joensuu

TABLE OF CONTENTS

1	INTRODUCTION	3
1.1	VIRTUAL DRIVE	3
1.2	MOTIVATION AND GOALS	5
2	SIMULATOR CONSIDERATIONS	7
2.1	VIRTUAL DRIVE ARCHITECTURE	7
2.2	VIRTUAL DRIVE USE CASES	9
2.3	SIMULATOR CHALLENGES	14
2.4	MODELS AND SIGNALS BETWEEN THEM	16
2.5	CONCLUSION	19
3	INTERNAL ABB SIMULATORS	22
3.1	OLD ABB SIMULATOR	22
3.2	CURRENT ABB SIMULATOR.....	23
4	NEW SIMULATOR FOR VIRTUAL DRIVE	24
4.1	PER UNIT VALUES.....	24
4.2	VECTOR CONTROL	26
4.3	VFD HARDWARE	31
4.3.1	Common DC link.....	34
4.4	MOTOR	35
4.5	LOAD	38
5	SIMULATOR COMPARISON	40
5.1	STEADY STATE.....	42
5.2	ACCELERATION.....	43
5.3	DECELERATION.....	48
5.4	SIMULATION SPEED.....	53
6	INTEGRATION TO VIRTUAL DRIVE.....	54
6.1	COMMON DC LINK.....	55
7	CONCLUSIONS	58
	REFERENCES.....	60

LIST OF SYMBOLS AND ABBREVIATIONS

FEM	finite element method
FMI	Functional Mock-up Interface
HW	hardware
PWM	pulse-width modulation
RMS	root mean square
VD	Virtual Drive
VFD	Variable frequency drive
VSI	Voltage source inverter

C	Capacitance
f	Frequency
I	Current
L	Inductance
pu	Per unit
p	Pole pair number
R	Resistance
t	Time
T	Torque
U	Voltage
ω	Rotational speed
Ψ	Flux

1 INTRODUCTION

Use of simulation in development of all kinds of industrial systems have become more and more important. Simulating a system in its development phase can be used to test the system before any part has been ordered and it helps optimizing the design and choosing most suitable components. It helps testing out possible problematic scenarios and problems with the design. It can also be used for training the operating personnel and monitoring the system for malfunctions or analyzing lifecycle of different parts. Therefore, simulation can help reduce risks, make the development phase faster and operation of the system more efficient. Also, the bigger the system, the bigger are the difficulties for optimizing its performance, energy efficiency and life cycle of its parts. Thus, simulation is especially useful for big and complex systems that are hard to design and operate efficiently.

Complex systems like a production line of a factory usually has different physical domains like electrical, mechanical and hydraulic. In the middle of the different domains there is often an electrical motor converting electricity to mechanical or hydraulic power and usually the motor is driven by a variable frequency drive (VFD, also just called *drive*). Motor and drive ties together automation software and electrical and mechanical domains and it can be argued that these are the most important parts to simulate in many kinds of systems. For these simulations ABB has created software called Virtual Drive (VD).

1.1 Virtual Drive

ABB Virtual Drive is a simulation software made for simulating a drive system — a system consisting of a drive, electric motor and a load. As the name implies it allows users to create, parametrize and run virtual drive systems. Simulated drives run the same firmware code as the physical devices to make them resemble the real device as well as possible. Figure 1.1 shows ABB Drive Composer commissioning tool connected to a VD instance.

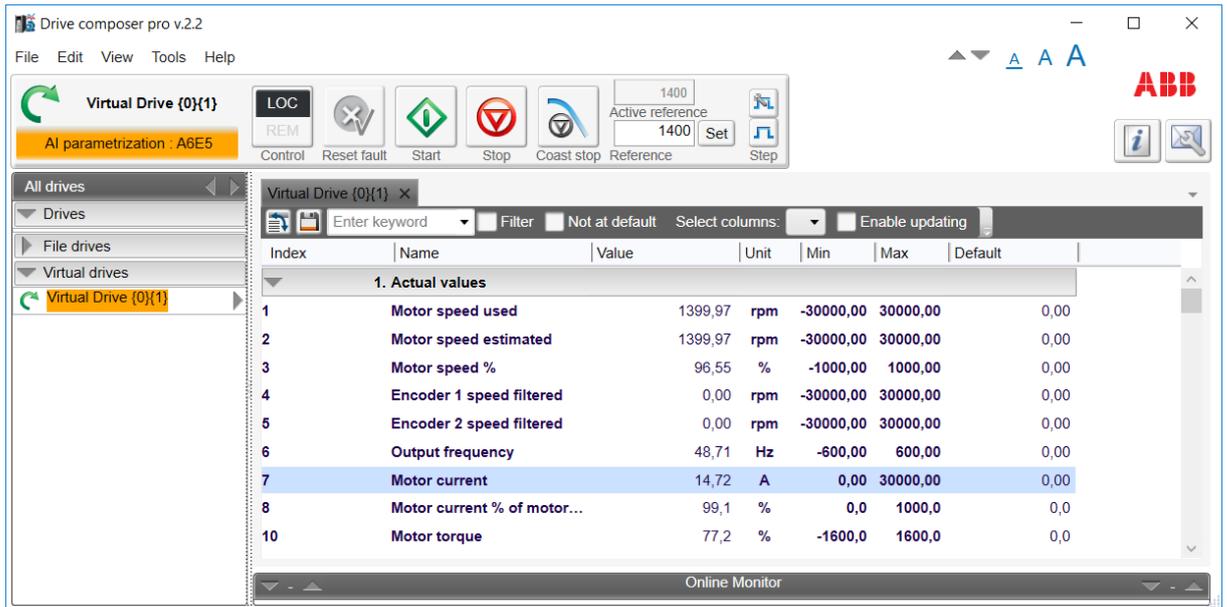


Fig. 1.1 Drive Composer program connected to a Virtual Drive instance.

As can be seen from figure 1.1 VD instance looks the same as a real drive in Drive Composer and you can change its parameters, give it speed or torque reference and see it's mechanical and electrical properties changing accordingly. This makes it possible to test different parameters for a drive and see how they affect its behavior. VD also has different virtual interfaces, like analog and digital inputs and outputs, which can be used to control the drive for example with automation software. There are also interfaces for connecting VD to other simulation tools and programs which can be used when simulating bigger and more complex systems like shown in figure 1.2.

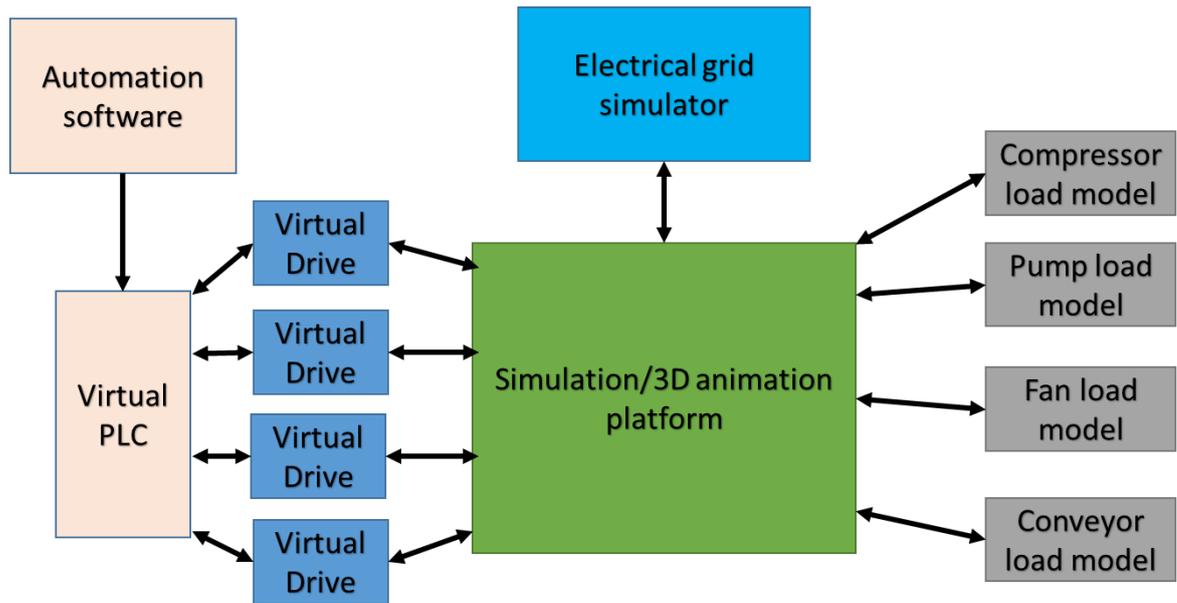


Fig. 1.2 Virtual Drive as part of system simulation.

The system shown in figure 1.2 could represent for example a production line of a factory which has multiple drives and different types of loads. The simulation/3D animation platform in the figure could for example be ABB RobotStudio which integrates all the different programs and models and provides ability to show 3D animations of the system. The animations could help visually see how a system works and could also provide feedback to load simulations. Because in many cases VD users have already selected simulation tools most suitable for their design domain, it's important that VD can be connected to them. One common way to connect simulation programs is by using Functional Mock-up Interface (FMI), which is a standardized interface made for this purpose. FMI is one of the interfaces that VD is also going to support.

1.2 Motivation and goals

Currently Virtual Drive has very simple and linear mechanical and electrical models of a motor and modeling of a drive is very basic with for example DC link voltage being fixed. Because VD aims to be as realistic as possible there is a need for making improvements in both the motor and drive modeling. In bigger systems with multiple drives (like in figure 1.2), often DC links of the drives are connected together, so simulating DC link voltage is a

must have improvement. VD also aims to be a flexible simulation tool that allows users to connect it to other simulation software or be able to use their own load models with it. So, in general there is also a need to make simulator in VD more modular to allow flexibility in how it can be used and interfaced with.

The main goal of this thesis is to make improvements to VD simulations. First step of this goal is to look at different VD use cases, consider general challenges for VD and find out what kind of improvements could be needed. From these considerations it was concluded, that a new drive system simulator should be created for VD. Finally, after realizing the simulator it was verified by testing it against ABBs high fidelity simulator.

2 SIMULATOR CONSIDERATIONS

For making improvements to VD simulations, it's necessary to first look at the general architecture of the software to see how the simulator part of VD connects to the rest of the software. Different use cases for VD are then looked at and general challenges regarding the VD simulator are discussed. Some possible solutions are then considered and finally conclusion is made about the next goal for this thesis.

2.1 Virtual Drive architecture

From the perspective of this work the general architecture of VD software can be split into two major parts: VFD firmware and the simulator. Simplified architecture diagram of VD is shown in figure 2.1.1 where these two parts can be seen, as well as how VD connects to external programs.

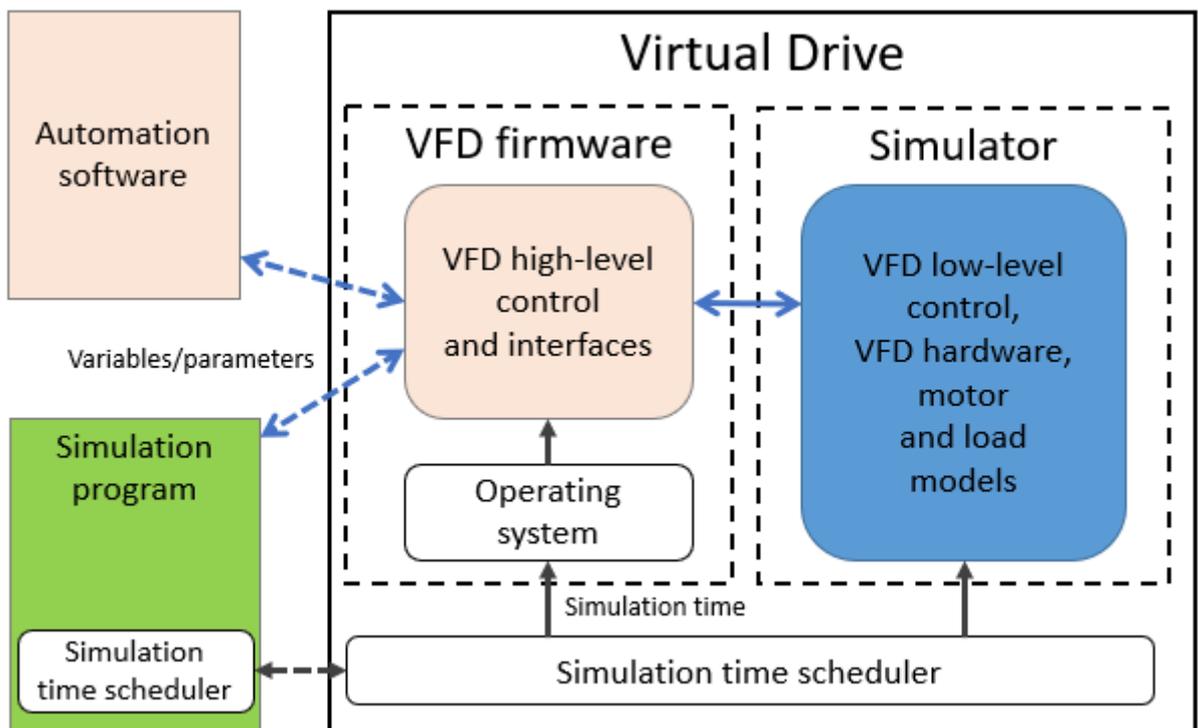


Fig. 2.1.1 Simplified diagram of Virtual Drive architecture and how it connects to external programs. Blue lines represent exchange of variables/parameters and black lines represent simulation time.

The function of the simulation time scheduler inside VD is to provide simulation time to VFD firmware and the simulator and run them in sync. As seen in figure 2.1.1 the VFD firmware includes the operating system, VFD high-level control and interfaces. These are the same ones that are used with the real devices. What isn't included in it though, is the real scalar and drive control software, to keep them safe from reverse-engineering. The high-level control in this case means speed control, reference ramping, limit checking etc. The behavior of high-level control is affected by the control parameters like torque limit or speed ramp time. Depending on if scalar or vector is chosen in the drive parameters, it will output frequency or torque reference to the simulator for execution. As such this frequency or torque reference is the main input to VD simulator. There are also many control parameters that should influence the simulated controls behavior and are also given to the simulator. Simulator outputs for example motor speed, torque, current and DC link voltage back to the VFD firmware. These could then be seen for example in Drive Composer just like in figure 1.1.

When VD is connected to external programs, it's done by connecting them to the VFD firmware interfaces as seen in figure 2.1.1. When VD is connected to some automation software, it could receive for example torque reference and control parameters from it and report back motor speed and torque. When connecting to a simulation program, VD and the program would exchange simulation variables like motor torque or VFD output voltage, which would depend on the simulation program type. They should also exchange simulation time information to keep in sync. To do it, either VD or the other simulation program would be the simulation master, which would then distribute simulation time and steps to the other program. VD simulators models and structure are shown in figure 2.1.2

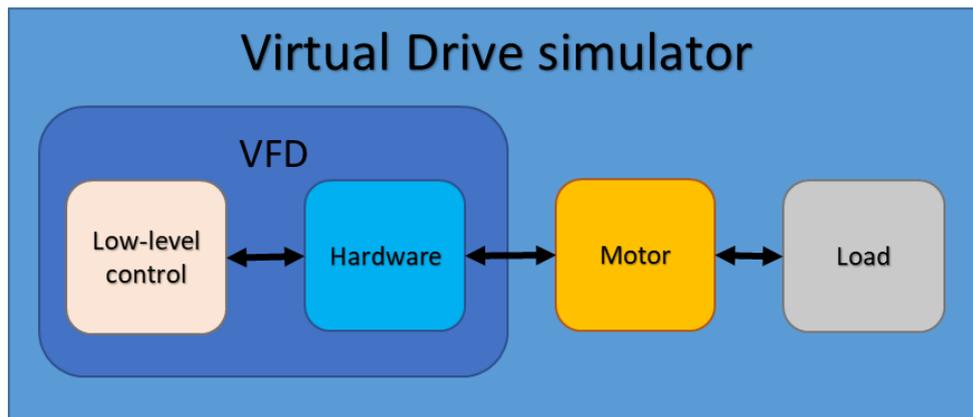


Fig. 2.1.2 Virtual Drive simulator structure and models.

As seen from figure 2.1.2 VD simulator includes models of VFD, motor and load. The VFD model can further be split to low-level control and hardware models. The torque or frequency reference mentioned earlier would be the input to the VFD low-level control model which can be thought as the starting point of the simulator. The control model would then simulate the control type chosen for the drive e.g. scalar or vector control, and then control VFD hardware to produce voltage to the motor which would then drive the load. The simulated variables of these models would then be sent back to VFD firmware.

2.2 Virtual Drive use cases

Use cases here will be separated based on different ways VD could be used and how it could be connected to other programs. The use cases represent some current and future possibilities with the program. There are also different variations and combinations of the use cases shown here, but they should show the most common ways VD could be connected. The simplest use case for VD is using it as standalone which is shown in figure 2.2.1.

