CODEN:LUTEDX/(TEIE-5234)/1-108/(2007)

Industrial Electrical Engineering and Automation

A Silicon Carbide Inverter for a Hybrid Vehicle Application

Oscar Haraldsson

Martin Andersson

Dept. of Industrial Electrical Engineering and Automation
Lund University

Abstract

This master thesis investigates how silicon carbide (SiC) bipolar junction transistors (BJT) can improve the power inverter in a hybrid car. Today's power electronics in a hybrid car needs a cooling system separated from the main cooling system. A separated cooling system increases the cost. The idea is to replace silicon transistors with silicon carbide transistors which can withstand higher temperatures. To study a complete three-phase inverter a PSpice model of a SiC BJT is built. The SiC BJT is provided from the company TranSiC that is developing a SiC transistor called BitSiC. The BitSiC model is used to simulate an inverter for a 5 kW electric machine (EM). The EM is intended for a Belt driven Alternator Starter (BAS) in a hybrid car of medium size. The SiC semiconductors has several interesting advantages compared to Si based semiconductors:

- SiC can withstand high temperatures, up to 600ºC.
- Higher thermal conductivity, lowering the junction to case thermal resistance.
- Positive temperature coefficient, easier to parallel.
- Faster and less reverse recovery, reducing switching losses

Through simulation this master shows that silicon carbide transistors can use the exiting cooling system of a car. With the SiC BJT low switching losses and low conduction voltage the efficiency reaches values of 98%.

Keywords: Silicon Carbide, SiC, Bipolar Junction Transistor, BJT, Inverter, Hybrid Car, BAS, PSpice.

Preface

Although the master thesis did not turn out as planed, it could not have turned out any better. Through hard work and wise guidance it was possible to simulate and evaluate a three-phase inverter based on silicon carbide (SiC) transistors. This master thesis was formed when professor Mats Alaküla (Professor, IEA), Bo Hammarlund (CEO, TranSiC) and Lars Hoffman (Project Manager, Saab Automobile AB) recognized the potential of SiC components in a hybrid car. The first plan was to build an inverter with SiC components. But as often when working with cutting edge technology it is necessary to adapt the goals to what is possible. Instead of building an inverter PSpice models of TranSiC's SiC transistors is built. This led to a simulation of a 5 kW inverter with SiC transistors.

It has been a joy to work on our master thesis and with the people at the department of Industrial Electrical Engineering and Automation (IEA) at Lund Institute of Technology. We would especially like to thank Mats Alaküla that has inspired and guided us through the entire project. We also would like to thank Bo Hammarlund, Martin Domeij (CTO, TranSiC) and Lars Hoffman for their support.

Last but not least we would like to thank Gunnar Lindstedt (Assoc Prof., IEA), Hans Bängtsson (guest lecturer), Getacew Darge (Research Eng., IEA), Thomas Bergh (Tech. Lic., IEA), Dan Hagstedt (PhD student, IEA), Jonas Johansson (PhD student, IEA) and the rest of the people at IEA. Thanks for the hospitality, we have been well received and felt us like one in the staff at IEA, we especially appreciate all the cakes that has been given to us.

Thanks to our families for all the support and encouraging our studies trough all the years.

Lund March 2007

Oscar Haraldsson Martin Andersson

_____ _____

Contents

Abbreviations

µC	Microcontroller
AC	Alternating Current
BAS	Belt Driven Alternator Starter
BJT	Bipolar Junction Transistors
CMOS	Complementary Metal Oxide Semiconductor
CSDF	Common Simulation Data Format
DC	Direct Current
DCB	Direct Copper Bonded
DIL	Dual In-Line
ECE	Economic Commission for Europe
EM	Electric Machine
EM	Electrical Machine
EMF	Electromotive Force
EMI	Electromagnetic Interference
EUDC	Extra Urban Driving Cycle
Ib	Basecurrent
Ic	Collectorcurrent
ICE	Internal Combustion Engine
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IGBT	Insulated Gate Bipolar Transistor
LED	Light Emitting Diode
MB	Megabyte
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
MVEG	Motor Vehicles Emissions Group
PCB	Printed Circuit Board
PWL	Piece-Wise Linear
PWM	Pulse Width Modulation
RISC	Reduced Instruction Set Computer
Rthch	Thermal resistance case heatsink
Rthha	Thermal resistance heatsink to ambient
Rthjc	Thermal resistance junction to case
sat	Saturation
SCC	Switched Current Controller
Si	Silicon
SiC	Silicon Carbide
TUT	Transistor Under Test
UDC	Urban Driving Cycle
Vbe	Base-Emitter Voltage
Vce	Colector-Emmiter Voltage
VSTIM	Controled voltage source in PSpice

1. Introduction.

This master thesis is planed to result in an inverter for a light hybrid vehicle. The inverters main component is silicon carbide (SiC) based power bipolar junction transistors (BJT). The advantage with silicon carbide semiconductors is that it has a larger bandgap, and it can withstand very high temperatures. In the vehicle industry this is especially interesting because it makes it possible to place the power electronics in the engine bay. In the engine bay the temperature can easily reach 120 degrees Celsius. With ordinary silicon semiconductors it is a problem to cool the electronics because the silicon cannot handle higher temperatures. With power electronics that can withstand higher temperatures one does not have to have a separate cooling system, which saves cost.

First the size of the Electrical Machine (EM) is determined. The EM is intended for a belt driven alternator starter (BAS). This EM will give both propulsion and act as a generator and starter for then internal combustion engine (ICE). The size of the EM will give the corresponding power ratings needed for the inverter. The optimal EM-size is obtained by different simulations with a hybrid car-Simulink program developed at IEA.

Due to a lack of SiC transistors the goal is to examine a SiC transistor and build a PSpice BJT model of it. With this model a complete three-phase inverter is simulated in PSpice.

Building the model of the transistor is rather difficult. The idea is to take a transistor model that have similar behaviour and “tweak” it to the behaviour of a SiC transistor.
A test rig is built. With the test rig a single switching can be done which gives the behaviour of the transistor. During the construction of the test rig many different types of driver-circuits is tried out.

With a working model switching signals for the control of the inverter is needed. In the car-Simulink program the maximum load of the EM-size is simulated, and thus giving us the working speed of the EM. With another Simulink program a three-phase inverter is simulated which gives us the control signals for different types of switching.

Simulating a whole driving cycle takes to long time because of the calculations of transients in each switching. With the chosen working point the switching parameters and back electromotive force (EMF) with the initial load current is extracted from the Simulink program. With all these signals and parameters one fundamental period of the EM is simulated in PSpice with a reasonable time and accuracy.

The next step is to design a complete three-phase inverter, the work is started with the same type of drive circuit that is used in the test rig. Now complete simulations are done with different switching frequencies. When a simulation is done the efficiency of the converter is calculated to see how much power that is needed to be cooled away by the cooling system.

2. The BitSiC

The BitSiC is a silicon carbide (SiC) bipolar junction transistor from the company TranSiC [1]. This means that instead of using a doped silicon (Si) wafer the BitSiC is made out of doped SiC wafer. SiC have its pros and cons.

2.1. Silicon carbide

One advantage with SiC is that it has about three times wider bandgap than Si. With a wider bandgap the electrons need higher activation energy to jump to the conduction band. Higher activation energy means that SiC devices can operate in a stable way at higher temperatures (over 200°C) with low leakage currents. Another advantage with SiC is that the electric breakdown field is about ten times higher than for Si and SiC can therefore withstand ten times higher electric fields than Si before breakdown. This advantage can be used in two ways, either is the SiC BJT made with a thinner and more highly doped collector region compared to a Si BJT or else it can be kept as the same size as the Si BJT. If the SiC BJT is made thinner it can withstand the same the electric fields as the regular Si BJT. A result of the thinner SiC BJT is that its turn-on and turn-off is made much faster as well as the on resistance becomes much lower. If the regular Si thickness is kept the SiC BJT will withstand much more electric field than the Si BJT [2].

2.2. The silicon carbide transistor structure

The SiC bipolar transistor (BitSiC) developed at TranSiC is made with epitaxial 4H-SiC wafers. The SiC BJT is built up with different doped layers as shown in figure 2.1.

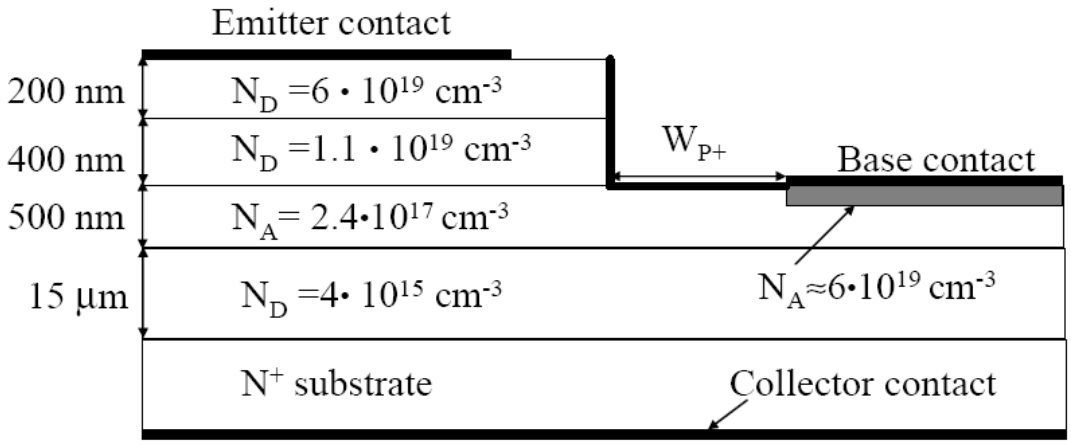


Figure 2.1 A cross section of the BitSiC [3]

During this project two samples of the BitSiC transistor is examined. The first BitSiC (referred in this thesis as BitSiC1) has a TO-247 package as can be seen in figure 2.2.

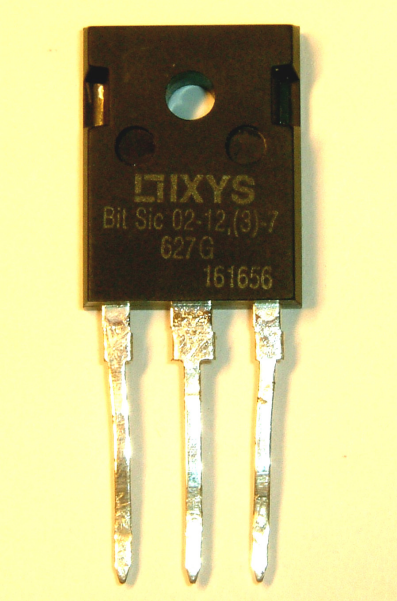


Figure 2.2 BitSiC1 in a TO247 capsule.

BitSiC1 have problems with contact resistances, this result in a rather low gain and power capability. The BitSiC1 is used to develop the test rig. When TranSiC have produced an updated version of the BitSiC, we were given a sample. The second sample of the BitSiC (referred in this thesis as BitSiC2) has an open capsule with a direct copper bonded (DCB) substrate (figure 2.3).

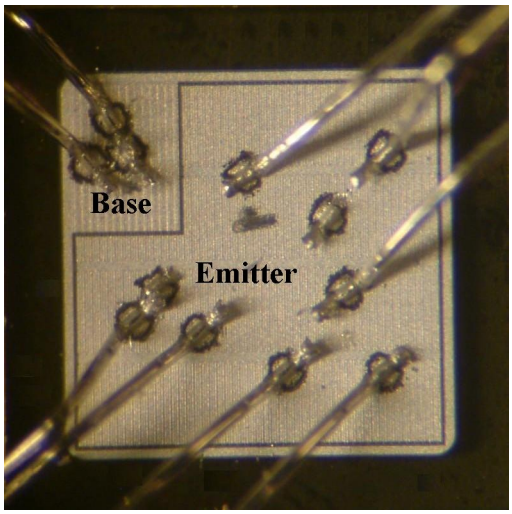


Figure 2.3 BitSiC2 magnified with bonding wires. The collector is glued with silver epoxy on the other side to the DCB substrate.

2.3. The SiC market

Currently there is no company that sells SiC transistors although there are a lot of researches being done. One example is the BitSiC that the company TranSiC is planning to start selling in the near future. However there are other SiC devices on the market. Currently two companies has released a SiC diode family, Infineon[4] and Cree[5]. Cree has released their Zero Recovery Rectifiers SiC family which include SiC diodes in three voltage ranges 10-20 A @300 V, 1-20 A @ 600 V and 5-20 A @ 1200 V. Zero Recovery refers to that a SiC diode has almost no reverse recovery due to the low capacitance. How this affect the switch and reverse recovery is seen in the figure 2.4. The figure is taken from an application note from Cree [6]

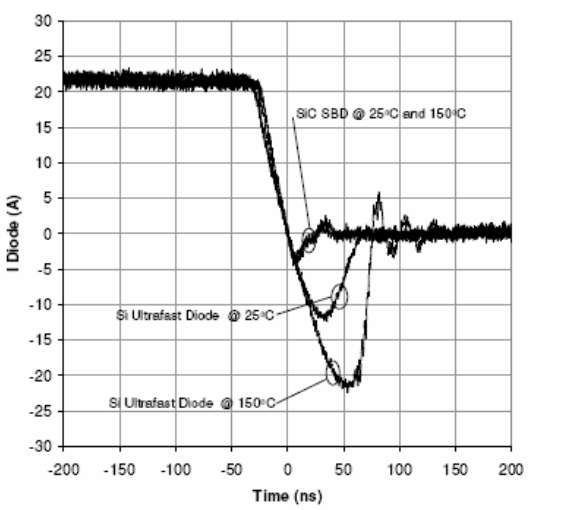


Figure 2.4 Reverse recovery current for a Si Ultra fast diode and a SiC Schottky diode @ 25 °C and 150 °C.

In the oscilloscope illustration one can see that the SiC Schottky diode has almost no reverse recovery. Neither does the SiC diode show any temperature dependence.

3. Path of simulation and design

A 5 kW inverter will require allot of components that there is no possibility to get hold of due to the low current production. This led to a new heading of the master thesis. This shifts the focus to simulating a 5 kW inverter, leading to a whole new array of questions and problems. The biggest problem is that there is no working model of the SiC BJT, so a spice model has to be built. The master thesis is expanding to a complex working path. Working with the full span from building a model, to simulating a three-phase inverter. This means that every thing from laboratory work i.e. building the test rig to simulating in several simulation programs has to be done. Because of this complex working path it is decided to write this chapter to help the reader to see the big picture. In figure 3.1 one can see the path used to produce the simulation result of this master thesis.

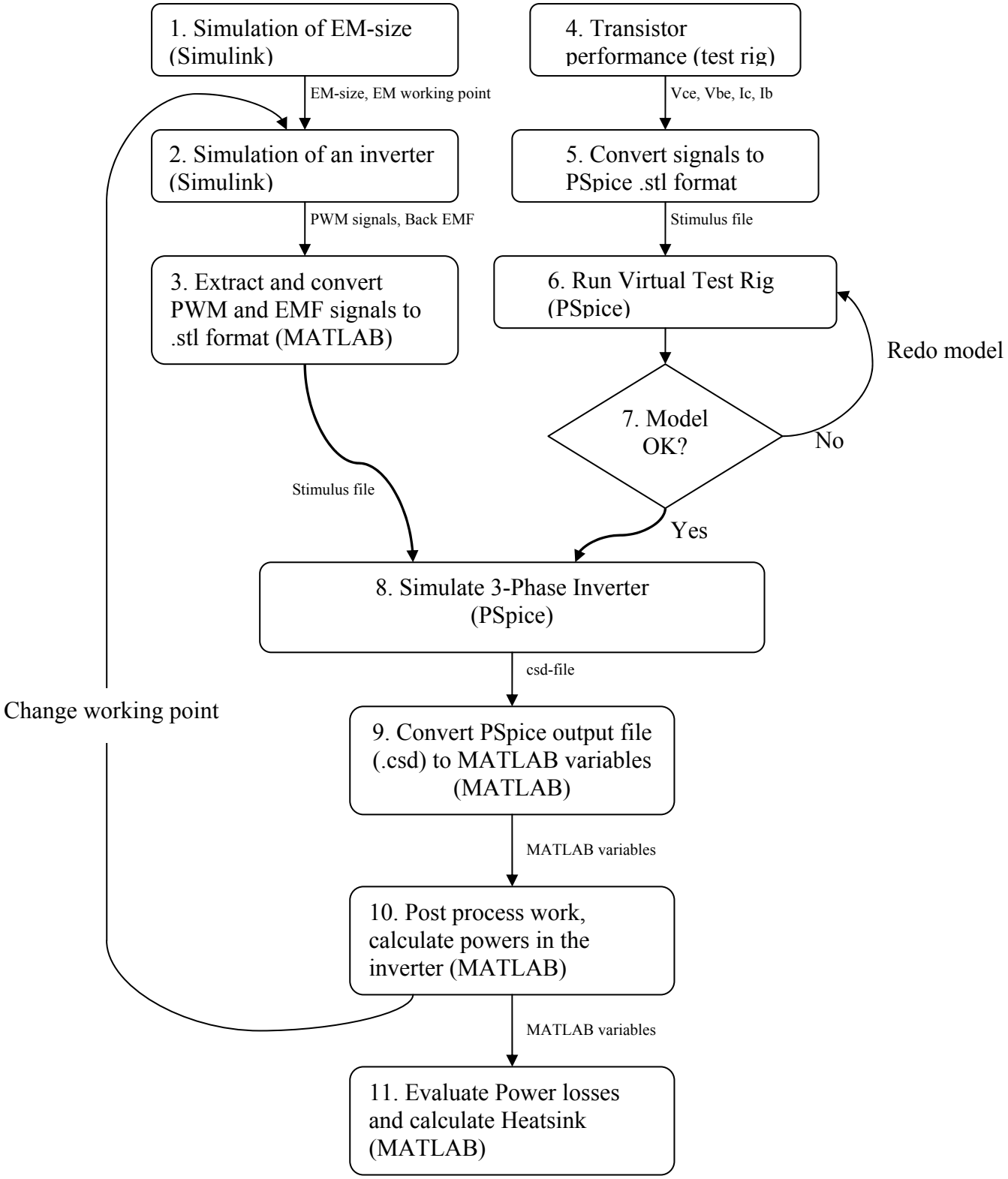


Figure 3.1 Path of simulation and design.

3.1. Steps of the simulation path

1. In the first step the optimal size for the electrical machine (EM) in a light hybrid car is simulated. To simulate a light hybrid vehicle a Simulink model developed at IEA is used. The simulation is simulating a parallel hybrid that can drive certain standard driving cycles. By monitoring the fuel consumption for different configurations an optimal size for the EM is given. The EM size in conjunction with the battery voltage gives the current rating for the three-phase inverter.

2. In the second step a simulated three-phase inverter with ideal switching is simulated in Simulink. The purpose of this simulation is to extract the pulse width modulation (PWM) signals and the back EMF of the EM. These control signals is later transferred to PSpice to control the "real" three-phase inverter. The back EMF is also extracted and transferred to PSpice to be implemented in the model of the electrical machine. In short the values of the battery voltage, modulation type and motor parameters are given to the Simulink program. In return the simulation is simulating the PWM signals and the back EMF for the electrical machine.

3. In PSpice there is a part named VSTIM. VSTIM is a voltage source that takes .stl files as input. The purpose of VSTIM is to give the PSpice user a tool for creating user defined voltage sources. These voltage sources is used as a bridge from MATLAB to PSpice. The .stl file format look something like the following:

```
.STIMULUS Sa PWL
+ TIME_SCALE_FACTOR = 1
+ VALUE_SCALE_FACTOR = 1
+    ( 0, 0 )
+    ( 0.002       ,  0.000 )
+    (+0.000000001,  1.000 )
+    (+0.001       ,  1.000 )
+    (+0.000000001,  0.000 )
```

From the STIM-file one can read out that the signal name is Sa and that it is a piece-wise linear (PWL) signal. The voltages are written in the following format + (time in seconds, voltage). The column of numbers will give a voltage signal as in figure 3.2.

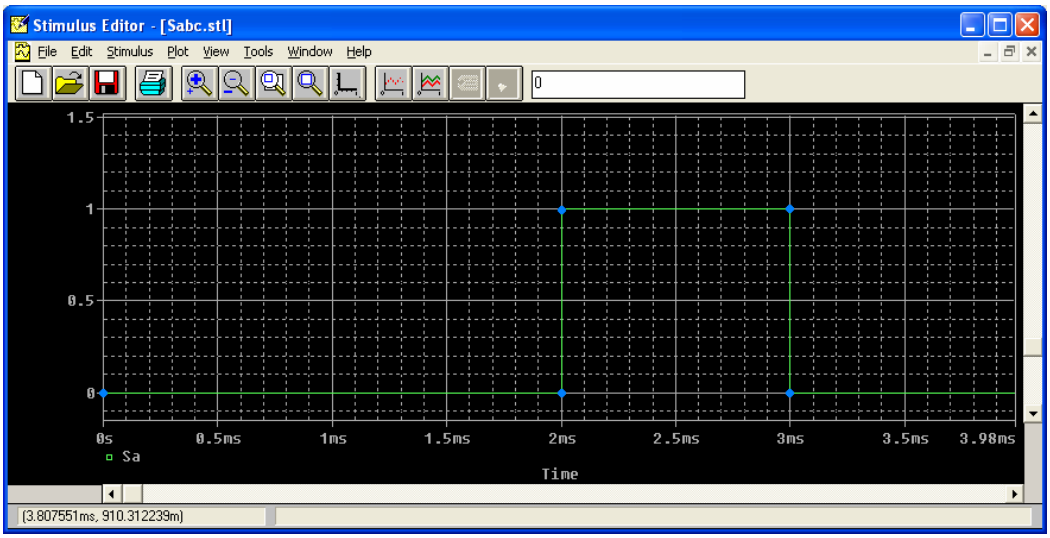


Figure 3.2 A view of the VSTIM signal "Sa" in the Stimulus editor (data points marked out).

The first data point is (0, 0). This data point will give 0 V at time 0. Next line (0.002, 0) will also give 0 V but at time 0.002 i.e. 2 ms. The time in this example is in absolute values, However as much freedom as possible is needed. For example the exact back EMF time is needed to be trimmed to fit the simulation. Because of this the signal is formatted with differential time steps. (+0.000000001, 1.000), indicated with a plus sign in front of the time signal. This means that the time is not an absolute time but the time is relative to the previous time step. In this case the voltage signal is making a voltage step at 2 ms from 0 V to 1 V. The time +0.000000001 will give a rise time of 1 ns. In the same manor will the next line (+0.001, 1.000) give a 1 V plateau with the length of 1 ms. This is the way that both the PWM-signal and the back EMF signal is built. A MATLAB m-file (Simulink2Pspice.m) that autonomously constructs the .stl files from MATLAB variables is written, see appendix A.

4. To be able to simulate a fully functional three-phase inverter with SiC BJT's it is needed to make PSpice models of the SiC BJT from TranSiC named BitSiC. To get a working and a realistic model of the BitSiC the performance and behaviour is extracted from a real transistors. These tests are made in a test rig. The test rig switches the transistor under test (TUT) and captures the voltages on an oscilloscope. When the transistor is switched the following signals are interesting: Vce, Vb voltages and the Ic, Ib current.

5. The captured signals are extracted from the oscilloscope with WaveStar [7], a program from Tektronix. The easiest way of extracting oscilloscope signals is to copy the signals from WaveStar and place them in a text document. The signals in the text document consist of two columns per signal. One time column with

the absolute time relative the trigger point and one column with the voltage values. This text document is then imported to MATLAB and (Tektronix2Pspice.m) converted to .stl files for PSpice, see appendix B for the complete m-file.

6. In this step a model of the test rig is simulated in PSpice. The real signals Vce, Ic, Vbe and Ib from step 5 are imported in to PSpice. In this way it is possible to evaluate the simulated switching versus the real switching. The process of modifying a BJT model to the BitSiC is a mixture of brute force and changing many of the parameters to get a feeling for what all the model parameters do.

7. When the model is changed the test rig is simulated. If the simulated switch does not match the real signals the model is changed a bit more. This process is looped allot of times until the BitSiC model matches the real signals. Cautions are advised when editing the model. The BJT is modelled as a stiff system. This means that there are several processes in work in different time scopes. On one side there is the dynamic properties i.e. switching behaviour and on the other side there is the DC-behaviour that for example sets the Vce (sat) voltage that slowly changes over time. All this is controlled with the same parameters.

8. When the model is satisfying it can be used in a more complex circuit. In this step the construction of the whole three-phase inverter with all the PWM-signals and back EMF is implemented. This also involves optimizing the drive circuit making sure it works as it is supposed to.

9. With a working three-phase inverter the simulation can store all the data in a "Common Simulation Data Format" (CSDF/CSD) file. When a PSpice simulation is calculated the solution data is saved with a time line in the .csd-file. This file tends to get massive and is saved in a sporadic format. This complicates the process of reading the file. MATLAB has no built in function for reading .csd-files so an m-file is written for this purpose. If the simulation is run for one fundamental period (1.6 ms) and saves the data in a .csd-file the file size will be in the reign of 1 Gigabyte (GB). This size of data file will kill even the best laboratory computer so a compromise is made. Only the most important signals such as power in the transistors and in the diodes etc are saved. With only the most important signals stored (approximately 26) the file size of the data file ends up in the region of 20 Megabyte (MB). This simulation file is red by our m-script (PSpice2Matlab.m) and converted to MATLAB variables. See Appendix C for the entire m-script.

10. When the PSpice signals is imported to MATLAB it is time for post processing work. By integrating the instantaneous power signals from i.e. a transistor the average power consumption for that transistor is calculated. With the total power consumption and power losses the efficiency can be calculated. This gives a pointer of how well the inverter works. If a different working point for the motor or a different modulation wants to be investigated the processes is repeated from step 2.

11. When all the simulation is done with a satisfying result the master thesis moves in to answer the core question. Is it possible to make an integrated belt driven alternator starter (BAS) that is cooled with a standard water cooling system with help of SiC technology? This question is answered with the help of calculating power looses and the heat sinks required.

4. Simulation of expected EM-size

The first Simulink studies are done to get an idea of an appropriate size of the Electrical Machine (EM). Knowing the size of the EM sets the power rating that the DC-AC inverter should be able to handle. By simulating a car under different conditions it is possible to estimate how much energy that is needed to drive a certain cycle. With the simulation it is also possible to determine the working point for the EM. The car model that is used in this thesis is developed in Simulink and uses maps and tables as input parameters. Simulink with its graphical interface and easy overview of complex systems rapidly gives a feedback of how the car model behaves.

4.1. Car model

The car model used is a typical medium car like "Opel Astra" with different motor types, diesel or gasoline. The vehicle model [8] was originally developed by Mats Alaküla and Karin Jonasson and later modified by Thomas Bergh and Dan Hagstedt to be more realistic for a belt driven alternator starter (BAS) car model, see figure 4.1.

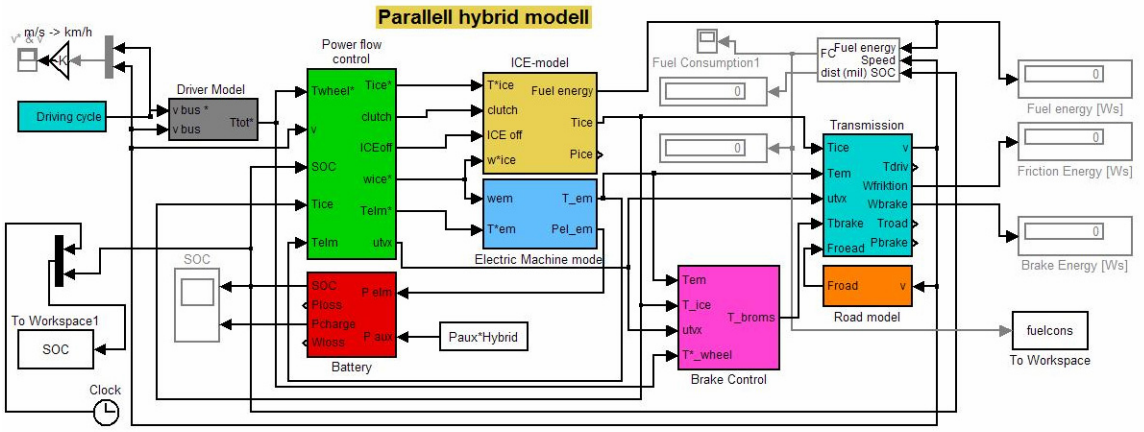


Figure 4.1 Parallel hybrid model in Simulink.

Technical specifications for the car:

```
Pice_max = 79000;    % ICEsize [W]
Mv = 1300;           % Vehicle weight [kg]
rw = 0.30;           % wheel radius [m]
Cd = 0.26;           % air_resistance
Cr = 0.007;          % roll resistance
Av = 2.56;           % Front area [m^2]
rho_air = 1.2;       % Air density [kg/m^3]
vmax=180/3.6;        % 180 km/h maximum speed [m/s]
```

4.1.1. BAS car model

The BAS is meant to replace the generator and starter in a car and is able to drive the car at low speeds and with zero-emission. When the model is simulated as a “regular” car of today the engine does not stop when the car is in stand still, and the EM does not help the car to get into motion. In BAS-mode the engine stops as soon as the car is at stand still. When the car start to move the EM gets both the engine started and helps the car accelerate. This type of hybrid-model is a light version of a parallel hybrid car. The hope is to create a cheap light hybrid car with little changes in the cars construction but with a significant reduction in fuel consumption.

4.2. Driving cycles

The driver is simulated as a PI-regulator with anti windup giving a simple and repetitive model of a driver. In the simulation different driving cycles is used, European standard ECE+EUDC test cycle and United States standard US06.

The ECE+EUDC also known as the MVEG-A cycle is used by the European light duty vehicle industry for emission certification. The cycle contains four ECE cycles without any interruption and is then followed by one EUDC cycle. The four ECE cycles is also known as the UDC (Urban Driving Cycle) and represents city driving cycles e.g. in Paris or Rome. This type of driving represents low speed, low engine load and low exhaust temperature. The EUDC (Extra Urban Driving Cycle) represents more aggressive and high speed driving modes. The ECE+EUDC-cycle is shown in figure 4.2.

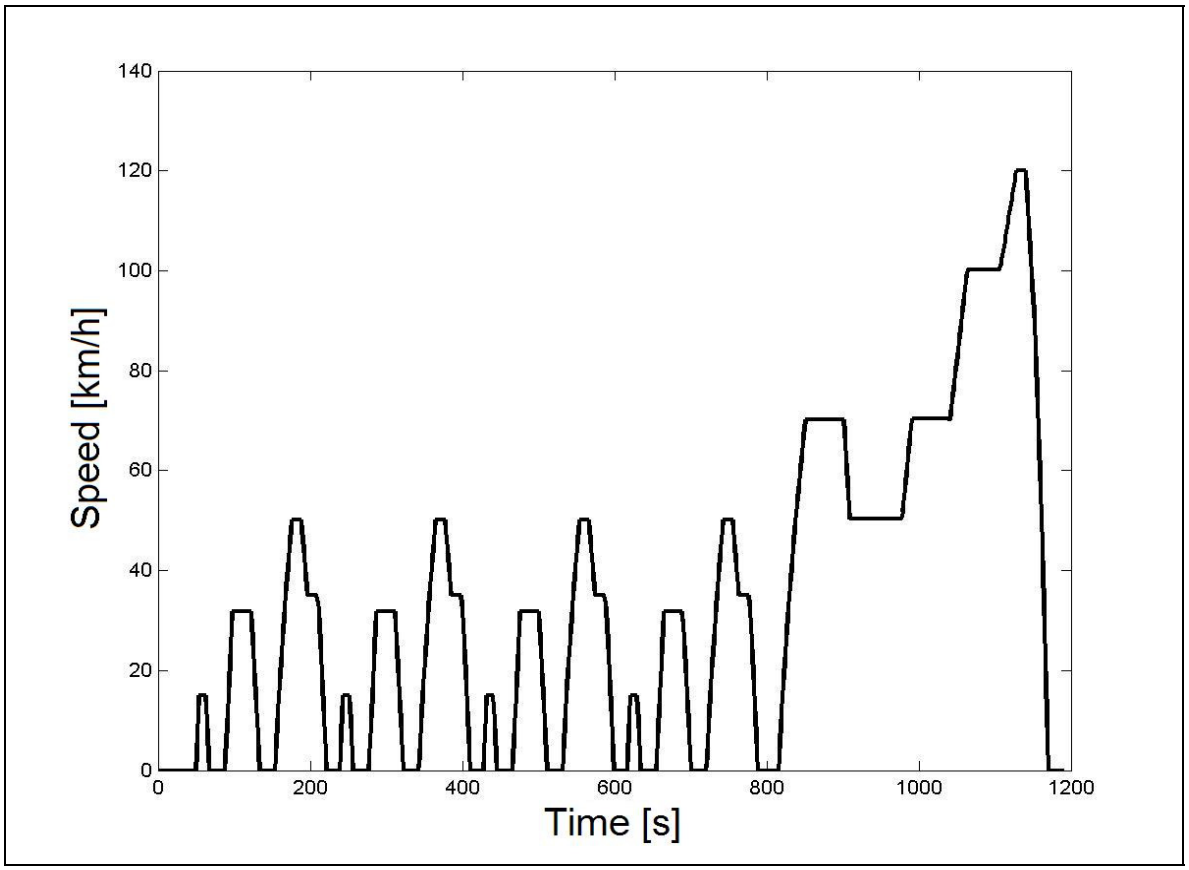


Figure 4.2 The ECE+EUDC velocity cycle model [8].

