

Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar

Automatizálási és Alkalmazott Informatikai Tanszék

Sipos Gergő

NEM az Elektronikus terelők

Kisfeszültségű folyamatirányítók

Konzulens

Dr. Érték Elek

BUDAPEST, 2018

Tartalomjegyzék

[Összefoglaló 3](#_Toc332798843)

[Abstract 4](#_Toc332798844)

[1 Bevezetés 4](#_Toc332798845)

[1.1 Formázási tudnivalók 4](#_Toc332798846)

[1.1.1 Címsorok 4](#_Toc332798847)

[1.1.2 Képek 4](#_Toc332798848)

[1.1.3 Kódrészletek 4](#_Toc332798849)

[1.1.4 Irodalomjegyzék 4](#_Toc332798850)

[2 Utolsó simítások 4](#_Toc332798851)

[Irodalomjegyzék 5](#_Toc332798852)

[Függelék 5](#_Toc332798853)

# Introduction

The main goal of my Hardware-in-The-Loop system is to make the development of an electric drive control unit easier. First, let’s take a look at how the electric drive system works.

The drive system that consist of the grid, a three-phase rectifier, a DC link, a three-phase two-level inverter model and an asynchronous machine.

The grid provides the three phase voltages, 230 Volt RMS each, with three grid inductivities. The three-phase converts this three-phase voltage into DC voltage around 540 Volts. The DC Link contains a capacitor, which reduces the voltage ripple and a reactor, which reduces the current ripple, thus makes a more stable DC supply. The voltage inverter, controlled by the drive control unit, converts the DC voltage into a controlled AC voltage for the asynchronous motor.

The control unit provides the control signals for the gate driver unit and regulates certain motor parameters. The gate driver unit gives voltage to the gates of the transistors in the inverter. This determines the output voltage of the inverter. Measured parameters, which make the regulation possible, are fed back to the control unit. If you want to test the control system’s algorithm, you can use non-real-time tests on a PC platform.

But if you want to test the control unit itself, you either need a testbench with an inverter and a motor, or a HIL system. The benefit of a HIL system is that it is much cheaper, and easier to test the control unit, and you can also monitor motor parameters, which are very difficult to measure.

It is very useful and important to be able to easily and cheaply test the drive control unit, therefore, it is used widely in the industry for advanced development.

# The fundamentals of real-time modelling

## Mathematical tools

### Simulation time

The FPGA clock cycles are discrete steps, so the HDL coder only supports discrete time codes. Therefore, the Matlab Simulink model is in discrete time as well.

### Number representation

There are two options for number representation, these are floating point and fixed-point data types.

Floating point data type numbers consist of three parts, a signum bit, exponent bits and fraction bits. The fraction bits represent the number itself, and the exponent bits determine where the binary point is. Because of this, you can represent numbers in an enormous scale.

Fixed-point numbers, on the other hand, have a signum bit and an integer part, with a pre-defined binary point. If we know the value of the representable number, fixed point representation is more accurate than floating point.

But if we don’t know the exact range, then the following problems may occur:

* Its range is much lower than using floating point. If the representable number is bigger than the limit, the fixed-point number overflows.
* If the representable number is small compared to the maximum representable value, it will contain a lot of unnecessary zeros, and it will be inaccurate.

I use only fixed-point data types in the model, because it is much faster for FPGA calculations, and it is pretty easy to know the ranges of the variables in the model (HDL Coding mostly supports fixed-point data types only).

### Discrete numerical approximation methods

For approximating differential equations solutions, we can use any of the Runge-Kutta methods. The first-order Runge-Kutta method is the Euler method. The explicit Euler method looks like the following:

Where *y* stands for the variable, *t* stands for time and *Ts* stands for the fundamental step size.

While the implicit Euler-method:

These are one-step solutions, which are really simple and can be calculated very quickly.

There are more complicated Runge-Kutta methods, like the fourth-order and the second-order Runge-Kutta. For example, the second-order explicit Runge-Kutta method is a two-step method, which calculates as follows:

The error of the Runge-Kutta methods are determined by the fundamental step size. The global error of each method is proportional to the step size raised to the power of the Runge-Kutta’s order.

Therefore, these higher-order Runge-Kutta methods are much more precise, than the Euler method, but they are more complicated, and are much harder to compute.

With using higher-order Runge-Kutta methods, the simulation can be more precise. However, with the Euler-method, the easier computations make me able to reduce the fundamental step size. This compensates the bigger error and reducing the step size is very good for the other calculations in the model.

For example, here is the differential equation of the inductivity, which I want to implement in my model in discrete time:

Examining a step size interval:

Converting the equation using the backward Euler method:

I use the implicit (forward) Euler method, because I want to determine the current of the inductivity for the same clock cycle, that is in the input voltage, not the previous one.

Ami hiányzik

Fix pontos számábrázolás, mik vele a gondok ( túlcsordulás, pontosság, műveletek költsége ( összeadás, szorzás, osztás)...)

Euler nél mi a jelentősége a TS nek ( lehetne ábra Newton módszerről ( derivált \* t + előző minta, ezen ábrán meg lehet mutatni a ha nagya lépés köz

Említsük runga-kutta ( elsőredű, másodrendű, mi a kapcsolata eulerrel -> miért használjuk az euler ezek helyett )

## Technical environment

For the implementation, I use the Zedboard (type!!). The Zedboard is a development kit, containing a high-performance FPGA and an ARM processor with an SD card. It also has plenty of interfaces, like micro-USBs, ethernet, Pmods and switches.

### FPGA

The FPGA stands for Field-Programmable Gate Array. It is an integrated circuit designed to be configurable. The basic components of the FPGA are flip-flops (registers) and lookup-tables (LUTs).