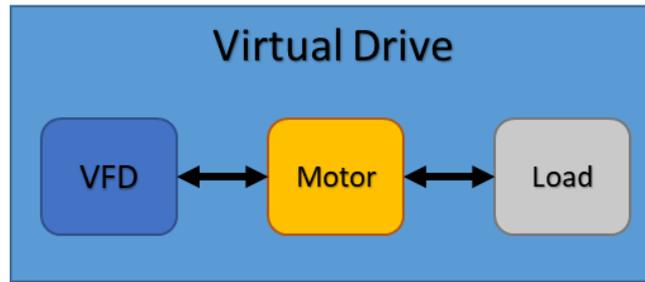


Fig. 2.2.1 Virtual Drive used as standalone where it simulates all parts of a drive system.

As figure 2.2.1 shows when used as standalone VFD, motor and load models are all simulated inside VD. For example, this would be used for fast general testing or learning basic drive concepts and seeing how drive setup generally works. Next use case would be connecting VD to external simulator or 3D animator program as shown in figure 2.2.2.

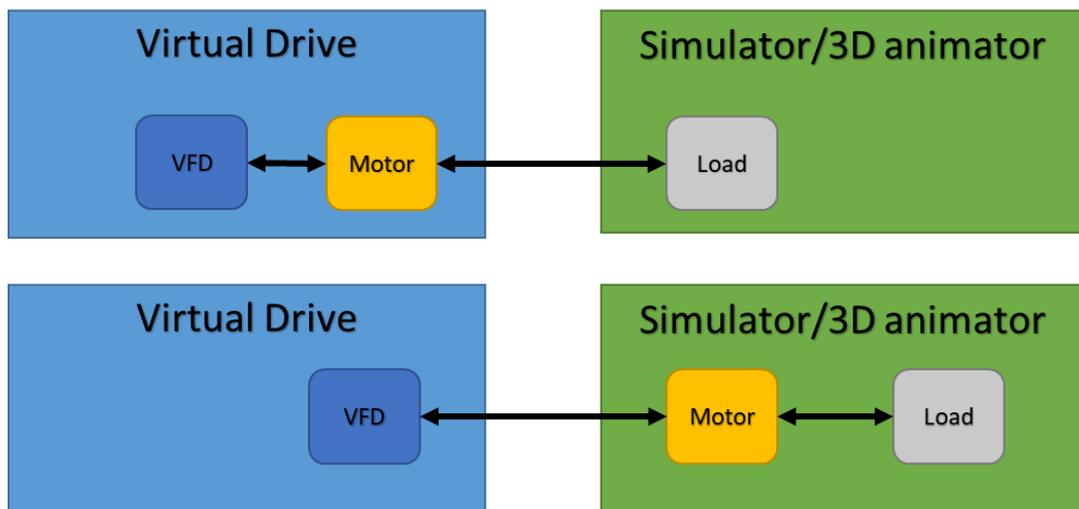


Fig. 2.2.2 Virtual Drive connected to external simulator or 3D animator.

As shown in figure 2.2.2 VD could be connected to external program differently depending on what is modeled in the external program. For example the external program could simulate both motor and load while VD simulates only VFD. This and the following use cases come up for example when users want to see how a drive would work with their system, test out different components and see how they work together. Another use case for VD would be to connect it to an electrical grid simulator as shown in figure 2.2.3.

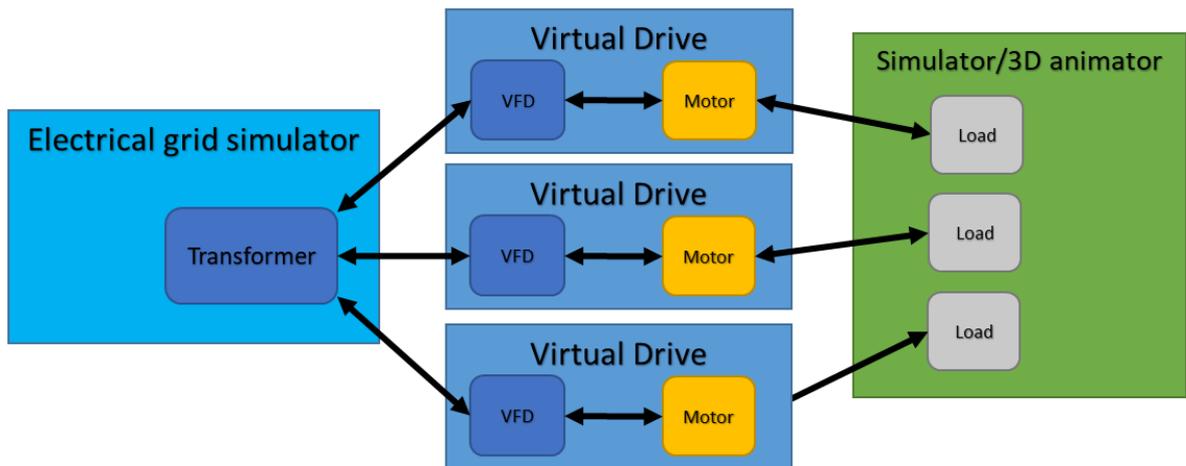


Fig. 2.2.3 Virtual Drive connected to electrical grid simulator.

As shown in figure 2.2.3 VD could be connected for example to a transformer in electrical grid simulator. In this case the transformer could provide the voltage to VFDs, which would in turn consume current, thus influencing the transformer and its voltage. This use case could come up for example when choosing optimal transformer to a factory or a system.

Last use case listed here is simulating a common DC system. Common DC system is a configuration where DC links of multiple drives are connected together, thus creating a common DC link. This is efficient way of balancing energy usage between multiple drives and is a common configuration in bigger systems. For example, when a drive in the system would sometimes be running as a generator, the produced electric energy could be used by the other drives. This way the energy wouldn't have to be gotten rid of in a braking resistor or a regenerative drive wouldn't have to be used (Drury, 2009). One common way of creating common DC system is shown in figure 2.2.5.

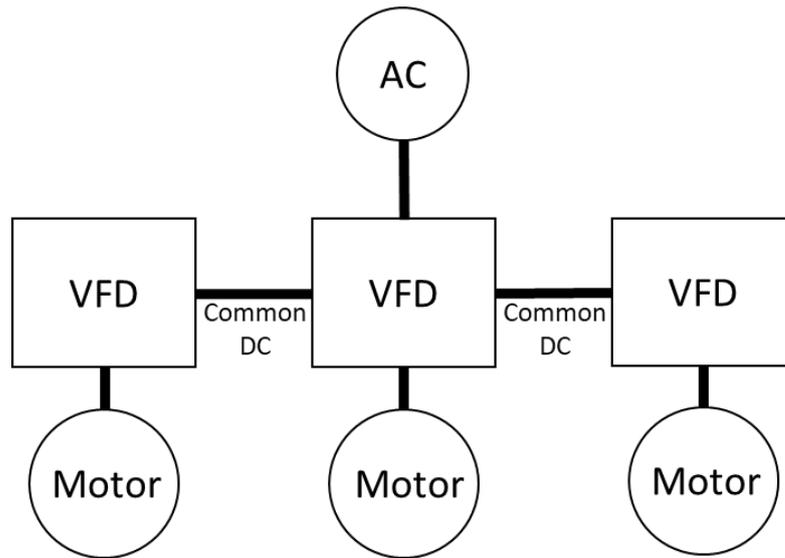


Fig. 2.2.5 Common DC system where one of the drives is supplying the DC current for all drives.

Like figure 2.2.5 shows, one option is that only one drive is connected to the network and it supplies DC current for all the drives. In this case the drive should be appropriately sized to accommodate the load from all drives. Another often used way to create a common DC system is shown in figure 2.2.6.

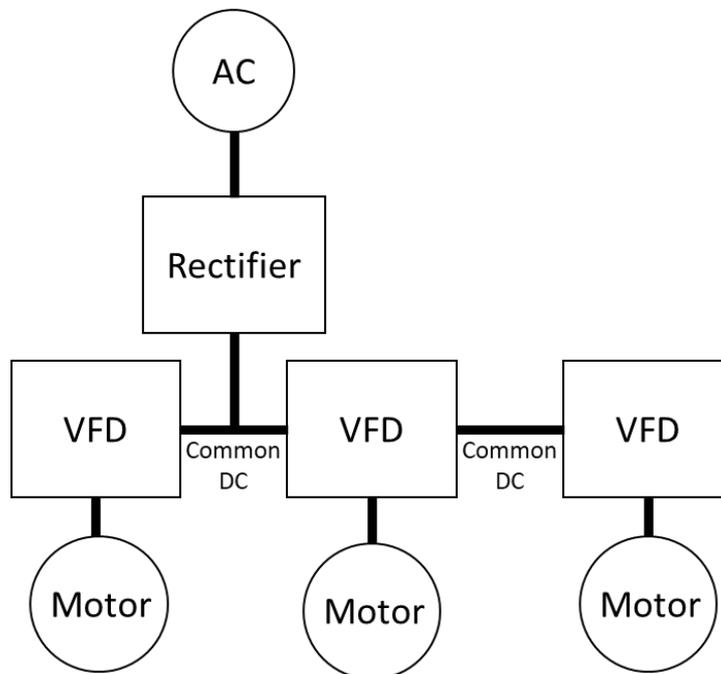


Fig. 2.2.6 Common DC system with a separate rectifier unit supplying the DC current.

As shown in figure 2.2.6 the other way is to have a separate rectifier unit that supplies the DC current for all the drives. There are also different types of common DC systems and different variations of the setups shown here. For example, there could be an additional capacitor bank connected to the common DC link, there could be a braking resistor, the rectifier or one of the drives could be regenerative etc. (ABB, 2014). Figure 2.2.7 shows two ways how simulating a common DC system with VD could look like.

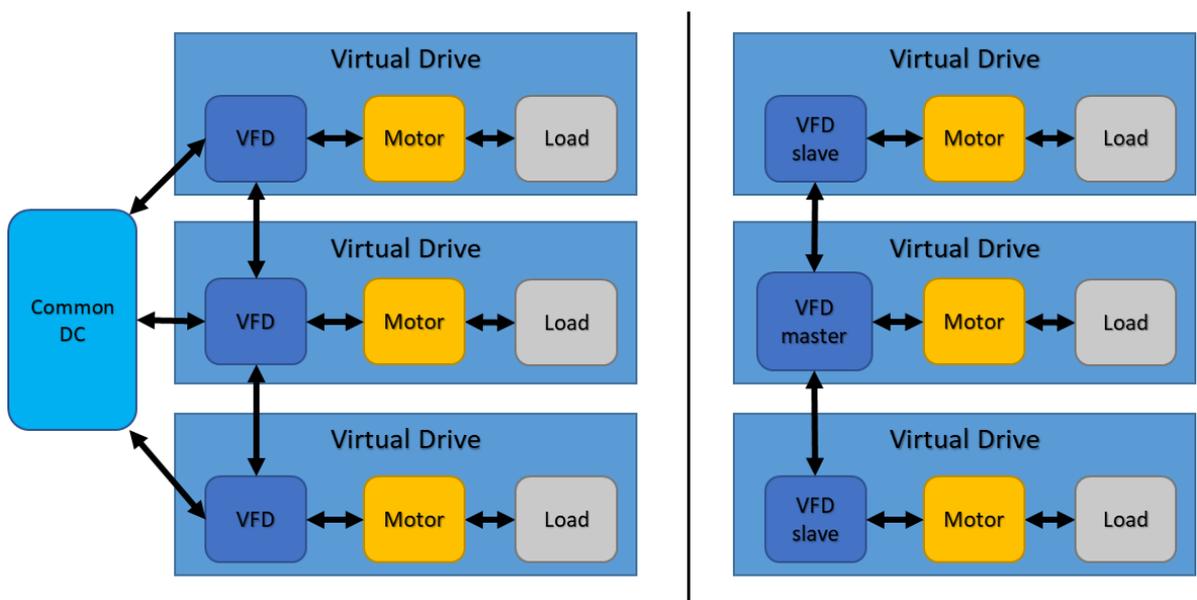


Fig. 2.2.7 Two ways of simulating a common DC system with multiple Virtual Drives. One is to have a separate common DC instance and other is to have a master drive instance that calculates the DC voltage for all the drives.

As shown in figure 2.2.7, simulating a common DC system with VD is unique in the way that there is at least two ways to realize it. The first way is to have a separate common DC instance that connects to all the drives and calculates the DC link voltage for them. Another way is to have master/slave principle between the drives, where the master drive would do the calculation for all the drives. Using this master/slave principle makes more sense in a way that there is no need to make an additional instance and a process for the common DC calculation. This way every drive could be configured to be standalone, common DC slave or common DC master. When in slave mode, a drive could communicate the DC current it

consumes to the master and get the DC link voltage back from it. The master drive would get DC current consumed from all drives including itself and calculate the DC link voltage according to its model. This requires that the DC link model should be flexible for the different common DC setups or even have different DC link models to fulfill all the different setups. In any case it makes sense to have the common DC link modeling in one place where user could choose the model and parameters used with it and not spread the calculations with all parallel running drives.

2.3 Simulator challenges

Many of the challenges posed to VD comes from its wide user base. First, the users can be split to internal and external ones. Internally VD could for example be used for designing drive control software, electric motors and as a tool for salespeople. Externally it could be used for designing and prototyping wide range of different systems. These systems could range from a single drive and motor setups to complex systems consisting of tens of drives and motors. With all these different users and use cases there could be different requirements for the simulations. In some cases, the simulations should be done in real-time, for example when monitoring the condition of a drive or when training operating personnel of a systems behavior. In other cases there is no real-time constraint. For example when optimizing a design of a system, getting the best accuracy is often what matters and not the simulation speed. The most realistic simulations would require simulating a drive on a microsecond or even a nanosecond time level. Getting the most realistic drive behavior would also require having the real drive control software running in VD. Because it's almost impossible to protect software from hacking and reverse engineering, VD with the real drive control software couldn't be openly distributed for external users. Though for internal usage, it could be done.

Continuing with the different requirements for simulations, it could be said that the more accurate simulations you need, the smaller time steps you need and the slower the simulations will get. Also, the bigger the system and the more drives and motors being simulated at the same time, again, the slower the simulations. Additionally the PC hardware that is used to run the simulations also affect the simulation speed. Thus, there always has to

be a trade-off between accuracy and speed of the simulations. In some cases you might want to maximize accuracy or speed, and sometimes something in between. Table 2.3.1 shows some simulation tasks and what needs they typically could have regarding the simulations. The needs are simply divided to two categories: if the task might require highest possible accuracy or fast/real-time simulation speed. In addition, the table shows the time levels, in which the user might be interested in looking at different phenomena in a drive system.

Table. 2.3.1 Typical simulation needs with VD for different tasks.

Task	Fast/ real-time	Highest accuracy	Simulation time scale			
			[s]	[ms]	[μ s]	[ns]
Testing/learning VD in mechanical domain	x		x	x		
Testing/learning VD in electrical domain	x		x	x	x	
Drive and motor sizing for a system	x		x	x		
Automation designing	x		x	x		
Operating personnel training	x		x	x		
System monitoring	x		x	x		
Drive control designing		x	x	x	x	
Drive hardware designing		x	x	x	x	x
Motor designing		x	x	x	x	x
Analyzing device life cycles		x	x	x	x	

As seen from table 2.3.1 there are many different tasks that VD could be used for and with different needs regarding the simulations. When testing out and learning about VD, real-time simulations would probably be preferred so when testing different drives, motors and parameters, the changes could be seen right away. When testing VD and looking at phenomena in mechanical domain, milliseconds could be the smallest time scale necessary for the simulations. For electrical phenomena, like inverter output voltage, microsecond time scale would be needed. When sizing a drive and motor for a system, things are mostly looked at mechanical domain, for example, is the acceleration/deceleration speed sufficient, can the motor control the load efficiently etc. In this case simulation time scale could be down to milliseconds and fast simulation speed would probably be preferred so different setups for a work cycle could be tested faster. For automation designing, again, mechanical properties are mostly the ones that are looked at and fast simulation speed preferred, so iterating over a design and making improvements could be faster. When running simulations parallel to a real system and monitoring it, real-time operation is a must. Same thing could be said for

training operating personnel running or monitoring the system. In both cases, milliseconds time scale should be the smallest one really needed for real-time operations. When designing drive control, getting the best accuracy certainly will become more important than speed. Electrical phenomena are then the ones looked at which could mean as low as microsecond time scale when for example DC voltage is looked at. For drive hardware designing, same things would apply, but even as low as nanosecond time level could be needed for example when looking at single transistors switching on and off. Getting the best accuracy is also needed when designing motors and time scale could again be as low as nanoseconds when looking at electromagnetic phenomena of a motor. For the last task of analyzing device life cycles, best accuracy is needed, and time scale could be as low as microseconds to simulate electrical phenomena.

To solve all the challenges mentioned previously, there might be only one way: to have multiple models within VD simulator with different types and levels of complexity. What is meant here with different types is that in addition to having electrical models of drives and motors, thermal ones could also be needed. For example when life cycle of different components is analyzed, simulating thermal phenomena might be necessary. For every model of the simulator — VFD control, VFD hardware, motor and load — there could be different choices listed in VD so users could choose the ones that suite their use case.