The US06 driving cycle represents highway driving with aggressive accelerations, high speed driving and rapid speed fluctuations giving transient fuel consumption. The US06 is developed because of the lack of aggressive driving in the United States standard FTP-75 cycle. The light duty vehicle industry uses FTP-75 for emission certification in the U.S. The US06 is shown in figure 4.3.

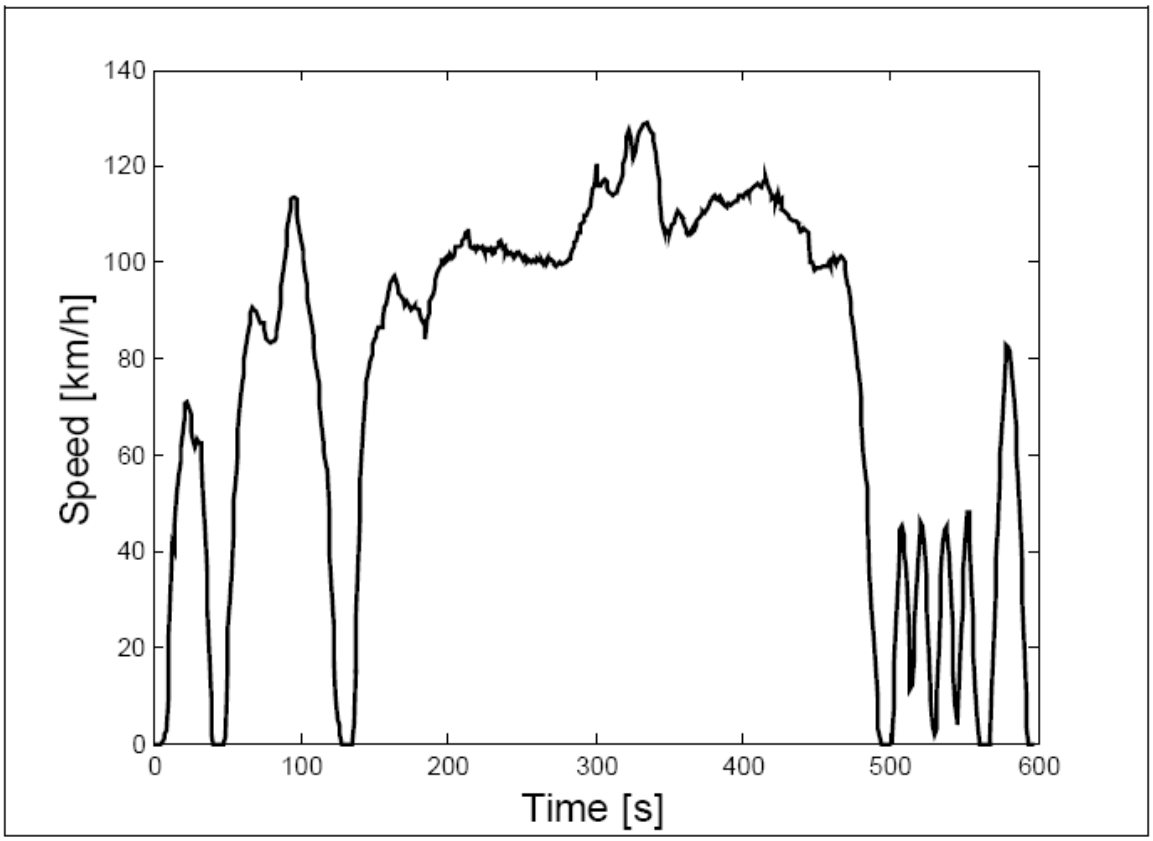


Figure 4.3 The US06 velocity cycle model [8].

4.3. Simulation.

To simulate the appropriate size of the power converter several simulations with different sizes of the EM is done, two types of engines (gasoline and diesel) and two driving cycles (ECE+EUDC and US06). The optimal solution varies with different motor types and different driving cycles.

With the European standard driving cycle the optimal BAS-size is around 7 kW for and gasoline engine and around 7.5 kW for a diesel engine see figure 4.4 and figure 4.5.

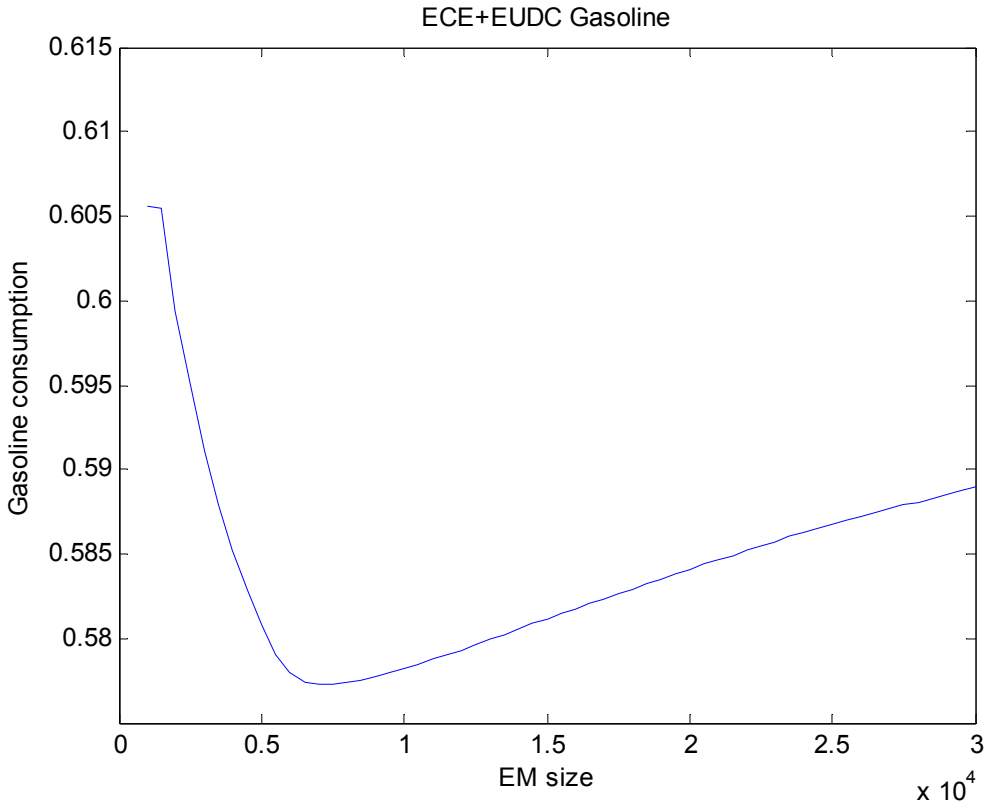


Figure 4.4 Different EM sizes with the ECE+EUDC cycle and a gasoline engine.

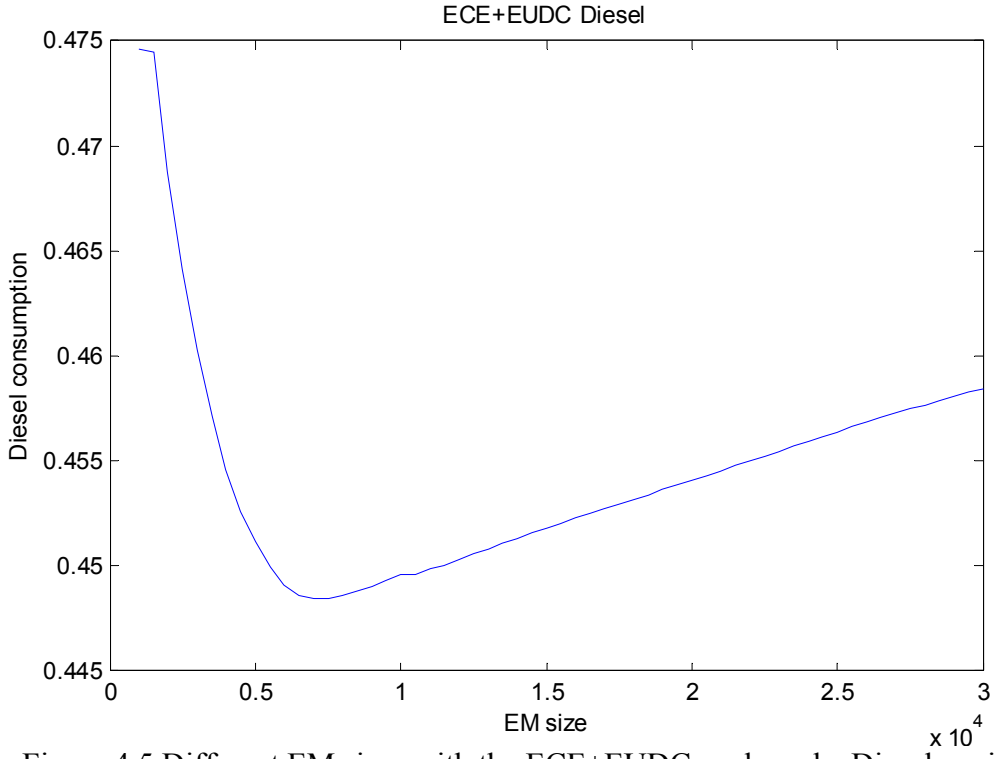


Figure 4.5 Different EM sizes with the ECE+EUDC cycle and a Diesel engine.

With the United States driving cycle the optimal BAS-size is around 8,5 kW for a gasoline engine and for a diesel engine, see figure 4.6 and 4.7.

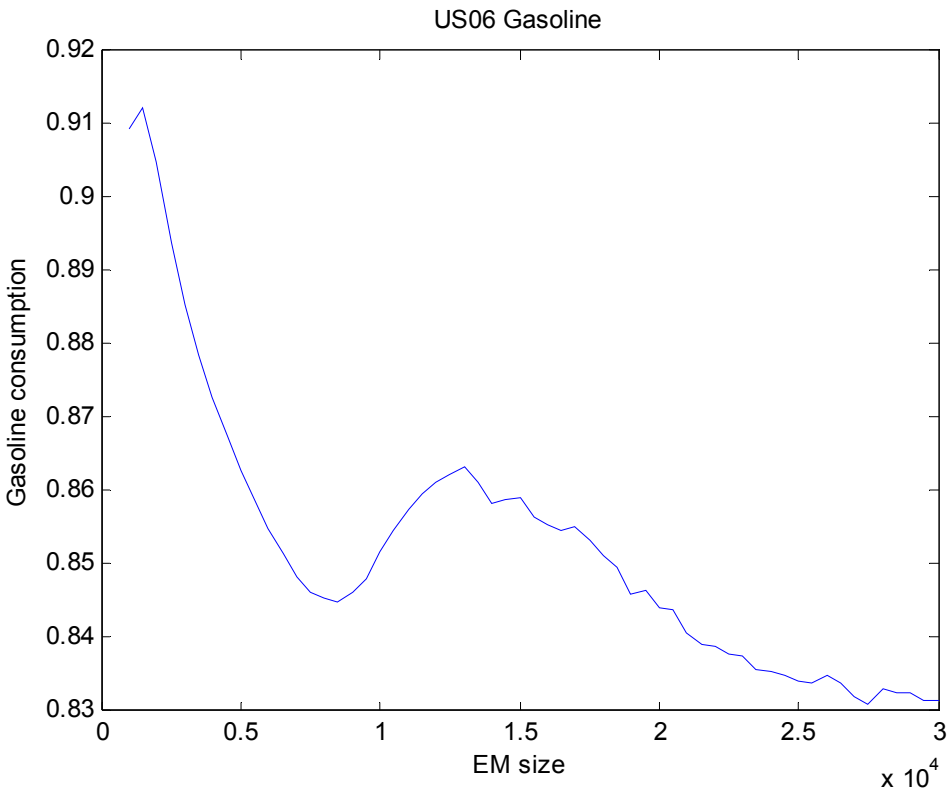


Figure 4.6. Different EM sizes with the US06 cycle and a gasoline engine.

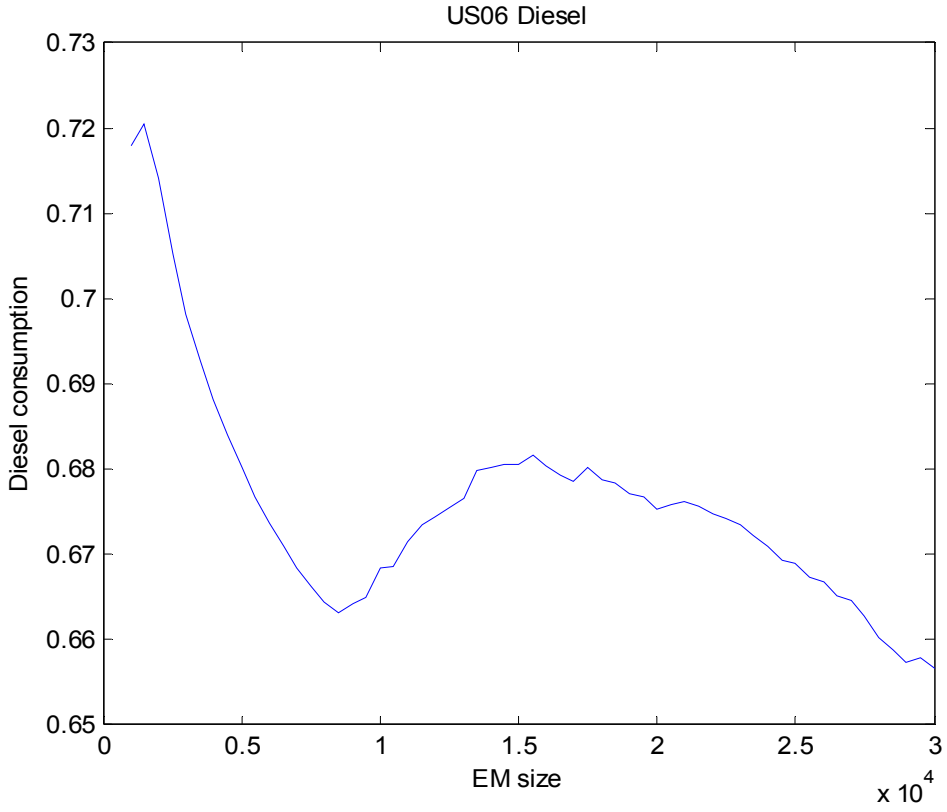


Figure 4.7. Different EM sizes with the US06 cycle and a Diesel engine.

4.4. Conclusion

Looking at the diagrams, the saving in fuel consumption is very small when going from an EM of 7-9 kW to a 5 kW EM. We come to the conclusion that an EM of size 5 kW is a good size for the EM. This will give a fair size of the EM as well as low fuel consumption.

4.5. Choice of working point

To simulate the inverter the EM-speed is needed. Running the first simulation of the hybrid car in Simulink, the car ran at average engine speed of 2000 rpm. This speed is then converted to the speed of the electrical machine so that the back EMF can be extracted. This together with the chosen PWM frequency gives the inverter's working point.

4.6. Improvements

The BAS model is used to get an idea of how the BAS-unit should be designed. There are many uncertain variables in the model that should be checked, for example the battery weight, how the battery should be charged, how the ICE efficiency map reflects a real engine and many more variables.

5. Transistor test rig

The purpose of this master thesis is to evaluate a new type of transistor (BitSiC). The idea is to test how the transistor will work in a motor driver for a belt driven alternator starter (BAS).

To be able to test the BitSiC transistor a high quality laboratory test rig is needed. The reason for this is to get high quality data and to be able to easily change or modify the drive circuits and the evaluated transistor. When brainstorming ideas for the test rig three main goals are decided.

1. The test rig shall control a real transistor at an accurate working environment to see the switching behaviour.
2. A high quality circuit is needed to shield the test rig from noise.
3. Last but not least a good interface for importing the measured data to a computer for post process in MATLAB and other computer programs is needed.

The idea of the test rig is to simulate a working environment for a transistor in a motor drive for BAS i.e. correct currents and load. To do this it is necessary to be able to change the load, voltages and currents in a controlled matter. To simulate the battery of a hybrid car the input to the test rig is made voltage stiff. Instead of a battery a regular power supply and a DC capacitor bank is used. To simulate an electrical machine a stiff current load is used. The test rig uses a big inductor as load, the bigger the better. The ideal is for the test rig to ramp up the current in the BitSiC as seen in figure 5.1.

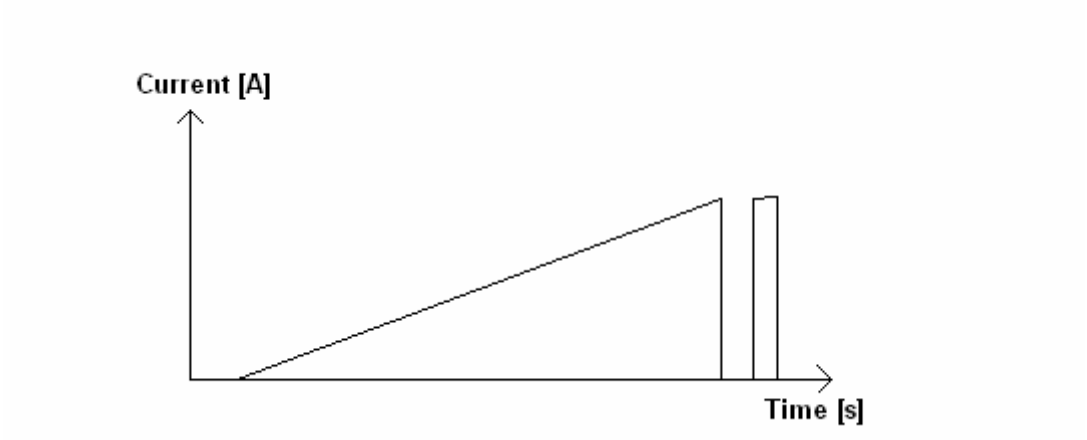


Figure 5.1 Collector current of the test transistor.

The test rig is able to ramp up the current in the inductor to a specific value specified by the working point that is investigated. Then the test rig makes a switch, i.e. turn on the transistor and turn it off again. This course of events is controlled by an AVR microcontroller (μC). The “switch” i.e. the switching after the ramp is the part that gives useful data. Datasheets of power electronic switches is specified for an inductive load. Since the test rig is using a large inductor (1340 mH) as a load the criteria of an inductive load is fulfilled.

The current ramp can be explained if one studies the equations for an inductor

$$u = L\frac{di}{dt} \cong L\frac{\Delta i}{\Delta t} \Leftrightarrow \frac{u * \Delta t}{L} = \Delta i \tag{5.1}$$

With a fixed voltage u, and a fixed inductance L, one time step can only increase the current a certain amount. This is the meaning of a current stiff load and is the reason for the current ramp. When the test rig is executing a switch at a specific current e.g. one ampere the test rig ramps up the current to one ampere and then makes the switch. If the switch is fast enough the current through the load will not have time to change because of the current stiffness. This will make sure that the current stays constant at one ampere through the load during the whole switch.

The test rig system is rather complex, the test rig system is shown in figure 5.2.

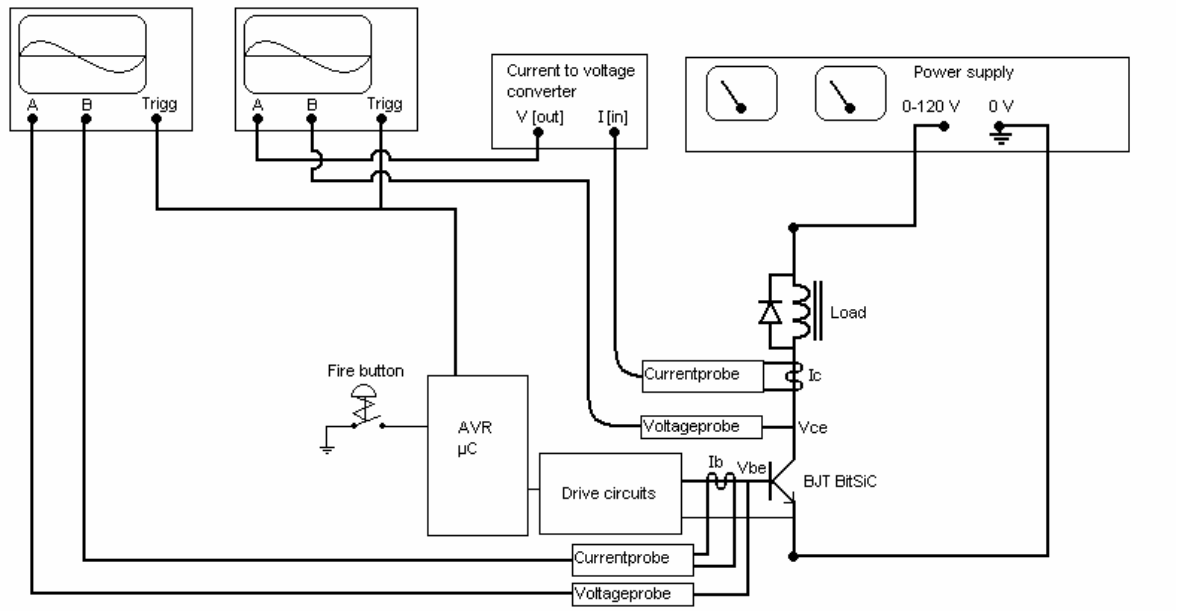


Figure 5.2 A schematic view of the most important instruments for the test rig.

In centre of the test rig system is an AVR microcontroller (µC). This µC controls the switching of the investigated transistor. When the BitSiC is turned on the current starts flowing in the load. With the help of the load and the free-wheeling diode the BitSiC gets a realistic switching. The switching behaviour of the BitSiC is captured by the oscilloscopes. This captured data is then imported to PSpice were the BJT model is built. To build the model the signals Vce, Ic, Vbe and Ib are captured. The Test rig makes it possible to extract behaviour from the transistor and import it to PSpice to create a model.

A detailed description of the design and outcome for the test rig can be red in appendix D1.

6. Choice of simulation program

The Department of Industrial Electrical Engineering and Automation (IEA) uses two programs for simulating electronic designs, the Orcad's PSpice [9] and Linear technology's SwitcherCAD III [10]. There are several other simulation programs, but in the industry the most widely used program is the cadence program suite including PSpice. Many of the programs are using a type of simulator called spice. Spice have a set of commands and certain ways of building different models of components, many of these programs have the same set of commands and models. The program SwitcherCAD III is license free, easy to use and powerful but PSpice have a well developed model editor and is the most used program in the industry. This is why the simulation of the three-phase inverter is done in PSpice.

6.1. Bipolar Junction Transistor (BJT) model

The most important BJT parameters [11] are listed below, a full list can be read in appendix E. These have a short description of what they do and some of them are self explaining just by reading the description. The theory for most of the parameters can be red in appendix F. A first attempt is made to measure the different parameters, this can be seen in appendix G.

Table 6.1 Parameter Description.

IS	transport saturation current
BF	ideal maximum forward beta
NF	forward current emission coefficient
VAF (VA)	forward Early voltage
IKF (IK)	corner for forward-beta high-current roll-off
ISE	base-emitter leakage saturation current
NE	base-emitter leakage emission coefficient
BR	ideal maximum reverse beta
NR	reverse current emission coefficient
VAR (VB)	reverse Early voltage
ISC	base-collector leakage saturation current
pNC	base-collector leakage emission coefficient
RB	zero-bias (maximum) base resistance
IRB	current at which Rb falls halfway to
RBM	minimum base resistance
RE	emitter ohmic resistance
RC	collector ohmic resistance
CJE	base-emitter zero-bias p-n capacitance

VJE (PE)	base-emitter built-in potential
MJE (ME)	base-emitter p-n grading factor
TF	ideal forward transit time
CJC	base-collector zero-bias p-n capacitance
VJC (PC)	base-collector built-in potential
MJC (MC)	base-collector p-n grading factor
XTB	forward and reverse beta temperature coefficient
XTI (PT)	IS temperature effect exponent

6.2. The use of the test rig

The idea is to use an existing model of a BJT and tweak it to get the desired behaviour. With the test rig it is easy to see the BitSiC switching behaviour. To see the difference between the model behaviour and the measured signal a simulation model of the test rig is built in PSpice, see figure 6.1.

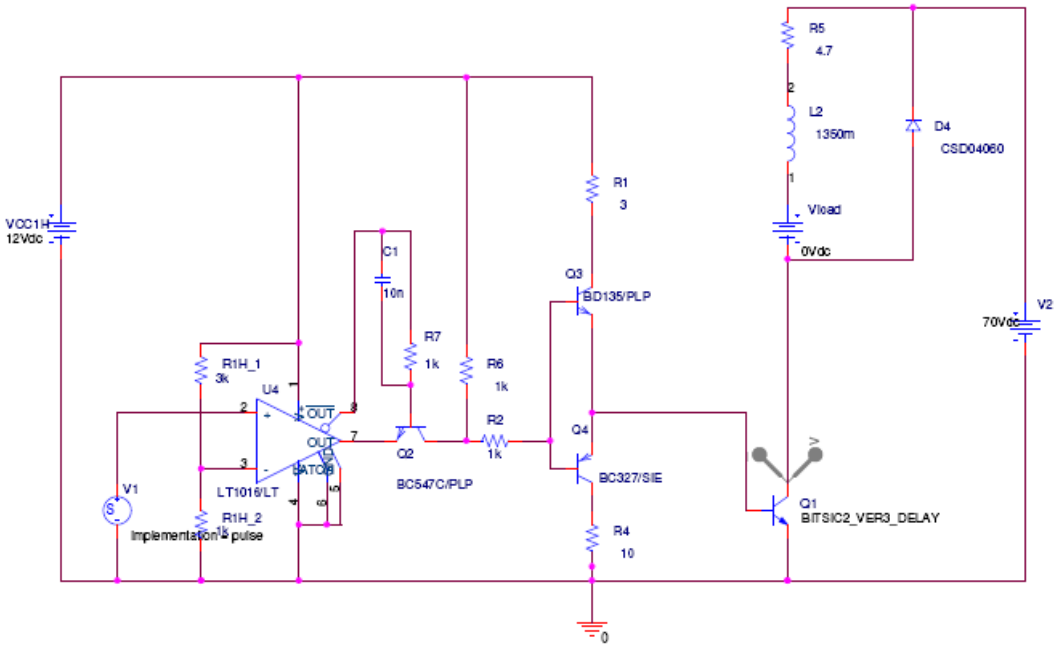


Figure 6.1 Simulated test rig in PSpice.

With both the simulated model behaviour and the measured signal in the same window it is easier to tweak the BJT model. The measured signals are captured with Tektronix WaveStar. WaveStar has different types of data capturing procedures; one can choose between a graphic picture just like the oscilloscope or get a data table. With the data table every signal matrix contains a time vector and a data vector. To get the measured

data into PSpice a working path is created. First one has to copy the matrix in to Microsoft Excel spreadsheet and then save it as a tabular structured text format, see the text structure below.

```
[Tek TDS200 Series].Data.Waveforms.CH 1,,[Tek TDS200
Series].Data.Waveforms.CH 2,,
S,Volts,S,Volts
tidVce,  Vce,  tidIc,  Ic
   -0.000104,  13.3599997,   -0.000104,0.00039999999
  -0.0001036,  13.2799997,  -0.0001036,0.00079999998
  -0.0001032,  13.4399996,  -0.0001032,-0.00039999999
  -0.0001028,  13.1999998,  -0.0001028,           0
```

The signals in the text document will consist of two columns per signal. One time column with the absolute time relative the trigger signal and one column with the voltage values. This text document is imported to MATLAB and via (Tektronix2Pspice.m) converted to a stimulus (.stl) file for PSpice. The .stl file can be used by the PSpice part VSTIM to create signals and be measured into the PSpice A/D program. A dummy resistor is used in series with the VSTIM, then the signals is measured over the resistor, see figure 6.2.

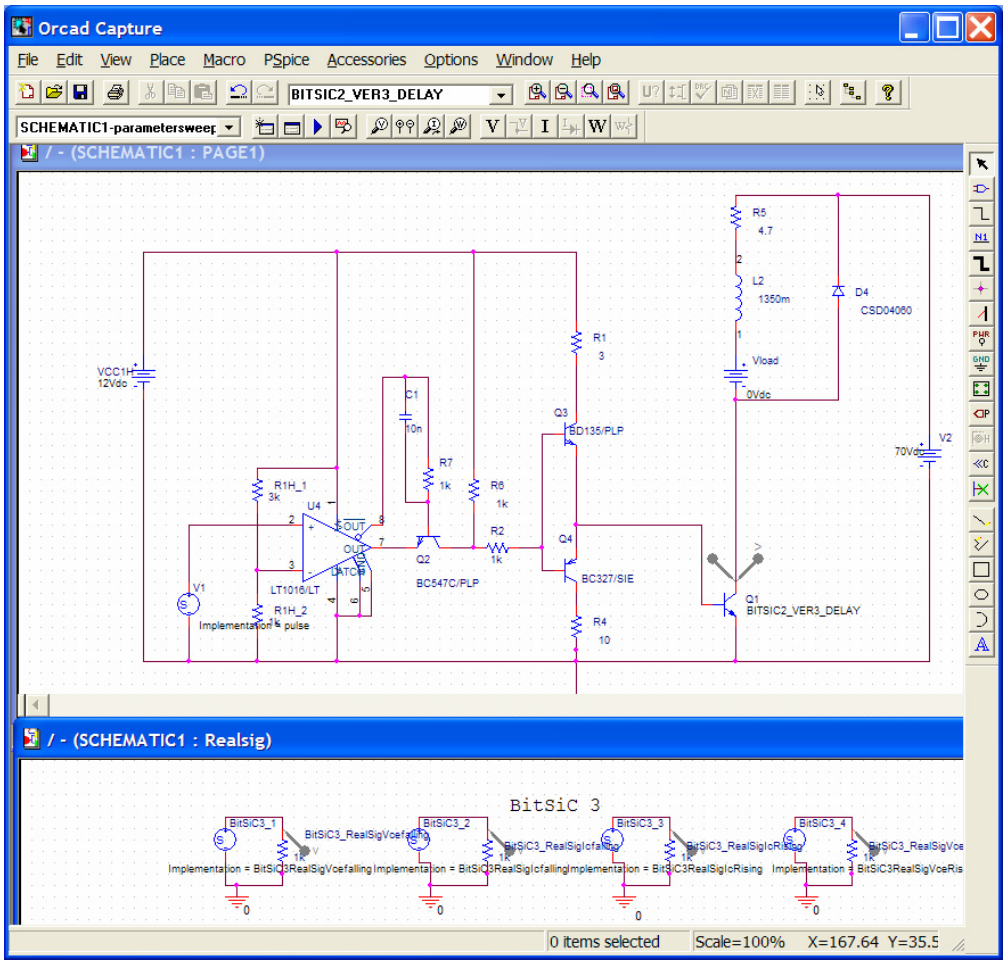


Figure 6.2 Picture showing the Orcad Capture with the test rig in the upper window, and the voltage sources to create the measured signals in PSpice A/D in the lower.

A search for a similar transistor to “tweak” to the wanted behaviour started. This was proven to be very difficult, starting with the BUX11A it did not give the desired result. After a lot of trail and error with other types of transistors, the idea of getting a working model of the SiC transistor seems unreachable.

6.3. The IEEE document.

While searching the internet for information on SiC BJT an IEEE report studying 4H-SiC power BJT is found [12]. The purpose of this paper is to study the effectiveness of the SPICE Gummel-Poon model in predicting the behaviour of a SiC BJT. Instead of tweaking an existing Si BJT model a default transistor containing the parameters found in the IEEE study is created. Surprisingly the model is behaving quite near the behaviour of the real transistor, see figure 6.3 and 6.4. Notice that in the picture of the simulation with original IEEE parameters the voltage is set to 12 V. If it would be set to 70 V the IEEE switching would not even remind of the real switching.

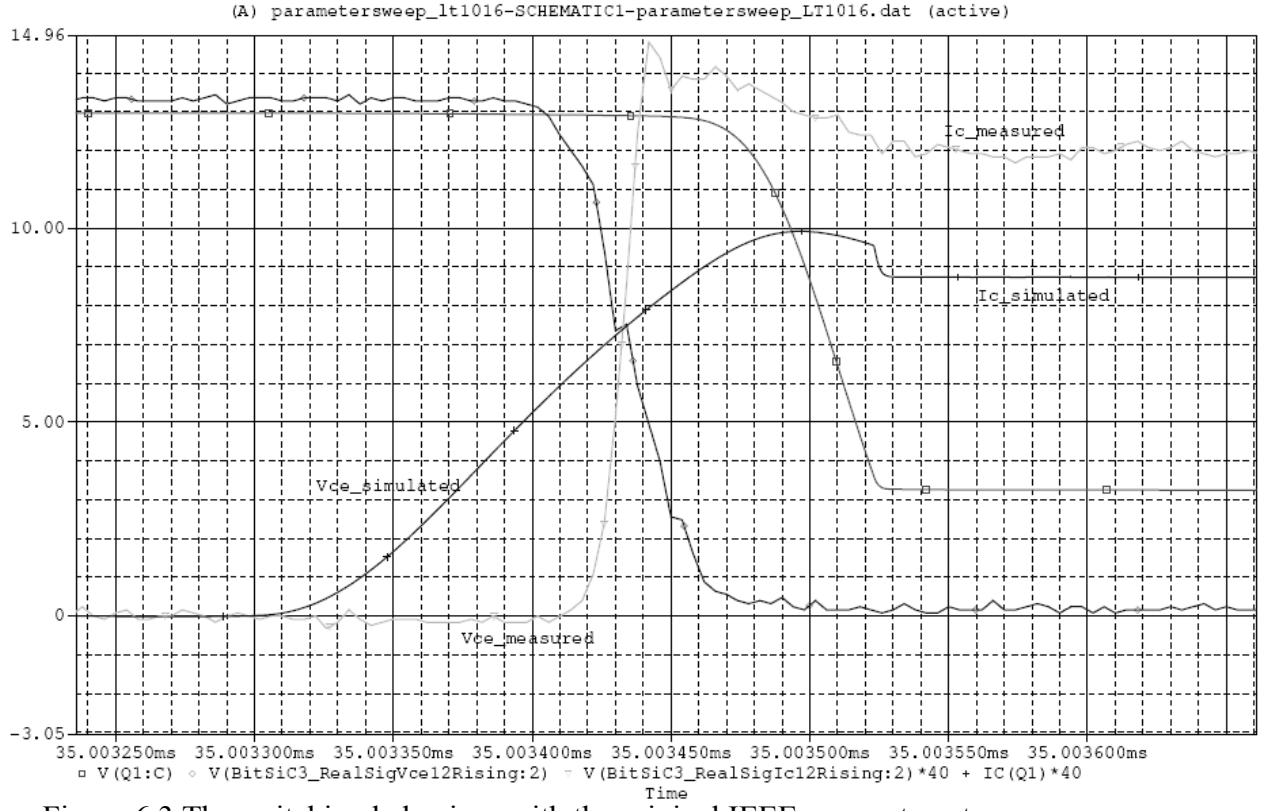


Figure 6.3 The switching behaviour with the original IEEE parameters, turn-on.