Flip-flops are responsible to store data between operations in the FPGA. They require a clock cycle

Lookup-tables are basic blocks in the FPGA, which can implement any logic functions of the input variables, it operates as a truth table.

Besides these fundamental blocks, the FPGA contains digital signal processing (DSP) blocks, which are made for fast arithmetic calculations (adding, multiplying). The DSP

It also contains Block RAMs (BRAMs), which can contain larger amounts of data.

# Creation of the system elements real-time models

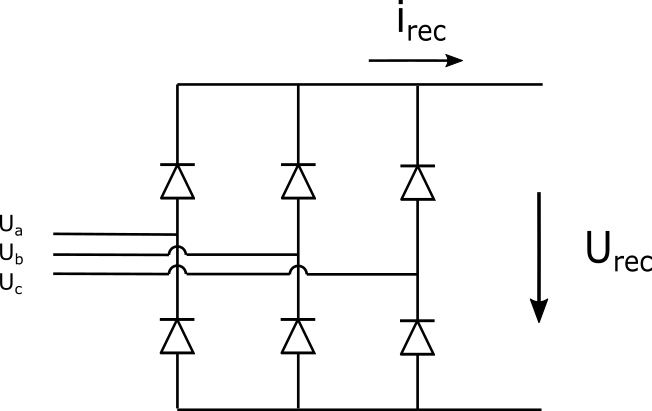
## Grid model

The Grid model has three phases. Each phase consists of a voltage generator and a grid inductivity.

The model calculates the grid phase voltages (Ua, Ub, Uc) as its output.

## Rectifier

The rectifier model contains 6 diodes, 2 for each phase. Its input is the 3 phase voltages (Ua, Ub, Uc) from the grid model and the rectified current (irec) from the DC link. Its output is the rectified voltage (Urec). The schematic circuit diagram of the rectifier:



1. Figure: Schematic of the rectifier

I use logic operations to compute the output voltage. The basic idea is that on the high side, the phase with the highest potential has the conducting diode. On the low side, the phase with the lowest potential has the conducting diode.

First, the model checks, if the Ua phase voltage is higher than the Ub and Uc voltages. If it is, the model passes it into the positive potential of the rectifier, Urec+. If not, the model checks whether Ub or Uc is higher and passes it to the Urec+ signal.

The low side of the bridge computes in a completely analogic way. The model checks if the Ua phase voltage is lower than the Ub and Uc voltages. If it is, the model passes it into the negative potential of the rectifier, Urec-. If not, the model checks whether Ub or Uc is lower and passes it to the Urec- signal.

Then, the model computes the original rectified voltage (Urec,org), with the following equation. This does not contain the diode voltage drops.

The average voltage Urec,avg of the rectifier can be calculated from the phase peak voltages (Upeak, assuming symmetrical phase voltages), with the following equation:

With my model having 325 V peak voltages, the result of the equation is around 537.5 V

The diode drops (Udiode) are determined by the rectified current. The model contains a lookup table, which assigns the voltage drop to the rectified current. I modelled the forward characteristics of a power diode:

A screenshot of a cell phone

Description generated with high confidence

2. Figure: Forward diode characteristics [2]

The 1-dimensional lookup table models the 25°C line. HDL Coder doesn’t support interploation (it would be too much computing anyway), so I filled up a 128 element lookup-table, so it is accurate without the extrapolation.

The diode drops at a certain moment are the same on the high and the low side, because the diodes are usually same (they are modelled as being equivalent), and the same current flows through them. Therefore, the calculated rectified voltage (Urec):

## DC link

The DC link model contains a choke inductivity and a DC capacitor. The models use the formerly presented Euler method. The DC link’s input is the rectified voltage from the rectifier and the DC current from the inverter model. Its outputs are the DC voltage and the rectified current.

<https://hu.mouser.com/ProductDetail/KEMET/C44HLGR6400AASJ?qs=sGAEpiMZZMsh%252b1woXyUXjzRu9w46NtHr%252bEdhPtGVC0k%3d>

<https://hu.mouser.com/ProductDetail/KEMET/SS26V-300028?qs=sGAEpiMZZMsVJzu5wKIZCRKkZmKdbMQOky%252b7rTdJ2X4%3d>

## Inverter

The inverter contains three bridge branches, each of them having a 2 transistor-diode pairs. Its input is the DC voltage from the DC link, the transistor gate signals from the control system, and the three phase currents from the motor model. It uses logic operations to determine its outputs. Its outputs are the three phase inverted voltages, the DC current, and the short-circuit signal.

## Asynchronous machine

The asynchronous machine model uses a reduced parameter equivalent circuit. The reduction is necessary, because it doesn’t require to model the rotor leakage inductance, which makes the modelling much easier. It operates in x-y coordinates, and performs d-q transformations in the end. Its inputs are the three phase voltages and the load torque. Its outputs are the three phase currents, the rotor angular speed and rotor position, and every electrical parameter in either x-y or d-q coordinates, which can be used for testing.

# Design of the FPGA environment

The FPGA does not support floating-point variables, so I used fixed-point variables in my models. The FPGA has digital outputs, and I used sigma-delta conversions for the model outputs.

# 5 Verification

I used the Matlab Simpower System toolbox for the model verification. To verify the implemented model, I made an RC filter to have analog output signals and measured them to see if the model fits the requirements.

Irodalomjegyzék

1. <https://www.xilinx.com/support/documentation/sw_manuals/ug998-vivado-intro-fpga-design-hls.pdf>
2. <https://www.infineon.com/dgdl/Infineon-FF650R17IE4-DS-v03_03-EN.pdf?fileId=db3a30431ff9881501201dcfe2a54986>