2.4 Models and signals between them

Regarding the different models, for the drive, there should probably be two types of models: electrical and thermal. For electrical one there should be at least one simple model and one complex one with much greater accuracy and level of detail. The simple one could use RMS (root mean square) values of voltage and current, and for example model the input current taken, DC bus voltage and voltage output for the motor. The more complex one could use instantaneous values of voltage and current, and thoroughly model everything in the drive e.g. all the input and output phases, commutation of rectifier diodes and PWM (pulse-width modulation) switching of inverter transistors. There could also be a use for a model in between the simple and complex ones but for a start it's probably best to do model for each extreme and later see if there is a need for the in between one. As for the thermal models,

there should be at least one accurate model which could for example be used for analyzing life cycle of different components of a single drive.

As for the motor, there is again probably need for two electrical models with the simple one using RMS values and the complex one using instantaneous values and modeling the motor more accurately. One accurate thermal model would be needed for analyzing for example bearing life cycle. In addition, there could be an option for using a FEM (finite element method) model of a motor for the most accurate electrical simulations.

As for the load models, few of the most common load models should probably be included. These could be for example pump, fan and compressor models that could also be parametrized to at least somewhat match the device user wants to simulate. When other load models or more accuracy is needed, users could connect VD to their own simulation programs and models.

For the control model, there could be one simple and one complex one just like with the drive and motor models. Both would get speed or torque reference as input and the simpler one could output voltage reference to the drive and the complex one could output switching logic for the drive's inverter.

If there should be different kinds of models in VD, the way to connect them to each other must be considered, because different signals could be used between them. Signals that could be used between simple simulation models are shown in figure 2.4.1.

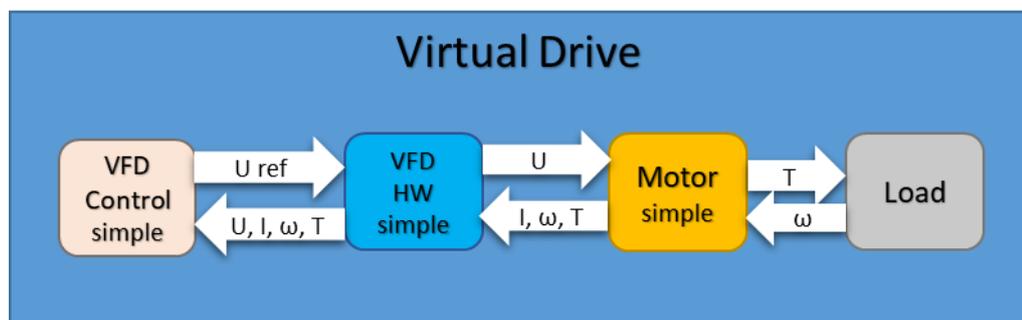


Fig. 2.4.1 Signals used between simple simulation models.

As shown in figure 2.4.1 control outputs voltage reference to drive hardware (HW) which outputs voltage to motor which produces torque to load. In the simplest HW model case, it probably wouldn't simulate modulator, so the voltage reference taken from the control model would be straight away the output for the motor. Load uses the motor torque to calculate rotation speed which it outputs to motor. Motor consumes current and outputs that information back to drive. Depending on what kind of a simple model the HW would be, it could give control model for example DC bus voltage and its output current. Motor would also give control model torque and speed through the HW model. In the simplest case all the current and voltage values could be RMS values. Signals that could be used between complex models in VD are shown in figure 2.4.2.

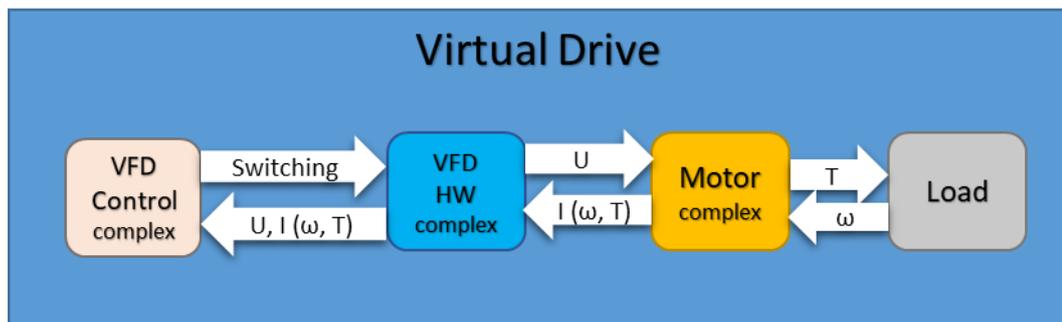


Fig. 2.4.2 Signals used between complex simulation models.

As shown in figure 2.4.2 the signals are the same as with the simpler ones, except control model outputs switching signal for the HW model, which should model the modulator. Another difference is that motor torque and speed isn't necessarily sent from motor to control model, but instead they could be calculated from motor currents by the control model. There is also the difference that voltage and current values should be instantaneous instead of RMS. Signals that could be used with thermal models are shown in figure 2.4.3.

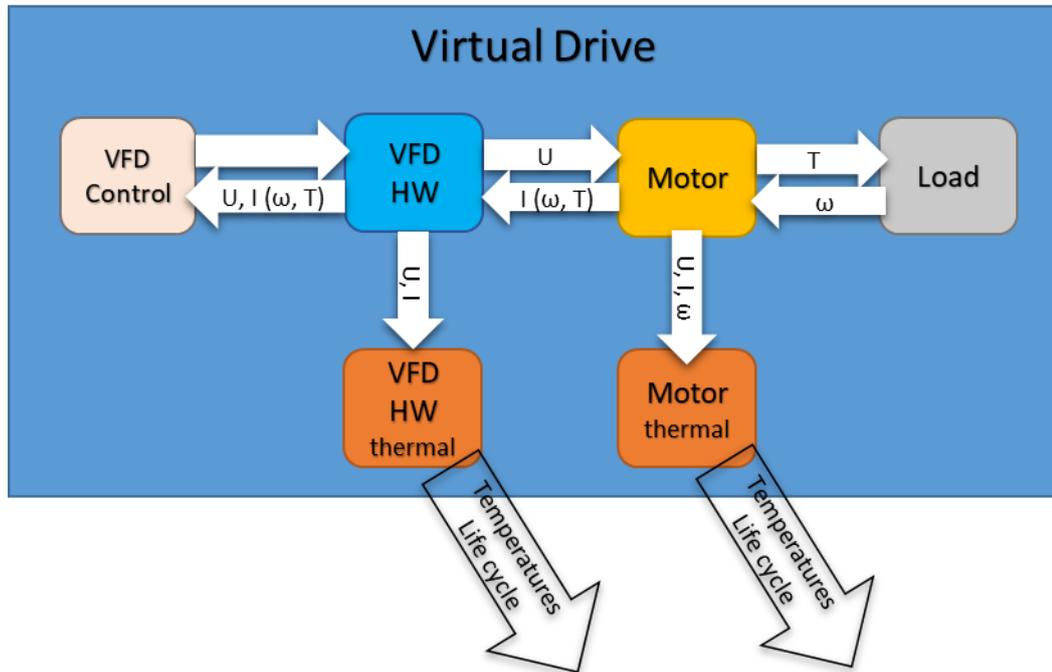


Fig. 2.4.3 Signals used with thermal models.

As shown in figure 2.2.3 thermal models should probably be run parallel to electrical drive and motor models. Thermal VFD HW model would get current and voltage information from the electrical model and output temperatures or life cycles of components. Thermal motor model would get current, voltage and rotational speed information from the electrical model and output temperatures or life cycles of components.

2.5 Conclusion

When having different changeable models in VD, it would make sense to couple the same complexity levels of VFD control, VFD HW and motor models together to match the instantaneous/RMS values between them. In most cases it probably wouldn't make sense to use simple VFD models with complex motor model or vice versa. Using the simple control model (which outputs voltage reference) with the complex HW model wouldn't even work because the HW model expects to get switching logic from it. Thus, at least same complexity levels of control and HW models should be coupled together. It could still be possible though to connect simple model that uses RMS values with a complex model that uses instantaneous values just by converting RMS signal to instantaneous one or vice versa. In any case, the

simulator part of the VD should still be modular, and every part should be changeable, so VD would be flexible when adding new models or when connecting it to external models. Figure 2.5.1 shows how modular simulator could look like in VDs architecture.

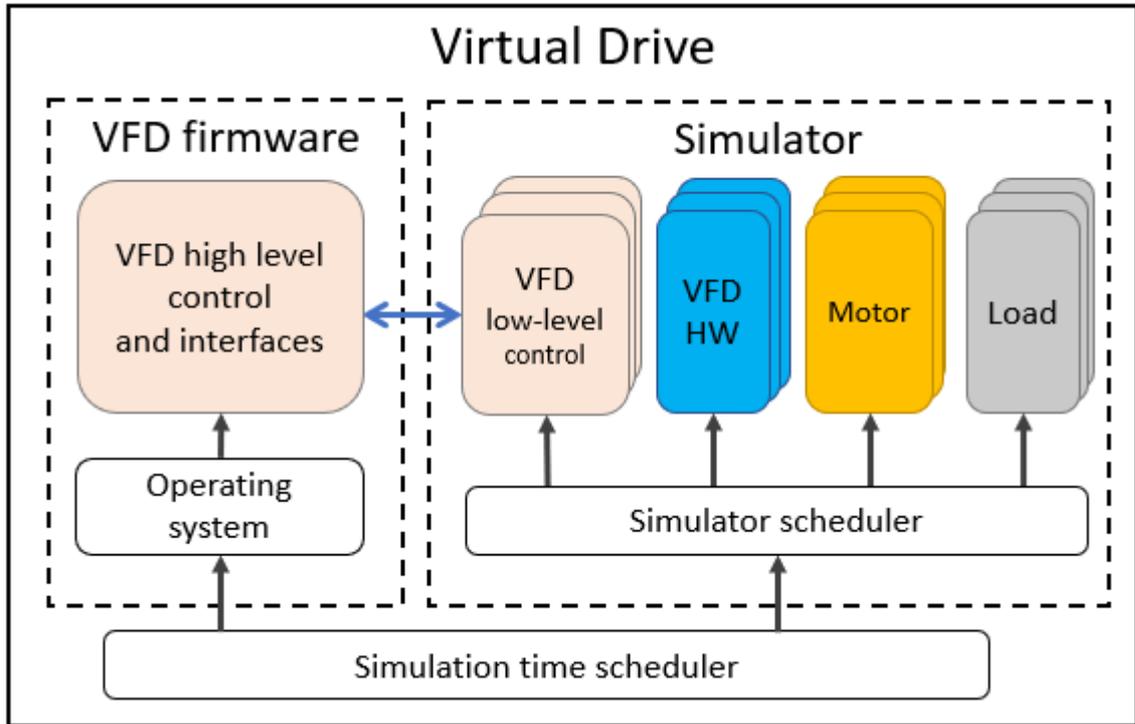


Fig. 2.5.1 Simplified diagram of VD architecture with changeable simulator models.

As shown in figure 2.5.1 the simulator would be modular, and a new simulator scheduler component would be added. Simulator scheduler could run the models that the user has chosen in the correct order and connect their inputs and outputs to each other. It could also handle the use of external models from programs that VD has connected to, instead of using the included models.

Based on the considerations made in this second chapter, it was decided that a new drive system simulator should be made for VD, which is the second goal for this thesis. Main objective for the simulator is to provide real-time simulations even when running multiple simulators simultaneously. As such it should be on the simpler side of the complexity spectrum. It could fulfill the needs for the tasks listed earlier in table 2.3.1, where fast or real-time simulation speed is required, and where drive system phenomena are needed to be

simulated up to a millisecond time level. One additional objective for the simulator is to be able to simulate changes in DC link voltage, which is currently missing in VD, and also provide ability to simulate a common DC system. Integrating the simulator to VD could showcase how modularity can be realized and how the simulator and models can be connected inside VD. Making a simulator that is on the simpler side will make this process faster. For a start the simulator could simulate only the most common type of a drive system e.g. setup with an induction motor and a non-regenerative drive with maybe an option for a braking chopper. Lastly, the simulator shouldn't contain any intellectual property code so it can be openly distributed.

3 INTERNAL ABB SIMULATORS

For this thesis ABB provided source codes for two high fidelity drive system simulators. Both are made for internal usage and aren't available publicly. In this thesis they will be named as old ABB simulator and current ABB simulator. Both were studied and used for making the simulator in this thesis. The current ABB simulator was also used in the simulator comparison in chapter 5.

3.1 Old ABB simulator

Old ABB simulator is a simulator that was developed in LUT University for ABB. It's not currently in active use and its development stopped in 2008. It was part of at least two master's theses where it was developed in (Aarniovuori, 2005) (Tiihonen, 2005) and one doctoral thesis (Aarniovuori, 2010) where it was extensively used for simulations and compared to real-world measurements.

Main features of the old ABB simulator:

- Simple network model
- Full drive modeling: diode bridge, DC link, inverter
- Drive losses modeling: input choke, diode bridge, DC link, IGBT switches
- Motor modeling: analytical or FEM modeling of an induction or permanent magnet motor
- Mechanics modeling
- Selectable control method: scalar or vector
- Different time steps are used with different parts of the simulator, smallest time step being 5 μ s
- Graphical interface
- Coded with C

As can be seen from the features, the old ABB simulator is a complete drive system simulator with for example different options for control method and motor modeling. It also has extensive parameters for the different models it has.

3.2 Current ABB simulator

Current ABB simulator is a simulator that is currently being actively maintained, developed and used inside ABB. It has also been extensively tested against real-world measurements.

Main features of current ABB simulator:

- Full drive modeling: diode bridge, DC link, inverter
- Motor modeling: analytical modeling of an induction, permanent magnet synchronous or synchronous reluctance motor
- Database for drive and motor parameters
- Mechanics modeling
- Selectable control method: scalar or vector
- Different time steps are used with different parts of the simulator, smallest time step being 2.5 μ s
- Graphical interface
- Coded with C++

One of the big features of the current ABB simulator is that it uses the same vector control as used with the real drives e.g. ABB ACS 880. Compared to the old simulator, it has more extensive parameters and options for drive and motor, even though it doesn't include option for FEM motor model. The smallest time step is also shorter and code-wise the project is considerably larger and more complex than the old simulator. There is also an option to choose motor and drive from a database which sets the correct parameters for them. When running the simulator there is an option to save the results to a csv file, which was used for making the chapter 5 comparison.

4 NEW SIMULATOR FOR VIRTUAL DRIVE

The simulator that was created for VD is based on the old and current ABB simulators with changes and additions where necessary. The general structure and signals used in the simulator are shown in figure 4.1.

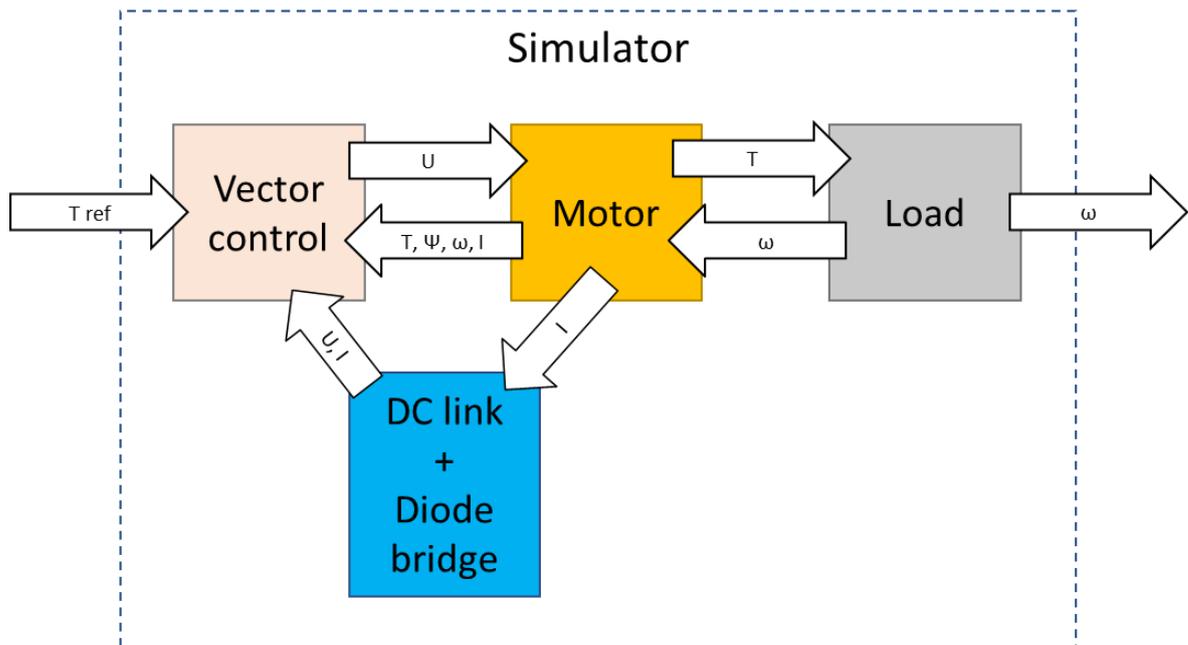


Fig. 4.1 Structure and signals used in the new simulator.