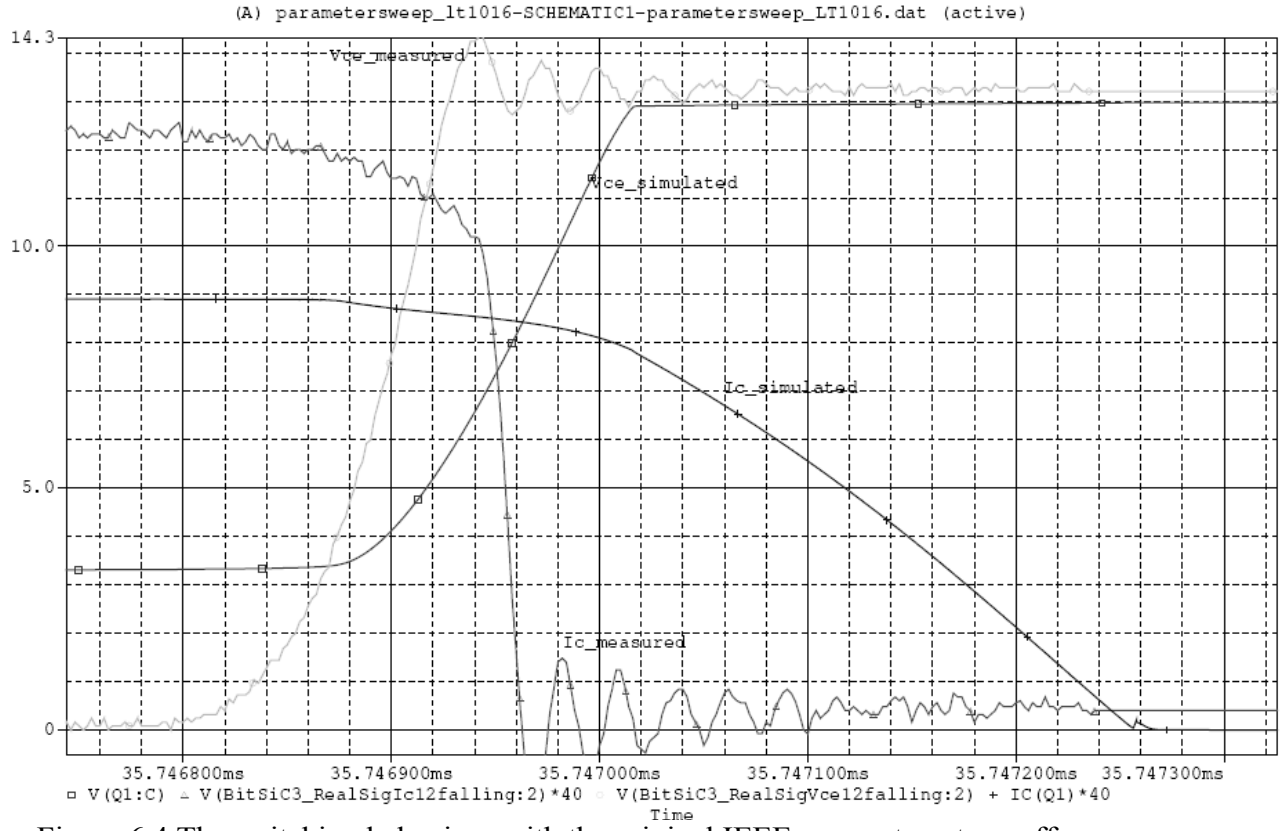


Figure 6.4 The switching behaviour with the original IEEE parameters, turn-off.

The parameters the IEEE study has extracted are in table 6.2, the rest PSpice parameters are set to default:

Table 6.2 Parameters in the IEEE report.

IS	=	4.31e-038 A
BF	=	18
NF	=	1.323
VAF (VA)	=	130 V
IKF (IK)	=	0.728 A
ISE	=	3.37e-019 A
NE	=	3.243
BR	=	0.09
NR	=	0.99
VAR (VB)	=	21 V
ISC (C4)	=	1.845e-016 A
NC	=	4.064
RB	=	23.85 ohm
IRB	=	0.005676 A
RBM	=	22 ohm
RC	=	11 ohm
CJE	=	4.96e-011 F
VJE (PE)	=	2.466 V
MJE (ME)	=	0.3451
TF	=	6e-8 s
CJC	=	1.658e-011

VJC (PC)	=	2.449 V
MJC (MC)	=	0.4058
XTB	=	-0.9356
XTI (PT)	=	73.04

6.4. The final model

With a lot of testing and tweaking the wanted behaviour of the transistor model is found. The result is four BitSiC models, the first is the one most similar to the first BitSiC we had. The other three versions are improvements made from the first BitSiC model to fit the second real BitSiC (BitSiC2) meaning that version three is the most similar to the real second BitSiC tested. Beneath there is a list of the changed parameters relative the IEEE report and the reason for the change, see table 6.3.

Table 6.3 Changed parameters relative to the original IEEE report

	IEEE01355311	**BitSiC2_ver3_delay**	**BitSiC2_ver3 changed relative IEEE original.**
BF	18	30	Increased to get higher gain.
IKF	0.728 A	40.728 A	Increased, moving up DC knee high current.
NR	0.99	1.3	Increased, making the Vce level to be higher.
RB	23.85 ohm	0 ohm	Decreased, to get better frequency response and noise
RC	11 ohm	0.16 ohm	Decreased, less Ic depending.
CJC	1.658e-011 F	1.558e-010 F	Increased, slower falling, faster rising
VJC	2.449 V	8.949 V	Increased to get an increasing slope in the switch moment.
MJC	0.4058	0.2558	A factor to the slope.
TF	6,00E-08	2.95e-010	Decreased, faster internal current fall.
TR	1,00E-08	1,00E-12	Decreased, lowering the time to turn on.
EG	1.11 eV	0.69 eV	Does not have any larger effect (fine adjustments).

The result is very close to the measured data, with the final version of the BitSiC model it is easy to simulate and verify it with 70V, see figure 6.5 and 6.6.

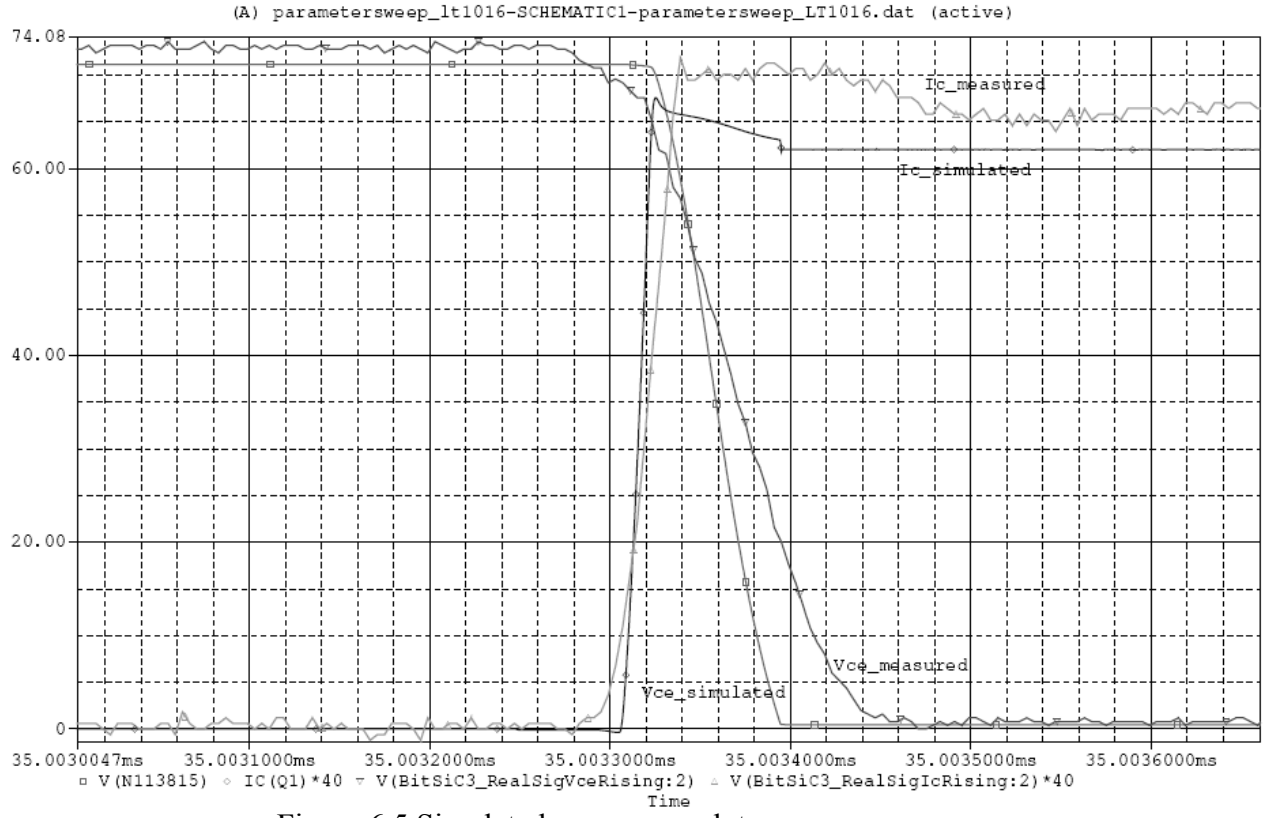


Figure 6.5 Simulated vs. measured, turn-on.

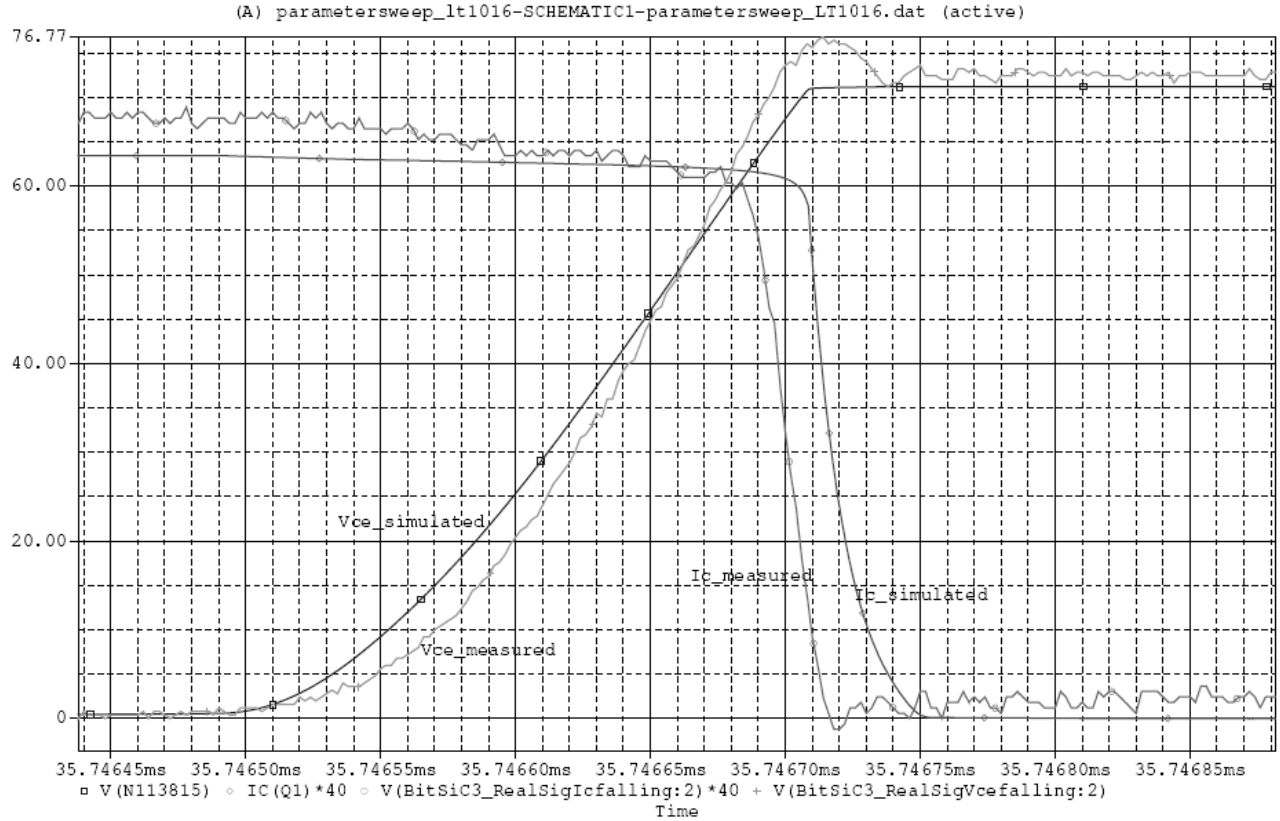


Figure 6.6 Simulated vs. measured, turn-off.

6.5. Uncertainty of measured data

There is a problem with the measured data that is used when trying to get a correct model. When measuring the real transistor the different instruments used has different delays. The current is measured with a current probe connected to an amplifier that is then connected to Tektronix oscilloscope, this setup can have a propagation delay of up to 30 ns with a rise time less than 7 ns. The voltage is measured with a 10x probe directly connected to the oscilloscope. Because of the different delays between the collected data we tried to alter the model and reduce the time difference. But the change did not succeed in the wanted way, this can effect later power loss calculations negatively. The goal was to verify the model with a 112 V simulation but because the lack of time the model could only get verified with 70 V.

6.6. Conclusions

This part of the master thesis, modelling the transistor is perhaps the most difficult. The idea is to tweak an existing transistor model to get the wanted behaviour. The IEEE report supplied a good starting point that made it possible to create a model that is behaving similar to the measured BitSiC transistor.

7. Three-phase simulation

With the BJT model in the foregoing chapter the work is now to build a complete three-phase inverter in PSpice. Most of the design work is already done while creating the test rig, the main design of the driver is copied directly from the test rig. To be able to handle the current levels when simulating with the BitSiC, the transistors are placed in parallel. When normal Si BJT's is placed in parallel there might be stability problems due to the negative temperature coefficient, however SiC has positive temperature coefficient. This makes it easier to place the BitSiC BJTs in parallel, the only problem is that the driver has to handle the higher base current. Our test rig driver can not handle that amount of current. Using power BJT is not that common today, a solution to the problem is found by studying old literature. The problem is that the driver has to be very fast with relative high current. There is not any single BJT fast enough that can handle enough current, the solution is to use a Darlington bridge with the BitSiC, see figure 7.1.

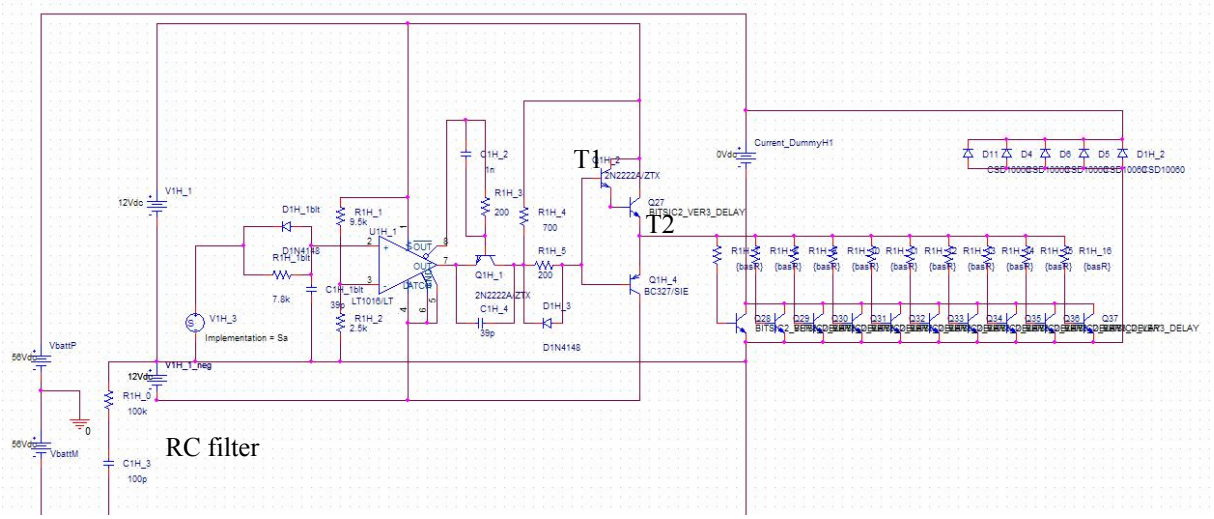


Figure 7.1 One sixth part of the three-phase driver circuit, highside.

The problem with this solution is that the voltage across the Darlington is equal to Vce at transistor one (T1) plus the Vbe of transistor two (T2), in our case the BitSiC. The problem is that the Vbe of the BitSiC is almost 3 V. This will cause a very high power loss in the Darlington with a large current. The next idea is to use a MOSFET instead of a BJT for turning on the base current. But this solution did not work in the simulation program. Doing simulations in PSpice the transient sometimes changes to fast. By reducing the accuracy in the current (ABSTOL) some of the small transients will disappear making it possible to simulate, see figure 7.2.

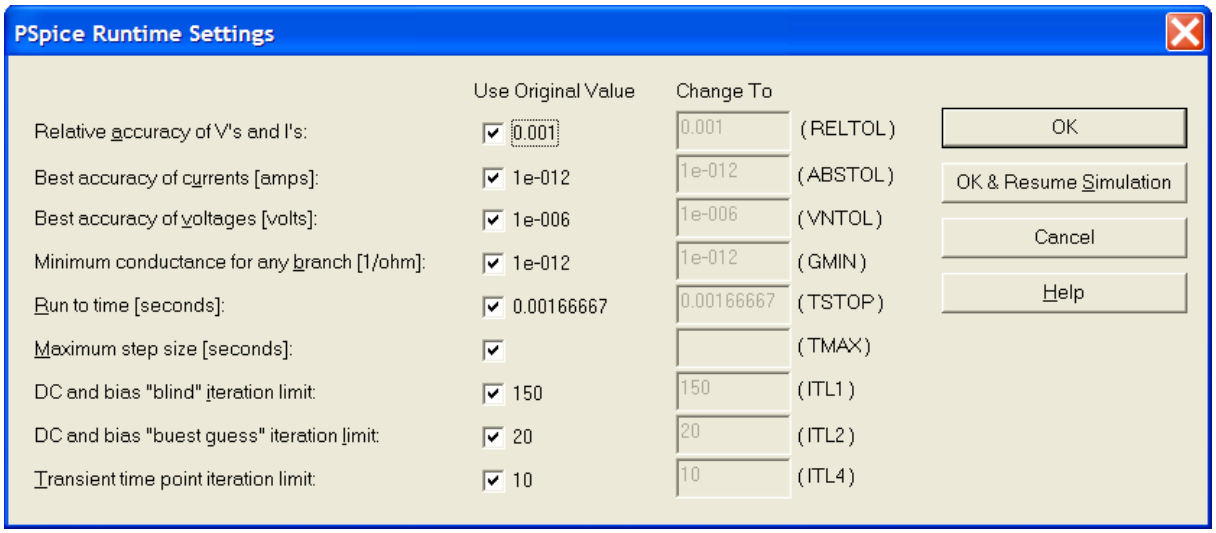


Figure 7.2 PSpice runtime settings.

With the simulation problems the accuracy sometimes has to be as low as 1e-005 A.
To reduce the transient problems a RC filter is made for the drivers, see figure 7.1. Running out of time the decision is to use the Darlington and concentrate on getting the whole simulation of the inverter to work. With the main idea of how the inverter shall be constructed a model of the EM is made. The specifications were already made, and the standard way of simulating an EM was used. An EM consists of three main parts, a resistance, an inductance and a back EMF for every phase, see figure 7.3.

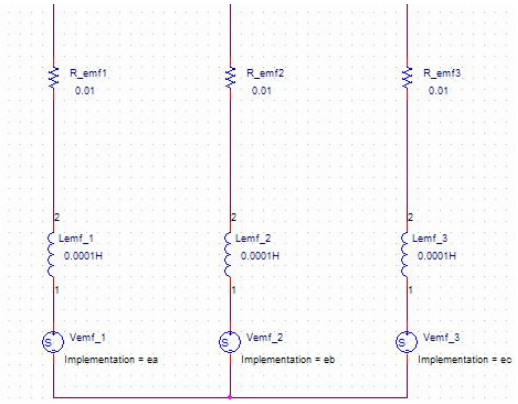


Figure 7.3 Showing a model of an electrical machine.

The resistance and inductance values are determined with the help from Mats Alaküla. The back EMF is extracted from a Simulink model, explained later. The whole inverter can be seen in appendix H.

After a lot of struggle everything started to work properly, the next step is to start the simulation in the selected working point as if it is in stationary running condition. Because the simulation takes a long time with all the detailed switching, the idea is to just simulate one fundamental period for the EM. To get the simulation to start as if in stationary, the initial currents and back EMF had to be extracted and set as initial condition in PSpice.

To get the right switching signals, back EMF and initial currents to the inverter a Simulink program used at IEA at Lund's University is used. The Simulink model simulates a three-phase inverter with different current control methods, see figure 7.4.

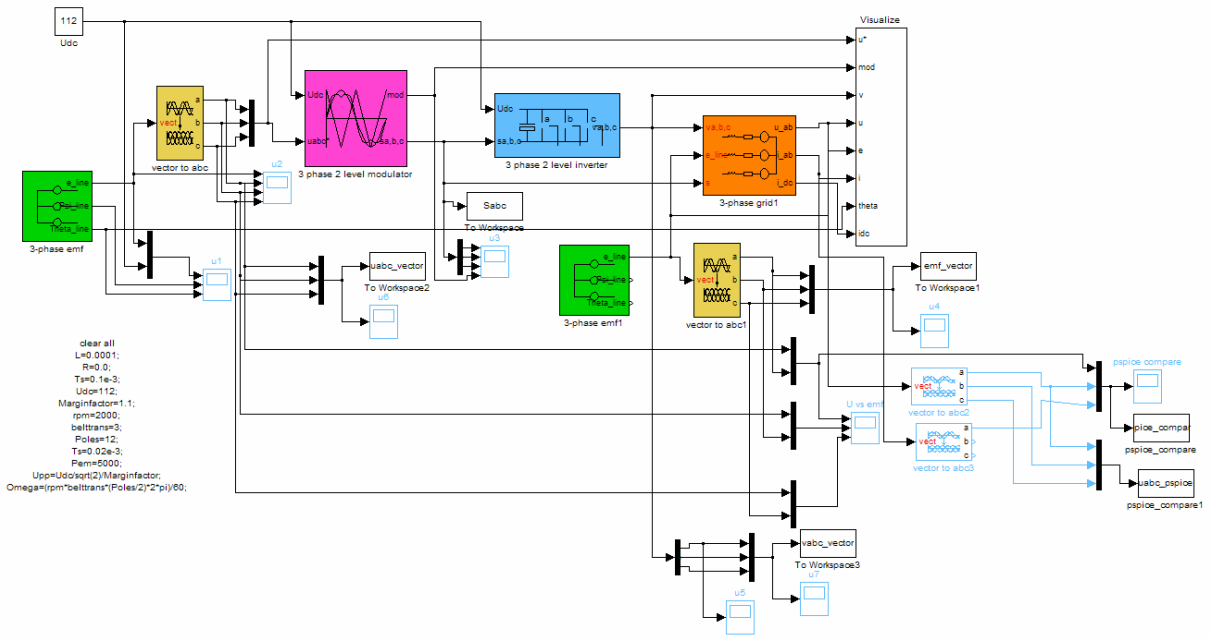


Figure 7.4 Simulink model of the three-phase inverter.

The model is changed to better fit the application. The graphs and data collections are inserted to collect the data that is needed for the PSpice inverter to work in the stationary point. This model can easily be changed to use other types of switching and different EM parameters. With the model it is easy to change the battery voltage and the EM-size. The selected parameters for the three-phase inverter are in table 7.1 below.

Table 7.1 Showing the selected three-phase inverter parameters.

clear all
L=0.0001;
R=0.0;
Udc=112;
Marginfactor=1.1;
rpm=2000;
belttrans=3;
Poles=12;
Ts=0.02e-3;
Pem=5000;
Upp=Udc/sqrt(2)/Marginfactor;
Omega=(rpm*belttrans*(Poles/2)*2*pi)/60;

Where:
L = inductance in the EM windings [H].
R = resistance in the EM windings [Ω].
Udc = battery voltage [V].

Marginfactor = Factor to keep the regulation within its boundaries.
rpm = revolutions per minute of the engine [rpm].
belttrans = Transmission factor between the engine and EM.
poles = number of poles of the EM.
Ts = sampling interval [s].

All of these parameters are easy to change, and the chosen settings are made together with Mats Alaküla [13] to get a realistic simulation. The battery voltage is given by GM powertrain, for a light hybrid car is in the region of 112 V. The three-phase 2 level carrier modulator uses symmetriezed fixed frequency modulation, this can also be changed. The fixed frequency modulation uses 1/(2Ts) as modulation frequency. With a sampling frequency of 50 kHz the PWM frequency is 25 kHz. Using the part VSTIM, mentioned earlier, both the switching pattern and the back EMF from the Simulink three-phase inverter is imported to the PSpice. With all the signals connected to the PSpice inverter model the simulation of the total behaviour started. The result is seen in figure 7.5.

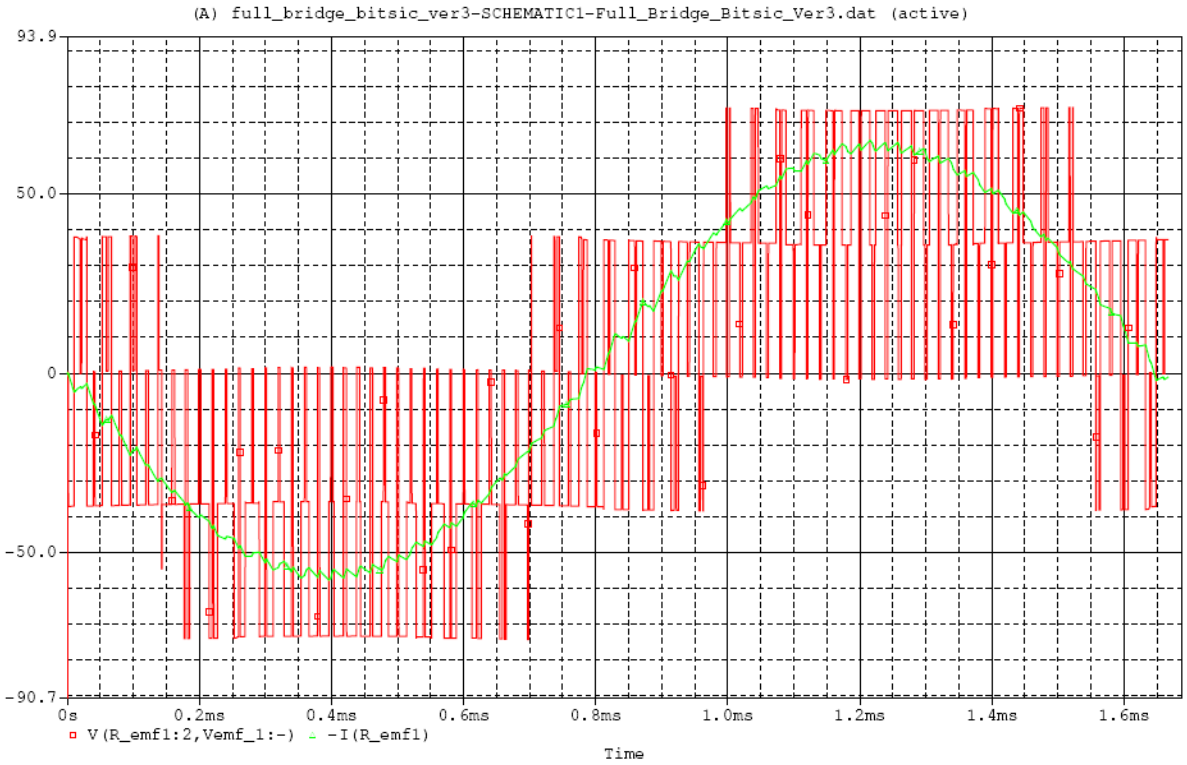


Figure 7.5 Voltage and current in one phase of the 5 kW EM at 25 kHz switching.

Figure 7.5 is showing one phase voltage and current. Both the driver and the BitSiC transistors are very fast and it is not any problem to increase the switching frequency. The extraction of the initial current in the EM inductors are a problem. Although the Simulink gives a value when using ideal switching it is not exactly the same for non-ideal switching. Seen in figure 7.6, when the initial current is not exactly correct there is a difference in the phase current. In the figure this phenomena results in that the top values differ slightly between the phases although the inverter is symmetrical.

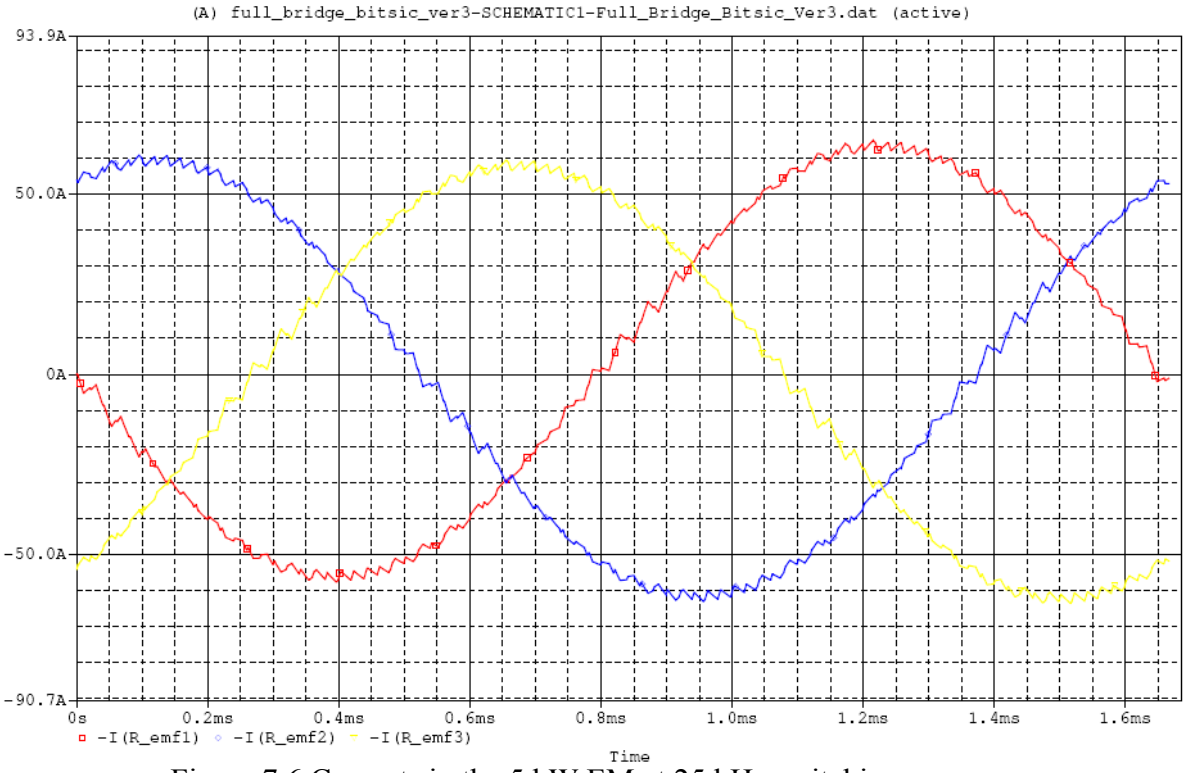


Figure 7.6 Currents in the 5 kW EM at 25 kHz switching.

With the three-phase inverter working properly the next step is to determine the efficiency of the inverter. Doing the simulations at a certain working point it is possible to calculate the efficiency and see how much power that is dissipated. With this data a calculation of how the cooling system shall be designed is possible.

8. Post processing of PSpice simulation

With the simulation of the three-phase inverter an evaluation of the performance is made. The post processing and evaluation of the performance is done in MATLAB. When running PSpice simulation there is an option to save the data points in a Common Simulation Data Format (.csdf/.csd). The csd-format will save all the data points that the simulation calculates with accordingly time stamps. To import the data points from the csd-file in to MATLAB variables an m-script file is written. MATLAB have a powerful set of mathematical scripts and inspections routines such as plotting.

8.1. Interfacing MATLAB with PSpice

Normally when PSpice runs a simulation it saves all data (voltage, current, power, digital and noise) in a binary .dat-file. This file is then red by PSpice A/D where the signals can be viewed. PSpice A/D offers a wide set of tools for viewing signals such as zoom options and capability of performing simpler mathematics on signals. To import the simulated signals in to MATLAB a script is made to read a data format called Common Simulation Data Format (.csdf/.csd). To make PSpice save the data point to a .csdf-file the simulation settings is changed as viewed in figure 8.1.

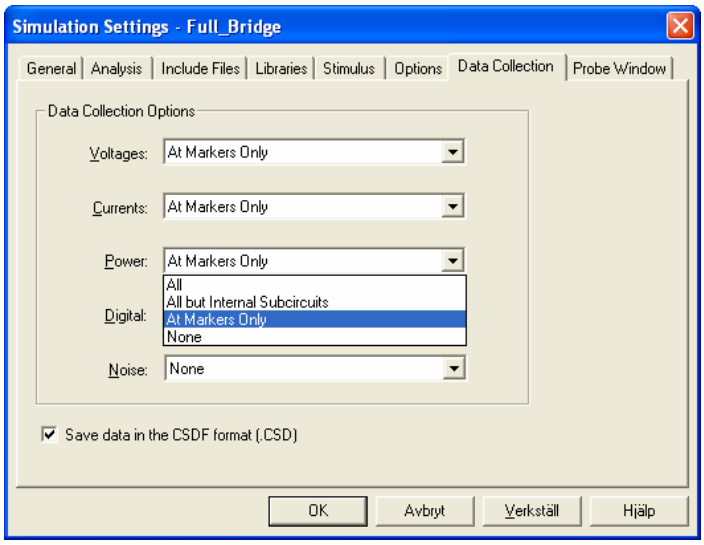


Figure 8.1 Simulation options for PSpice A/D.

With the simulation settings it is decided what types of signals to save and in what format to save them in. These options are of great importance. In a regular simulation of the three-phase inverter there will be a massive size of the data file. If one fundamental period of the load (1.6 ms) is simulated the file size will be around 1 GB. This is far too much data to be handled smooth and easy. To be able to do some post processing work on the data, it is crucial to get rid of all the unnecessary data to get the file size down. This is done by configure the simulation setting (figure 8.1). By only saving the data from voltage-, current- and power-markers the size is reduced. A

marker is a probe that is placed in the schematic to view a specific signal. By limiting the signals, only the most important signals (around 29) will be saved this cut down the file size to around 20 Megabytes.

This file is then read by the MATLAB script "Pspice2Matlab.m". It soon became clear that it was not that easy to develop a MATLAB script to import the csd-file. The csd-file is not consistent in its file structure i.e. on every line there is a random number of variables. This made it hard to develop a MATLAB script that could read the csd-file. One other thing also makes it more complicated. In PSpice it is possible to use a differential voltage probe. This means that signal is the difference in voltage between the two differential probes. This is not represented with one variable per data point in the csd-file but with two data points relative ground. The script needed to recognize when a variable is a differential variable and subtract the two data point to get a true differential signal. To let the script know that a variable is a differential variable the variable names is fitted with an index notation. For example can a name line in the csd-file look like the following line:

name: wVbattP wQ1H_5 iR_emf1 iR_emf2 iR_emf3 d1emf1 d2emf1

The script is able to tell what type of signal it is by checking the first letter.

- w noted that the signal is a power signal
- v noted that the signal is a voltage signal
- i noted that the signal is a current signal
- d1 noted that the signal is the first potential of a differential probe
- d2 noted that the signal is the second differential probe

In the case of the differential probe MATLAB creates the true differential probe by subtracting the data points:

$$d(t)=d1(t)-d2(t) \tag{8.1}$$

In figure 8.2 the result from the differential probe over one of the phases on the load is seen.

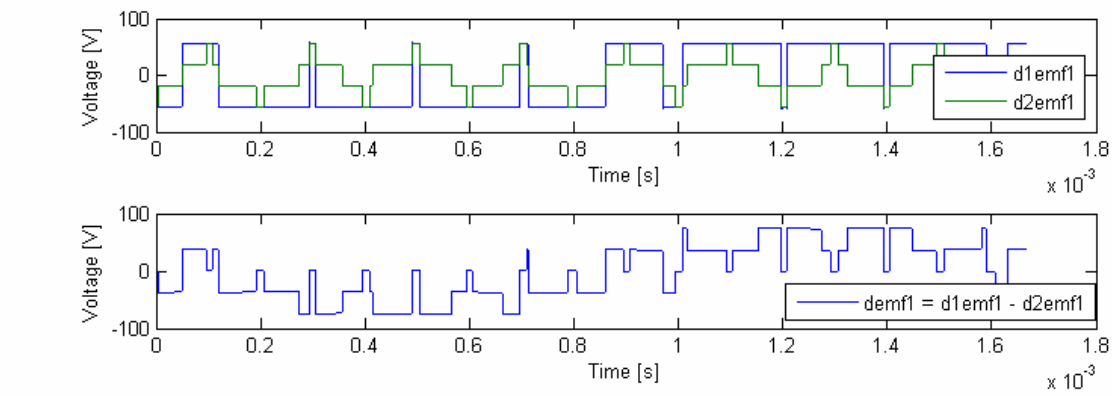


Figure 8.2 The differential signal in the csdf-file and the calculated true differential signal.

One can see that the zero and the switching becomes a sinusoidal like wave when subtracted.

To get a full insight of how the three-phase inverter consumes power a certain signals is exported to MATLAB. First off the power is divided into two categories, the power providers and the power consumers. All the probes selected and imported in to MATLAB can be viewed in figure 8.3.

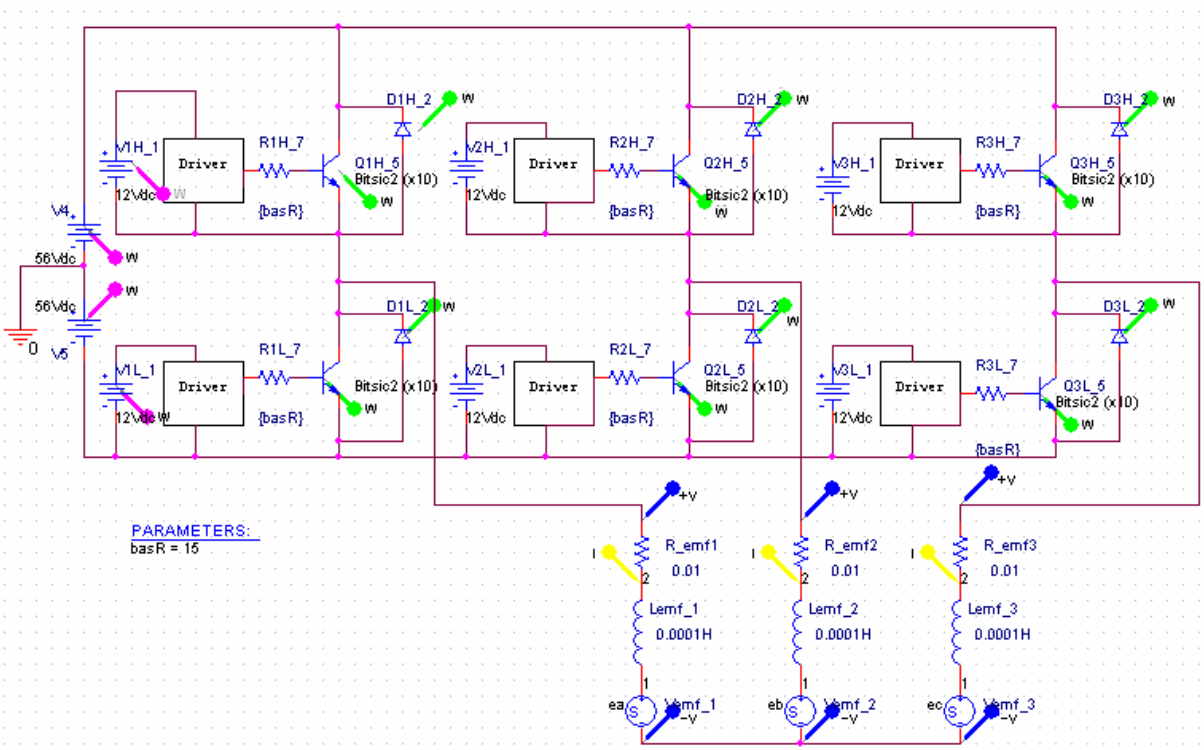


Figure 8.3 A simplified view of the three-phase inverter with marker probes.

I figure 8.3 all the power providing components are pink. For example the main batteries (V4, V5) and the driver batteries (V1H1, V1L_1) give power to the circuits and the load. The driver's battery power is consumed inside the driver circuits, the base resistant and in the BitSiC transistors. The base current gives the power contribution Ib*Vbe in the transistors. This base power contribution is the only part that influences the power in the inverter as a whole. The main battery power is consumed in the transistors, the diodes or the load. This is the reason why there are green power probes at the transistor and the diodes. These probes register all the power consumed in the transistors and the diodes. Power probes are not used in the load because the load does not consist of one component. Instead the voltages over the phases using (blue) differential probes and the current in the phases using current probes (yellow) is saved. When these are multiplied they give the power consumed in the load.

8.2. Efficiency Calculations

When the power of the upper transistor in the first leg is imported to MATLAB it looks like figure 8.4.

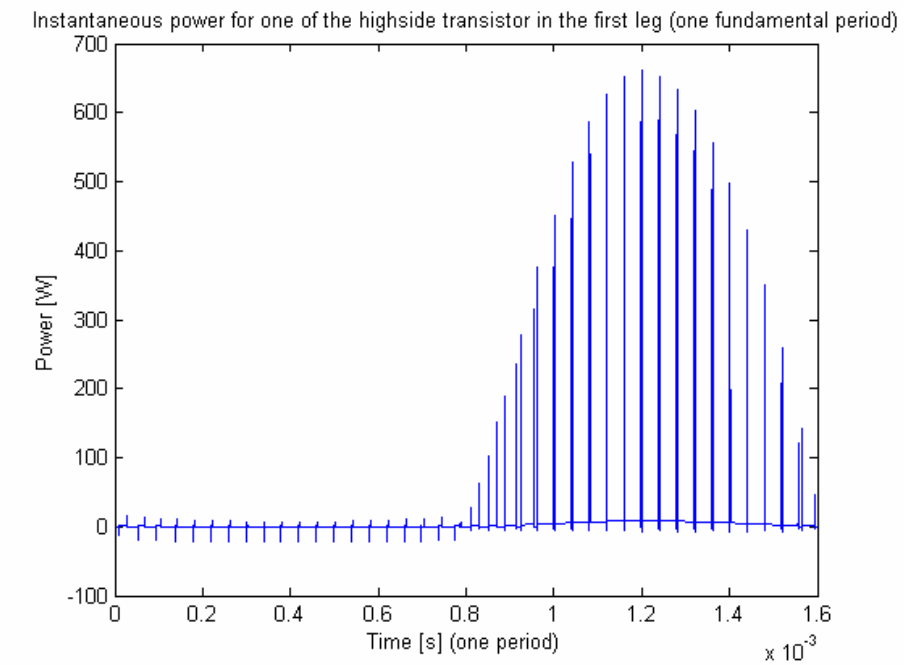


Figure 8.4 Instantaneous power for Q1H_5 (high side transistor for one of the phase).

The signal is represented in MATLAB with two data vectors one containing the power data and one containing the time. The time vector is the same for all the probes. The solver in PSpice A/D calculates the data points for all the markers at different time samples. It is of most importance to remember that the solver uses variable time step. When the signals changes the time step is smaller than when the signal is constant.
Figure 8.4 shows the instantaneous power consumption in the transistor Q1H_5. Notice that during the first half period the transistor almost does not consumes any power. Instead it is the diode in anti parallel with the transistor that leads the current and consumes power. To calculate the power equation 8.2 is used.

$$P = V * I \tag{8.2}$$

However equation 8.2 uses average values of current and voltages. In this case it is necessary to integrate the average power for one second to get the true average power consumption, see equation 8.3.

$$P = \int_{t=0}^{t=1} p(t)dt \tag{8.3}$$

Because the power signal in figure 8.4 is not a continues signal it is needed to approximate the integral with a sum of the signal.

$$P = \int_{t=0}^{t=1} p(t)dt \approx \sum_{k=1}^{k=N} (p(t) * dt) \tag{8.4}$$

In equation 8.4 the signal is approximated with a sum. This approximation can be viewed in figure 8.5. In Figure 8.5 there is a "made up" signal to illustrate the problems with approximations.

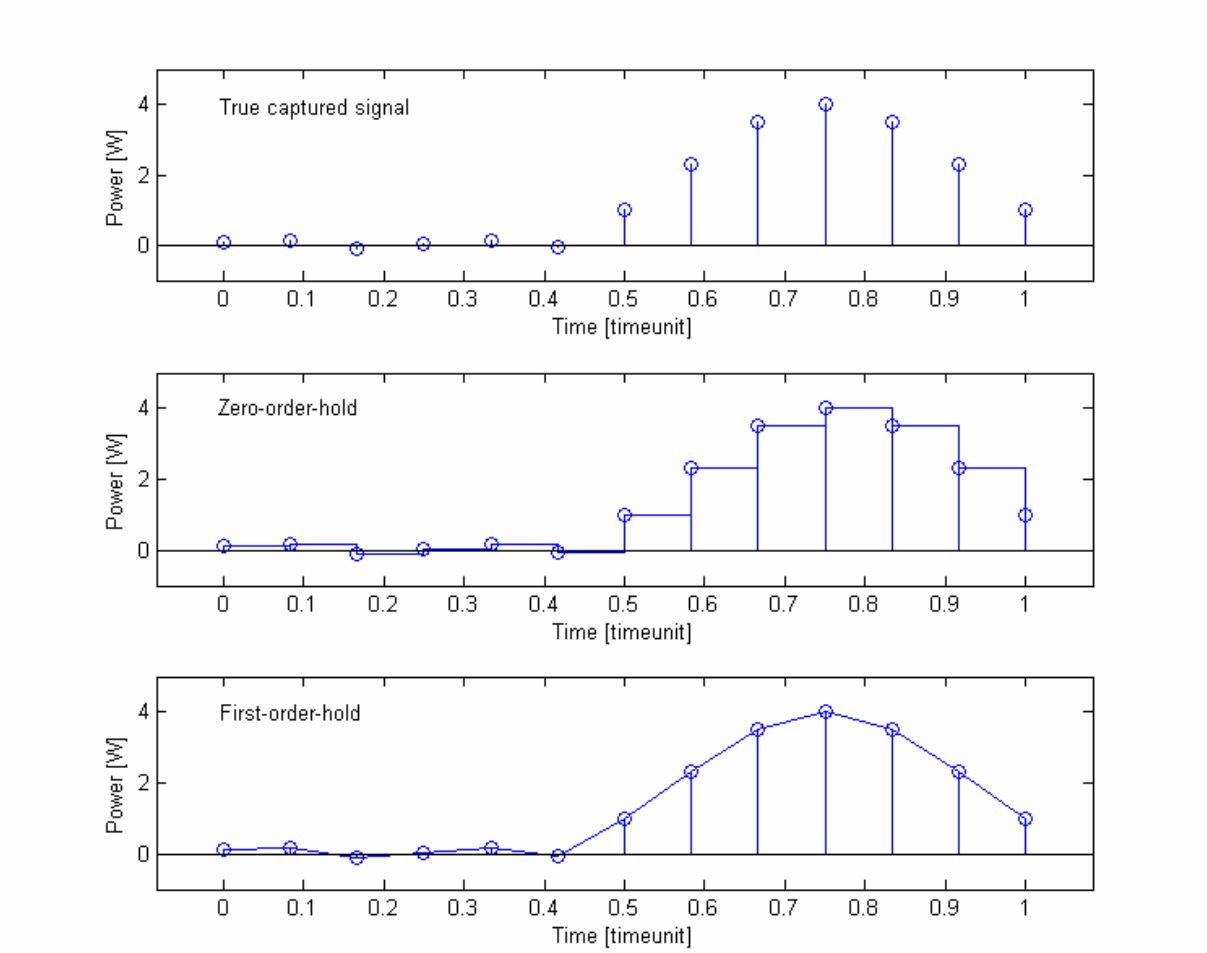


Figure 8.5, Top: True sampled signal. Mid: Zero-order-hold approximation. Bottom: First-order-hold approximation.

In the sampled signal of figure 8.5 there is no power defined because it is a discrete signal. It is of course not true that there is no power in the signal. To calculate the signal an approximation of how the signal looks in between the samples is made. One way of doing this is shown in the middle of figure 8.5. The signal is approximated with a zero-order-hold. With zero-order-hold one sample is used and prolonged with no derivate to the next sample. With a zero-order-hold it is possible to calculate the power in the example according to equation 8.5.

$$P_{ZOH} = \sum_{n=1}^{N} p(n) * \Delta t = 1.4083 \text{ W} \tag{8.5}$$

However a zero-order-hold approximation leaves triangle gaps, it is not likely that the signal look like a zero-order-hold. The higher the derivate or the time step the more error a zero-order-hold approximation will give. This is the reason to use another approximation method to get a more realistic approximation. This is done via a first-order-hold. In a first order hold the first derivate is calculated and added to the samples. This results in triangles being added to the zero-order-hold. To use a first-order-hold to approximate an integral is called a trapezoidal numerical integration [14]. MATLAB has a built in commando for a trapezoidal numerical integration called "trapz".

$$P_{FOH} = \sum_{n=1}^{N} \frac{\Delta t}{2}(p(n) + p(n+1)) = 1.4458 \text{ W} \quad [15] \tag{8.6}$$

One other thing that is different from ordinary analytic maths is that there are not equivalent time steps. Because the time step changes over time this has to be taken in to account. The "trapz" command has an optional input argument, the time vector. With this optional time vector the trapz function can calculate the true trapezoidal integration and taking the variable time in to account. The trapz function calculates the integral as equation 8.7.

$$P_{FOH} = \sum_{n=1}^{N} \frac{t(n+1) - t(n)}{2}(p(n) + p(n+1)) = 1.4458 \text{ W} \tag{8.7}$$

Now all the tools for calculating the power in the three-phase inverter are found.

To calculate the efficiency the equation 8.8 is used.

$$\eta = \frac{Useful\ power\ output}{Total\ Power\ input} \tag{8.8}$$

The "useful power output" is the power that the load draws. The "total power input" is the power that the batteries give. In this case the total "power input" can be divided into two components, the main battery consisting of two 56 V voltage sources (V4, V5) and the batteries for the drive circuits. The efficiency is presented in both with and with out the driver. This has to do with the efficiency of the driver. The driver is not optimized for efficiency and has efficiency around 26%. This is the reason for giving the efficiency numbers with and without driver to give a more independent and fair number of the BitSiC's performance. This concept will give two definitions of the efficiency; see equation 8.9 and equation 8.10.

$$\eta_{With\ driver} = \frac{Load\ Power}{Main\ Battery\ Power + Driver\ Power} \tag{8.9}$$

$$\eta_{Without\ driver} = \frac{Load\ Power}{Main\ Battery\ Power} \tag{8.10}$$

Three different PWM frequencies are simulated 5 kHz, 25 kHz and 100 kHz. These frequencies show how versatile the BitSiC is. PWM frequency of 25 kHz will give a good performance in regard to the load ripple and three-phase inverter performance. 5 kHz is perhaps considered a low-end value for the load, with another load or different ripple ratings the PWM frequencies can be lowered considerably lower than 5 kHz. However it shows how the efficiency increases at lower PWM frequencies. The High-End value 100 kHz shall not be considered a max operating PWM frequency. The choice of 100 kHz is made because it is the PWM frequency of SEMIKRON's [16] high-end drive circuits. The BitSiC showed good performance at this high frequency. The PWM frequency can probably increased to a higher value, but at 100 kHz the other element than the BitSiC showed signs of being in need to get optimized or changed in order to increase the PWM frequency. The result of the simulation can be viewed in table 8.1. For a full investigation of power consumption and efficiency for the three-phase inverter the reader is referred to Appendix I.

Table 8.1, Efficiency with and without diver for different PWM frequencies.

	5 kHz	25 kHz	100 kHz
efficiency with driver	0.94	0.94	0.91
efficiency without driver	0.98	0.96	0.92

An efficiency of 0.91 to 0.98 seams to be a rather good number. It is hard to make comparison because the efficiency is so dependent on the working point and what to include. But it seams like the SiC technology can fulfil the prediction made in [17], that the SiC can improve the efficiency from today's 0.93.5 % to 0.98%.

8.2.1. Efficiency improvements

There are several things that lower the efficiency, the most apparent is the driver. But the rather low main voltage (112 V) will also give higher losses. Thinking in terms of efficiency a BJT might seem as a bad choice in comparisons with a MOSFET or IGBT, but it is not as bad as one might think. Let's study BJT driving in detail (figure 8.6).

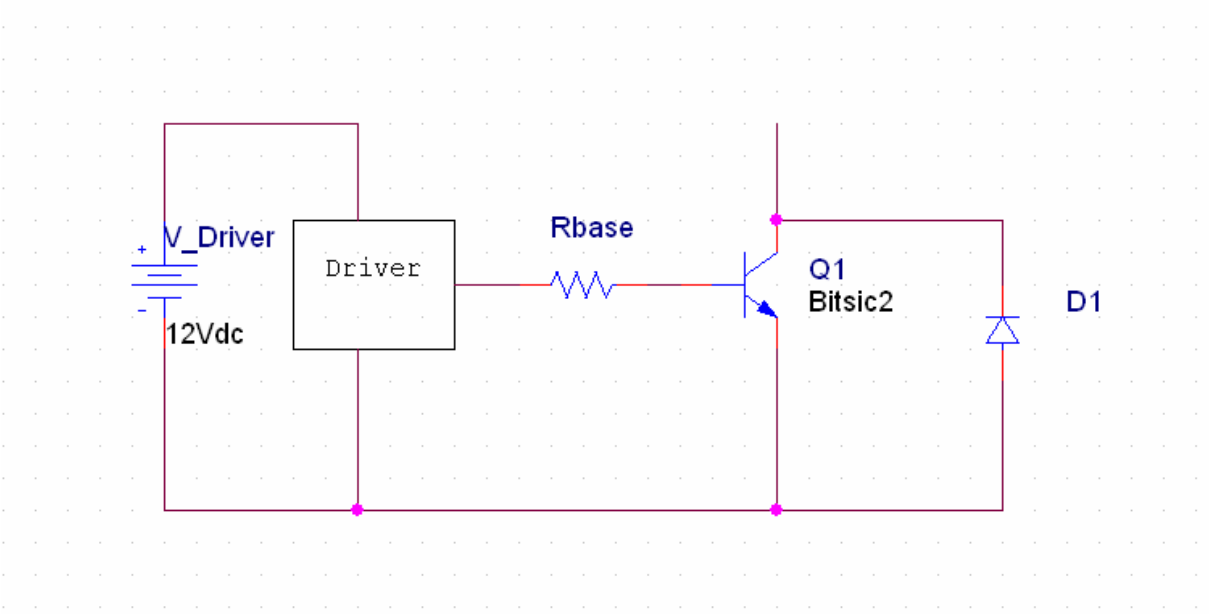


Figure 8.6 Schematic figure for driving the BJT.

The problem with a BJT is that one need continues base current to drive the transistor. The driver selected in this master thesis has a rather low efficiency. This can however certainly be greatly improved. The required power that the BJT needs to be driven can be calculated as in equation 8.11.

$$P_{\text{required}} = Vbe * Ib \tag{8.11}$$

The total power ratings for the six drivers are 128.30 W. With a 25 kHz switching frequency the total power dissipated in the BitSiC transistors (Required) is 33.74 W. The rest of the power is dissipated in the base resistor and in the Darlington transistor that drives the base resistor. There are several things that can be done to improve the efficiency of the driver. One way is to decrease the voltage of the driver (V_driver). If the voltages is decreased it will lower the voltage drop over the base resistor and thereby remove most of the unnecessary power losses in the driver. One other thing to do is to remove the base resistor entirely. The current still needs to be restricted at 3.44 A per position and leg. This could perhaps be solved with working the transistors in the driver box in the active area. But this will also give high losses. One method that improves the driver efficiency is to build the driver as a switched current controller (SCC). This is perhaps the method that will give the best driver efficiency. With a SCC-driver the efficiency would probably be greater than 90%. A SCC will work like the three-phase inverter but as a miniature. The SCC will in principle be constructed like in figure 8.7.

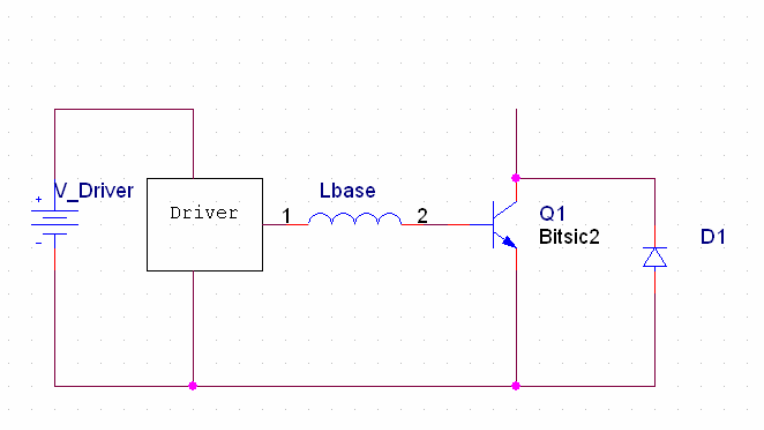


Figure 8.7 Switched current controller driver.

In a SCC the base resistor will be replaced with an inductor. This inductor shall then be driven with a PWM-pulse to regulate the current in the inductor. The efficiency **with driver** will approach the numbers as calculated **without a driver**. Using a SCC will also solve the problem of supplying the driver with a high current 12 V supply. With a SCC it is possible to drive the base inductor with the main DC voltage. This will greatly decrease cost and space for the drivers DC-DC converter.

One other technique to improve the efficiency is to increase the main voltage of the inverter. With the help of a rough and general calculation of the static conduction power loss the benefit of an increased voltage can be seen.

Calculation:
With a setup of a main voltage of 112 V and two transistors in series, each with a voltage drop of approximately 1.4 V and a current of 45 A DC. It is possible to calculate the power loss in the transistors:

$$P_{phase} = 2 * V_{ce(sat)} * I_{load} = 2 * 1.4 * 45 = 126\,W \tag{8.12}$$

If the remaining voltage is assumed to end up across the load, the load voltage will be:

$$V_{Load} = V_{Main} - 2 * V_{ce(sat)} = 112 - 2 * 1.4 = 109.2\,V \tag{8.13}$$

This will give an efficiency of:

$$\eta_{With\ driver} = \frac{V_{Load} * I_{Load}}{V_{Load} * I_{Load} + 2 * V_{ce(sat)} * I_{load}} = \frac{109.2 * 45}{109.2 * 45 + 2 * 1.4 * 45} = 0.975 \tag{8.14}$$

If the voltage now is increased three fold to 336 V it is possible to decrease the current three fold, which will give a DC current of 15 A. This will give a new power loss in the transistors:

$$P_{phase} = 2 * V_{ce(sat)} * I_{load} = 2 * 1.4 * 15 = 42\,W \tag{8.15}$$

In the same way as before, the load voltage will be:

$$V_{Load} = V_{Main} - 2 * V_{ce(sat)} = 336 - 2 * 1.4 = 333.2\,V \tag{8.16}$$

and the efficiency will be:

$$\eta_{With\ driver} = \frac{V_{Load} * I_{Load}}{V_{Load} * I_{Load} + 2 * V_{ce(sat)} * I_{load}} = \frac{333.2 * 15}{333.2 * 15 + 2 * 1.4 * 15} = 0.992 \tag{8.17}$$

This example shows how the efficiency is increased when the voltage increases. The change from 0.975 to 0.992 might seem trivial, but this small change will have a big effect on reduction of the power dissipation in the switching transistors, as a matter of fact if one increase the voltage three times it decreases the static power dissipation in the transistor by three times.

This is why the Toyota's hybrid car Prius is using a voltage step-up converter [18].
The Prius battery has a nominal voltage of 201.6 V. The battery voltage is then transformed with a buck-boost converter to maximum 500 V that then drives the EM. With the bigger bandgap of the SiC BJT it is also more appropriate to work with higher voltages than for regular Si BJT's.

9. Thermal calculations

To get a good reliable design it is necessary to estimate the power loss that occurs in the inverter so that it can be cooled in the right way. To keep the electronics reliable, working and not overheating it has to be kept under a certain temperature. This temperature level is different for different types of components, also there are other problems with higher temperatures like thermal stress. In this master thesis it is especially interesting to look at the power electronics components, these are made of silicon carbide. Silicon carbide can withstand much higher temperatures than ordinary silicon components.

Calculating the losses in the inverter is done in different ways. One way is to calculate the losses with datasheets and get an estimation of the dissipated power. Another way is to simulate the switching losses in a simulation program. A third way is to build a test rig and measure the losses for one switching under different currents and voltages. TranSiC does not have any datasheets for the BitSiC so a test rig is built. With the test rig it is possible to measure the voltages and currents and use it to do a model of the transistor. With the test rig it is easy to run a single switching for different types of drive circuits and for different transistors. After the model is built, the design of the entire three-phase inverter is simulated.

To get an approximation of the power dissipation that has to be cooled away, several three-phase simulations are performed. The power dissipation turns into undesired heat that travels thru different layers of material with different properties in the BJT. Convention, radiation and conduction are the three different types of heat transfer. Convention is the most complicated process to get an accurate model for. For both radiation and conduction there exist accurate models.

The packaging of common power electronics component is not suitable for higher temperatures than up to 175 °C. This is a disadvantage for silicon carbide. Ordinary silicon BJT components losses its current gain abilities at temperatures around 150 °C. Silicon carbide can easily operate in temperatures above 250 °C, see [19]. The problem is to make a reliable packaging of the chip, with today's technology it will be possible do a package that can withstand temperatures up to 200 °C. This will mean that it will not be the SiC junction that limits the maximum temperature. TranSiC has at current time not decided on what type of case the BitSiC component will have. A car manufacture will get the highest performance if they design a special made case like the one in Toyota Prius, see figure 9.1.

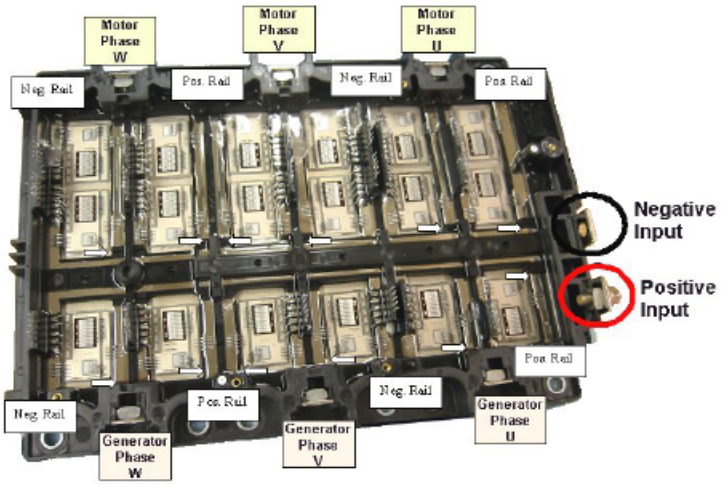


Figure 9.1 Picture of Toyota Prius module [18].

Because the BitSiC do not have a capsule, and thereby do not have any packaging specification. The choice is to take temperature coefficients for an IGBT-module that will fit the specifications. The thermal specifications for SiC and Si are not the same. SiC has around three times better thermal conductivity than Si, but today the Si-chip wafers can be made almost three times thinner than SiC wafers, also the Si transistor chip has large area. So an expected SiC module going out on the market at this time are likely to have the same thermal conductivity as for a Si –module [20].

The SEMIKRON's SKM 75GD123DL IGBT-module has about the same current / voltage ability as expected for the SiC module. The module has 0,32 °C/W as thermal conductivity. Looking at the junction temperature one must see if the temperature changes fast or if it can be seen as stationary, depending on the time constants of the cooling system. The highest thermal capacitance in the module has a time constant of 0.08 s. This is far less than the power fluctuation for a BAS. In this time range it is possible to see the calculations as stationary and thereby exclude thermal impedance.

9.1. Calculating the maximum thermal heatsink resistance

Approaching the problem with the undesired heat it is interesting to see if it is possible to cool the power inverter with the combustion engines cooling water. After studying different methods to calculate the maximum value of the thermal resistance for the heatsink, the method taught at the power electronics course at LTH Lund's University is chosen. This method follows three steps.