As seen from figure 4.1 the simulator has four parts: vector controller, motor, load and VFD hardware (DC link and diode bridge). Vector controller, diode bridge, motor and load models are based on the old ABB simulator and DC link model is based on the current one. The models ended up being more complex than the simplest ones discussed in chapter 2.4, where it was discussed that the simplest models could use RMS values, instead of instantaneous ones, which were used with this simulator. The simulator also has $50 \mu s$ step time and as shown later in chapter 5.4 the simulator still ended being fast to run.

4.1 Per unit values

The simulator uses per unit values (abbreviated p.u. or pu) with all its calculations. It means that all the variables are scaled by dividing them with some base value which come from the

nominal values of the motor (Pyrhönen et al. 2008). Voltage, current and rotational speed are the ones that IEC-60034 standard determines from which all the other base values can be derived from. The base value for voltage is the nominal peak phase voltage of the motor and can be calculated by

$$U_b = U_n \sqrt{\frac{2}{3}}, \quad (4.1.1)$$

where U_n is the nominal line voltage of the motor. The base value for current is the nominal peak phase current of the motor and can be calculated by

$$I_b = I_n \sqrt{2}, \quad (4.1.2)$$

where I_n is the nominal RMS phase current of the motor. The base value for rotational speed is

$$\omega_b = 2\pi f_n, \quad (4.1.3)$$

where f_n is the nominal stator frequency. Base value for impedance can be calculated by

$$Z_b = \frac{U_b}{I_b}. \quad (4.1.4)$$

Base value for inductance can be calculated by

$$L_b = \frac{U_b}{\omega_b I_b}. \quad (4.1.5)$$

Base value for capacitance can be calculated by

$$C_b = \frac{I_b}{\omega_b U_b}. \quad (4.1.6)$$

Base value for torque can be calculated by

$$T_b = \frac{3 p U_b I_b}{2 \omega_b}. \quad (4.1.7)$$

Base value for inertia can be calculated by

$$J_b = \frac{p T_b}{\sqrt{\omega_b}}, \quad (4.1.8)$$

where p is the pole pair number of the motor. Base value for flux can be calculated by

$$\Psi_b = \frac{U_b}{\omega_b}. \quad (4.1.9)$$

When using pu values, also the time should be scaled. Base value for time is

$$t_b = \frac{1}{\omega_b}. \quad (4.1.10)$$

Still, inside the simulator time is kept in SI units as its easier to understand than the scaled per unit one. This means that for example when integrating, the time must be divided by this base value, which means multiplying it with the base rotational speed value.

4.2 Vector control

Basic idea of a vector controller, as presented in (Trzynadlowski, 1994) and (Leonhard, 2001), is to separately control motor flux linkage and its torque. This is done by controlling the currents that produce flux and torque. To make this possible, the instantaneous current values of each three phases of a motor are first transformed to a two-dimensional vector with α -axis and β -axis components. This is done by Clarke's transformation and it is also useful for motor voltage and fluxes as they can also be more easily manipulated as two-dimensional vectors. Clarke's transformation in matrix form is

$$\begin{bmatrix} x_\alpha(t) \\ x_\beta(t) \\ x_0(t) \end{bmatrix} = \begin{bmatrix} 2/3 & -1/3 & -1/3 \\ 0 & 1/\sqrt{3} & -1/\sqrt{3} \\ 1/2 & 1/2 & 1/2 \end{bmatrix} \begin{bmatrix} x_a(t) \\ x_b(t) \\ x_c(t) \end{bmatrix}, \quad (4.2.1)$$

where x_a , x_b , and x_c are the values from each phase and x_α , x_β and x_0 are the α , β and zero components. When the currents are assumed to be symmetrical and with 120° phase difference between them, the sum of the currents is always zero and the zero component of the vector can be ignored. After Clarke's transformation the current vector is presented in a stationary frame. For controlling the torque and flux the currents must be transformed to a moving frame of the rotor flux. This is done by Park's transformation which in matrix form is

$$\begin{bmatrix} x_d(t) \\ x_q(t) \end{bmatrix} = \begin{bmatrix} \cos\theta(t) & \sin\theta(t) \\ -\sin\theta(t) & \cos\theta(t) \end{bmatrix} \begin{bmatrix} x_\alpha(t) \\ x_\beta(t) \end{bmatrix}, \quad (4.2.2)$$

where θ is the rotor flux angle and x_d and x_q are the d-axis and q-axis components. Regarding the current components, d-axis component produces the flux linkage in the motor and q-axis component produces the torque. So, by controlling these d-axis and q-axis currents, the flux and torque of the motor can be controlled separately. As the reference frame for the vector in this case is the rotor flux and the currents are the main control variables, this vector control strategy is called rotor-flux-oriented current control. This process of transforming three phase values to a two-dimensional vector and changing the reference frame is shown visually in figure 4.2.2.

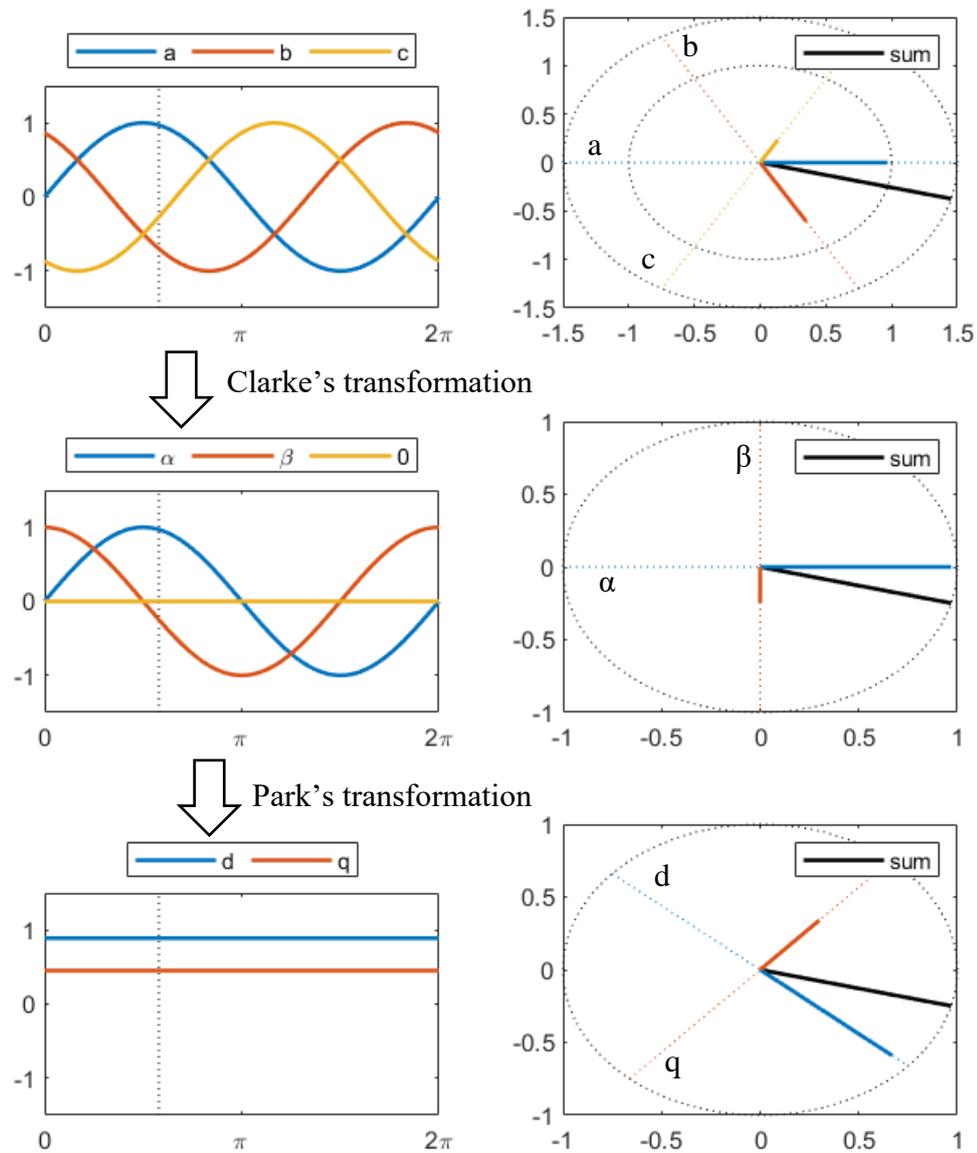


Fig. 4.2.2 Visual presentation of Clarke's and Park's transformation, where Park's transformation is done to an arbitrary reference frame. On the left are graphs of the instantaneous values and on the right vector presentation of the values at an arbitrary point in time.

Figure 4.2.2 shows how 3-phase values are simplified to two components and how the zero component, as its name implies, is zero when an ideal 3-phase current is transformed. Typically, Clarke's transformation is made so that the peak values match between 3-phase and two-dimensional ones. This makes it so that on the top right corner of the figure, the

sum of the phase values is 1.5 times the peak phase value, and also 1.5 times bigger than the sum with two-dimensional ones. The two-dimensional values can be scaled to match this RMS value by multiplying them with 1.5 if needed. The vector controller structure of the simulator is shown in figure 4.2.2.

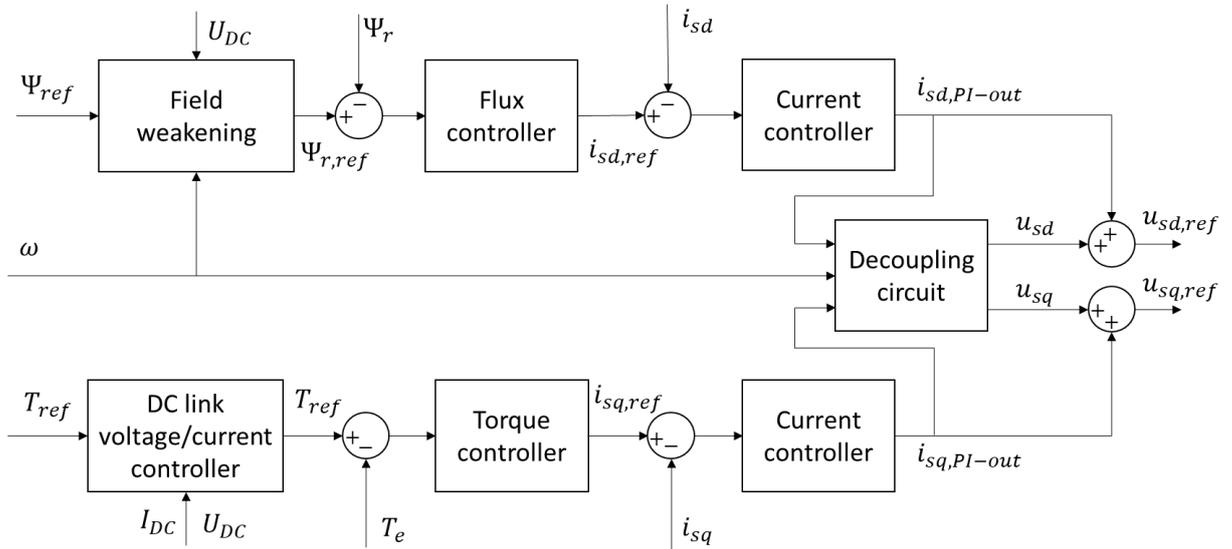


Fig. 4.2.2 Structure and signals used in the simulators vector controller. Adapted from (Aarniovuori, 2010).

As seen from figure 4.2.2, the vector controller consists of current, flux and torque controllers, field weakening calculation block, DC link voltage/current controller and a decoupling circuit. User inputs for the controller are flux and torque reference, which the controller tries to control by controlling the stator d- and q-axis currents. Output of the controller is stator d- and q-axis voltages. All the controllers are simple PI controllers and flux, stator currents, torque, DC link voltage and DC link current are controlled in closed loop. In real drives, every input variable used in the controller are measured or estimated using some model, but in the simulator these values come straight from the motor and HW models. As such, this can be thought as an ideal vector controller that knows always the exact values of all the variables.

Idea of the field weakening block is to reduce the flux reference when motor is wanted to be driven over its nominal speed (Nam, 2019). Field weakening calculations are

$$\Psi_{ref} = \Psi_{ref} \frac{U_{max}}{\omega_{ref}}, \quad (4.2.1)$$

when the speed reference $\omega_{ref} > U_{max}$. The max output voltage U_{max} is calculated by

$$U_{max} = \frac{U_{DC}}{\sqrt{3}}, \quad (4.2.2)$$

where U_{DC} is DC link voltage.

Objective of the DC link voltage/current controller block is to reduce braking torque when DC link voltage or DC link current is over a defined limit. Also accelerating torque can be limited if DC link voltage is under or the DC link current is over a defined limit. These will affect the torque reference only after the limit is crossed and will stop limiting the torque after the PI controller's integrator reaches zero. Parameters for the block are minimum and maximum DC voltage limits and maximum DC current limit.

Because the d- and q-axis currents should be controlled separately, a decoupling circuit is necessary. Without it, the output voltages of the controller for both axes would somewhat affect each other which is not wanted. The decoupling circuit is from (Vas, 1998) and the equations are

$$u_{sd} = -\omega_r i_{sq,PI-out} \left(L_s \frac{L_m^2}{L_r} \right) \text{ and} \quad (4.2.3.)$$

$$u_{sq} = -\omega_r i_{sd,PI-out} \left(L_s - \frac{L_m^2}{L_r} \right) + \omega_r |i_{mr}| \frac{L_m^2}{L_r}, \quad (4.2.4)$$

where $i_{sq,PI-out}$ and $i_{sd,PI-out}$ are outputs from the current controllers, L_s , L_r and L_m are motor induction parameters (explained in chapter 4.4), ω_r is rotor flux linkage rotation speed and i_{mr} is rotor magnetizing current. Output from the decoupling circuit are u_{sd} and u_{sq} which are added to current controller outputs.

4.3 VFD hardware

Most of the modern drives including ABB drives are of a type called voltage source inverters (VSI), which is also the one that is modeled in the simulator (Hughes and Drury, 2013). Basic structure of a VSI drive is shown in figure 4.3.1.

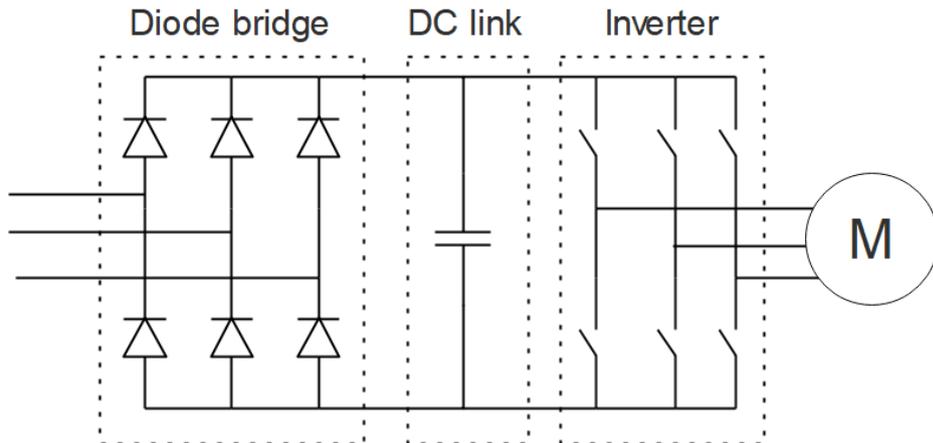


Fig. 4.3.1 Basic structure of a voltage source inverter. Adapted from (Hughes and Drury, 2013).

As seen from figure 4.3.1 VSI has three parts: diode bridge, DC link and inverter. To make the simulator faster and simpler, it was decided that inverter would be left out and it wouldn't be modeled. The function of the diode bridge is to take AC voltage and convert it to DC voltage. Input to the diode bridge model is an ideal three phase AC voltage with each phase being 120 degrees out of phase with each other. The phase voltages U_1 , U_2 and U_3 can be written as

$$\begin{aligned}
 U_1 &= U_p * \sin(2\pi ft) \\
 U_2 &= U_p * \sin \left(2\pi ft + \frac{2\pi}{3} \right) \\
 U_3 &= U_p * \sin \left(2\pi ft + \frac{4\pi}{3} \right),
 \end{aligned} \tag{4.3.1}$$

where U_p is the peak phase voltage, f is frequency and t is simulation time. If the diode bridge would be connected to an external network simulator, these phase voltages could be an input

to the model. Another option would be to have the peak phase voltage and the voltage frequency as an input from the network simulator. Output from the diode bridge is the maximum voltage difference between all the phases. This is because the diodes in the diode bridge allow the current to flow only one way. So, the output voltage from the diode bridge can be written as

$$U_{DC,in} = \max(U_1, U_2, U_3) - \min(U_1, U_2, U_3). \quad (4.3.2)$$

The function of a DC link is to even out the voltage $U_{DC,in}$ coming from the diode bridge. The model of a DC link is shown in figure 4.3.2.