1. Determine the needed heatsink temperature for all components with a maximum junction temperature. For each component calculate.
   $$T_{hj} = T_{Jj} - P_{Dj} * (R_{thjc,j} - R_{thch,j}) \quad (9.1)$$
2. The component requiring the lowest heatsink temperature will determine the maximum heatsink temperature. Use this as the heatsink temperature.
   $$T_h = \min(T_{hj}) \quad (9.2)$$
3. This selection of heatsink will make some components to have lower silicon temperature than the maximum allowed. The maximum allowed thermal

heatsink resistance can be calculated as:

$$T_h = T_a + R_{thha} * \sum_{j=1}^{n} P_{dj} \Rightarrow R_{thha} = \frac{T_h - T_a}{\sum_{j=1}^{n} P_{dj}} \quad (9.3)$$

In worst case the temperature in the cooling water will reach up to 130 °C. Maximum temperature for a SiC-junction is 250 °C and for a Si-junction 150 °C. In practical designs a margin of 20-40 °C below the rated maximum junction temperature is desired. For a SiC component it is possible that the capsule will restrict the maximum temperature. Because of this a restriction for the maximum temperature for the SiC component will be 200 °C. With a margin of 25 °C for the SiC module the junction temperature will be 175 °C. Using the lowest margin (20 °C) for a Si module the junction temperature will be at 130 °C. This shows that it is not possible to use Si-module as a reliable power electronic component for the inverter cooled with the combustions engine cooling water. Some of the surrounding electronic components can withstand this high temperature, and those that cannot, should be shielded with a heat shield and other precautions. The most important is that the power handling part of the inverter is close to the EM, minimizing the losses in cables and with the engine bay shielding of the Electromagnetic interference (EMI).

Calculation of the thermal heatsink resistance (25 kHz):

1. With a junction temperature of 175 °C.

$$T_{Transistor} = 175 - 171.6 * (0.32 - 0.14) \approx 161.84\,°C \quad (9.4)$$

$$T_{Diode} = 175 - 47.3 * (0.6 - 0.14) \approx 169.17\,°C \quad (9.5)$$

2. This shows that the transistor needs the most cooling and the heatsink maximum temperature will be:

$$T_h = 161.84\,°C \quad (9.6)$$

3. The maximum allowed thermal heatsink resistance is:

$$R_{thha} = \frac{161.84 - 130}{218.92} \approx 0.145\,°C/W \quad (9.7)$$

This is a rather high value and will not be any problem to meet for a liquid cooled heatsink. Looking at standard cooling plates, Austerlitz liquid cooling has a cooling plate (75-631-17 at ELFA's homepage [21]) that has 0.025 °C/W thermal resistance at 10 l/min. There are many other specialized companies in cooling electronics that has lower thermal resistance, Tykoflex [22] has cooling plates that has thermal resistance below 0.008 °C/W at 5 l/min.

9.2. Calculate the junction temperature

The calculation in the previous section resulted that there is no problem to find a heatsink for a SiC three-phase inverter. In the following calculation the goal is to step backward and start with a specified heatsink and then calculate the junction temperature.

For the calculation the following is given:
Temperature for the ambient water: Ta = 130 °C
Heatsink thermal resistance: Rthha = 0.025 °C/W (ELFA)
Power dissipation for the transistors: Pbjt = 171.61 W
Thermal resistance for junction-case: Rthjc = 0.34 °C/W
Thermal resistance for case-heatink: Rthch = 0.6 °C/W

The maximum temperature of the heatsink can be calculated according to equation 9.8.

$$Th = Rthha * Pbjt + Ta = 0.025 * 171.61 + 130 = 134.29\ °C \tag{9.8}$$

The junction temperature will be:

$$Tj = Th + Pbjt * (Rthjct + Tthch) = 135.47 + 218.92 * (0.34 + 0.6) = 147.45\ °C \tag{9.9}$$

What this means is that with the given data, a standard water heatsink from ELFA the junction temperature for the SiC transistors will be 147 °C. This is far from the maximum junction temperature for the SiC at 250 °C. Even the most standard BJT or IGBT is rated up to 150 °C. Although there is one major problem with standard Si components, at degrees above 100 °C the performance is so degraded that it is almost impossible to find a transistor that work's.

10. Conclusions

This master thesis started with the task to figuring out an appropriate size of the electrical machine in a BAS. After many Simulink simulations with different types of engines and driving cycles, an EM-size of 5 kW is a good size, giving both low fuel consumption as well as keeping the EM-size to a minimum.

Knowing the appropriate size of the EM the design of an actual three-phase inverter started. The main component in the inverter is supposed to be a SiC-based transistor (BitSiC). But the limited access to components forced us to do a simulation of the inverter instead. When doing simulations a model of the transistor is needed.

To see the BitSiC behaviour a test rig is constructed.

The test rig uses an AVR microcontroller to control and regulate the transistor under test. With the help of making a PCB, a high quality circuit is made to keep the noise to a minimum. With the test rig, the capture of the BitSiC switching behaviour is made possible. The switching behaviour is then imported to PSpice were a model of the BitSiC is constructed. The model turned out to be a success. The model shows good resemblance of the real BitSiC both in the transient and static behaviour.

With a good model of the BitSiC a three-phase inverter is built in PSpice. To control the inverter, signals from a three-phase model in Simulink are used. From the Simulink simulation the PWM signals is extracted as the control signals to driver and back EMF-vector for the EM. In this way it is possible to simulate a complete inverter with everything from the control signals to the electrical machine. The PSpice simulation made it possible to view the instantaneous power in the components of the inverter. Without doing any changing or trimming the inverter it is possible to push the PWM frequency up to 100 kHz. However with the efficiency in mind a more appropriate frequency is 25 kHz or 5 kHz.

With Vce(sat) approximately at 1 - 1.4 V the inverter saves a lot of power if compared to a IGBT. This fact together with the fast and efficient switching resulted in a high efficiency for the inverter. With a PWM-frequency of 5 kHz the inverter got an efficiency of 98 % (not including the drivers), at 25 kHz the efficiency dropped a bit to 96 % (without the drivers).

The SiC BJT's superior performance at high temperatures makes the SiC BJT a good choice for use in an inverter for a BAS. The calculation shows that even a standard water heatsink would be enough to cool the inverter. With the SiC-technology it will be possible to cool the inverter with the regular water cooler system of a hybrid car. This will not be possible for a Si-based inverter because of the Si's poor performance at even moderate high temperatures.

11. Future work

Despite how far this master thesis has come there is still a lot of work to do before the world can see a SiC based BAS in a hybrid car. One of the most obvious things to do is to confirm the simulation by building a real three-phase inverter. If the aim is to build a high performance three-phase inverter with the BitSiC, the problems faced in this master thesis has to be considered. To get a high efficiency an increase of the DC voltage should be considered. The driver also needs to be the state of the art. The temperature tolerance of common Si components has stagnated. With the new fields of SiC components which can tolerate a junction temperature of 250 °C and above require a whole new type of capsule. The SiC transistor BitSiC from TranSiC will also keep evolving. This means that the model of the BitSiC will become obsolete as the real BitSiC keeps getting better. However with this master thesis as a foundation it is easier to adjust a model for newer transistors with better performance.

12. References

[1] TranSiC: www.transic.se 2007-04-15

[2] "Process Technology for Silicon Carbide Devices, Docent seminar" by Carl-Mikael Zetterling March 21st, 2000

[3] "Geometrical effects in high current gain 1100 V 4H-SiC BJTs" by M. Domeij, H-S. Lee, Student Member, IEEE, E. Danielsson, C-M. Zetterling Senior Member, IEEE, M. Östling Fellow, IEEE and A. Schöner

[4] Infineon: www.infineon.com 2007-04-15

[5] Cree: www.cree.com 2007-04-15

[6] "Hard Switched Silicon IGBTs Cut Switching Losses in Half with Silicon Carbide Schottky Diodes" by Jim Richmond http://www.cree.com/products/pdf/CPWR-AN03.B.pdf 2007-04-15

[7] WaveStar: www.tek.com/site/ps/0,,60-12123-INTRO_EN,00.html 2007-03-04

[8] Master thesis: Karin Jonasson (2002), Analysing hybrid drive system topologies, TEIE-1031

[9] Orcad: www.orcad.com 2007-04-15

[10] LT SwitcherCAD III: www.linear.com 2007-04-15

[11] "A guide to circuit simulation and analysis, Using PSpice" by Paul W.Tuinenga ISBN: 0-13-834607-0

[12] "IEEE Gummel-Poon model for 1,8kV SiC high voltage BJT" http://ieeexplore.ieee.org/iel5/9371/29755/01355311.pdf 2007-04-15

[13] Alaküla Mats, professor Dept of Industrial Electrical Engineering and Automation, Lund Institute of Technology. 2006

[14] MATLAB: http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml 2007-04-15

[15] "Scientific computing An Introductory Survey 2:nd Edition" by Michael T. Heath ISBN 0-07-239910-4 2002

[16] Semikron: www.semikron.com 2007-04-15

[17] "Technology Shift in power electronics and electric motors for hybrid electric vehicles-A study of silicon carbide and iron powder materials – Master thesis" by Henrik Horrdin Eva Olsson p. 28

[18] “Evaluation of 2004 Toyota Prius Hybrid Electric Drive“ http://www.osti.gov/bridge/servlets/purl/890029-WIfqPO/890029.PDF 2007-04-15

[19] “4 kV, 10 A Bipolar Junction Transistors in 4H-SiC” http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=34863&arnumber=1666128&count=105&index=81 2007-04-15

[20] Domeij Martin CTO, Founder and Co-founder of TranSiC and Ph. D. in Solid-State Electronics at KTH Royal Institute of Technology with 12 years experience of Si and SiC power device research.

[21] Elfa: www.elfa.se 2007-04-15

[22] Tykoflex: http://www.tykoflex.se/ 2007-04-15

[23] Atmel: www.atmel.com 2007-04-15

[24] MCS electronics: http://www.mcselec.com/ 2007-04-15

[25] “Power Electronics, Converters, Applications, and Design” by Ned Mohan, Tore M. Undeland, WillIam P. Robbins, ISBN 0-471-58408-8

[26] “Electrical Engineer's Reference Book” M A LAUGHTON, D F Warne ISBN 0-7506-4637-3

[27] Tektronix: www.tektronix.com 2007-04-15

[28] Cadsoft Eagle: http://www.cadsoft.de 2007-04-15

[29] “Gummel-Poon bipolar model modeldescription parameterextraction” http://eesof.tm.agilent.com/docs/iccap2002/MDLGBOOK/7DEVICE_MODELING/3TRANSISTORS/1GummelPoon/GP_DOCU.pdf 2007-04-15

[30] “Principles of semiconductor devices” by Sima Dimitrijev ISBN 0-19-516113-0

[31] http://www.ecs.csun.edu/~jrb/ver9/OrcadExtras/document/rel92pdf.pdf Orcad onlinemanual r9.2 2007-04-15

13. Appendix A- Simulink2Pspice.m

```
%Converts a Simulink Structor to a Pspice stimulus file
%Modify the m-file to your needs
clc
clear FileIndex SigIndex
StructTimeDiff=zeros(3,1);
StructSigDiff=zeros(3,1);

 FileIndex=[ '* Stimulus Generated by Matlab          \n';
             ';!Stimulus Get                          \n';
             ';! pulse Analog Sa Analog Sb Analog Sc \n';
             ';!Ok                                    \n';
             ';!Plot Axis_Settings                    \n';
             ';!Xrange 0s 36ms                        \n';
             ';!Yrange 0 6                            \n';
             ';!AutoUniverse                          \n';
             ';!XminRes 1ns                           \n';
             ';!YminRes 1n                            \n';
             ';!Ok                                    \n'];

SaIndex=[ '.STIMULUS Sa PWL              \n';
          '+ TIME_SCALE_FACTOR = 1       \n';
          '+ VALUE_SCALE_FACTOR = 1      \n'];

SbIndex=[ '.STIMULUS Sb PWL              \n';
          '+ TIME_SCALE_FACTOR = 1       \n';
          '+ VALUE_SCALE_FACTOR = 1      \n'];

ScIndex=[ '.STIMULUS Sc PWL              \n';
          '+ TIME_SCALE_FACTOR = 1       \n';
          '+ VALUE_SCALE_FACTOR = 1      \n'];

Struct='Sabc'; %Name of structor
StimFile='Sabc_500V.stl';%Name of the created file

StructTime=evalin('base',[Struct '.time']);
StructSig=evalin('base',[Struct '.signals.values']);

%differates the structor
for k=1:3
    j=1;
    for i=1:(length(StructSig)-1)
        if (StructSig(i+1,k)~= StructSig(i,k))
            StructTimeDiff(j,k)=StructTime(i);
            StructSigDiff(j,k)= StructSig(i,k);

            if((StructTime(i+1)-StructTime(i))<1e-9)
                StructTimeDiff(j+1,k)=StructTime(i+1)+1e-9;
            else
                StructTimeDiff(j+1,k)=StructTime(i+1);
            end
            StructSigDiff(j+1,k)= StructSig(i+1,k);
            j=j+2;
        end
    end
end
```

```
StructTimeDiff2=[0 0 0; StructTimeDiff];
for i=1:(length(StructTimeDiff))
    StructTimeDiff3(i,:)=StructTimeDiff2(i+1,:)-StructTimeDiff2(i,:);
end

%Converting output from the signals [-1:1]  to  [0:2]
for i=1:(length(StructSigDiff))
     for k=1:(length(StructSigDiff(1,:)))
        StructSigDiff(i,k)=StructSigDiff(i,k)+1;
     end
end

%Want the signals to from 0to5  multiplinge with 2.5
StructSigDiff=StructSigDiff.*2.5;


StructTimeDiff3 = abs(StructTimeDiff3);

fid = fopen(StimFile,'wt');
fprintf(fid,FileIndex);

fprintf(fid,SaIndex);
fprintf(fid,'+    ( 0, 0 )\n');
fprintf(fid,'+    (+%9.9f, %6.3f )\n',[StructTimeDiff3(:,1)
StructSigDiff(:,1)]');

fprintf(fid,SbIndex);
fprintf(fid,'+    ( 0, 0 )\n');
fprintf(fid,'+    (+%9.9f, %6.3f )\n',[StructTimeDiff3(:,2)
StructSigDiff(:,2)]');

fprintf(fid,ScIndex);
fprintf(fid,'+    ( 0, 0 )\n');
fprintf(fid,'+    (+%9.9f, %6.3f )\n',[StructTimeDiff3(:,3)
StructSigDiff(:,3)]');

fclose(fid);
disp('Conversion Done')
```

14. Appendix B - Tektronix2Pspice.m

```
load('BiTSiC2_Vce12V_Ic_baraswitchförloppet_raw_medpushpull_utan_neg_070102
_medkondensatorbank_trueIc.mat') %Raw switch data from the oscilloscope

VceLow= downsample(Vce,10)';
timeVceLow= downsample(tidVce,10)';
IcLow= downsample(Ic,10)';
timeIcLow= downsample(tidIc,10)';

for i=1:(length(timeVceLow)-1)
time_Vcediff(i)=timeVceLow(i+1)-timeVceLow(i);
time_Icdiff(i)=timeIcLow(i+1)-timeIcLow(i);
end

VceLow_diff=[time_Vcediff; VceLow(1:(length(timeVceLow)-1))];
IcLow_diff=[time_Icdiff; IcLow(1:(length(timeIcLow)-1))];

 fid =
fopen('pSpice_BiTSiC2_Vce12V_Ic_baraswitchförloppet_raw_medpushpull_utan_ne
g_070102_medkondensatorbank_trueIc.txt','w');
 fprintf(fid,'+    (+%11.11f, %6.3f )\n',VceLow_diff);
 fprintf(fid,'\n############ [Ic ] ################\n');
 fprintf(fid,'+    (+%9.11f, %6.3f )\n',IcLow_diff);
 fclose(fid)

 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%5

load('BiTSiC2_Vce12V_Ic_falling_raw_medpushpull_utan_neg_070102_medkondensa
torbank_trueIc.mat')

VceLow= downsample(Vce,10)';
timeVceLow= downsample(tidVce,10)';
IcLow= downsample(Ic,10)';
timeIcLow= downsample(tidIc,10)';

for i=1:(length(timeVceLow)-1)
time_Vcediff(i)=timeVceLow(i+1)-timeVceLow(i);
time_Icdiff(i)=timeIcLow(i+1)-timeIcLow(i);
end

VceLow_diff=[time_Vcediff; VceLow(1:(length(timeVceLow)-1))];
IcLow_diff=[time_Icdiff; IcLow(1:(length(timeIcLow)-1))];

 fid =
fopen('pSpice_BiTSiC2_Vce12V_Ic_falling_raw_medpushpull_utan_neg_070102_med
kondensatorbank_trueIc.txt','w');
 fprintf(fid,'+    (+%11.11f, %6.3f )\n',VceLow_diff);
 fprintf(fid,'\n############ [Ic ] ################\n');
 fprintf(fid,'+    (+%9.11f, %6.3f )\n',IcLow_diff);
 fclose(fid)

 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%5

load('BiTSiC2_Vce12V_Ic_helaswitchförloppet_raw_medpushpull_utan_neg_070102
_medkondensatorbank_trueIc.mat')

VceLow= downsample(Vce,10)';
```

```
timeVceLow= downsample(tidVce,10)';
IcLow= downsample(Ic,10)';
timeIcLow= downsample(tidIc,10)';

for i=1:(length(timeVceLow)-1)
time_Vcediff(i)=timeVceLow(i+1)-timeVceLow(i);
time_Icdiff(i)=timeIcLow(i+1)-timeIcLow(i);
end

VceLow_diff=[time_Vcediff; VceLow(1:(length(timeVceLow)-1))];
IcLow_diff=[time_Icdiff; IcLow(1:(length(timeIcLow)-1))];

 fid =
fopen('pSpice_BiTSiC2_Vce12V_Ic_helaswitchförloppet_raw_medpushpull_utan_ne
g_070102_medkondensatorbank_trueIc.txt','w');
 fprintf(fid,'+    (+%11.11f, %6.3f )\n',VceLow_diff);
 fprintf(fid,'\n############ [Ic ] ################\n');
 fprintf(fid,'+    (+%9.11f, %6.3f )\n',IcLow_diff);
 fclose(fid)

 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%5

load('BiTSiC2_Vce12V_Ic_rising_raw_medpushpull_utan_neg_070102_medkondensat
orbank_trueIc.mat')

VceLow= downsample(Vce,10)';
timeVceLow= downsample(tidVce,10)';
IcLow= downsample(Ic,10)';
timeIcLow= downsample(tidIc,10)';

for i=1:(length(timeVceLow)-1)
time_Vcediff(i)=timeVceLow(i+1)-timeVceLow(i);
time_Icdiff(i)=timeIcLow(i+1)-timeIcLow(i);
end

VceLow_diff=[time_Vcediff; VceLow(1:(length(timeVceLow)-1))];
IcLow_diff=[time_Icdiff; IcLow(1:(length(timeIcLow)-1))];

 fid =
fopen('pSpice_BiTSiC2_Vce12V_Ic_rising_raw_medpushpull_utan_neg_070102_medk
ondensatorbank_trueIc.txt','w');
 fprintf(fid,'+    (+%11.11f, %6.3f )\n',VceLow_diff);
 fprintf(fid,'\n############ [Ic ] ################\n');
 fprintf(fid,'+    (+%9.11f, %6.3f )\n',IcLow_diff);
 fclose(fid)
```

15. Appendix C – Pspice2Matlab.m

```
%Converts PSpice csd-files to matlabvaiables
% Csd-file need to start with
%#H
%[number of varibles] (Not Nessesary)
%[linenumber of the data start] (Not Nessesary)
%Name: name1 name2 ... [name of variables, d1=diffient probe+,d2=diffient
probe-]
%...data

clear
clc
tic

%Edit to the right filename
y=textread('1_070119_noblt_10bitsic_1diode_full_bridge_070105-SCHEMATIC1-
Full_Bridge.csd','%s','whitespace','','delimiter','\n');

%**Remove line if the csd-file is formated according heading>
name=char(strread(cell2mat({[  'wVbattP ' 'wVbattM ' ' wV1H_1 ' ' wV1L_1  '
'wQ1H_5 ' ' wQ1L_5 ' ' wD1H_2 ' ' wD1L_2 ' ' wQ2H_5 ' ' wQ2L_5 ' ' wD2H_2 '
' wD2L_2 ' ' wQ3H_5  ' 'wQ3L_5 ' ' wD3H_2 ' ' wD3L_2 ' ' iR_emf1 ' '
iR_emf2 ' ' iR_emf3 ' ' d1emf1  ' 'd2emf1 ' ' d1emf2  ' 'd2emf2 ' ' d1emf3
' ' d2emf3']}),'%s'))
%**
t(1)=toc;
tic

start=0;%Startline
e=1;
while start == 0
    tempstr=cell2mat(y(e));
    if strmatch('Name:', tempstr)
        name=char(strread(cell2mat({tempstr}),'%s'));
        name=name(2:size(name,1),:)
    elseif strmatch('name:', tempstr)
        name=char(strread(cell2mat({tempstr}),'%s'));
        name=name(2:size(name,1),:)
    end

    if strmatch('#C', tempstr)
        start=e;
        NbrOfVar=str2num(tempstr((max(strmatch(' ',
tempstr'))+1):(length(tempstr))));
    end
    e=e+1;
end
%^^Reads NbrOfVar and Start line and Names


for i=1:NbrOfVar % Removes ( ) from names
    name(i,:)=strrep(name(i,:),'(','_');
    name(i,:)=strrep(name(i,:),')',' ');
end


time=zeros(((max(size(y))-start)/(1+ceil(NbrOfVar/10))),1);%Creating void
variables
```

```
value=zeros(((max(size(y))-
start)/(1+ceil(NbrOfVar/10))),NbrOfVar);%Creating void variables

k=1;
x=0;
for n=start:(1+ceil(NbrOfVar/10)):(max(size(y)-2))%Step=1 line +number of
10's of variable
    ytemp1=y{n,:};
    ytemp2=y{(n+1),:} ;
    ytemp3=0; %temp variable for multiple lines of variable

    ytemp1=strrep(ytemp1,'#C ','');%Formating timeline for reading
    ytemp1=strrep(ytemp1,[' ' num2str(NbrOfVar)],'');%Formating dataline
for reading
    if NbrOfVar<=10 % For 9 or less data variable

            for n=1:NbrOfVar
                ytemp2=strrep(ytemp2,[':' lower(dec2hex(n))],'');%Formating
dataline for reading
            end

    else % For multipel data lines
        for i=1:(ceil(NbrOfVar/10)) % Number of datalines
            if rem(10,NbrOfVar-(i-1)*10)==10 % Finds the last dataline
                if hex2dec(ytemp2((max(strmatch(':',
ytemp2'))+1):(length(ytemp2)-1)) )==[] % Finds out how long the dataline
is?
                    jEnd=ytemp2((max(strmatch(':',
ytemp2'))+1):(length(ytemp2)-1)) % Dataline ends in a number
                else
                    jEnd=hex2dec(ytemp2((max(strmatch(':',
ytemp2'))+1):(length(ytemp2)-1)) );%Dataline ends in a HEX letter
                end
                for j=(10*i-9): jEnd

                    ytemp2=strrep(ytemp2,[':' lower(dec2hex(j))],'');
                end
                ytemp3=[ytemp3 ytemp2];
                ytemp2=y{(n+i+1),:} ;
            else %Last dataline
                for m=(10*(i-1)+1):(10*(i-1)+mod(NbrOfVar,10))% Slurk tal
                    ytemp2=strrep(ytemp2,[':' lower(dec2hex(m))],'');
                end
                ytemp2=[ytemp3 ytemp2]; %Converts all dataline as one
dataline
            end



        end

    end

    time(k)=str2num(ytemp1)';
    value(k,:)=str2num(ytemp2);
  k=k+1;
end

i=1;
while i<=NbrOfVar
    if strcmp(name(i,1), 'd')==1
```

```
        evalin('base',[name(i,1) name(i,3:length(name(i,:))) '=value(:,i)-
value(:,i+1);']) %Import the data to variblels named according to the
inputfile
        i=i+2;
    else
        evalin('base',[name(i,:) '=value(:,i);'])
        i=i+1;
    end
end
t(2)=toc;
disp(['File read in ' num2str(t(1)) 'and converted in ' num2str(t(2)) '
seconds'])
clear k n i y ytemp1 ytemp2 j jEnd m x ytemp3 t e tempstr
plot(time,value)
```

16. Appendix D1 The test rig.

16.1. phase 1 – design study

The most important work and decision in phase one is in what way the driver logic should be separated from the power BJT. This can seem trivial especially if one study the final design of the test rig and discover that it has no galvanic isolation between the logic and drive circuit. However the technique of galvanic isolation is perhaps one of the most important factors when designing a drive circuit.

There are two main techniques used for galvanic isolation optocoupler and signal transformer. Both are in use today and both have its pros and cones. Galvanic isolation is used for blocking the high noise levels at the power side, and thus preventing the spreading of noise from the power side to the logic and control side of the inverter. One more use of galvanic isolation is as a safety precaution. If something breaks in the high power side the galvanic isolation make sure there is no chance for high current or high voltage to cross over to the control/logic side. This could destroy the control circuits or even worse lead high power to the user with the risk of injure. The most common way of galvanic isolation today is the optocoupler, see figure 16.1.

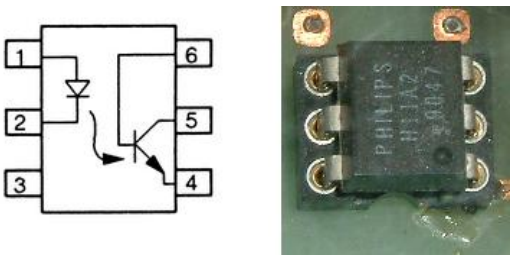


Figure 16.1. Schematic and a real-life component view of an optocoupler.

The optocoupler used a light emitting diode (LED) to transmit the signal from one side to another. On the other side of the galvanic isolation is a phototransistor that reacts on the light signal and amplifies the signal to a useful electric signal. One of the best features of an optocoupler that is really simple to use. There is a wide range of price and performance of optocoupler. Most optocoupler have an isolation voltage of a couple of thousands volts and come in regular dual in-line (DIL) packages. One advantage of an optocoupler is that it is very noise resistant. The signal can only travel from the LED to the phototransistor and not the other way. This blocks all noise from the powerside and makes it impossible for it to travel back to the control circuits. The downside for this is that the optocoupler cannot transfer power. The power has to be supplied both on the control and the power side.

However a signal transformer does not have this disadvantage. A signal transformer can transfer both signal and power. A signal transformer can be viewed in figure 16.2.

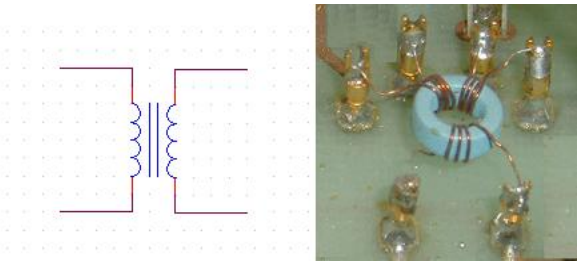


Figure 16.2. Schematic and a real life component view of a signal transformer.

A signal transformer usually consists of a ferrite ring core with two windings. The signal is applied on the ingoing winding and magnetically transferred to the out going winding. Because of the nature of transformers one can only transfer an AC signal from the control side to the power side and vice versa. This makes the signal transformers a bit difficult to use. A DC control signal has to be modulated with a high frequency signal. This high frequency signal is then transmitted over the transformer and converted back to an appropriate control signal. The advantages to be able to transfer power can be realized if the half bridge in figure 16.3 is studied.

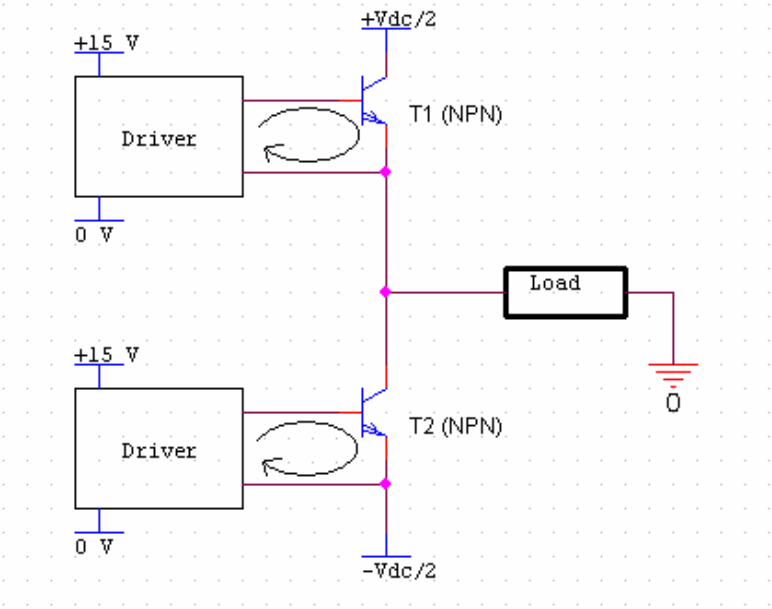


Figure 16.3. Half-bridge with drive circuits.

The drive circuits drive the BJT with a current going in to the base and returning after the emitter. If all the Earth points are connect together it will cause a short-circuit between the low side transistor (T2) collector and the load. Because of this one have to use different earth points for high and low side drivers. This can be done in several ways. One way is to create two independent floating 15 V voltage supplies that are used to drive the high and low side. With this solution it is possible to use an optocoupler or a signal transformer to transfer the signal in to the drive circuits. However this solution is bulky and requires two separated voltage supplies to every half bridge. If a signal transformer is used one can make the drive circuits so that all the power required to

switch the transistors is provided via the signal transformer. If the power is supplied via signal transformers there is no need for separated power supply's, which can save money and space. However things are not always this black and white. The BJT requires a continuous current when the BJT is turned on. This makes it very hard to design an appropriate signal transformer and drive circuit for a BJT.

16.2. Phase 2 – testing the ideas

Now it is time to take all the ideas from the previous chapter and put them to the test. During the master thesis TranSiC gave us two samples of the BitSiC 1206 transistor. At the start of the master thesis only one sample of a BitSiC transistor was available. Because of this it is very important that the transistor is not damaged. Most of the time a stunt double transistor is used. The stunt double is used to test the circuitry and the BitSiC is only used when the circuitry is thoroughly tested. During the development of driver circuits etc. a multiboard seen in figure 16.4 is used to rapidly test circuits and easily modify the circuit to the requirements.

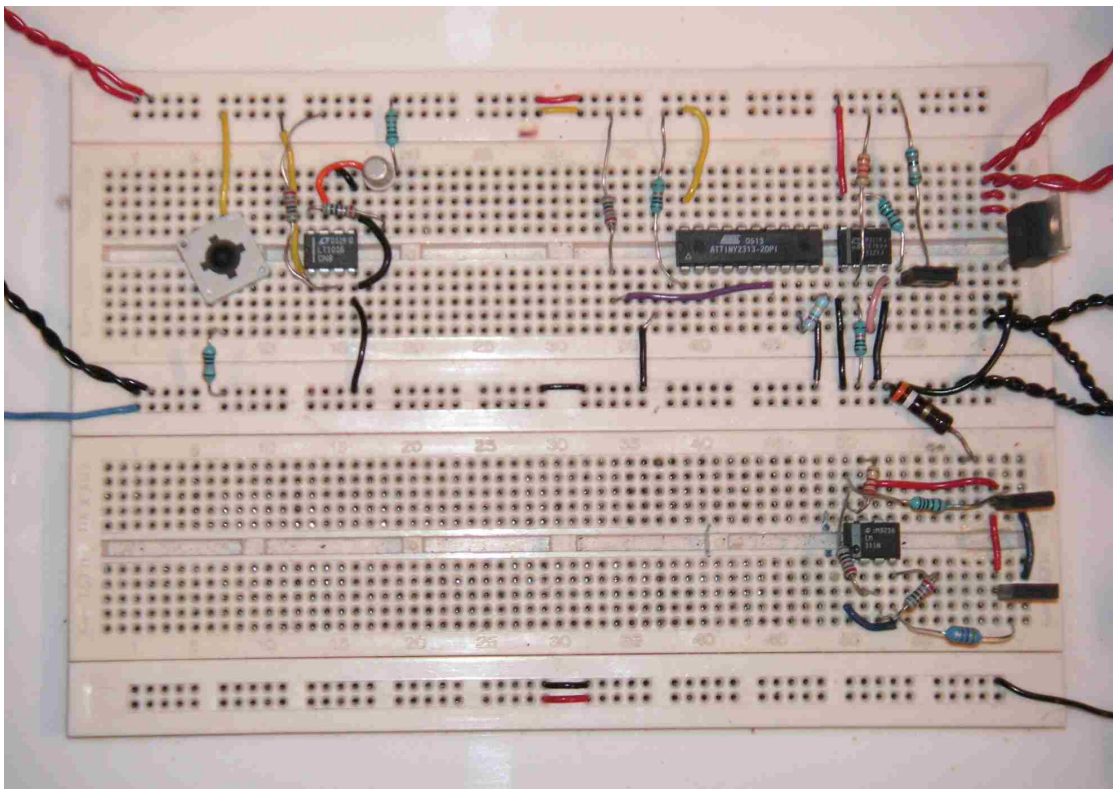


Figure 16.4. Rapid test made on a multiboard.

The first thing that is implemented in the driver is a "brain" to make it possible to control the BJT. It is decided to use a microcontroller (µC) to control and regulate the events in the test rig. The µC is regulating the BJT to a determent working point i.e. right current and voltage. The µC is then turning on the BJT for a predetermined time and turning it off again. It is not desired to turn on the transistor continually. This will heat up the BJT and make the transistor to behave differently from when it is cold. It is also important to regulate the current so that the BJT is not turned on for too long time and become overheated.

16.2.1. Microcontroller

Because of earlier experience of AVR microcontrollers from Atmel [23] this processor family is chosen. Within this family the AVR ATtiny2313 microcontroller is picked. The ATtiny2313 is an 8-bit CMOS processor with Reduced Instruction Set Computer (RISC) architecture. Some of feature of the ATtiny2313 is listed bellow.

- Operating Voltages: 2.7-5.5 V
- Speed: 0 - 20 MHz, up to 20 MIPS @ 20 MHz
- Flash EPROM: 2 kB
- EEPROM: 128 B
- SRAM: 128 B
- 32 General purpose working registers
- Number of I/O's: 18
- One 8-bit Timer/counter with separate prescaler and compare mode
- One 16-bit Timer/counter with separate prescaler and compare mode
- Four PWM channels
- In-system programable via SPI port
- Internal calibrated oscillator

The ATtiny2313 microcontroller has 120 assembler instructions. To the AVR 8-Bit RISC family line Atmel offers a free professional Integrated Development Environment (IDE) program named AVR Studio 4. AVR Studio 4 contains an assembler and a simulator that together with a chip programmer offers a powerful program suite to tackle all kind of programming challenges. Although AVR Studio 4 has been used in the past it is decided to go for a BASIC Compiler from MCS electronics [24] named BASCOM-AVR. BASCOM will give the advantage of really fast and easy being able to program an AVR µC. In differ from assembler language, in BASIC there is the possibility of using variables and other high language features. BASCOM has almost all features of the AVR µC's implemented as functions in basic language. In figure 16.5 one can view the BASCOM programming environment.

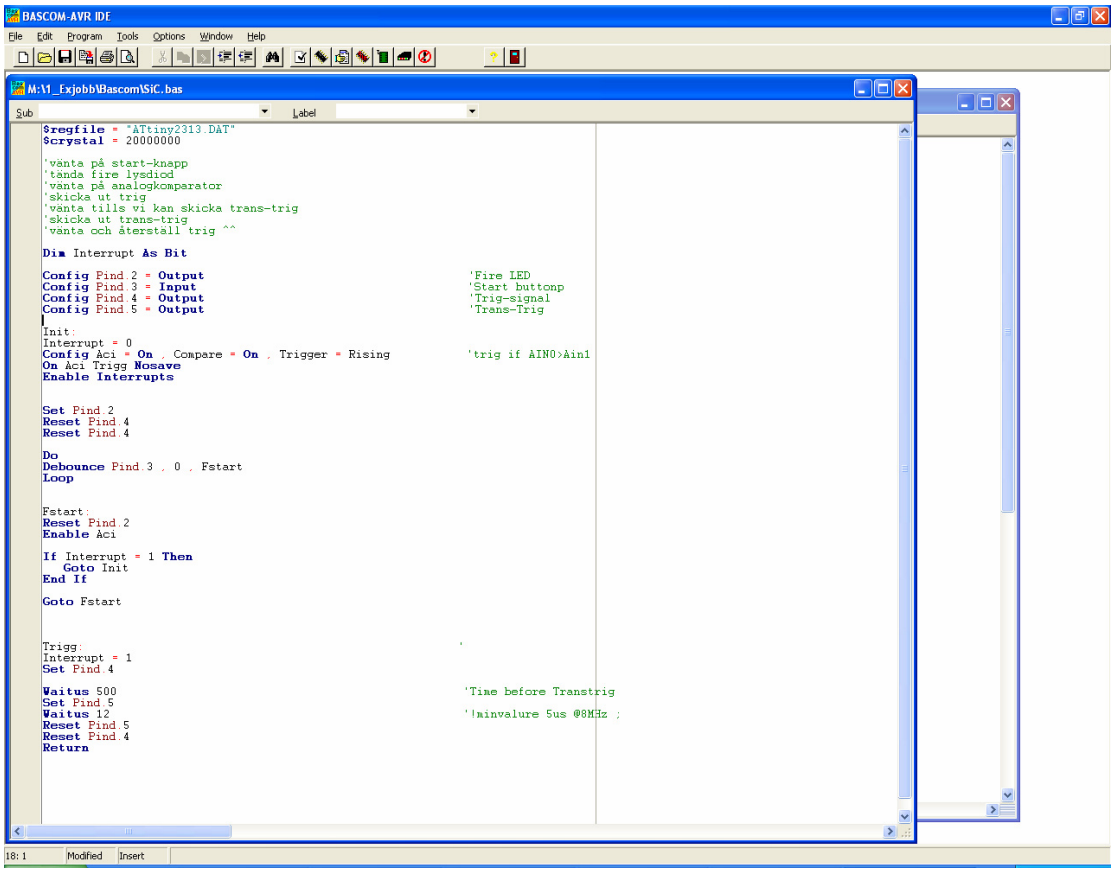


Figure 16.5. BASCOM program layout.

BASCOM allows one to use such programming commands as

- IF, ELSE statements
- WHILE, FOR and LOOP statements
- SET, RESET, DEBOUNCE and PRINT statements
- BIT, BYTE, INTEGER, WORD, LONG, SINGLE and STRING variables
- WAITMS, GOTO, POWERDOWN, IDLE, CALL and SUB statements

These commands are only a few selected samples of what BASCOM can offer. Just the possibility of using variables and variable names makes the code a lot simpler. BASCOM also has a fully integrated simulator that allows the coder to compile the code and run it as if in real-time. In the simulator one can monitor virtual chip-outputs as well as internal registers and memory. This makes it fast and easy to test code and ideas. When a program is written one can straight away check how it runs in the simulator and discover logic errors and get an idea of the performance of the code. In the simulator it is also possible to simulate hardware and easily interact with the simulator as if it was running in a real complex circuit.

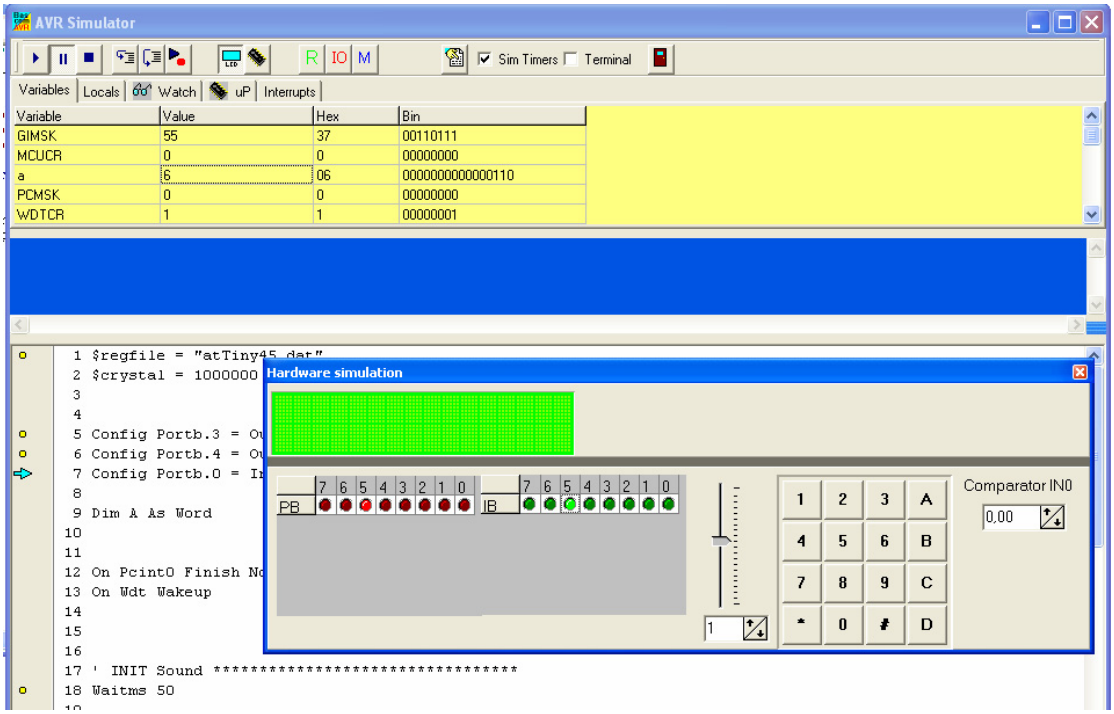


Figure 16.6. BASCOM simulator.

To explain the written code, it is easiest to follow along in the real code as the statements are explained. The full BASCOM code is in Appendix D2.

The code starts by instructions to the compiler

```
$regfile = "ATtiny2313.DAT"
$crystal = 8000000
```

The first line declares what type of microcontroller that the code is implemented on and thus giving what type of resources and alias that the compiler can use. The second line tells at what frequency the clock runs at. Next follows remark lines that declares for the reader what the program is suppose to do.

```
'Wait for firebutton
'Turn on Transistor
'Wait for the current to ramp up
'Turn off the transistor
'Send Trig-signal
'Make the switch i.e. turn on, turn off
```

The program will generate the logic signals as viewed in figure 16.7.
Next in the program come the configuration commands

```
Config Pind.2 = Output          'Fire LED
Config Pind.3 = Input           'Firebutton
Config Pind.4 = Output          'Trig-signal
Config Pind.5 = Output          'Trans-Trig (inverted)

Config Debounce = 50
```

These lines set the port pins as outputs and inputs. If a pin is set as an output it can sink or source about 20 mA which is enough for driving the LED or trigger the comparator in the drive circuit. The last line configures the debounce to 50 ms. The debounce command is used for getting rid of false triggers when using a switch. If a mechanical switch is used as an electrical input there will be contact bounces for a few ms when the switch is toggled. This is taken care of with the debounce command. When the

debounce command first registers a high logic level on the input it waits for a predetermined time or in this case 50 ms (25 ms is default). If the input still is high when the time has expired the debounce command executes a jump. If the program did not wait for 50 ms before making the jump, the program could do jumps at every bounce of the switch.

```
Beginning:

Set Portd.2                     'Init condition turn off LED
Reset Portd.4                    'Turn off Trig-signal
Reset Portd.5                    'Turn off Trans-Trig (inverted)

Do
Debounce Pind.3 , 0 , Fstart
Waitms 1                        ' Wait for Firebutton
Loop
```

The “Beginning:” command is a label that is used for looping the program. Next do all the used pins get set to their initial values. When the program has come this far it enters a loop. This loop will loop on forever unless the fire-button is pressed. If the fire button is pressed and it passes the debounce routine, the routine will make a jump to the “Fstart” label.

```
Fstart:
Reset Portd.2                    'Turn on LED
Set Portd.5                     'Turn on transistor

Waitus 2200                      'Ramp transistor

Reset Portd.5                    'Turn off transistor

Waitus 120                       'Time between ramp and switch
Set Portd.4                     'Trig oscillioscope for switch
Set Portd.5                     'Turn on Transistor foe the Switch
Waitus 50                       'Wait for ending the switch

Reset Portd.4                    'Turn off Trig-Signal
Reset Portd.5                    'Turn off Switch

Goto Beginning                      'Start over and wait for Firebutton
```

This code segment is best described with a logic graph viewed in figure 16.7.

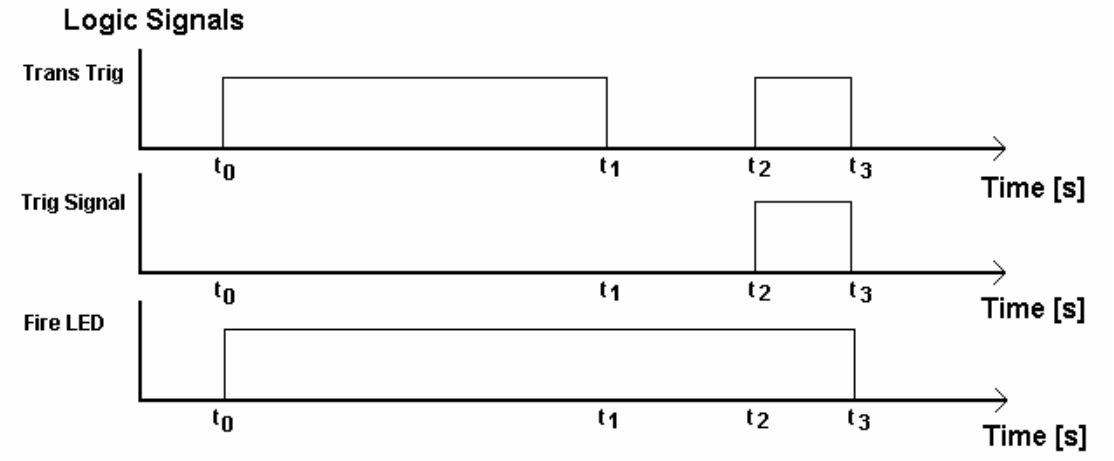


Figure 16.7. Logic output from the microcontroller.

When the program counter enters the Fstart label at t_0 the µC will turn on the transistor with "Trans trig". This will start to ramp the current in the load. At the same time the µC turn on the light emitting diode (LED) which indicates that the BitSiC has been turned on. The program waits for a predetermined time to (t_1) as the load current ramps up to the right value. At t_1 the program closes the BitSiC transistor. This will force the current to freewheel through the freewheeling-diode. Because all these events happen in the microseconds scale and the load is so big the current will stay relatively constant. Next the program opens the transistor for the switch. At the same time the program sends a trigger signal (Trig Signal) to the oscilloscopes so they can trigger and capture the events and fall/rise-times. When the BitSiC has been tuned on for 50 µs the program turns off the BitSiC, trigger signal and the LED. Now the switch has been executed and the program loop back to the start and waits for another go.

16.2.2. Push-Pull

A push-pull driver (fig 16.9) might at first view look simple. However the truth is far from it. Many hours is spent on doing laboratory work and discussing the "to be or not to be" of a push-pull driver.

The main question boils down to "how can the BitSiC be switched with as little energy loss as possible?". To answer this question one first need to get an insight of how a BJT switch. In a BJT it is all about moving charge. When turning on a BJT one have to start by driving a current in to the base. Initially the base-emitter junction will be reverse biased or at zero bias. Then the base current starts, the base-emitter junction becomes forward biased with a voltage equal to the junction built-in potential. As this happens, carrier injection starts and electrons from the emitter can diffuse through the base and reach the collector. It is at this point that the collector current starts to rise. To keep a certain collector current there is a need for a minimum continues base current. This base current will maintain a stored charged distribution. If the base current is increased from this minimum the base current will start to build up an excess of charge storage in the base, for a power bipolar transistor this also occurs in the collector. This stored charge will in fact decrease the turn-on time. The excess of charge have to be removed at turn-off which will slow down the turn-off. In the turn-off all the charge that is stored in the BJT will have to be removed. If the base current stops flowing, the charge in the transistor will be removed by the internal recombination process [25]. However this process of turning off the transistor is slow. In conventional power electronic driving techniques of BJT's one is advised to reverse the base current and draw a negative current out from the base to remove the stored charge. This will cut off the transistor faster. When all the stored charge in the transistor has been drown out and the Vbe voltage has become negative the transistor is fully cut off.

Now with the knowledge of how a BJT switches one can look at how the base current shall be shaped for optimal switching [26]. The waveform for the base current is shown in figure 16.8 .

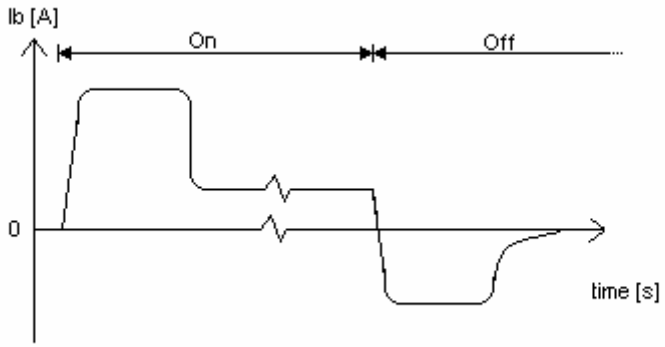


Figure 16.8. Waveform for an optimal base current.

Figure 16.8 shows the base current rises rapidly to a high value at the start of the turn-on. This high current makes sure that the BJT get a fast build up the storage in the transistor. When the transistor is turned on, the current is reduced to keeping the current at a minimum for maintaining the stored charge distribution. This reduces undesired excess of charge. At the turn-off the driver draws a negative current from the base. As the base-emitter junction will be reversed biased the Vbe voltage will get negative and act as a blocking diode and cut off the transistor and base current. This is the practice of driving power BJT's, at least for a silicon (Si) BJT. If one uses these higher initial base currents the turn-on and turn-off will be shorter. With a SiC BJT one has to rethink this concept. A SiC BJT has far less diffusion-capacitance and storage charge than a regular Si BJT. Because of this it is not that obvious that a negative current to remove charge by carrier sweep-out is needed.

After initial talks with TranSiC it became clear that a negative base current might not be necessary. Laboratory work also pointed in this direction, at least it did not show any improvement when the BitSiC is switched with a negative base current. This initial testing of the BitSiC is not done at the right working point. This is the reason for building the test rig with the capability to be driven with a negative voltage.

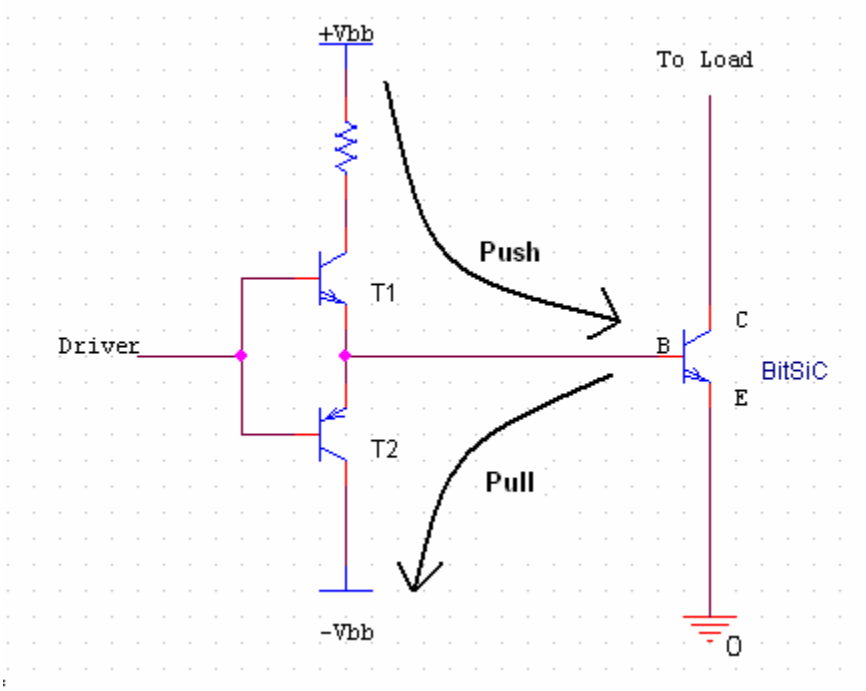


Figure 16.9. A Push-Pull driver for the BitSiC.

It is at this point a push-pull driver comes in to the picture. To create a current waveform as shown in figure 16.8 the driver need the capability to draw current from the base. One of the simplest push-pull configurations is viewed in figure 16.9. There are a lot of details of the driver that has to be solved for the push-pull driver to work. Taking aside such non trivial things like positive and negative drive voltage ($+V_{BB}$, $-V_{BB}$), there is still things that need to be carefully designed for the push-pull diver to work. The first and possibly the hardest thing is to get the lower transistor (T2) to open when the upper transistor (T1) closes. This requires that the drive circuit deliver both negative and positive voltage. The push-pull driver consists of one high-side transistor and one low-side transistor. These are of different types, the high-side transistor is an N-type transistor and the low-side transistor is a P-type transistor. When the control signal is high the N-type transistor (T1) will conduct and push current in to the base of the BitSiC. When the control signal toggles and becomes negative, the upper transistor turns off as the lower transistor of P-type (T2) starts to conduct and pulls out the current from the BitSiC base. In the switch moment there is a possibility that both transistors are turned on for a short period of time. This will result in a short-circuit and draw excessive power or in worst-case lead to a breakdown of the transistors. Therefore is the choice of transistors of great importance. Fortunately there are transistor pairs made to be complementary to each other. For example if one look up BC368 or BC369 (71-000-68/ 71-000-92) [21] on ELFA's homepage one will see that BC368 and BC369 is a complementary pair and is recommended for the push-pull configuration used in figure 16.9. Still it is not given that one will find a complementary pair with suitable performance. In this case it is important to examine how the transistor pair works together. There is one area that this thesis has overlooked so far, the current limiter resistor. In figure 16.10 resistors has been put in the circuit to limit the current at the collector of the upper transistor (T1). This is the normal way of driving a load with a BJT. But there are other ways to limit the current.

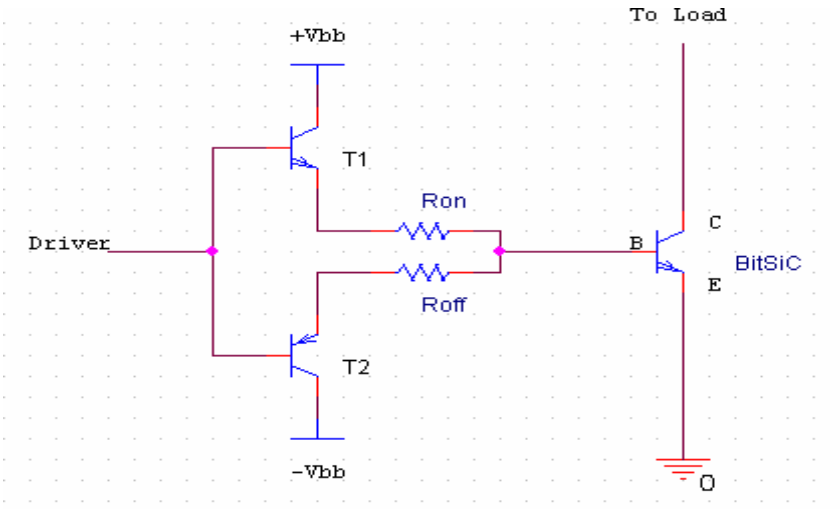


Figure 16.10. One way of putting in current limiter resistors.

Figure 16.10 shows two separate resistors, one for limiting the turn-on current and one for limiting the turn-off current. This configuration is quite common especially for driving MOSFET's. Using this way of limiting the current has to be made with caution. The voltage over the resistors, for example V (R_{on}) will be part of a feedback loop. The feedback is shown with the following equations:

$$Vbe_{(T1)} = Drivervoltage - R_{on} * Ib_{(BitSiC)} - Vbe_{(BitSiC)} \quad (16.1)$$

In figure 16.9 there is no feedback loop, instead the expression will be:

$$Vbe_{(T1)} = Drivervoltage - Vbe_{(BitSiC)} \quad (16.2)$$

Now the term $R_{on} * Ib_{(BitSiC)}$ is not in the expression and thus will not give any feedback. The lesson learned is that one has to be careful when the resistors is placed directly to the power transistors base.

16.2.3. Comparator

A comparator is a good thing to incorporate in the final design. The comparator will speedup the switching and converts signals between different logic levels. If a comparator is used as input to the driver circuit, the driver will have a more independent interface. Anything that can generate a voltage can control the driver circuit without the need of being able of provide much power. This means in reality that the driver can be controlled with a microcontroller, logic circuit or directly with a computer. A comparator can also guarantee a certain performance. If a slow control signal is used as in the case of an AVR microcontroller one always get a high performance after the comparator. It does not matter if the control signal is slow because the comparator only trigger at a certain trigger point and transforms the slow control signal to a fast output signal independent of the input signal. At the start of the laboratory work, most of the work was done with the LM311 comparator that the faculty had in stock. Although LM311 was fine for initial testing of the drive circuit it soon became clear that there was a need of a high performance comparator. LM311 has a rise time of 100 ns and a fall time 400 ns. This is significant slower than the target

rise/fall time for the BitSiC. Therefore a comparator with a rise and fall time that matches the BitSiC is needed. After searching the net for an appropriate comparator the LT1016 was found to fulfil the requirements. LT1016 has a rise/fall time less than 10 ns.

16.2.4. The Baker clamp

When looking into how the BitSiC switch one discovers that the BitSiC have a turn-off that is about two times slower that the turn-on. The literary study showed on a technique that is used for shortening the turn-off the Baker clamp [25]. The Baker clamp is an anti-saturation network used for BJT's. This means that the anti-saturation network keeps the BJT from going too much into saturation. If a BJT is driven too much in to saturation the BJT takes up additional charge that slows the turn-off. This will make the BJT consume unnecessary power. A Baker clamp is shown in figure 16.11.

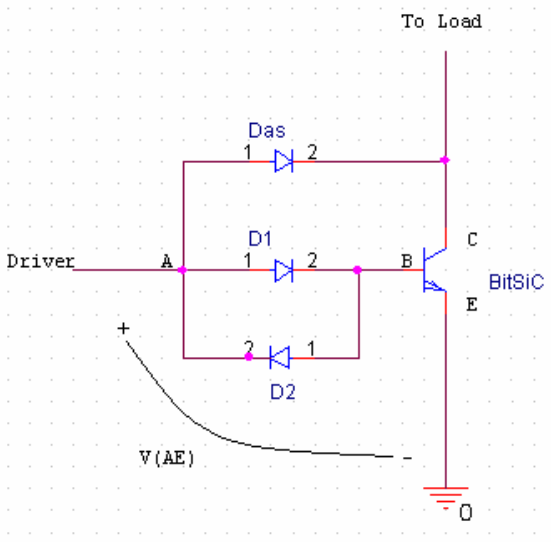


Figure 16.11. The Baker clamp.

The Baker clamp works by clamping the V_{CE} voltage slightly above the forward saturation voltage $V_{CE(sat)}$. If you study figure 16.11 you see that:

$$V_{AE} = V_{BE(on)} + V_{D1} = V_{CE(sat)} + V_{Das} \tag{16.3}$$

If the diodes have the same forward voltage drops, the expression will be:

$$V_{BE(on)} = V_{CE(sat)} \tag{16.4}$$

Since $V_{BE(on)}$ is generally larger than $V_{CE(sat)}$ you will keep the transistor from going to deep in saturation and thus reducing the storage time. However there is one down side of the Baker clamp, the Baker clamp will increase the on-state losses. This results in that the baker clamp is mostly used in high frequency circuits, were the increased on-state losses gets lower than the improved turn-off losses. There is also a way to regulate how deep the transistor goes in to saturation. One can increase the forward voltage of

D1 by adding multiple diodes in series with D1. This will keep the transistor further away from saturation and reduce the turn-off time even further.

Testing the baker clamp on the BitSiC there was no improvement of turn-off switching behaviour. This could be explained by of the much smaller stored charge in BitSiC. A SiC BJT has far less stored charge, which makes it switch faster than a regular Si BJT. Even if the BitSiC is driven hard into saturation there is far less charge to be removed in the turn-off, which makes a baker clamp unnecessary for this application.

16.2.5. Soft switching versus hard switching

In the push-pull chapter the optimal waveform for the base current is described. The main idea is to create a waveform like the one in figure 16.8 to make the switching as fast as possible. However raw speed is not always the optimal solution. Although higher speed minimizes the time for a switch and there by get a low average power there might be better ways of getting low power consumption. In the previous examples, the base current has been drawing directly to the negative base voltage $-V_{BB}$. This is called hard switching. In soft switching one does not want to draw the current hard to the negative drive voltage supply. Instead the current derivative is limited for the purpose to form a fall and rise contour to a more benefiting switching. One way of limiting the base current is to insert an inductor as can be seen in figure 16.12.

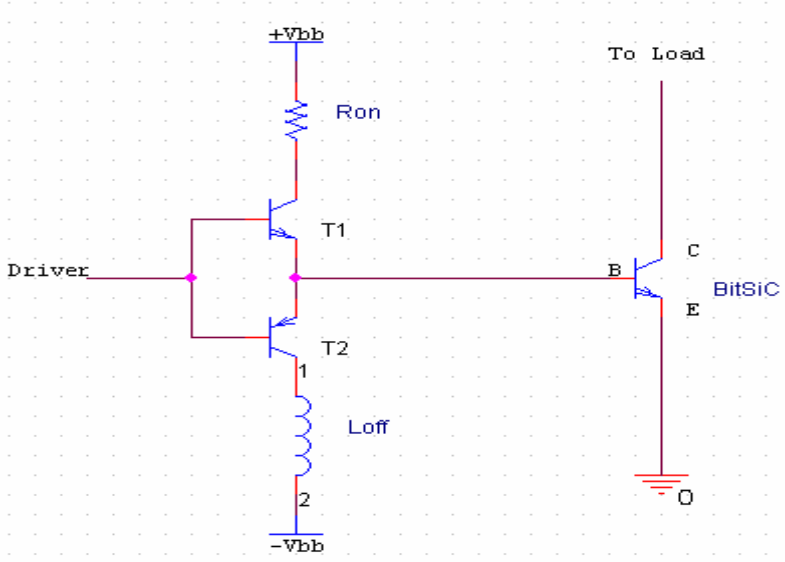


Figure 16.12. Soft switching by adding an inductor.

With an inductor in the turn-off path the currents negative derivate will be restricted. This will affect the waveform of the current in that way that it restricts the negative flanks. There is one major flaw with closing the transistor as fast as possible with a direct negative voltage. If the current is drawn out too fast from the transistor there might be charge trapped inside the drift region. The only way for this charge to escape the transistor is via internal recombination [25] and this is a very slow process. The effect of this trapped charge can be seen in figure 16.13.

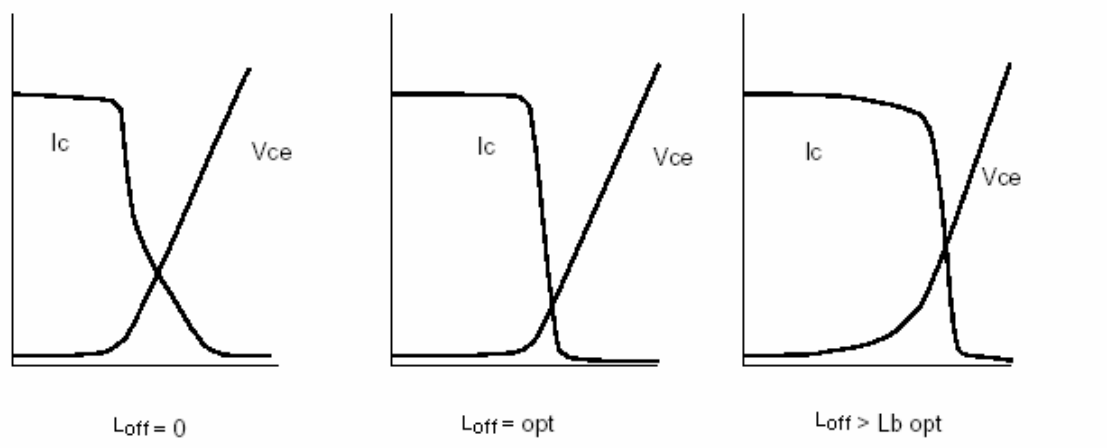


Figure 16.13 Waveforms with different inductor (Loff) at turn-off.

In the leftmost picture (figure 16.13), there is no inductor to regulate the negative current derivate. There one can see that after a fast slope the current starts to level out causing a current tail. This current tail is the result of charge being trapped inside the transistor and prolonging the turn-off. By regulating the negative current derivate one can make sure that all of the charge in the transistor is removed by carrier sweep-out. Choosing an optimal value on the inductor one can produce a turn-off like in the middle picture. With an optimal value one do not get a current tail and in the same time minimize the power consumed in the turn-off. This can clearly be seen by inspecting the area beneath Ic and Vce.

In the laboratory there is not any improvements seen of using soft switching. The reason for this could be that the SiC transistor has less stored charge than a regular Si transistor. In the tests made one can see that the BitSiC give a current tail in some conditions. This current tail can often be corrected by changing the base resistor or changing the rise time or fall time of the driver.

16.3. Phase 3 – Building the test rig

In the previous phase all the laboratory work is done that will result in a correct driver circuit to fulfil the requirements. It is in this phase that the test rig system is constructed. The test rig system is rather complex, the test rig system is shown in figure 16.14.

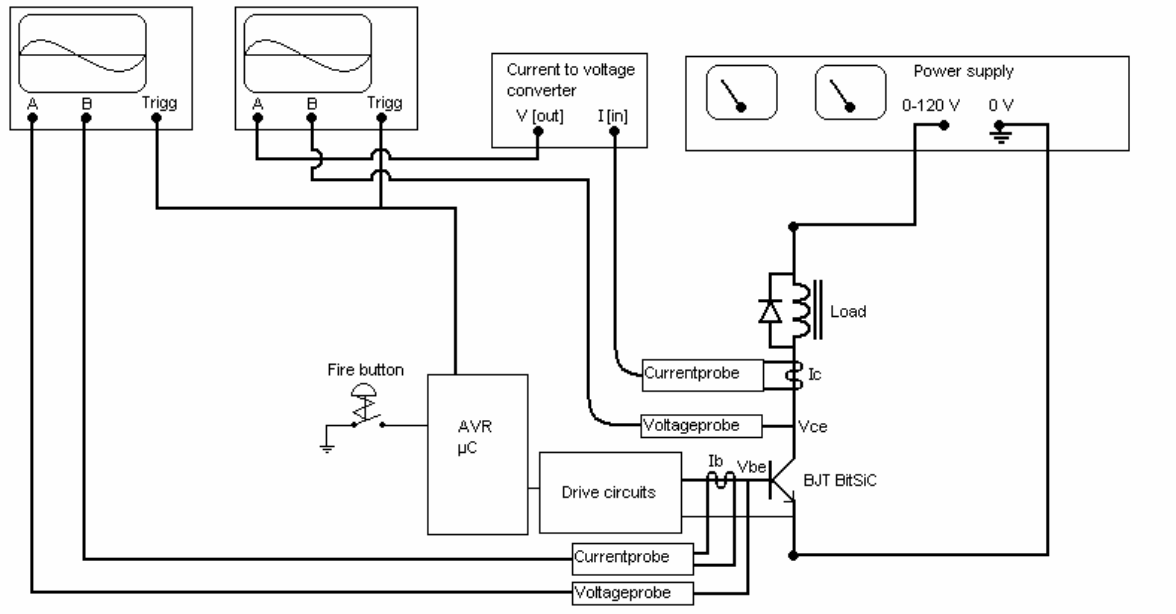


Figure 16.14. A schematic view of the most important instruments for the test rig.