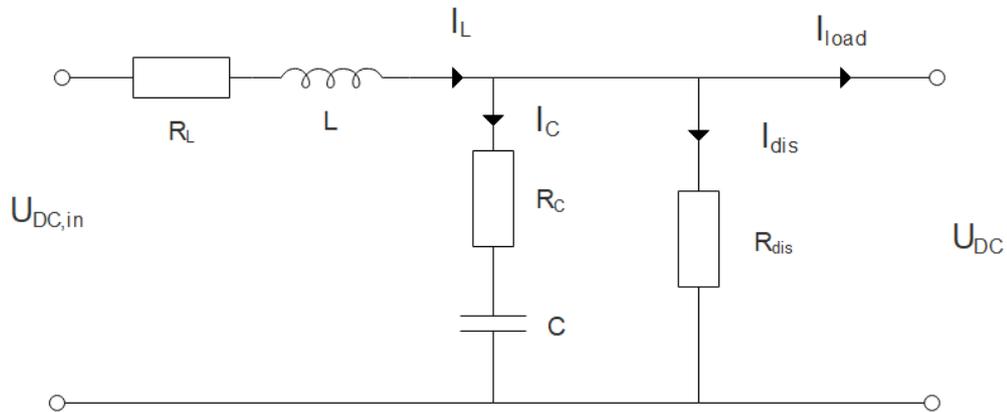


Fig. 4.3.2 DC link model used in the simulator.

From figure 4.3.2 we can see the components and parameters used in the model which are: input resistance R_L , input inductance L , DC resistance R_C , DC capacitance C and discharge resistance R_{dis} . The figure also shows DC link input current I_L , DC current I_C , discharge current I_{dis} and load current to motor I_{load} . Current flow through the input inductor can be calculated by integrating voltage over it

$$I_L = \frac{1}{L} \int U_L dt, \quad (4.3.3)$$

where U_L is voltage over the inductor. The voltage over the inductor can be calculated by subtracting input resistance voltage and DC link voltage and from diode bridge output voltage

$$U_L = U_{DC,in} - U_{DC} - I_L R_L. \quad (4.3.4)$$

The DC link capacitor voltage can be calculated by integrating current through it

$$U_C = \frac{1}{C} \int I_C dt. \quad (4.3.5)$$

The current through the capacitor can be calculated by subtracting load and discharge currents from the input current

$$I_C = I_L - I_{load} - U_{DC} R_{dis}, \quad (4.3.6)$$

where I_{load} is the current that motor consumes from the DC link. There is also an option to use a braking chopper that consumes current from the DC link. It has parameters for activation and deactivation voltage levels and the braking chopper resistance. If braking chopper is used and DC voltage level gets higher than the activation level, braking chopper will be activated until the voltage drops below deactivation level. Current through the capacitor when braking chopper is used is calculated by

$$I_C = I_L - I_{load} - U_{DC} R_{dis} - U_{DC} R_{bc}, \quad (4.3.7)$$

where R_{bc} is the braking chopper resistance. Finally, the DC link voltage can be written as

$$U_{DC} = U_C + I_C R_C, \quad (4.3.8)$$

where I_C is calculated depending on if braking chopper is used or not by equations 4.3.6 or 4.3.7.

4.3.1 Common DC link

Simulating a common DC link was realized using the same idea presented in chapter 2.2 where there is a simulation parameter for controlling mode of the drive. There are three modes the drive model could be in: standalone, slave and master. In the standalone mode VFD simulates its own diode bridge and DC link in the same way presented previously in this chapter. In the slave mode the drive doesn't simulate the diode bridge or the DC link but sends the current consumed by the motor to the common DC master instance and should get the DC link voltage from there. If there is no communication to the master instance, slave uses fixed DC voltage. In the master mode the drive simulates the DC link of the whole setup. Figure 4.3.1.1 shows the common DC link model.

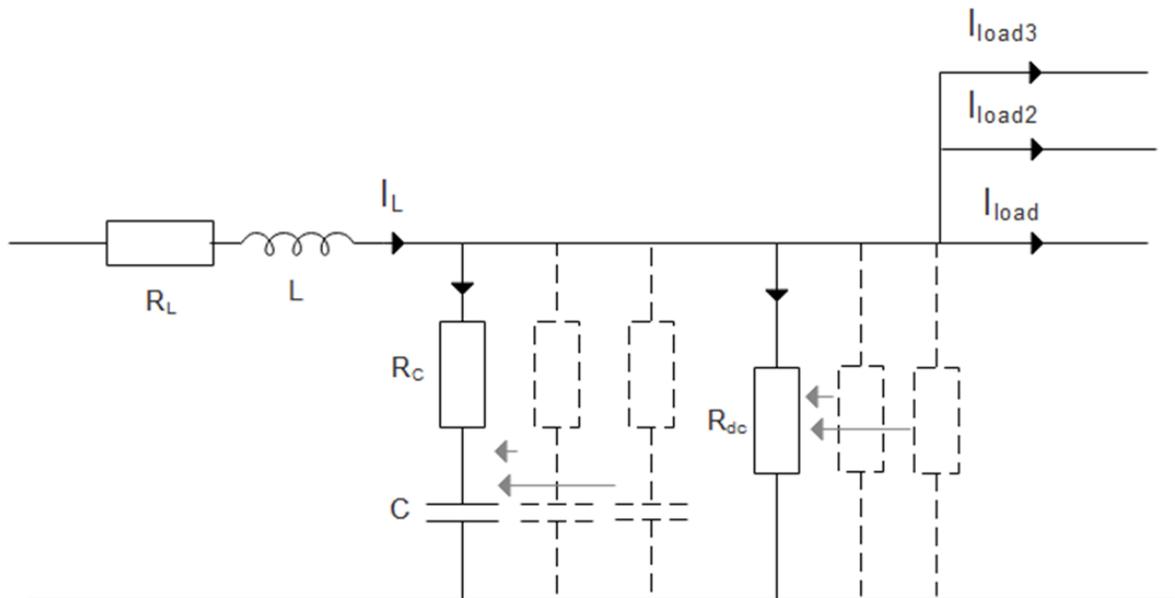


Fig. 4.3.1.1 Common DC link model where parameters of multiple drives are combined to make a single equivalent circuit.

The common DC link setup is simulated with the same model presented in this chapter by combining the parameters from the drives used with the setup. When the common DC setup is the kind of a setup shown in figure 2.2.5 where one of the drives is connected to the network, the DC link parameters C , R_c and R_{dis} from all the drives should be combined and used as the parameters for the master drive instance. If the common DC setup is the kind of a setup shown in figure 2.2.6, where there is a separate rectifier unit providing the DC

current, depending on the rectifier the DC link could also be simulated with this same model. The DC link parameters in the master drive would of course then match the rectifier. If the setup includes for example an additional capacitor bank, the capacitance and resistance of the capacitors could be combined to the C and R_c parameters. When the simulation is running, the master drive should get the DC link current consumed from all the drives, add them together and use the sum with the equation 4.3.6 as the I_{load} .

4.4 Motor

Motor model in the simulator is based on the old ABB simulator and work done by Aarniovuori (Aarniovuori, 2005) (Aarniovuori, 2010), which was also used as the reference for this chapter. The most common way to model an induction motor is by using so called T-equivalent circuit model shown in figure 4.4.1.

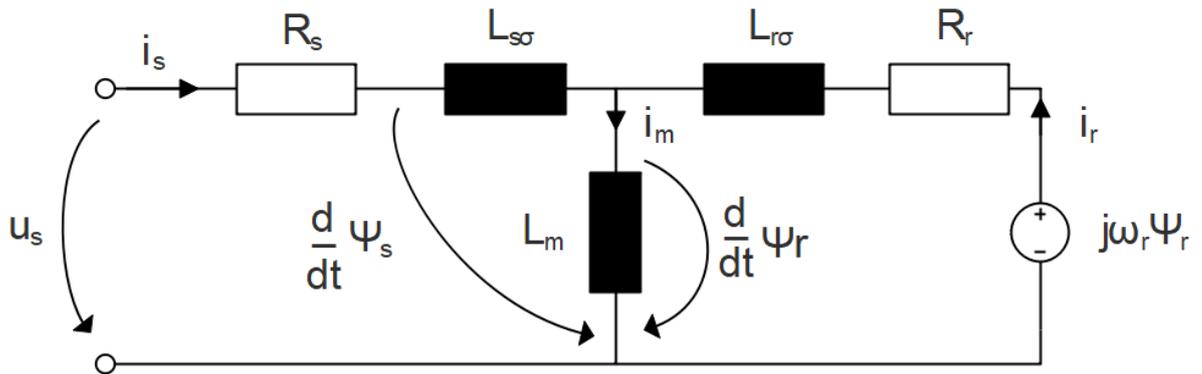


Fig. 4.4.1 T-equivalent circuit model of an induction motor.

As figure 4.4.1 shows, T-equivalent circuit has five motor parameters: stator resistance R_s , rotor resistance R_r , stator leakage inductance $L_{s\sigma}$, rotor leakage inductance $L_{r\sigma}$ and magnetizing inductance L_m . These are the parameters often given for a motor or they can be obtained with short-circuit and locked rotor tests (Trzynadlowski, 2001). The motor model used in the simulator is called inverse Γ -equivalent circuit which is derived from the T-equivalent circuit (De Donker and Novotny, 1994). The advantage with inverse Γ -equivalent circuit is that the rotor leakage inductance has been moved to stator side and combined with stator leakage inductance, thus reducing one parameter. This also simplifies the calculations

as everything can be calculated in stator reference frame and the need for moving between stator and rotor reference frame is removed. Inverse Γ -equivalent circuit is shown in figure 4.4.2.

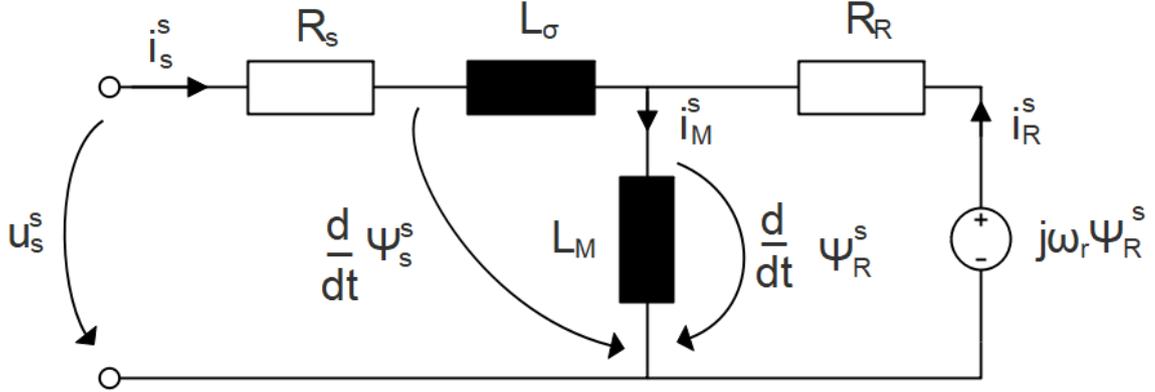


Fig. 4.4.2 Inverse Γ -equivalent circuit model of an induction motor.

As figure 4.4.2 shows, inverse Γ -equivalent circuit has four parameters: stator resistance R_s , rotor resistance R_R , leakage inductance L_σ and magnetizing inductance L_M . The stator resistance is the same as with the T-equivalent circuit and the other ones can be calculated from T-equivalent parameters. Rotor resistance can be calculated by

$$R_R = \left(\frac{L_m}{L_m + L_{r\sigma}} \right)^2 R_r. \quad (4.4.1)$$

The magnetizing inductance can be calculated by

$$L_M = \frac{L_m^2}{L_m + L_{r\sigma}}. \quad (4.4.2)$$

The leakage inductance can be calculated by

$$L_\sigma = L_{s\sigma} + \frac{L_m L_{r\sigma}}{L_m + L_{r\sigma}}. \quad (4.4.3)$$

The voltage equations for the stator and rotor are

$$u_s = i_s R_s + \frac{d\psi_s}{dt} \text{ and} \quad (4.4.4)$$

$$0 = i_R R_R + \frac{d\psi_R}{dt} - j\omega\psi_R, \quad (4.4.5)$$

where u_s is the stator voltage, ψ_s is stator flux linkage, ψ_R is rotor flux linkage and ω is speed of the motor. The stator voltage is the input to the motor model. The equations for the stator and rotor currents are

$$i_s = \frac{\psi_s - \psi_R}{L_\sigma} \text{ and} \quad (4.4.6)$$

$$i_R = \frac{\psi_R}{L_M} - i_s. \quad (4.4.7)$$

Stator and rotor flux linkages can be calculated by

$$\frac{d\psi_s}{dt} = u_s - i_s R_s \text{ and} \quad (4.4.8)$$

$$\frac{d\psi_R}{dt} = i_R R_R - j\omega\psi_R. \quad (4.4.9)$$

Torque of the motor can be calculated by

$$T_e = \frac{3}{2}p(\psi_s \times i_s), \quad (4.4.10)$$

When calculated in per unit values the torque equation simplifies to

$$T_{e,pu} = \psi_s \times i_s. \quad (4.4.11)$$

Motor torque T_e is the output of the motor model to the load. Another output of the motor is how much current it draws from DC link, as there is no inverter model between them. The current consumed by the motor is estimated by equation

$$I_{load} = i_{sq}\omega + \frac{i_{sd}^2 R_s + i_{Rd}^2 R_R}{U_{DC}}, \quad (4.4.12)$$

where i_{sq} is stator current q-axis component, i_{sd} is stator current d-axis component and i_{Rd} is rotor currents d-axis component. Every variable in the equation is in per unit value and the current components are scaled to match their RMS values. The $i_{sq}\omega$ component in the equation takes into account the torque producing current and is linear with respect to the motor speed. Scaling the q-axis current with speed is comparable to basic motor power equation where the power produced is also linear with respect to the motor speed

$$P = T\omega. \quad (4.4.13)$$

The second part of the equation takes into account the resistive losses from the flux producing d-axis current in stator and rotor. In the equation, power consumed by the stator and rotor resistance are summed and then divided by the DC link voltage to get the current DC link needs to output for making up these losses.

4.5 Load

With loads and rotational speed calculations there are two different options. With both options the load model could be an internal one or an external one connected through some interface. First option is that speed is calculated by the motor model by giving it load torque. In this case the load model could be given motor speed if it needs it for the load torque calculation. Speed would in this case be calculated with equation

$$\frac{d\omega}{dt} = \frac{p}{J}(T_e - T_{load}), \quad (4.5.1)$$

where T_{load} is torque from load and J is combined inertia of the motor and load. Second option is that the motor torque is given to a load model which then calculates the speed. In this case there are two ways to handle inertia. Simplest one is to have the motor inertia included in the load model. If it isn't included in it, the load would accelerate faster than it should, though if the load inertia is much higher than the motor inertia it could be unnoticeable. Other solution is to not include the motor inertia in the load but reduce the effect of motor inertia from the motor torque. This can be calculated by basic torque equation

$$T_{inertia} = J_m \frac{d\omega}{dt}, \quad (4.5.2)$$

where J_m is inertia of the motor. This torque would be reduced from the motor torque before giving it to a load model. This also requires the simulator to calculate the derivative of speed between load model steps. This way of calculating speed is less accurate but in most cases shouldn't have noticeable effect if the load model time step isn't too big compared to its inertia.

5 SIMULATOR COMPARISON

The new simulator was compared to the current ABB high fidelity simulator. The current ABB simulator is actively being developed, used and tested inside ABB and it includes the same vector control code used with the real drives. As such it should be a good software to compare the new simulator against. There are also plans for integrating the current ABB simulator to Virtual Drive which also makes the comparison useful. There were four tests made for comparing these simulators: steady state, dynamic acceleration, dynamic deceleration and simulation speed test. In the first three tests the internal variables that were of interest were saved in 50 μs intervals with both simulators. This was the simulation time step that the new simulator uses for all its parts. The current ABB simulator uses different time steps for different parts of the simulator and the smallest time step is 2.5 μs which it uses for example with some parts of motor and drive. This means that the current ABB simulator could show more finer details about motor current or DC link voltage that are not shown in chapter 5.2 or 5.3 figures. Still, for the comparison it made more sense to have the same 50 μs variable saving interval for both simulators. For these tests both simulators used the same 37 kW motor and 45 kW drive parameters. The motor parameters are shown in table 5.1.

Table. 5.1 Motor parameters used with comparison tests.

Type	Induction
Poles	2
Nominal frequency [Hz]	50
Nominal voltage [V]	400
Nominal current [A]	68.9
Nominal power [kW]	37
Inertia [kg·m ²]	0.385
Nominal speed [RPM]	1480
Stator resistance [Ω]	0.0629
Rotor resistance [Ω]	0.0439
Stator leakage inductance [mH]	0.4833
Rotor leakage inductance [mH]	1.003
Magnetizing inductance [mH]	28.88

The motor was chosen because it was the default one in the old ABB simulator and included all the necessary parameters. Because the current ABB simulator uses mechanical time constant instead of inertia, the time constant was matched so that both simulators' motors accelerate at the same rate when same torque is applied. Another thing changed in the current ABB simulator was removing motors fan and friction torque losses. This way the load torque in different tests could be matched exactly in both simulators. The same fan and friction losses could also have been as easily added to the new simulator. Parameters for the drive are shown in table 5.2.

Table. 5.2 Drive parameters used with comparison tests.

Nominal power [kW]	45
Nominal current [A]	87
Line voltage [V]	400
Grid frequency [Hz]	50
DC link capacitance [mF]	1.545
DC link resistance [mΩ]	10.7
Input inductance [mL]	0.5
Input resistance [mΩ]	18
Discharge resistance [Ω]	33300

The drive was chosen from a parameter database included in the current ABB simulator and its power rating was closest to the chosen motor. Table 5.3 shows the pu values used in the graphs and tables of following chapters.

Table. 5.3 Per unit values used in the graphs and tables of the chapters 5.1, 5.2 and 5.3.

Voltage [1 pu]	Current [1 pu]	Torque [1 pu]	Speed [1 pu]
325 V	97.4 A	238.7 Nm	1480 RPM

In the table 5.3 the per unit voltage is the nominal peak phase voltage of the motor. Per unit current is the peak nominal current of the motor. Per unit torque is the nominal torque of the motor calculated from its nominal power and speed. Per unit speed is the nominal speed of the motor. Neither torque or speed are actually per unit values according to chapter 4.1 equations or IEC standard, but they are actually per unit values scaled to nominal values to better suit the graphs and tables.

5.1 Steady state

For the steady state tests simulators were run with different loads ranging from 1.25 to 0.25 of the motors nominal torque. The load was linear relative to speed and at the motors nominal speed it would match the given value. Simulators were given the same torque reference as this load value. Simulations were running until they were in a steady state after which saving the internal variables started and the simulation was ran for one more second. Averages were taken from these variables and the results are shown in table 5.1.1.

Table. 5.1.1 Steady state comparison results. In the table *ABB* means the current ABB simulator and *New* means the simulator made in this thesis.

Load		1.25	1.00	0.75	0.50	0.25
Torque	ABB	1.2315	0.9768	0.7290	0.4809	0.2509
	New	1.2495	0.9996	0.7497	0.4997	0.2498
	Difference	1.44 %	2.28 %	2.76 %	3.77 %	-0.47 %
Speed	ABB	0.9972	0.9909	0.9841	0.9752	1.0157
	New	0.9996	0.9996	0.9996	0.9995	0.9991
	Difference	0.24 %	0.87 %	1.55 %	2.42 %	-1.67 %
Current	ABB	1.123	0.9043	0.709	0.540	0.415
	New	1.1066	0.8960	0.6890	0.4864	0.3111
	Difference	-1.45 %	-0.93 %	-2.98 %	-11.07 %	-33.31 %
DC link voltage	ABB	1.6492	1.6502	1.6532	1.6693	1.6851
	New	1.6492	1.6502	1.6511	1.6665	1.6853
	Difference	0.00 %	0.00 %	-0.13 %	-0.17 %	0.01 %
Motor efficiency	ABB	0.9712	0.9778	0.9826	0.9890	0.9874
	New	0.9938	0.9924	0.9900	0.9852	0.9710
	Difference	2.27 %	1.47 %	0.74 %	-0.39 %	-1.69 %

From the table 5.1.1 results it can be seen that the current ABB simulator and the new simulator are quite close to each other and the DC link voltage is almost exactly the same with both simulators. For some reason the current ABB simulator never quite reached the torque reference given to it when the reference was higher than 0.25. That's why the new simulators torque and speed results are a few percent off. This also has a small effect on the other results. Reason for why the current ABB simulator didn't produce exactly same torque as the reference was unsolved. Another difference is in motor efficiency. The new simulator shows highest motor efficiency with the biggest load and goes smaller with smaller loads. It also shows unrealistically high motor efficiency, for example 99.38 % when load is 1.25. On

the other hand, current ABB simulator shows best efficiency of 98.9 % when the load is 0.5 of the motors nominal torque. Thus, motor efficiency with both simulators seem a bit off. Biggest difference between the simulators is with motor currents when using smaller loads. When the load was 0.5, the new simulator showed 9.9% smaller motor current than the current ABB simulator and when the load was 0.25 the difference was 27%. Thus, it can be said that the motor model of the new simulator shows too small currents with smaller loads. Close to nominal load the difference between the simulators is small.

5.2 Acceleration

For the acceleration test motor was at a standstill but already magnetized by the drive. Then nominal torque of the motor was given as a reference value for the controller. There was also a load that was linear relative to the speed of the motor and the load torque matched the nominal torque at the nominal speed. Because all the variables are influenced by each other, the results are presented first and reasons for any differences between the simulators are discussed at the end of the chapter. Figure 5.2.1 shows behavior of motor speed in the acceleration test.

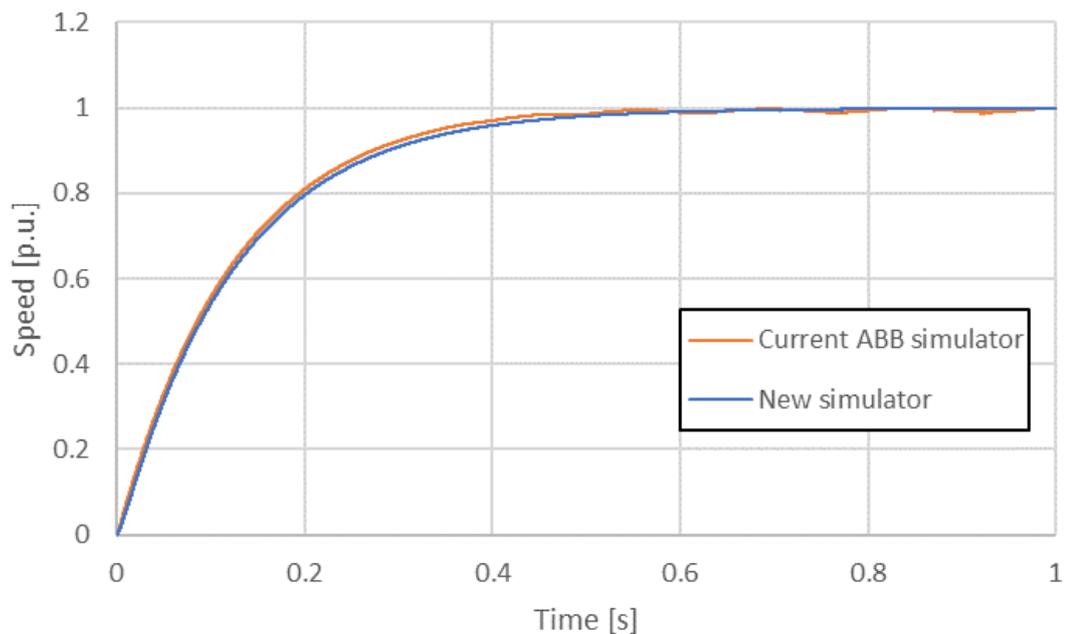


Fig. 5.2.1 Speed as a function of time in acceleration test.

From figure 5.2.1 it can be seen that the speeds match almost exactly between the current ABB simulator and the new simulator. Figure 5.2.2 shows behavior of torque in the acceleration test.

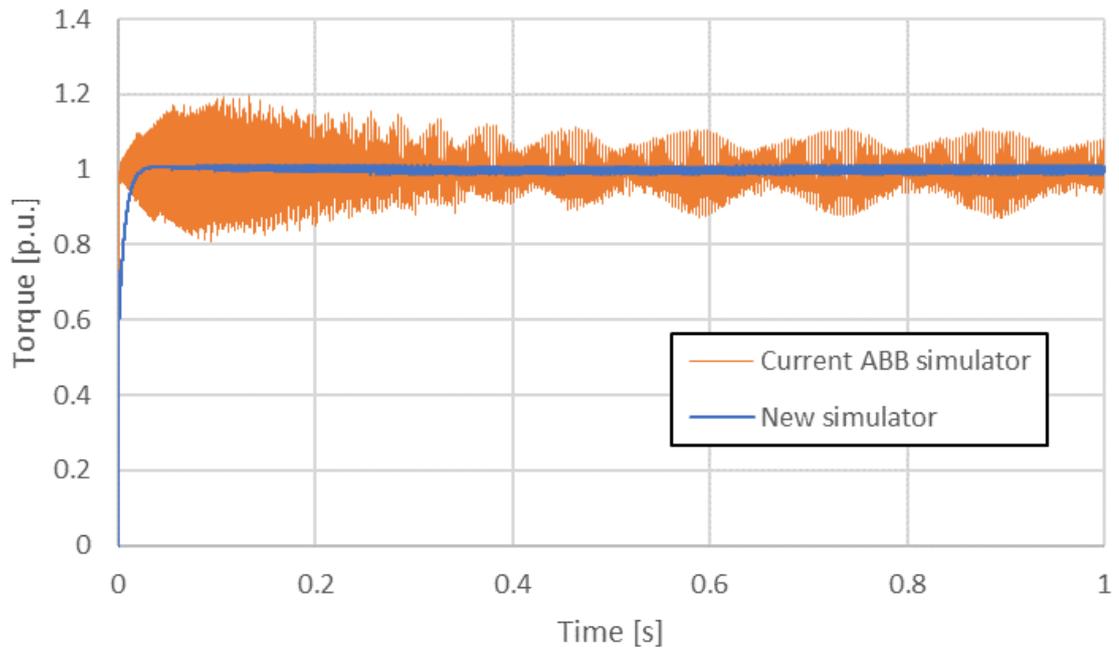


Fig. 5.2.2 Torque as a function of time in acceleration test.

Figure 5.2.2 shows that the average of the torque in both simulators are close to each other and seem to match the nominal torque reference given to them. Because the new simulator doesn't model modulator, it doesn't have as much fluctuation in torque as the current ABB simulator. There is also a difference in rise time of the torque. The new simulator torque rises a bit slower than the one with the current ABB simulator. Figure 5.2.3 shows zoomed plot of the torque at the start of the acceleration.

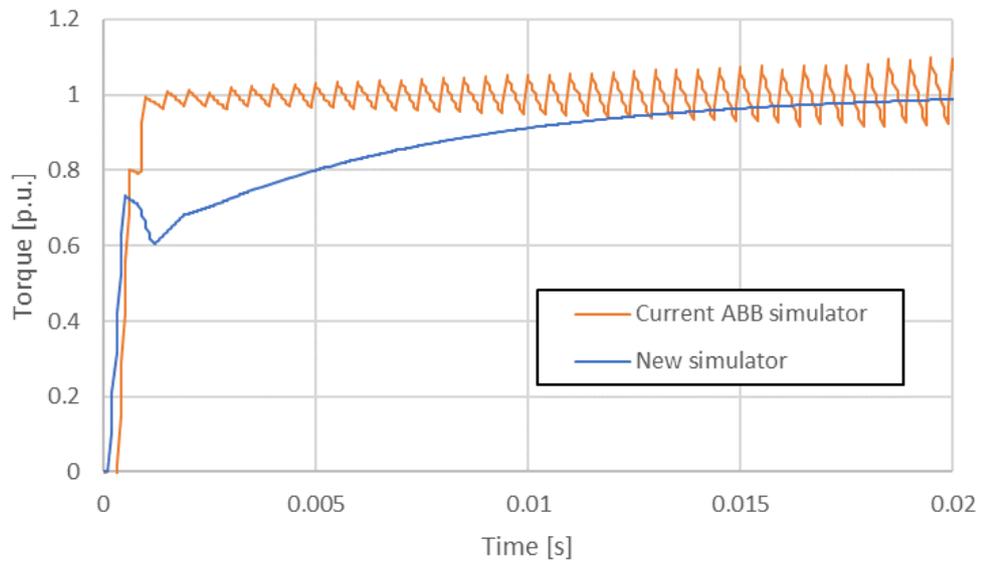


Fig. 5.2.3 Torque at the start of the acceleration test.

Figure 5.2.3 shows that torque rise time with the new simulator is hundredth of a second slower than in the current ABB simulator and there is also a small notch at the start. The notch and the rise time could maybe be improved by tuning the control parameters of the vector control, though hundredth of a second probably won't matter in most of the use cases with VD. Figure 5.2.4 shows behavior of motor current in the acceleration test.

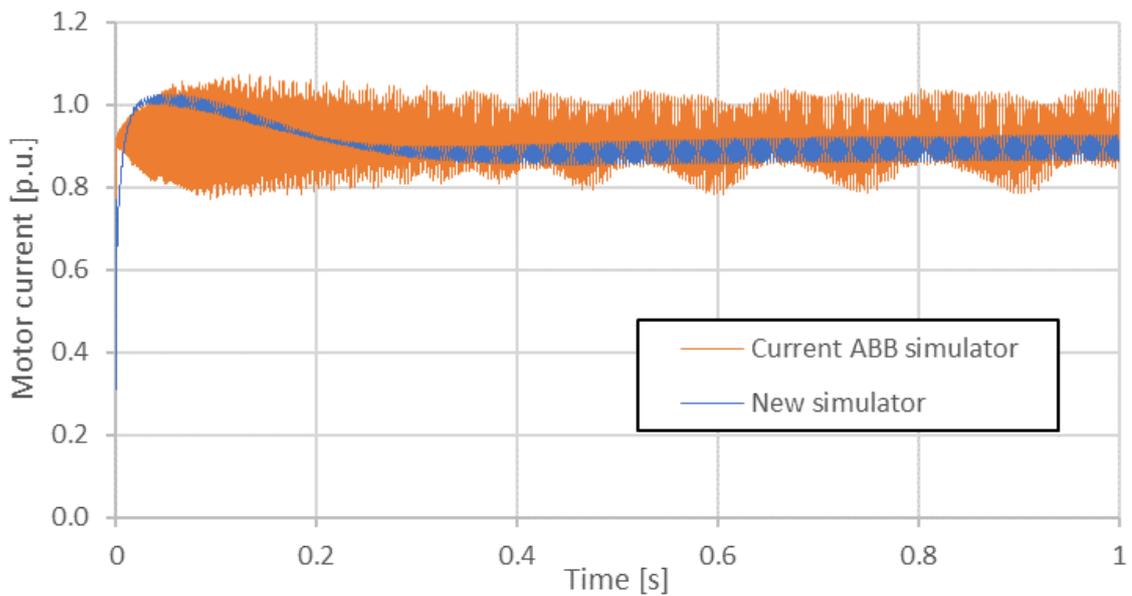


Fig. 5.2.4 Motor current as a function of time in acceleration test.

Figure 5.2.4 shows that just like with the torque, the motor current with the ABB simulator has more fluctuation than the new simulator. At the start the new simulator shows somewhat higher motor current than the ABB simulator but then settles close to the same level. Figure 5.2.5 shows behaviour of DC link voltage in the acceleration test.

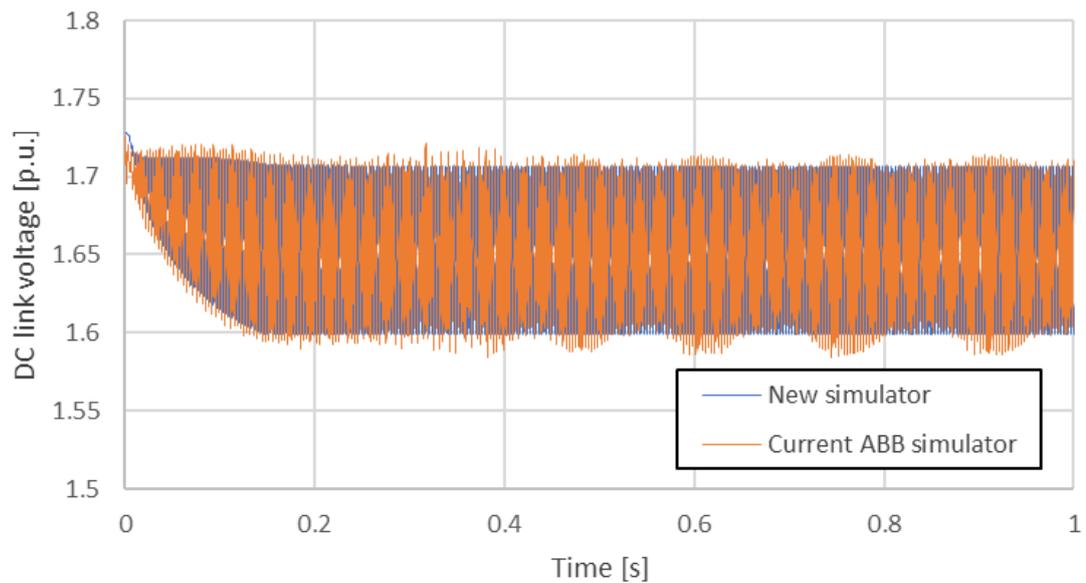


Fig. 5.2.5 DC link voltage as a function of time in acceleration test.