Figure 16.14 shows the circuits and instruments that are used for the investigation of the BitSiC transistor. The "brain" of the set-up is the AVR microcontroller. This µC controls the drive circuit. Pressing the fire button the µC will send an initial control signal to the drive circuit, which ramps the load current to a specific working point. When the predetermined collector current has been reached the µC turns off the BitSiC and sends out a trigger signal to the sampling oscilloscopes. After this the µC makes a switch i.e. a turn-on and a turn-off with a specific turned on time. When the BitSiC opens it will start to draw current through the load. The load consists of a 1350 mH inductor with a laminated iron core. A rather big inductor is chosen to make the load as inductive as possible. With a large inductor the load will be more current stiff. A more current stiff load will slow the current derivate down which will result in more time to do the switch. When the BitSiC has driven up the collector current through the load it closes. The current will continue to flow inside the inductor. To avoid a voltage build up at the collector an anti parallel diode (flyback diode) is added. To stick with the theme of the master thesis a SiC Schottky diode were chosen as a flyback diode. For practical reasons is a SiC diode from Infineon chosen for the test rig, more exact SDT06S60 (600 V, 6 A) that is in the same power range as BitSiC.

To measure the collector current a current probe is used. The current probe (P6302) is connected to a Tektronix AM 503 Current probe amplifier, shown in figure 16.15. The current probe amplifier converts the current to a scalable voltage that is readable for an oscilloscope. To measure the base current for the BitSiC a differential voltage probe is used. The differential voltage probe is connected over a known base resistance. In this way it is possible to get a scaled value of the base current. The voltages Vce and Vbe are measured with a normal oscilloscope 10x voltage probe.
Pressing the fire button the AVR µC will start to ramp up the current to the correct working point. Just before the µC makes the switch it sends out a trigger-signal to the oscilloscopes. The Oscilloscopes are set for "Single trigger mode" which means that the oscilloscopes wait for a trigger and then captures the signal. After the oscilloscope has captured the signal it freezes so one can examine the signal. To capture a new sequence

the oscilloscope has to be reset. The test rig system has two TDS210 oscilloscopes from Tektronix [27]. Ideally a four channel oscilloscope should have been used. However a four channel oscilloscope was not possible to get hold of. Instead the test rig system is improvised with two oscilloscopes. To make the oscilloscopes trigger at the same time the trigger signal is dived via coaxial cables. If the same external trigger is used on both oscilloscopes the same time frame is captured on the both oscilloscopes.

Figure 16.15 Tektronix current probe (P6302) and amplifier (AM 503).

16.3.1. Building the PCB

The next step in the phase is to do a schematic in a PCB CAD program. When working with a multiboard it is noticeable that there is some problem with stray inductance. The stray inductances causes the current to oscillate when the BitSiC switch. This is solved by improving the circuit with a PCB. The main goal for the PCB is to improve the quality of the captured signals.
To make the PCB a CAD program from Cadsoft [28] is used named EAGLE Layout Editor v4.16r2. The Eagle program consists of two parts one schematic part (figure 16.16) and one Board part (figure 16.17). A PCB project is started by drawing the schematics. Eagle comes with a lot of library's that allows one to quick and easy find

the components that is needed. In the case of the BitSiC, Eagle does not have a library with the BitSiC transistor so a TranSiC library is made with the BitSiC component.

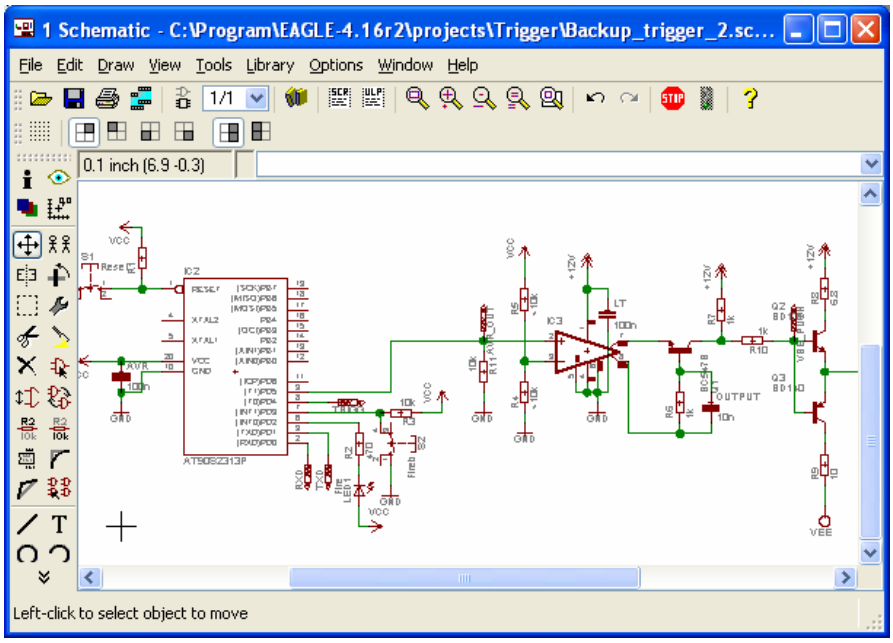


Figure 16.16. Showing the schematic part of Eagle.

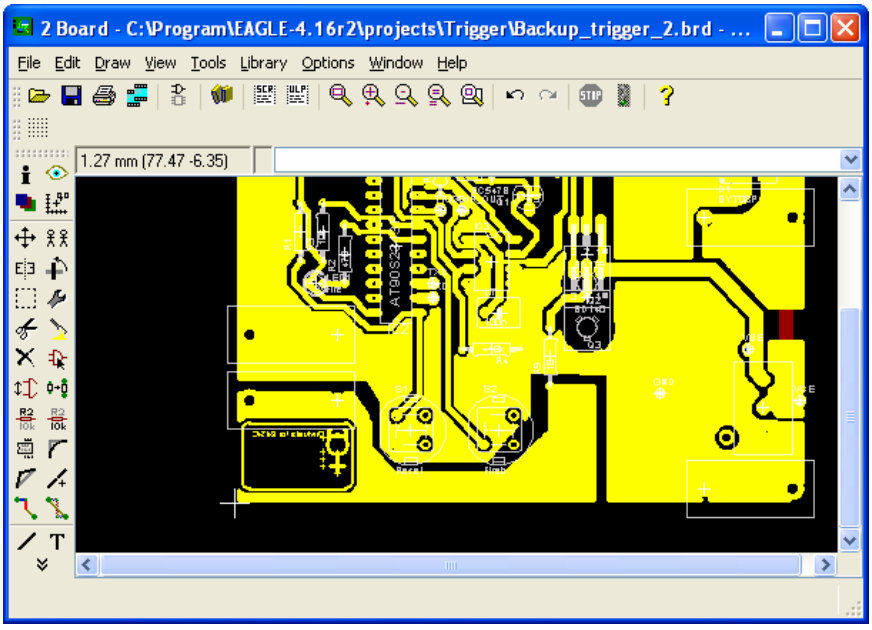


Figure 16.17. Showing the Board part of Eagle.

When the schematic is drawn it is time to switch over to the Board part of the Eagle program. All the parts in the schematic will be imported to the board version. In the board program the components are placed at the desired position on the PCB. Then the traces i.e. the copper routes between the parts are drawn. All the connections in the schematic will be represented in the board version with lines. All the connections lines are then drawn with copper routes to complete the PCB.

To minimize interference the PCB is divided in several sections.

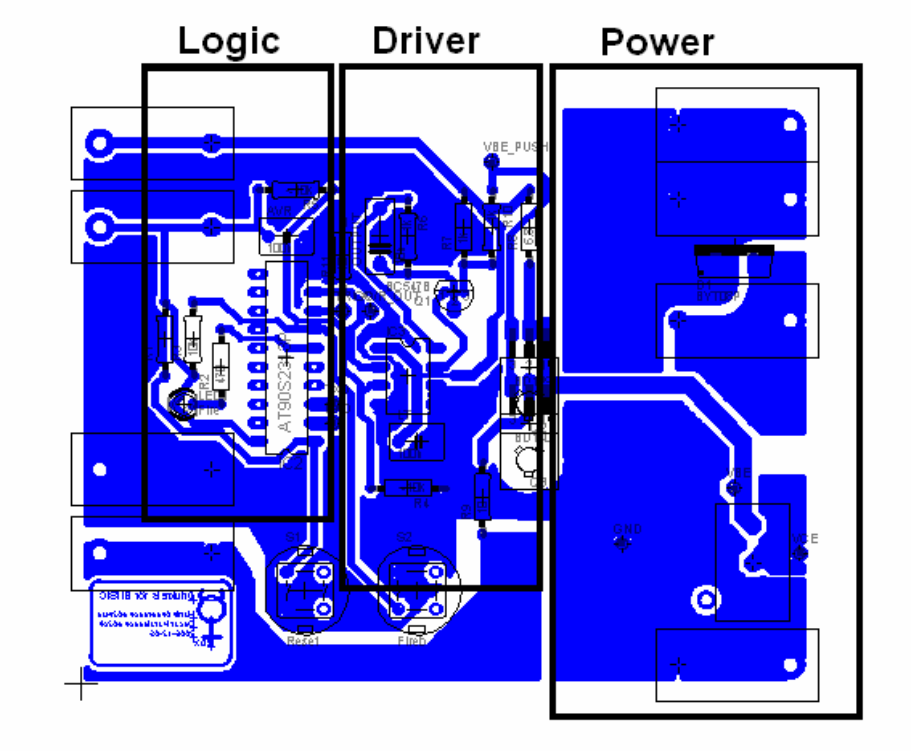


Figure 16.18 The sections of the PCB.

In figure 16.18 it is shown how the PCB is divided in the sections Logic, Driver and Power. To further separate the sections and prohibit interferences several different voltage sources are used in different section. The logic section consists of the AVR µC that is the brain of the circuit. The µC is driven by a +5 V voltage source. The driver section is driven with a positive 12 V and a negative 6 V. In the drive section the logic signal from the µC if amplified to a powerful base current that drives the BitSiC in the power section. In the Power section do the BitSiC switch the load current so the oscilloscopes can capture the behaviour of the BitSiC. Also notice the width of the PCB traces in the logic section, the traces are relative small. Then in the driver section the traces are bigger. Finally in the Power section the traces are so big that they are whole fields of cupper. This has to do with the stray inductance. The bigger current that flows in the traces, the bigger effect has the stray inductance on interference. This is also the reason for the ground plates that fills up the area between traces.

One other way of getting rid of unwanted oscillation is to make the voltage source as voltage stiff as possible. It is important to get the voltage stiffness as close to the switching as possible. Placing capacitors as close to the switching as possible does this, see figure 16.19.

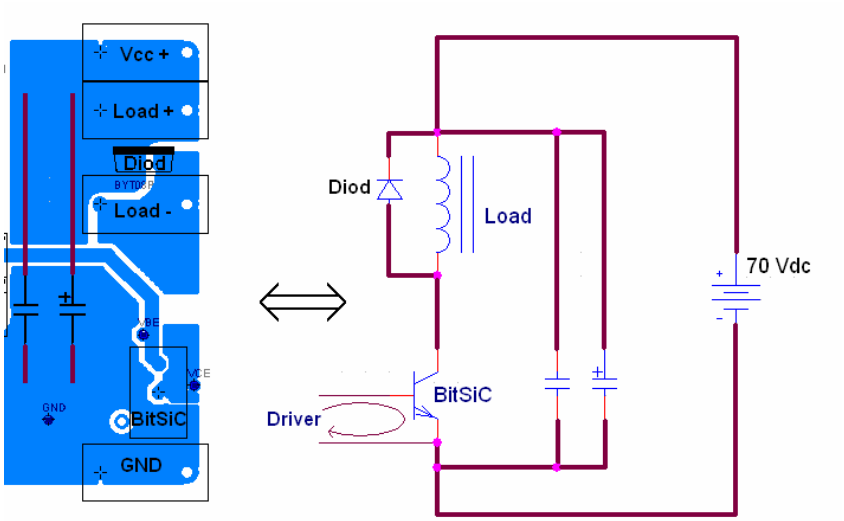


Figure 16.19. Capacitors placement.

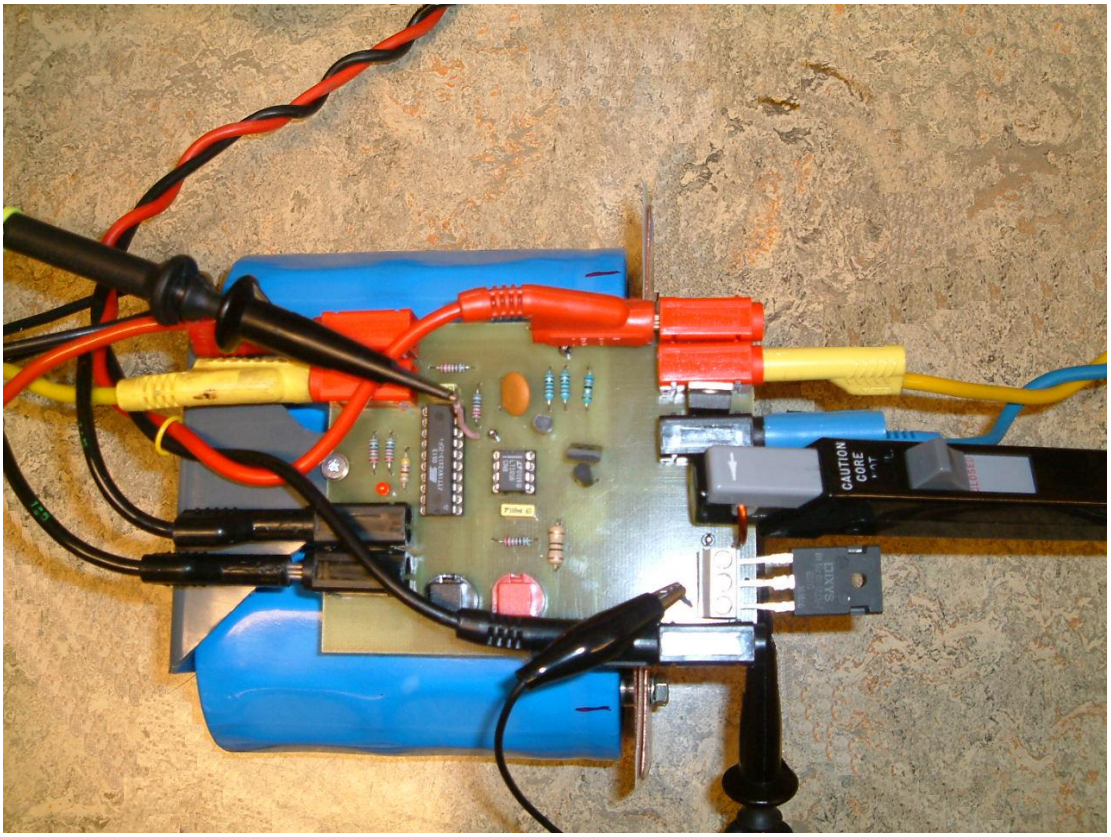


Figure 16.20. Close up of the test rig.

As seen in figure 16.20 everything is made as cramped as possible this reduces the stray inductance. It is also important to use two different types of capacitors. First there are two big polarized electrolyte capacitors that make up a big capacitance bank to supply the switch with energy. However electrolytic capacitors have quite poor performance at high frequency. This is the reason why a polypropylene capacitor is added. This smaller capacitor takes care of the fast transients. To fixture the capacitors to the PCB a sandwich technique is used. This will make the connection between the capacitors and the switching parts as capacitive as possible see figure 16.21.

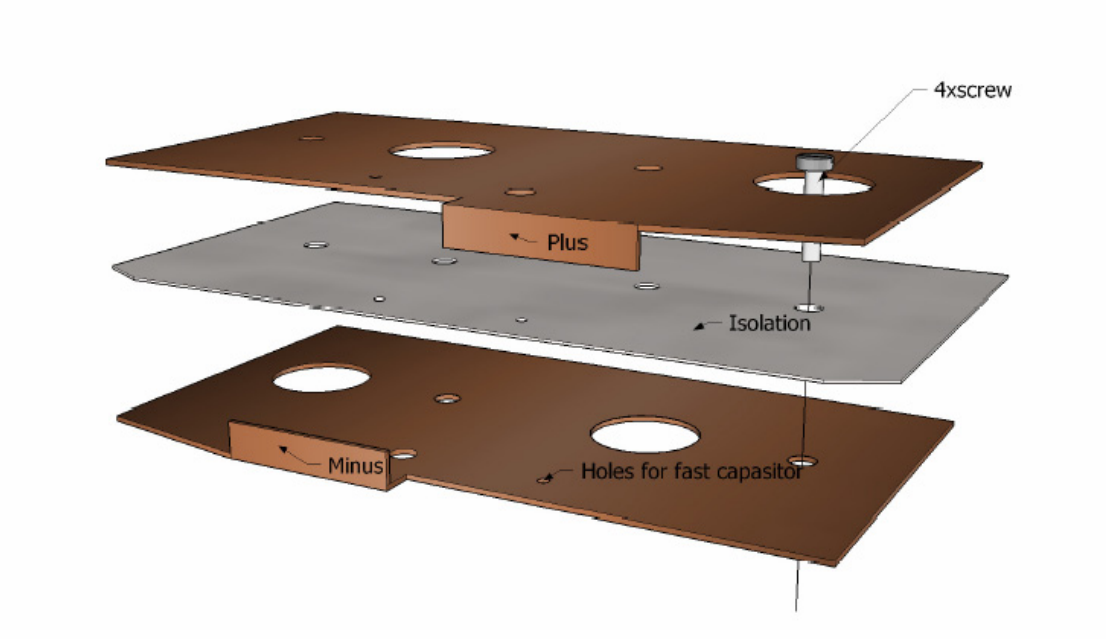


Figure 16.21. Sandwich technique to fixture the capacitors.

The sandwich technique uses two sheets of copper plates that are placed together. In between the copper plates is a dielectric sheets placed as an isolation material. The different poles of the capacitors are then bolted to each copper plate. The cupper plates are soldered directly to the PCB as close to the switching as possible. This configuration is in it self as a capacitor and thus is the stray inductance kept to an absolute minimum.

One other practical thing to keep in mind when designing a PCB for power switching is to keep the control loop as separated as possible from the power loop. What this mean can be illustrated in figure 16.22.

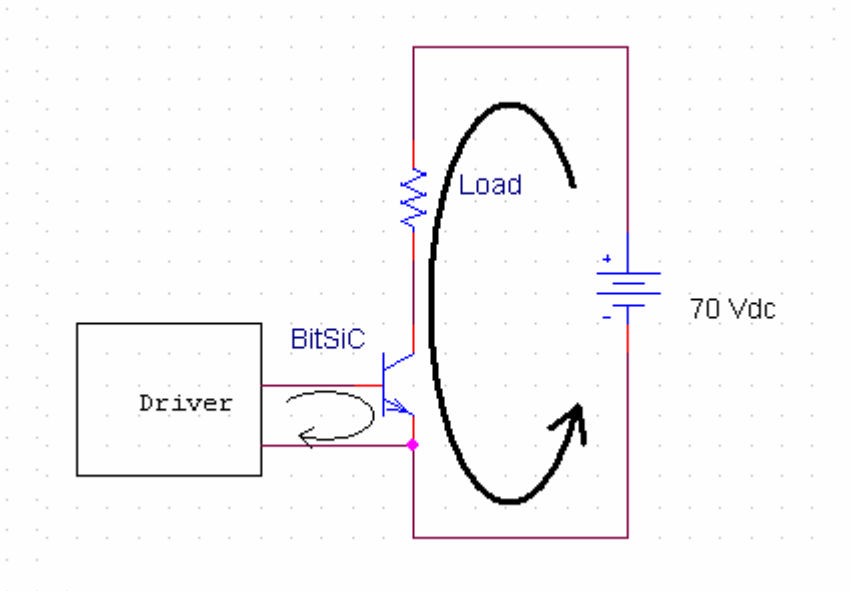


Figure 16.22. Control loop and power loop.

A power transistor such as BitSiC shall be treated as a four-port i.e. with an input/output for the driver and an input/output for the power connection. Although a

regular BJT only has 3 legs the emitter shall be separated into two parts, one part for the driver (driver emitter) and one part for the power loop (power emitter). The Driver emitter shall be separated from the power loop as soon as possible after the junction. This means that the driver emitter should in an ideal case be separated inside the capsule. This is the case for power components in a higher power range. The reason to use a four-port is that if a four-port is not used there will be a voltage drop over the emitter wire. This voltage drop will be dependent on the load current and give unwanted feedback to the driver. In the worst case the feedback can alter the working point for the transistor and cause erratic behaviour.

17. Appendix D2 - The full BASCOM code

```
$regfile = "ATtiny2313.DAT"
$crystal = 8000000

'Wait for firebutton
'Send Trig-signal
'Turn on Transistor
'Wait for the current to ramp up
'Turn off the transistor
'Make the switch

Dim Interrupt As Bit

Config Pind.2 = Output                    'Fire LED
Config Pind.3 = Input                     'Firebutton
Config Pind.4 = Output                    'Trig-signal
Config Pind.5 = Output                    'Trans-Trig (inverted)
Config Pinb.0 = Input                     'Ai0 to input
Config Pinb.1 = Input                     ' - "-"
Config Debounce = 50

Init:
'Interrupt = 0                            'Removed: No compare is used
'Config Aci = On , Compare = On , Trigger = Rising          'trig if AIN0>Ain1
'On Aci Trigg
Disable Interrupts

Begining:

Set Portd.2                               'Init condition turn off LED
Reset Portd.4                             'Turn off Trig-signal
Reset Portd.5                             'Turn off Trans-Trig (inverted)

Do
Debounce Pind.3 , 0 , Fstart
Waitms 1                                  ' Wait for Firebutton
Loop

Fstart:
Reset Portd.2                             'Turn on LED
Set Portd.5                               'Turn on transistor

Waitus 2200                               'Ramp transistor

Reset Portd.5                             'Turn off transistor

Waitus 120                                'Time between ramp and switch

Set Portd.4                               'Trig oscillioscope for switch
Set Portd.5                               'Turn on Transistor foe the Switch
Waitus 50                                 'Wait for ending the switch

Reset Portd.4                             'Turn off Trig-Signal
Reset Portd.5                             'Turn off Switch

Goto Begining                             'Start over and wait for Firebutton
```

18. Appendix E - BJT parameters

Model parameters[31]	Description	Units
AF	flicker noise exponent	
BF	ideal maximum forward beta	
BR	ideal maximum reverse beta	
CJC	base-collector zero-bias p-n capacitance	farad
CJE	base-emitter zero-bias p-n capacitance	farad
CJS (CCS)	substrate zero-bias p-n capacitance	farad
CN	quasi-saturation temperature coefficient for hole mobility	
D	quasi-saturation temperature coefficient for scattering-limited hole carrier velocity	
EG	bandgap voltage (barrier height)	eV
FC	forward-bias depletion capacitor coefficient	
GAMMA	epitaxial region doping factor	
IKF (IK)	corner for forward-beta high-current roll-off	amp
IKR	corner for reverse-beta high-current roll-off	amp
IRB	current at which Rb falls halfway to	amp
IS	transport saturation current	amp
ISC (C4) †	base-collector leakage saturation current	amp
ISE (C2) †	base-emitter leakage saturation current	amp
ISS	substrate p-n saturation current	amp
ITF	transit time dependency on Ic	amp
KF	flicker noise coefficient	
MJC (MC)	base-collector p-n grading factor	
MJE (ME)	base-emitter p-n grading factor	
MJS (MS)	substrate p-n grading factor	
NC	base-collector leakage emission coefficient	
NE	base-emitter leakage emission coefficient	
NF	forward current emission coefficient	
NK	high-current roll-off coefficient	
NR	reverse current emission coefficient	
NS	substrate p-n emission coefficient	
PTF	excess phase @ 1/(2□·TF)Hz	degree
QCO	epitaxial region charge factor	coulomb
QUASIMOD	quasi-saturation model flag for temperature dependence if **QUASIMOD** = 0, then no **GAMMA**, **RCO**, **VO** temperature dependenceif **QUASIMOD** = 1, then include **GAMMA**, **RCO**, **VO** temperature dependence 0	
RB	zero-bias (maximum) base resistance	ohm
RBM	minimum base resistance	ohm
RC	collector ohmic resistance	ohm
RCO ‡	epitaxial region resistance	ohm
RE	emitter ohmic resistance	ohm
TF	ideal forward transit time	sec
TR	ideal reverse transit time	sec
TRB1	RB temperature coefficient (linear)	$°C^{-1}$
TRB2	RB temperature coefficient (quadratic)	$°C^{-2}$
TRC1	RC temperature coefficient (linear)	$°C^{-1}$
TRC2	RC temperature coefficient (quadratic)	$°C^{-2}$

TRE1	RE temperature coefficient (linear)	$°C^{-1}$
TRE2	RE temperature coefficient (quadratic)	$°C^{-2}$
TRM1	RBM temperature coefficient (linear)	$°C^{-1}$
TRM2	RBM temperature coefficient (quadratic)	$°C^{-2}$
T_ABS	absolute temperature	°C
T_MEASURED	measured temperature	°C
T_REL_GLOBAL	relative to current temperature	°C
T_REL_LOCAL	relative to AKO model temperature	°C
VAF (VA)	forward Early voltage	volt
VAR (VB)	reverse Early voltage	volt
VG	quasi-saturation extrapolated bandgap voltage at 0° K	V
VJC (PC)	base-collector built-in potential	volt
VJE (PE)	base-emitter built-in potential	volt
VJS (PS)	substrate p-n built-in potential	volt
VO	carrier mobility knee voltage	volt
VTF	transit time dependency on Vbc	volt
XCJC	fraction of **CJC** connected internally to Rb	
XCJC2	fraction of **CJC** connected internally to Rb	
XCJS	fraction of **CJS** connected internally to Rc	
XTB	forward and reverse beta temperature coefficient	
XTF	transit time bias dependence coefficient	
XTI (PT)	IS temperature effect exponent	

19. Appendix F Theory of modelling components in PSpice

To model different components in PSpice there are a set of equations with different parameters that have to be set. This master thesis is only going to concentrate on the BJT models most important parameters, because the semiconductor that is to be modelled is a BJT. The idea is to take a transistor that has similar behaviour and "tweak" it to get the behaviour wanted.

To model a BJT one first has to understand a simple diode and remember that simulation modelling is a curve fitting "game". Even if one has measured the part to calculate the different parameters, one still have to adjust the spice-parameters to get the right behaviour of the model.

19.1.1. Semiconductor diode model

It is difficult to explain every equation and parameter in the diode model, but starting with the most basic behaviours a more complex model will arise. The PSpice diode model contains a nonlinear current source that follows the Shockley equation, see equation 19.1.

$$Id = Is * \left(e^{Vd/(n*Vt)} - 1\right) \tag{19.1}$$

Where: Vd is the voltage across the junction
Vt is the thermal voltage (k*T/q)
n is the forward current factor against the voltage.

This equation does not include the non ideal behaviour of the diode at low currents (<1 nA), these low currents will be ignored by PSpice but can be seen in reality.
The Is-parameter controls the forward and backward current relative the voltage. Changing the Is-parameter to different values one can obtain characteristics of different types of diodes, see figure 19.1.

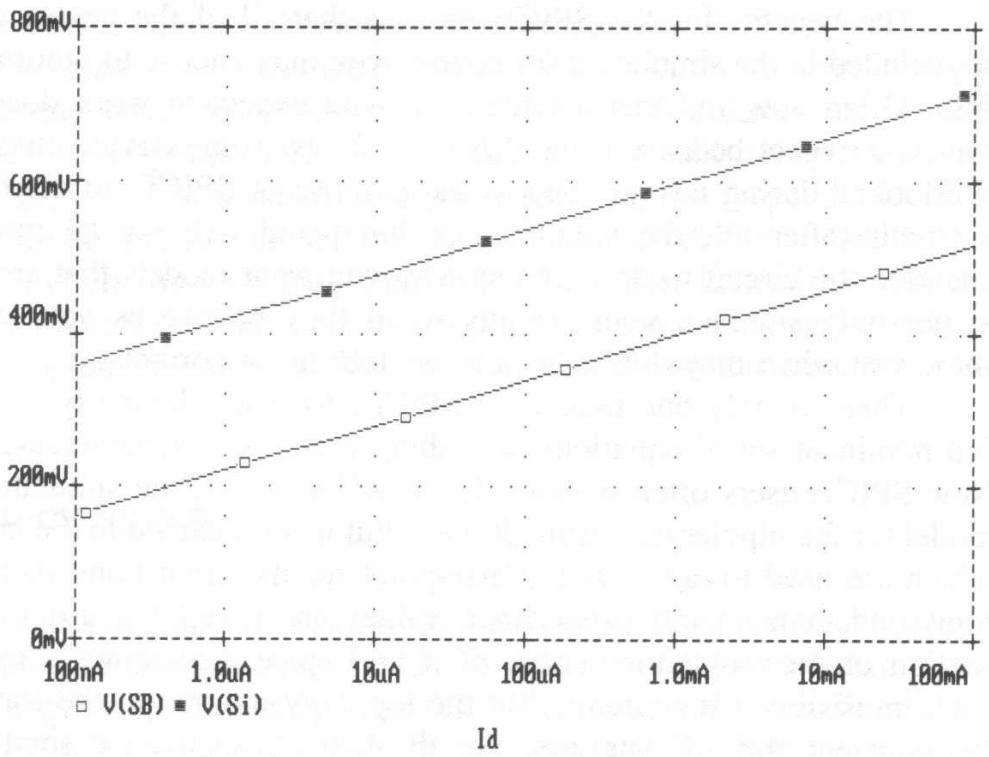


Figure 19.1 Device voltage vs. current with typical IS values for Schottky-barrier and diffused-junction diodes [11].

.

The diode has a series resistance (RS) parameter that controls the forward voltage drop at high currents. By adding RS one can limit the exponential effect of the equation and get a more reality like behaviour at high currents, see figure 19.2.

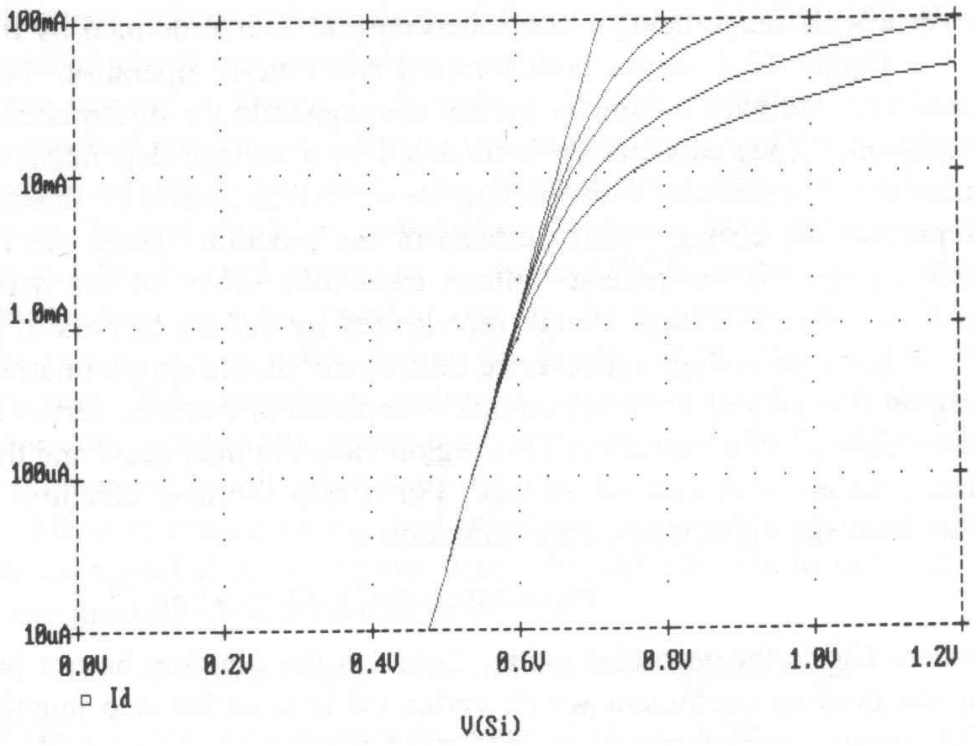


Figure 19.2 Device current vs. voltage with different values of RS [11].