Figure 5.2.5 shows that DC link voltage behaves the same way with both simulators. At the start DC link voltage is around 1.7 pu and when the acceleration starts the average voltage starts to drop and stabilizes around 1.65 pu. The voltage also starts to fluctuate and stabilizes to fluctuating between 1.6 and 1.7 pu. Though there is a minor difference that the current ABB simulator has a little bit more fluctuation in the voltage than the new simulator. Figure 5.2.6 shows zoomed plots of the DC link voltage in the acceleration test.

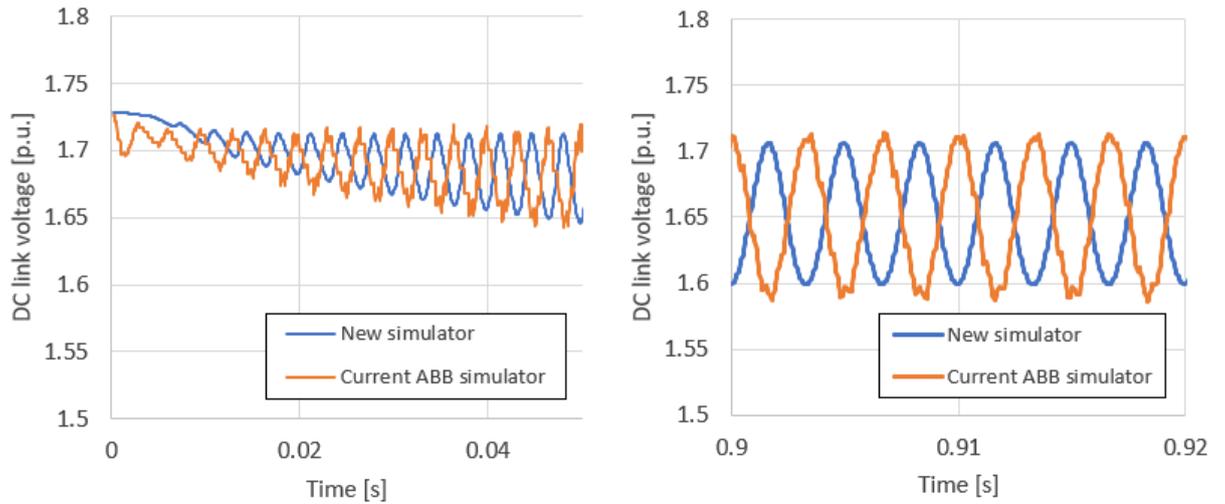


Fig. 5.2.6 Zoomed plots of DC link voltage in the acceleration test.

Figure 5.2.6 shows minor difference between the simulators right at start of the acceleration where the DC link voltage drops a bit faster with the current ABB simulator. At the end of the simulation where DC link voltage is stabilized the voltages look almost the same. There is of course a phase difference between the voltages which doesn't matter, as they could be made to match by changing the phases of the voltages to the diode bridge.

In general, in the acceleration test the new simulator seems to produce same kinds of results as the current ABB simulator. Motor speed and DC link voltage behaved almost the same way with both simulators. Torque was on average the same with both simulators though torque rise time was a little bit slower with the new simulator. Motor current seemed to rise a little bit higher at the start of the acceleration with the new simulator but then settled close to the same level as the current ABB simulator. The biggest difference with the torque and motor current was bigger fluctuation with the current ABB simulator. This can be explained by the current ABB simulator modeling inverter and its switches, whereas the new simulator doesn't model the inverter at all.

5.3 Deceleration

For the deceleration test the motor was running at the nominal speed with torque reference being the nominal torque of the motor. Controller was then given negative nominal torque as the torque reference which made the motor decelerate until the speed reached zero and the motor would then accelerate to the opposite direction. There was also a load which was linear relative to the speed and the load torque matched motors nominal torque at the nominal speed. Both simulators were given maximum DC link voltage limit of 1.9 pu after which they should start limiting the braking torque. Because all the variables are influenced by each other, the results are presented first and reasons for any differences between the simulators are discussed at the end of the chapter. Figure 5.3.1 shows behavior of motor speed in the deceleration test.

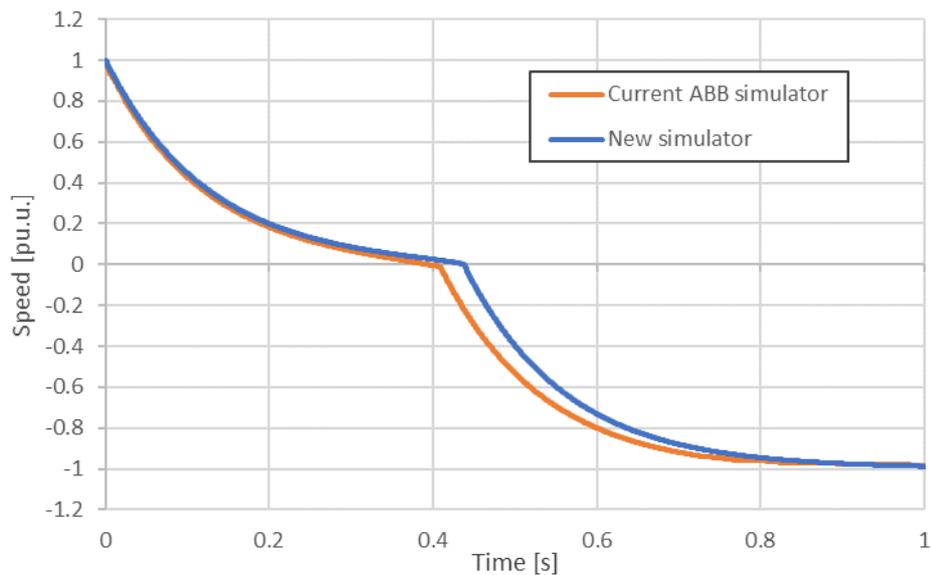


Fig. 5.3.1 Speed as a function of time in the deceleration test.

Figure 5.3.1 shows that the speed curves with both simulators look very similar. The curves are almost identical until 0.2 s after which the current ABB simulator seems to slow down a bit faster and reaches zero speed few hundredths of a second faster than the new simulator. Accelerating to the other direction seems to be a bit faster with the new simulator even though with the acceleration test both simulators seemed to accelerate at the same rate. Figure 5.3.2 shows behavior of torque in the deceleration test.

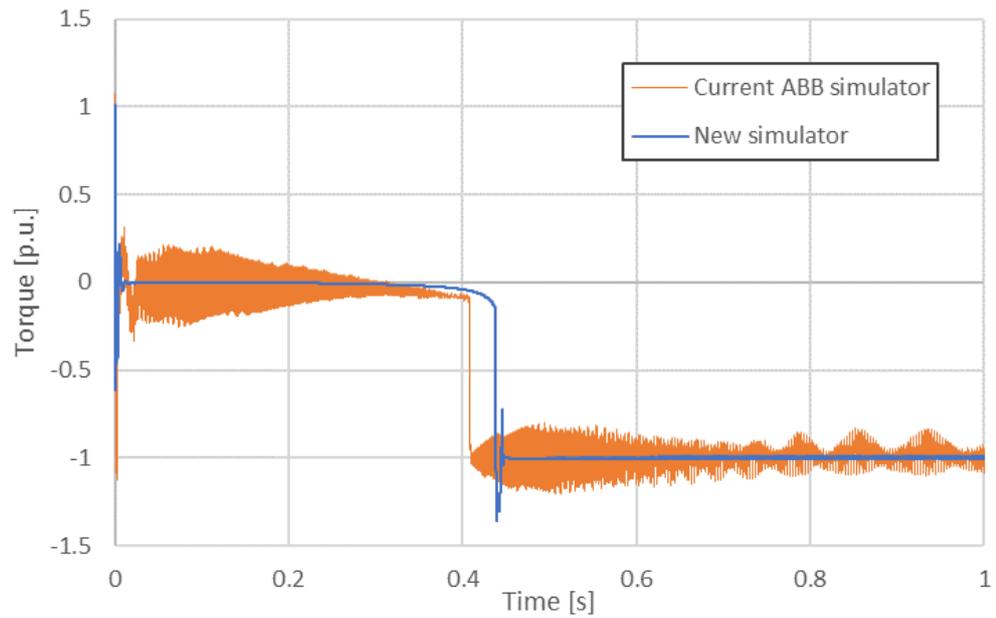


Fig. 5.3.2 Torque as a function of time in the deceleration test.

Figure 5.3.2 shows that the average torque with both simulators seem to match quite well. The current ABB simulator has again generally more fluctuation in the torque and after 0.2s it seems to start producing more braking torque than the new simulator which also explains why it reaches zero speed faster. Figure 5.3.3 shows zoomed plots of the torque in the deceleration test.

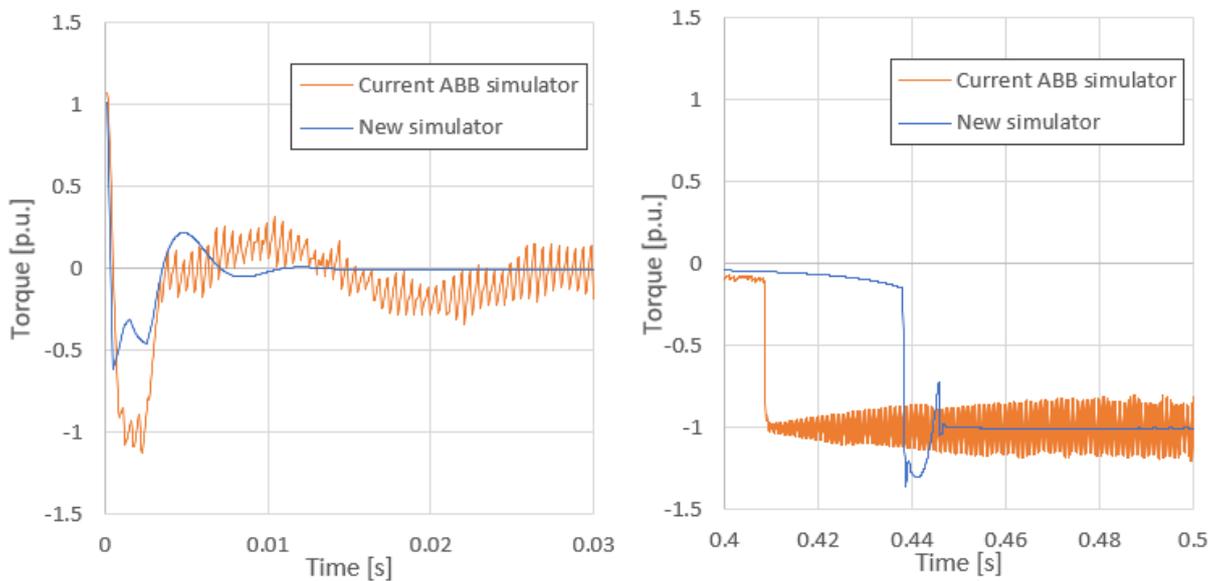


Fig. 5.3.3 Zoomed torque plots in the deceleration test.

Figure 5.3.3 shows that at the start of the deceleration the torque curves behave a little bit differently where the current ABB simulator has a bigger spike in the torque. When the acceleration to the other direction starts the new simulator has some overshoot in torque and stabilizes to the reference torque slower than the current ABB simulator. Figure 5.3.4 shows motor current behavior in the deceleration test.

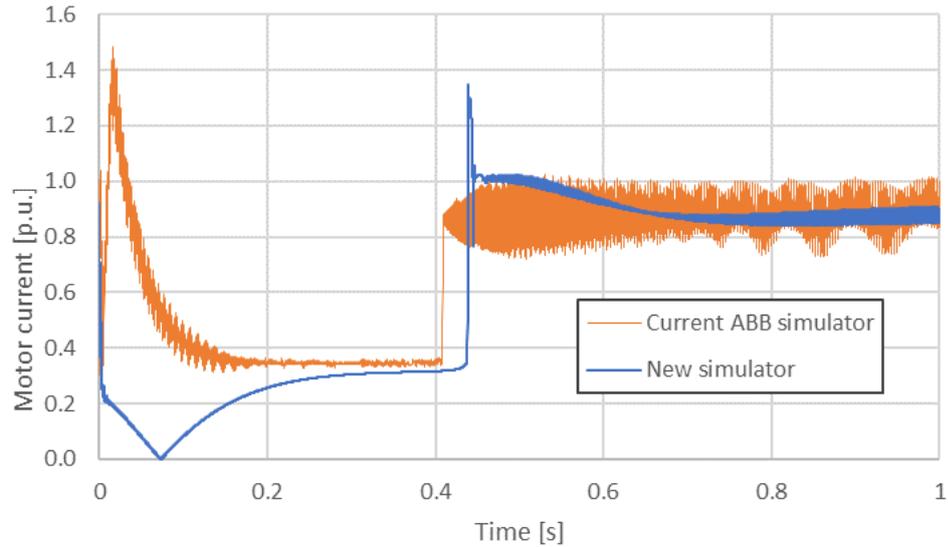


Fig. 5.3.4 Motor current as a function of time in deceleration test.

Figure 5.3.4 shows that the motor current with the new simulator behaves quite differently from the current ABB simulator. At the start of the deceleration the motor current drops to zero at 0.1 s after which it rises slowly to seemingly stable level a little bit lower than the current ABB simulator. When the acceleration to other direction starts there is a big spike in the current after which the motor current seems to behave the same way as with the acceleration test. The current ABB simulator on the other hand has a big spike of around 1.5 pu at the start of the deceleration before stabilizing to a lower level. Figure 5.3.5 shows behavior of DC link voltage in the deceleration test.

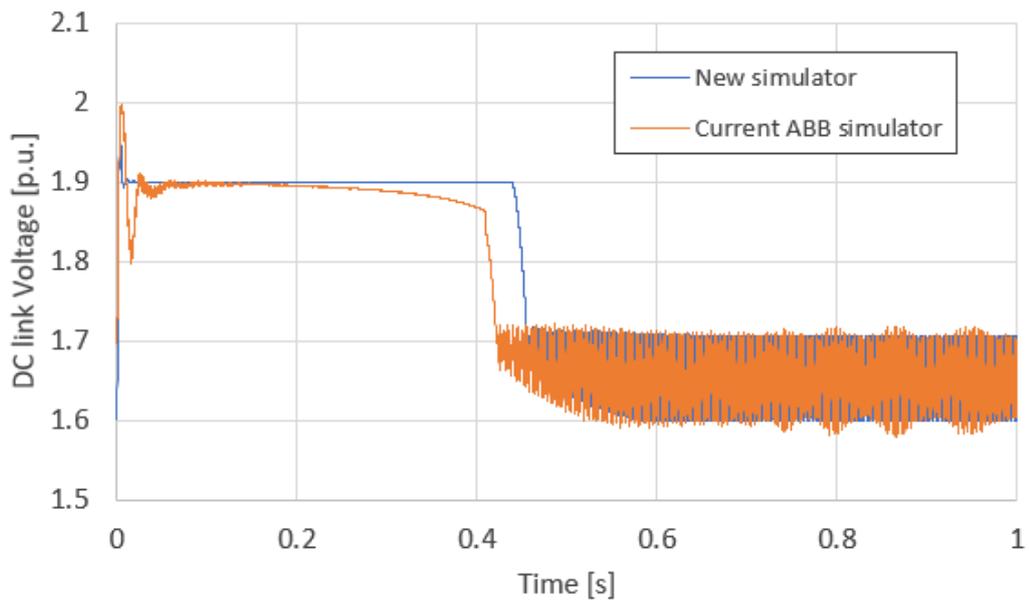


Fig. 5.3.5 DC link voltage as a function of time in deceleration test.

Figure 5.3.5 shows that the DC link voltage behaves quite similarly with both simulators. At the start voltage rises very fast and both simulators start limiting the braking torque to keep the DC link voltage at the specified 1.9 pu. When zero speed is reached and acceleration starts, the DC link voltage drops fast and starts to fluctuate with an average of 1.65 pu. The bigger difference between the simulators is that the new simulators voltage stays at 1.9 pu until it starts accelerating to other direction whereas voltage with the current ABB simulator starts to drop slowly at around 0.15 s while still decelerating. This allows for more braking torque and faster deceleration. Figure 5.3.6 shows zoomed plot of the DC link voltage at the start of the deceleration.

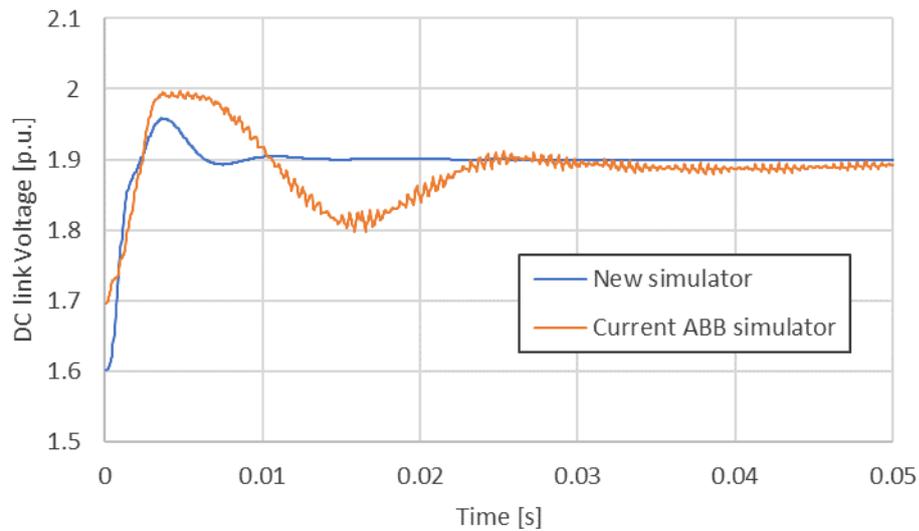


Fig. 5.3.6 DC link voltage as a function of time in deceleration test.

Figure 5.3.6 shows that with both simulators the DC link voltage rises very fast at the start of the deceleration and overshoots a little bit before stabilizing at 1.9 pu. The new simulator stabilizes a bit faster and with smaller overshoot than the current ABB simulator. Though it must be noted that DC link voltages at the steady state seemed to be at different phase, like with the acceleration test in figure 5.2.6. This explains why the simulators have different voltage at the start of the deceleration.

Generally, in the deceleration test both simulators showed similar kind of results where the DC link voltage rises very fast and braking torque gets limited until the speed gets to zero. Biggest difference between the simulators was with the motor current. It was tested that when decelerating, the flux controller in the new simulator was very sensitive to different tunings and would show very different motor current results with different tunings. It is also the reason for the spike in the motor current when the rotation direction changes and probably the reason for the spike in the torque as well. There was also a difference between the simulators in DC link voltage, which is why the current ABB simulator had more braking torque and decelerated a bit faster than the new simulator. The reason for this probably lies in the way that the current ABB simulator calculates the output current of the DC link, as it models the inverter and uses the switch positions and motor currents for calculating the output current. This is inherently different from the way the new simulator estimates the DC

link output current. Still, it must be said that this kind of fast deceleration and direction change is a very challenging task to simulate which makes it a good test for a simulator, but probably isn't that common of a use case without using a braking chopper or a drive with an ability to output the generated electricity back to the network.

5.4 Simulation speed

For the simulation speed test both simulators were made to simulate 100 s and the real time it took to do that was measured. This was done 10 times to get an average of it to reduce the effect of fluctuation in the simulation speed. Time was measured directly in the code by using the same a C++ time tracking function (`std::chrono::high_resolution_clock`) with both of the simulators. The time was measured so that only the time used for the simulation was measured and for example the time used for processing the graphical interface of the current ABB simulator didn't affect the time. Nominal torque reference was given for both simulators and the same linear load was used as with the acceleration and deceleration tests. The simulators were run on a laptop that had Intel Core i7-4810MQ which has 4 cores, max frequency of 3.8 GHz and was launched in 2014. Table 5.1.1 shows the results of the simulation speed test.

Table. 5.1.1 Real time used in simulating a drive system for 100 seconds.

	Time taken for simulation [s]										Average
ABB	99,10	99,69	94,42	97,34	94,00	94,17	93,28	95,22	94,73	93,97	95,59
New	1,016	1,063	1,047	1,031	1,188	1,063	1,078	1,094	1,047	1,141	1,077

Table 5.1.1 shows that the new simulator took on average 1.077 s and the current ABB simulator 95.59 s to simulate a drive setup for 100 s. The ratio of the times is about 1:88 e.g. the new simulator is 88-times faster than the current ABB simulator. So at least compared to the current ABB simulator, the new simulator is very fast to run. The new simulator should also be able to fulfill the goal of simulating even tens of drive systems at the same time in real-time speed. This doesn't of course take into account the resource usage of the whole Virtual Drive program but at least from the simulator perspective it should be possible to simulate multiple drives parallel in real-time, especially with a newer desktop processor.

6 INTEGRATION TO VIRTUAL DRIVE

As part of the task of integrating the new simulator to Virtual Drive, the whole old ABB simulator and its code was integrated to Virtual Drive project. The graphical interface and connections to it was removed. The parts of the old ABB simulator that the new simulator uses were modified accordingly to match what was described in chapter 4. Source code of the current Virtual Drive simulator was modified to accommodate the new simulator and its function calls. Figure 6.1 shows the new simulator running inside Virtual Drive instance that is connected to Drive composer.

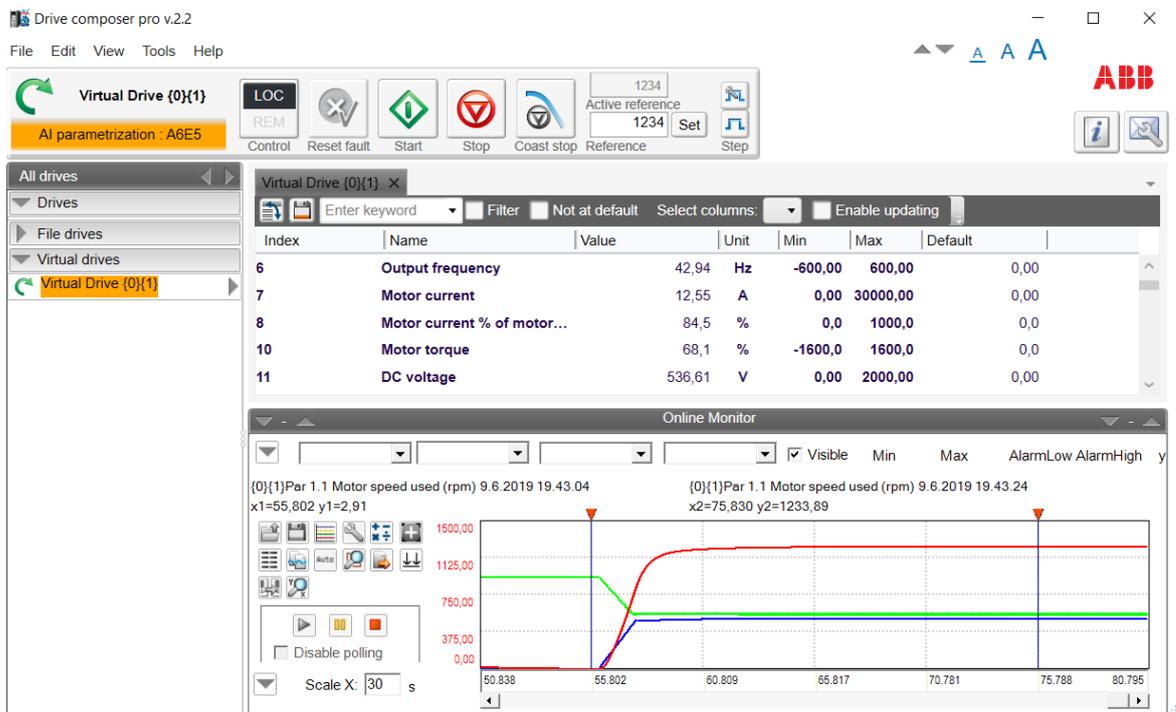


Fig. 6.1 Virtual Drive instance with the new simulator connected to Drive Composer. The plot in the screenshot shows motor speed, torque and DC link voltage changing after changing the speed reference for the drive.

Figure 6.1 shows the new simulator successfully integrated to Virtual Drive and reacting to speed reference given to it. Getting these different simulation variables like torque, DC link voltage and motor current to show in Drive Composer, required connecting the new simulators internal variables to the *high-level control and interfaces* module of the Virtual Drive program shown previously in figure 2.1.1. Some of the drive control parameters that

can be given to a drive were also connected to the new simulator e.g. DC link max voltage for the vector controller.

There were some improvements left that should be made to the simulator in the future. The motor parameters are hardcoded and interface through which to give these parameters should be added. DC link parameters are currently estimated by getting the DC link capacitance value from an internal Virtual Drive variable, and then estimating the other parameters by using a function made to match values from the current ABB simulator database. This also could be changed by having an interface to give these parameters. Better yet for both of these would be to have a list of motors and drives to choose from with the corresponding parameters like in the current ABB simulator.

The old ABB simulator software with vector control was also briefly tested to work within Virtual Drive. In the future it could also be an option to use with Virtual Drive.

6.1 Common DC link

For making the common DC link possible, Virtual Drive instances should have a way to directly communicate between each other. Currently there is no way to directly communicate between the instances and should probably be made possible in the future. For the common DC link, the communication speed is critical because the drive in the new simulator is simulated in 50 μ s time steps. For getting the most accurate results, communicating the voltage and currents between slaves and master should be made after each time step. Depending on what kind of interface is realized and how many drives are used, communicating after every 50 μ s might prove to be too slow in some cases. Other option is to still use the same model but communicate less frequently e.g. after each 1 ms. This produces less accurate results and might cause some oscillations or unwanted behavior which could be fixed by using a filtered DC link voltage with all the drives and not the instantaneous one with lots of fluctuation.

Because there is no communication between Virtual Drive instances, the common DC link model was tested by using RobotStudio, to which there is an interface. Three Virtual Drive

instances were launched and connected to Robot Studio which handled inputs and outputs of these instances. One was set as the common DC master and other two as slaves. Common DC link master had input for the external load currents and output for DC link voltage which were connected to the two slave instances. RobotStudio updated these inputs and outputs for every 1 ms of simulation time. The DC link voltage that was shared between them was filtered for making it more stable. Figure 6.1.1 shows the three Virtual Drive instances connected to Drive Composer.

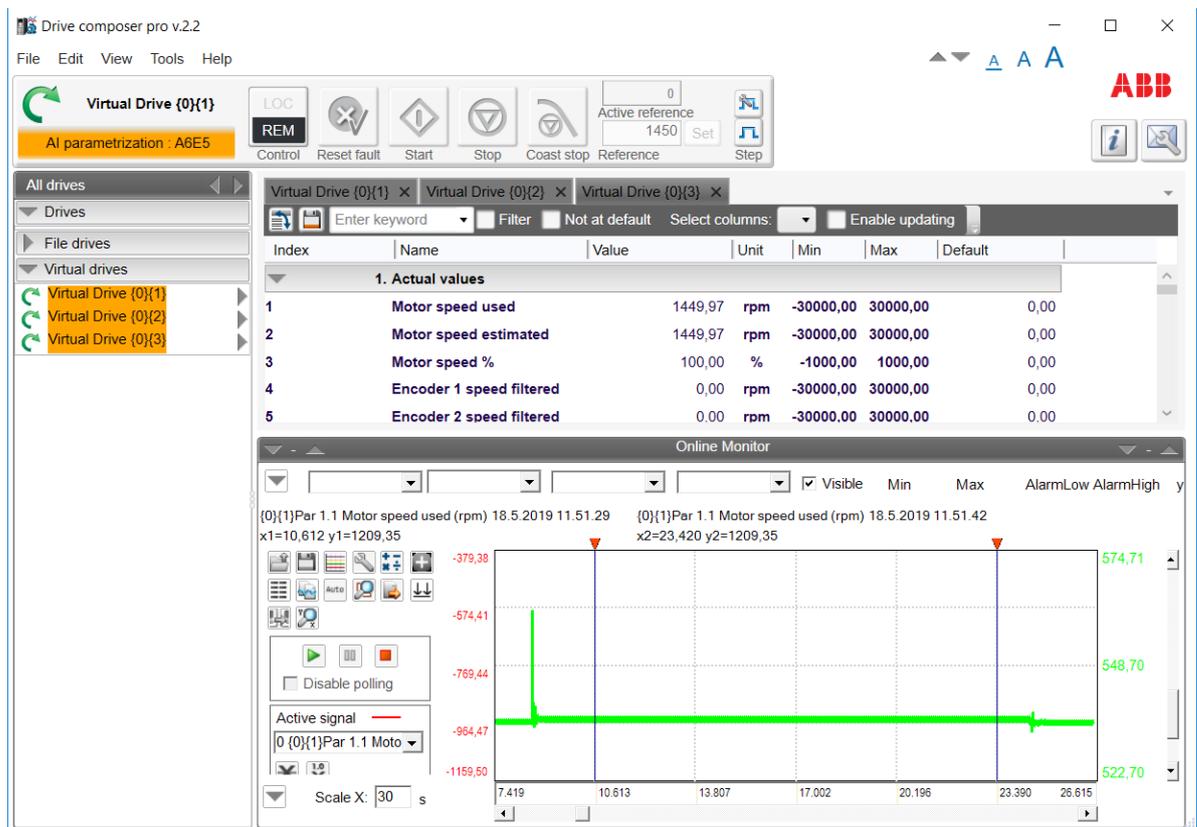


Fig. 6.1 Three Virtual Drive instances connected to Drive Composer and running a common DC link setup. The green line chart on the bottom of the screenshot shows DC link voltage that was shared between all drives.

Figure 6.1 shows that at least on principle the setup and model can be seen to work. At first all three drives were running on nominal speed with 80% of the nominal load. Then one of them decelerated to zero, from which there is a big spike in the chart. Later, the same drive accelerated back to nominal speed, from which there is a smaller spike in the chart. The DC

link voltage level also seems to change depending on if the one drive was running or not. Different kinds of runs were performed with the drives accelerating and decelerating and the system seemed to stay stable, though further testing might be required and comparison to a real setup should also be made.

7 CONCLUSIONS

Different use cases for ABB Virtual Drive software were studied and different solutions for realizing them were considered. Because of multiple different use cases and users, it was concluded that Virtual Drive should also have multiple different simulation models or a way to add or connect different models to it. In some cases, there might be a need to simulate multiple drives parallel in real-time and on the other extreme there might be a need to simulate single drive as accurately as possible, and everything in between. Special attention was given to realizing a common DC link setup and it was concluded that it also could require multiple models of a DC link to cover all the cases. For covering many of the use cases and opening the way for improvements in Virtual Drive simulations, a new simulator was created for Virtual Drive. Main objective for the simulator was to provide real-time simulations even when running multiple simulators simultaneously, and also to provide ability to simulate a common DC system. Two ABBs internally used simulators were presented and used as a base for the new simulator. The new simulator uses vector control and models the motor, mechanics and drive. Simulating the inverter and its switches was omitted to keep the simulator simpler and faster. One way of creating a common DC link setup was realized to the new simulator. The new simulator was compared against ABBs high-fidelity simulator and was found to produce very similar results. In a steady state test, from the variables studied, only motor current with smaller loads differed noticeably from the ABB simulator. In transient situations with acceleration and deceleration tests, again motor current was the variable that had biggest differences from the ABB simulator. There was also much less fluctuation in torque and motor current with the new simulator because it doesn't model the inverter and it's switches. Simulation speed was also tested, and the new simulator could simulate almost hundred times faster than real-time or the ABB high-fidelity simulator. The new simulator was integrated and tested to work correctly within Virtual Drive. Creating a common DC link system with multiple Virtual Drive instances was also tested and found to work at least in principle.

For future work there is lot to be improved inside Virtual Drive simulator. More models and ways for connecting to external models could be added. The new simulator could be tested more thoroughly with different motor and drive parameters. Vector controller and its tuning

parameters could be improved and tested more thoroughly, especially the flux linkage controller. Common DC link systems could be studied more and additional DC link models for them could be made. Also, real testing and comparison of a common DC link system could be made. ABBs internally used simulators could be integrated to Virtual Drive. Especially the older one that was somewhat integrated already as part of this thesis and was also tested to be working.

REFERENCES

Aarniovuori, L. 2005. Sähkökäyttösimulaattori ja sen suorituskyky (Electric drive simulator and its performance). LUT University. URL: <http://lutpub.lut.fi/handle/10024/29947>

Aarniovuori, L. 2010. Induction Motor Drive Energy Efficiency - Simulation and Analysis. LUT University. URL: <http://lutpub.lut.fi/handle/10024/63346>

ABB. 2014. Application guide: ACS880-01 drives and ACS880-04 drive modules Common DC systems. URL: https://www.convertingsystems.com/uploads/2/6/8/5/26859557/acs880_common_dc_bus.pdf

De Doncker, R.W. and Novotny, D.W. 1994. The universal field oriented controller. IEEE Transactions on Industry applications. Vol 30. No.1. pp. 92–100

Drury, B. 2009. The Control Techniques Drives and Controls Handbook. London. The Institution of Engineering and Technology

Hughes, A., Drury, B. 2013. Electric Motors and Drives: Fundamentals, Types and Applications. Oxford. Newnes

Leonhard, W. 2001. Control of Electrical Drives. Berlin. Springer-Verlag

Nam, K. H. 2019. AC Motor Control and Electrical Vehicle Applications. Florida. CRC Press

Tiihonen, T. 2005. Taajuusmuuttajan häviöidensimulointityökalun kehitys. LUT University. URL: <http://lutpub.lut.fi/handle/10024/29985>

Trzynadlowski, A. 2001. Control of Induction Motors. San Diego. Academic Press

Trzynadlowski, A. 1994. The Field Orientation Principle in Control of Induction Motors. Boston. Kluwer Academic

Pyrhönen J., Jokinen, T., Hrabovcova, V. 2008. Design of Rotating Electrical Machines. Wiltshire. J. Wiley & Sons

Vas, P. 1998. Sensorless Vector and Direct Torque Control. New York. Oxford University Press