A real diode has leakage across the junction in reverse operation, the Shockley equation gives to small value in reverse operation. To simulate the leakage, a conductance is connected in parallel, set by the parameter GMIN. See the figure 19.3, with and without the conductance.

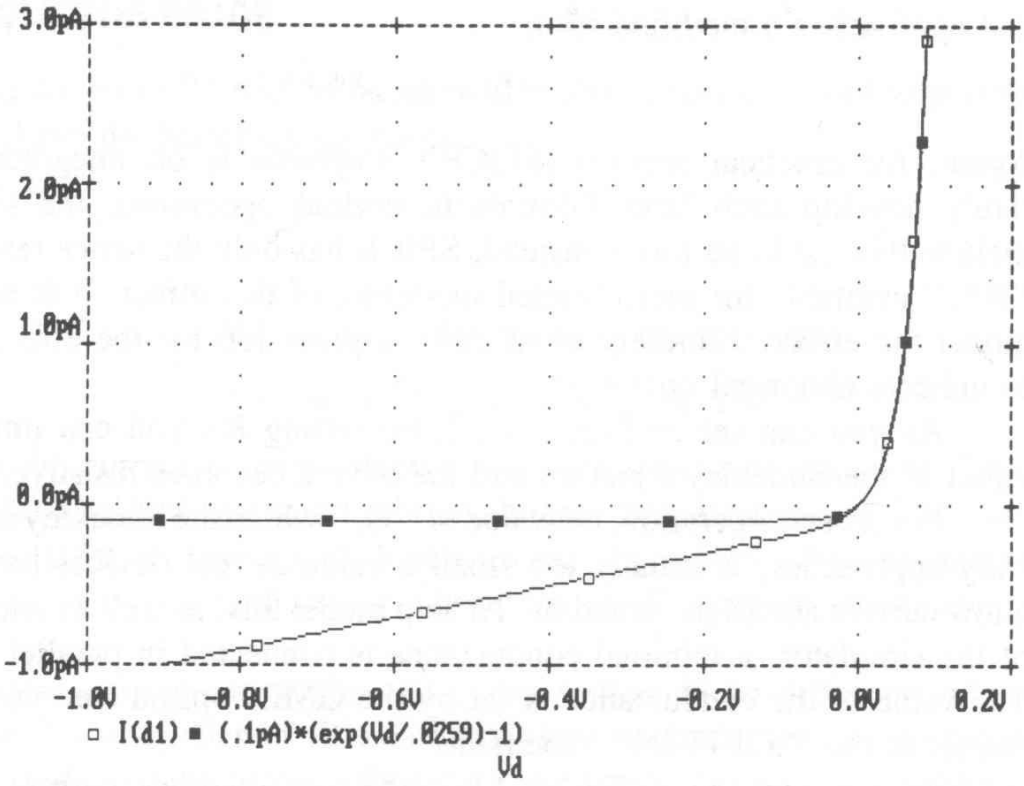


Figure 19.3 Reverse current and Schockley equation vs. voltage [11].

To simulate reverse breakdown and other types of diodes (zenerdiodes) another equation is used. See equation 19.2.

$$breakdown\_current = Ibv * e^{-(Vd+Bv)/Vt} \tag{19.2}$$

The equation adds a breakdown to the regular diode for negative voltage. This can be seen in the figure were the reverse current increase dramatically around minus five voltage, see figure 19.4.

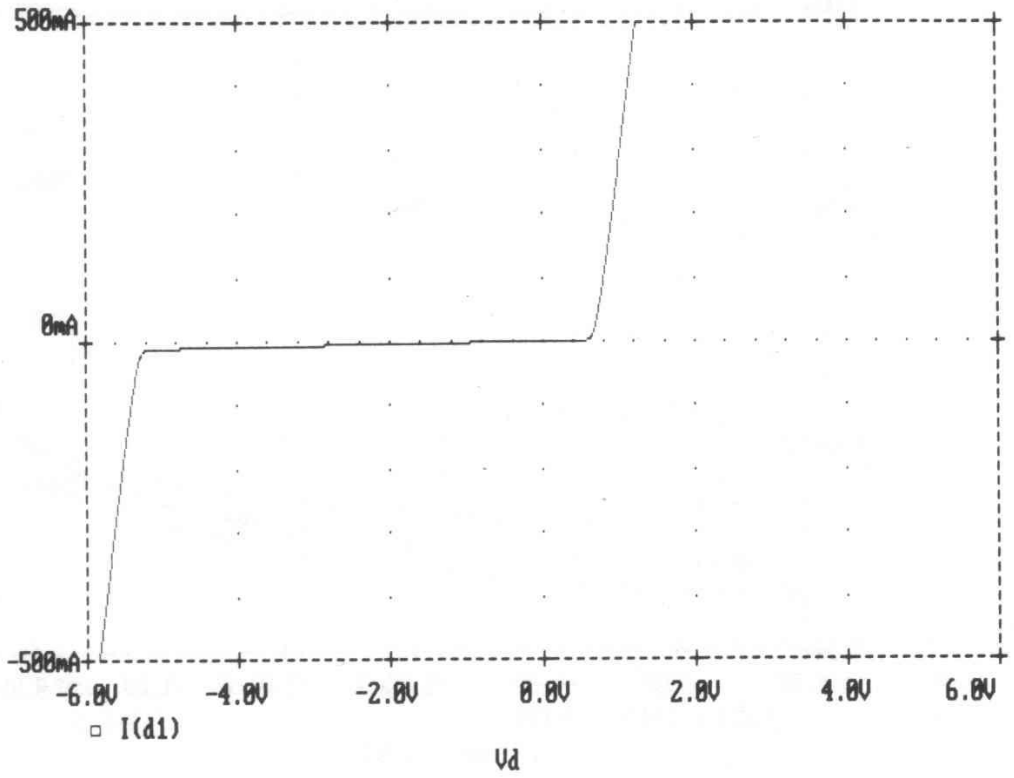


Figure 19.4 Device voltage vs. current [11].

To model the capacitance in the diode a voltage-dependent capacitor is connected in parallel with the current source (id). The capacitance can be divided in two components:

1. Reverse voltage capacitive effect of the depletion region.
2. Forward voltage charge represented by mobile carriers in the junction.

The reverse voltage capacitive effect:
This capacitance is an approximation of the area depleted in the junction and varies with the voltage see equation 19.3.

$$capacitace = Cjo/(1 - Vj/phi)^{M} \tag{19.3}$$

Where: Cjo is the zero bias value.
Phi is the junction barrier potential.
Vj is voltage across the intrinsic diode only.

M is the grading coefficient, which varies. If M has a value of 1/2 this will give a step like response (step junction), a value of 1/3 will give a linearly graded junction. Usually the value of M is some were in between.

By setting M to different values the slope will change as the reverse-bias capacitance change, see figure 19.5.

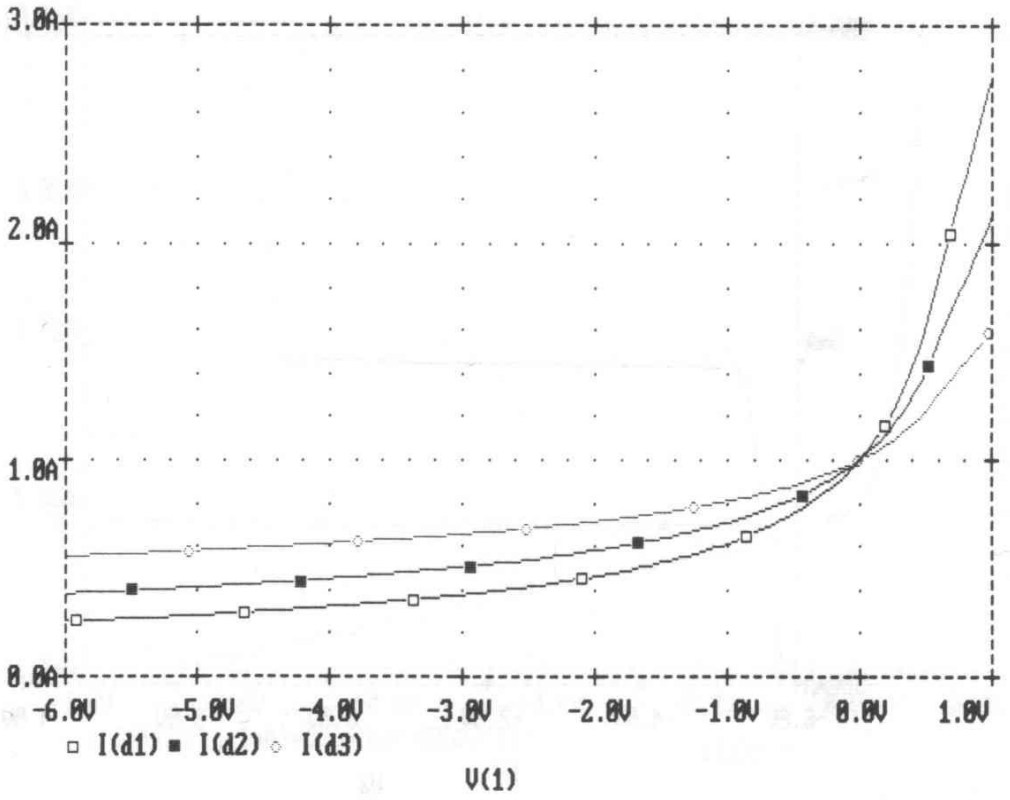


Figure 19.5 Junction capacitance vs. voltage [11].

The forward voltage charge:
When the diode is in forward biased the voltage charge varies with the current. This is modelled as a transportation time for the carriers to cross the junction. The forward voltage charge (diffusion charge) is the transportation time (TT)*device current (Id). The diode capacitance is its derivative with respect to the bias voltage see equation 19.4.

$$\text{capacitance} = \text{TT} * \frac{Is}{(\text{n} * \text{Vt})} * \text{e}^{\text{Vd}/(\text{n}*\text{Vt})} \tag{19.4}$$

This forward charge works as the storage time when the diode starts to block. When the current flows in the forward direction the junction first gets charged, when trying to change the current direction one first has to discharge the junction. The system can be solved to an equation for the transportation time TT parameter, see equation 19.5.

$$TT = \frac{storage\_time}{\ln\left((iF - iR)/-iR\right)} \tag{19.5}$$

The forward voltage charge dominates the reverse recovery characteristics, see figure 19.6. In the figure it is possible to see that at the last part the diode gets reversed biased and the reverse voltage capacitance causes a small tail.

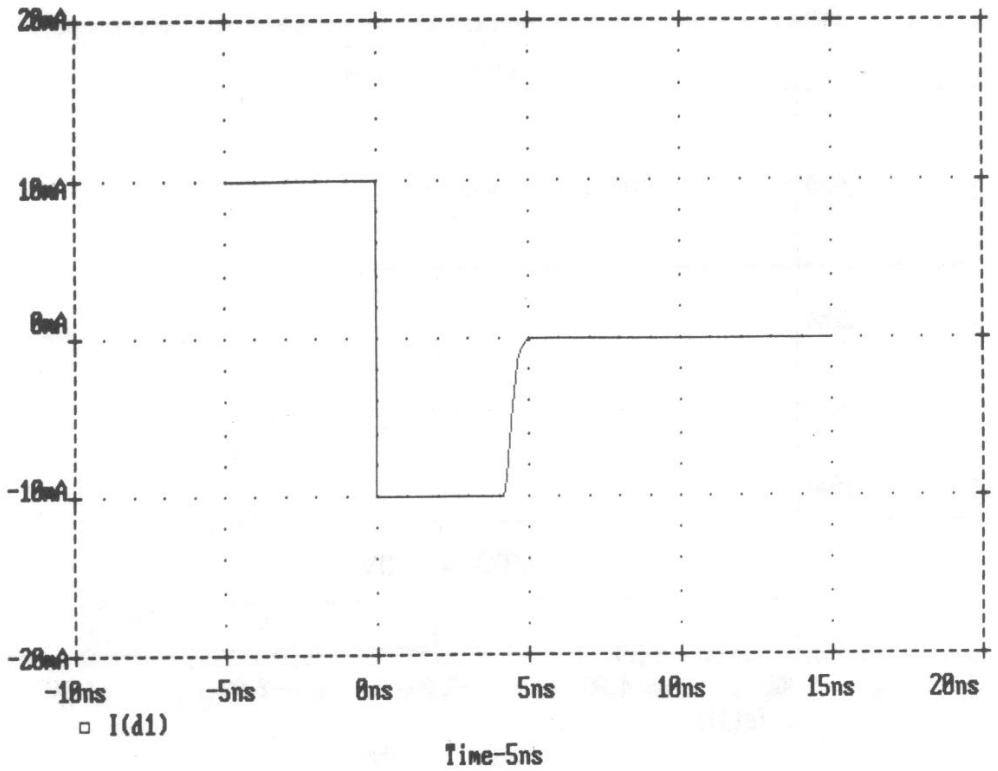


Figure 19.6 Reverse recovery current [11].

These parameters and equation are the most important in the PSpice model for the diode.

PSpice BJT model is an enhanced Gummel-Poon model. The PSpice model consist of several levels, were the activation of higher levels increase the complexitivity of the model. By using the default parameters one gets the lowest level of the model and by changing the parameters one can access the higher levels of the model.
Using default values the simplest Ebers-Moll DC model is simulated. With higher level of the model all of the junction capacitances follow. Right modelled it can give good AC and transient simulation result. Because both the Ebers-Moll and Gummel-Poon models are symmetrical there are forward and reverse parameters, some of them associated with base-emitter or base-collector junction. Many of the parameters are of the same types, with the different that one changes the forward characteristic and the other changes the reverse characteristic.

The simplest Ebers-Moll I-V curve can be seen in the figure 19.7.

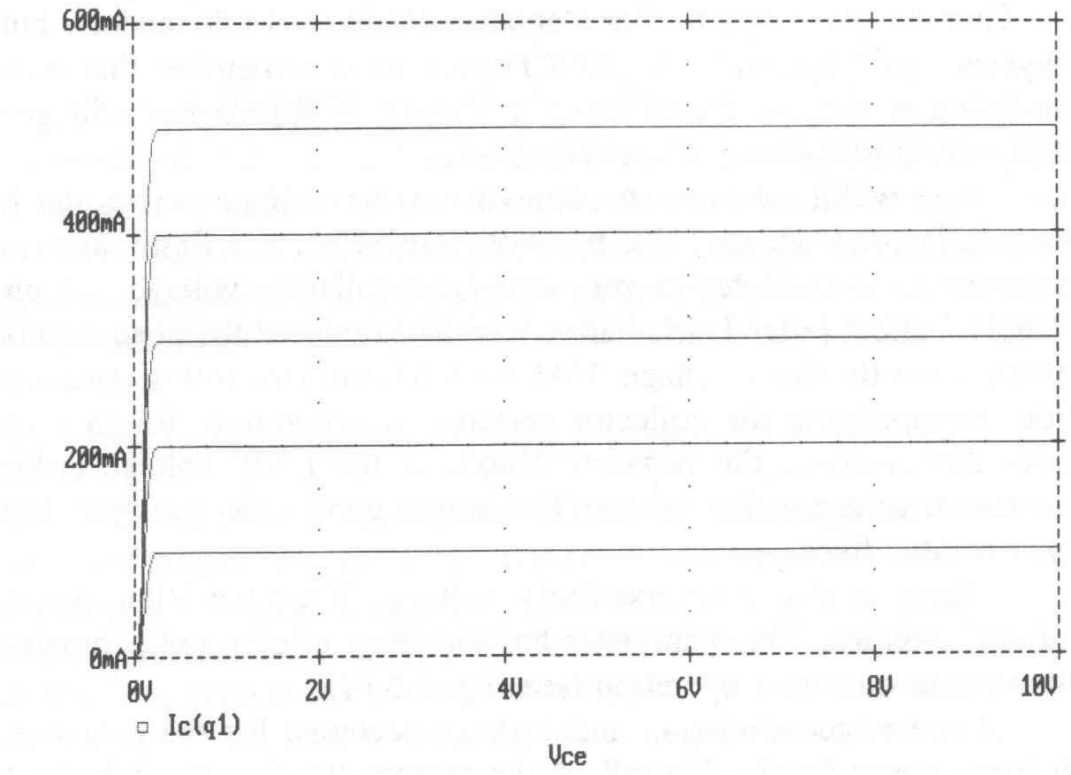


Figure 19.7 Collector current "curve family" for the bipolar transistor [11].

To get the next level of model the capacitances and parasitic effects for each terminal is added. By looking on the transistor model as two diodes (base-emitter and base-collector junctions) one can apply the same theory as described for the diode model. In this level the model simulates the frequency, transient and switching behaviour.

These are the parameters with its corresponding diode parameter.

- CJE and CJC, which are equivalent to CJO for the diode
- VJE and VJC, which are equivalent to VJ for the diode
- MJE and MJC, which are equivalent to M for the diode
- FC, which is for b-e and b-c junctions, and equivalent to FC for the diode (FC is the forward-bias depletion capacitance coefficient)
- TF and TR, which are equivalent to TT for the diode

Emitter (RE) and collector (RC) resistances model the terminal characteristics. These resistors decrease the slope of collector current for low collector-emitter voltages, and the base resistance (RB) changes the frequency response and noise dependency.

The last level of the model is the carrier recombination and base-width modulation that gives a reality like gain variation. The voltage across the base-emitter and base-collector junctions gives the base-width modulation. Finite output conductance is one obvious effect one other phenomena is the "Early" effect that regulates the collector current with the collector-base voltage. VAF is the parameter for the forward Early voltage, extrapolating the collector

currents (in saturation) forms converging lines that intersects with the negative X-axis at the forward Early voltage (expressed as positive, although it is negative), see figure 19.8.
VAR is the parameter for the reverse Early voltage, extrapolated in the same way as the forward Early voltage for reverse transistor operation.

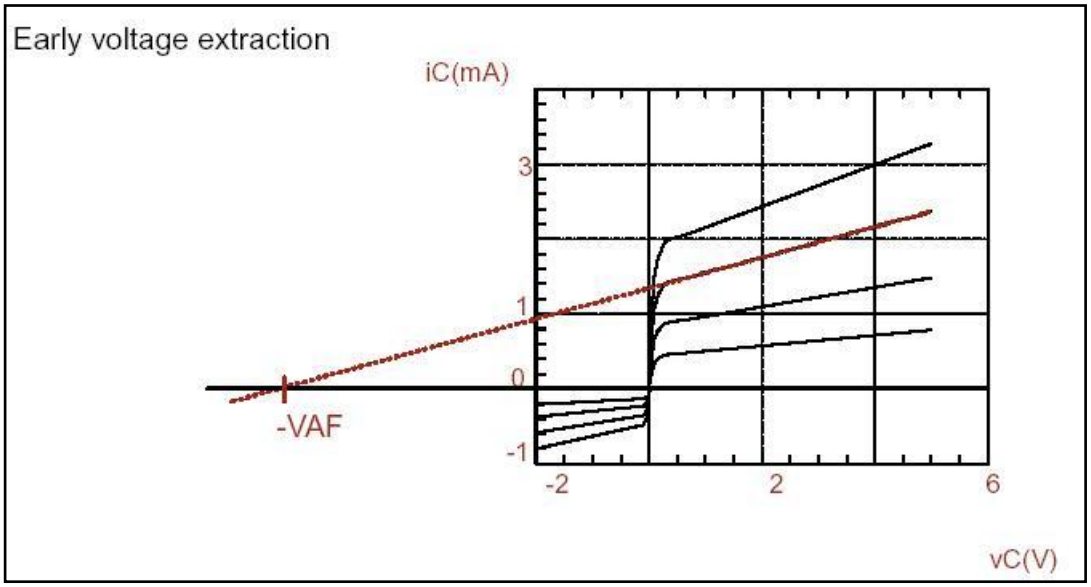


Figure 19.8 Showing the early voltage extraction [29].

The base current has two parasitic effects, base current leakage and recombination. The current that is not lost in these two ways participates in the amplification action of the transistor. The leakage (eq. 19.6) and recombination (eq. 19.7) currents are voltage dependent similar to the Shockley equation.

$$Leakage\_current = IS_L * \left(e^{V/(4*Vt)} - 1\right) \tag{19.6}$$

$$\mathrm{Re}combination\_current = IS_R * \left(e^{V/(2*Vt)} - 1\right) \tag{19.7}$$

In the Gummel-Poon model these equations arise from the non-ideal diode (leakage diode). The leakage diode is connected in parallel with the ideal diode. The ideal diode current is the current that participate in the transistor gain, this current is multiplied by beta to generate the collector current. The non-ideal current does not participate in the transistor action. The two non-ideal equations are simplified to one non-ideal diode, see equation 19.8.

$$Leakage\_diode\_current = ISE * \left(e^{V/(NE*Vt)} - 1\right) \tag{19.8}$$

Where: ISE is the saturation current for the non-ideal base-emitter diode.
NE is the emission coefficient for the non-ideal base-emitter diode.

And for the ideal diode the equation is:

$$Ideal\_diode\_current = IS * \left(e^{V/(NF*Vt)} - 1\right) \tag{19.9}$$

Where: IS is the saturation current for the ideal base-emitter diode.
NF is the emission coefficient for the ideal base-emitter diode.

Because there are two diodes in the model at a certain point there will flow equal current in them, when this happens the gain in the transistor will be reduced to half.

The effects of ISE:
By sweeping the value of ISE and base current, one can see that the ISE parameter sets the point for the reduction, see figure 19.9 and figure 19.10.

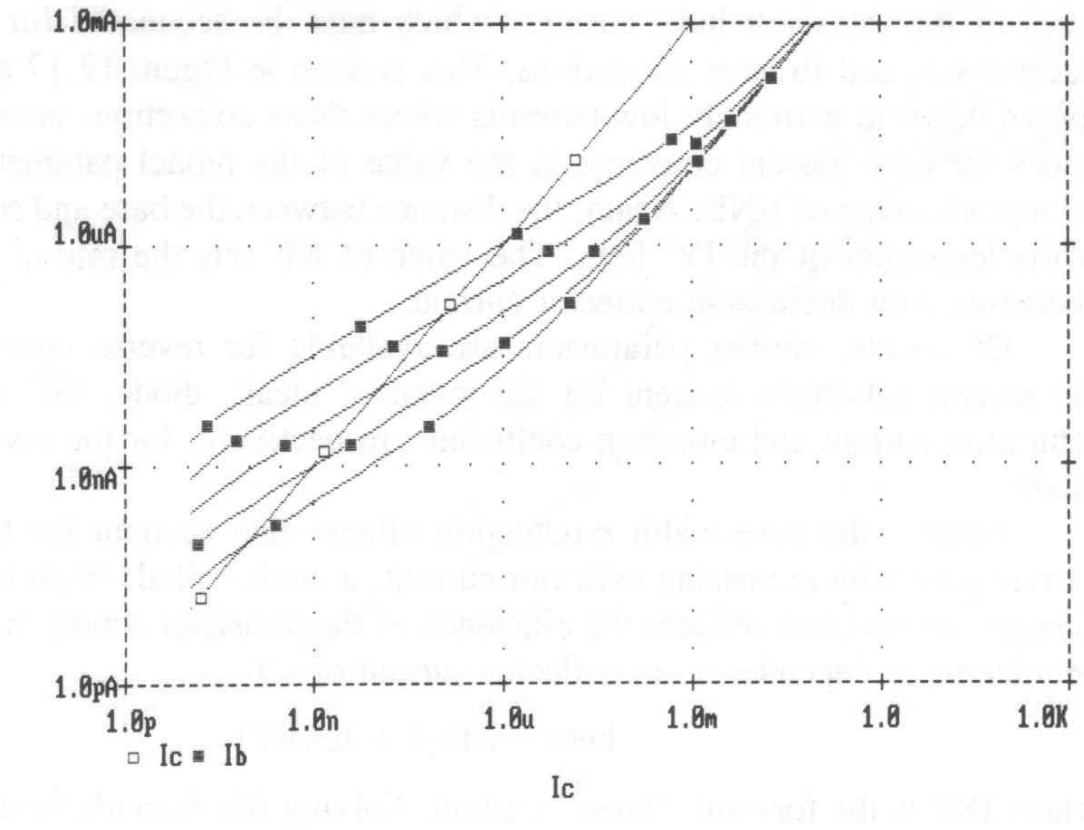


Figure 19.9 Transistor currents, varying ISE parameter [11].

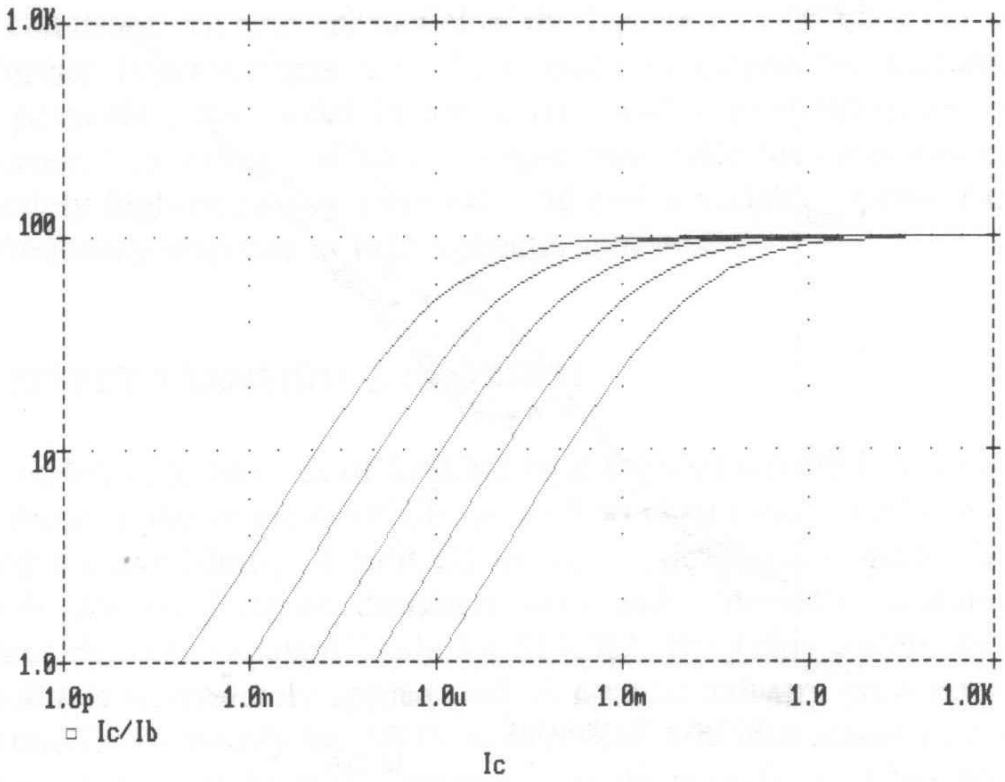


Figure 19.10 Transistor DC beta, varying ISE parameter [11].

The effects of NE:
By sweeping the value of NE and the base current, one can see that the NE parameter sets the rate at which DC beta decreases with decreasing collector current, see figure 19.11.

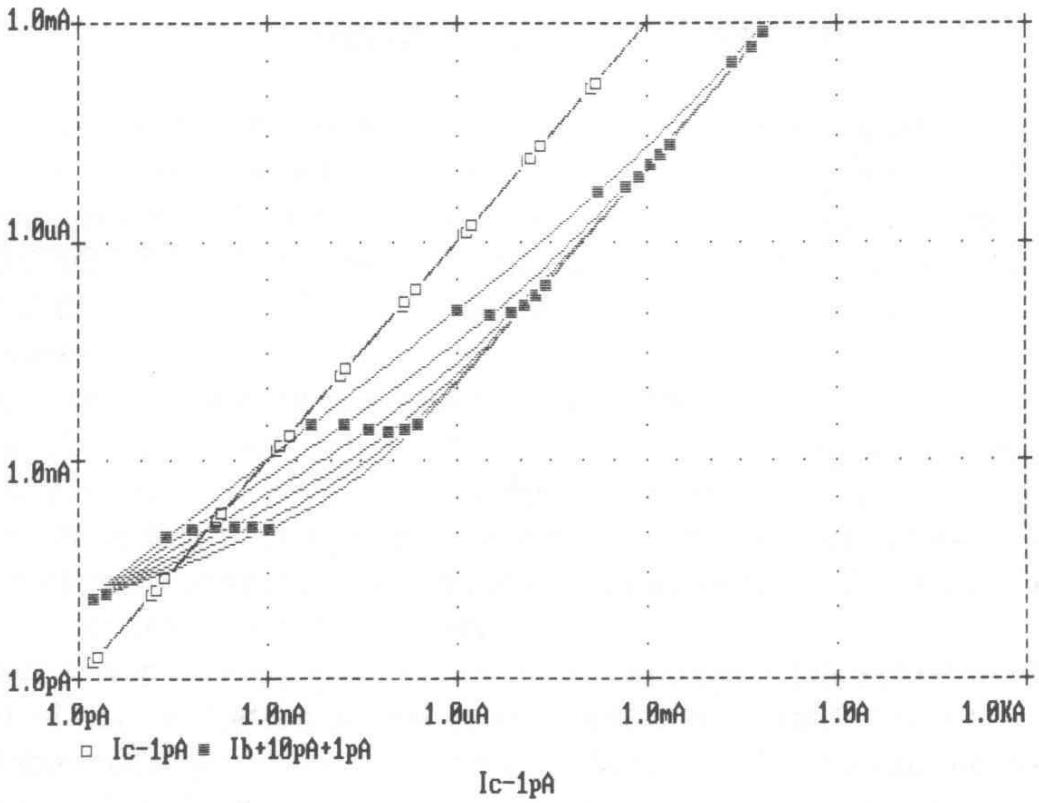


Figure 19.11 Transistor current, varying the NE parameter [11].

There are similar effects for the reverse operation (the value of ISC and NR).

The base-width modulation effects also accounts for the reduction in current gain with increasing collector current (high-level injection). The efficiency of the transistor is reduced by charge conservation at high current. Beta has a dependency on the collector current according to equation 19.10.

$$beta=BF/(1+Ic/IKF) \qquad (19.10)$$

IKF is the forward "knee" current. DC beta is the ratio of Ic to Ib, in the figure 19.12 the vertical distance between Ic and Ib. At low currents beta is in unity, at high currents beta changes with changing IKF. There is a similar parameter IKR for the reverse operation.

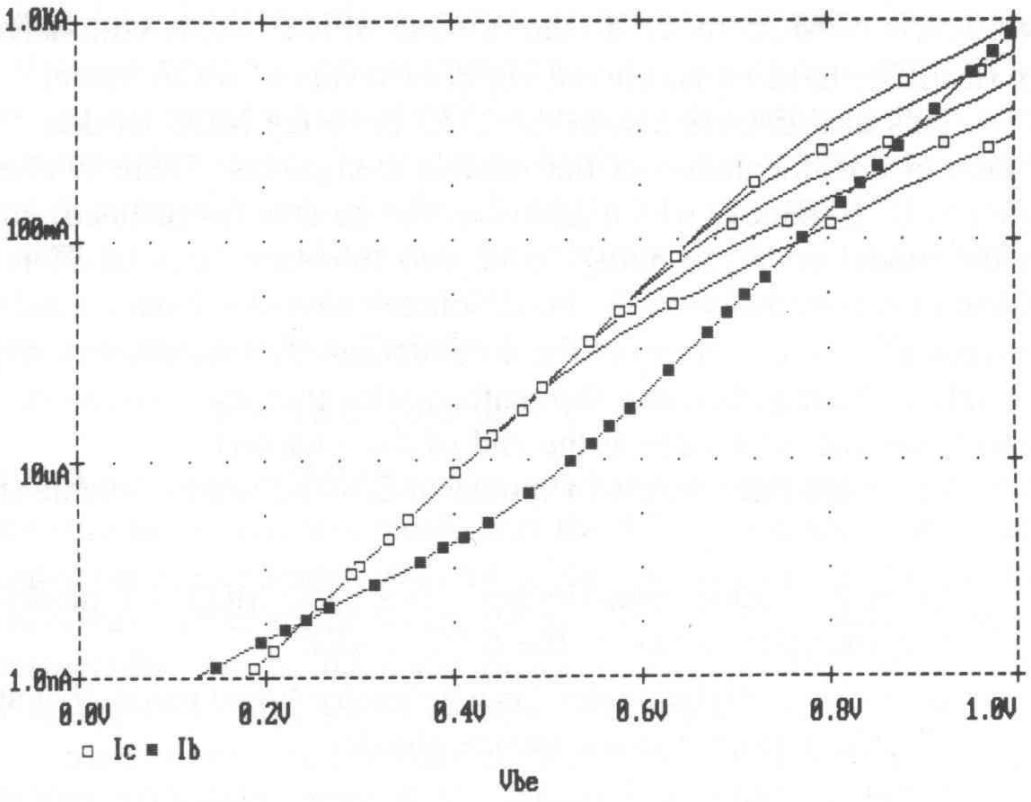


Figure 19.12 Transistor current, varying the IKF parameter [11].

20. Appendix G Measuring model parameters

The first BitSiC from TranSiC does not represent the later samples from TranSiC. This transistor has high contact resistance. The parameters that is going to be measured is VAF, Is, Bf, Br, Cje, Cjc. The methods for extracting different parameters are read from the book “Principles of semiconductor devices” [30] and from “Gummel-Poon Bipolar model description parameterextraction” [29].

20.1.1. I-V curve

TranSiC had at that time not an instrument to measure I-V curves for high currents but they had ordered one. Because the limited access to instruments at IEA contacts is made with the other faculties to see what types of instruments they has. After a lot of searching a semiconductor parameter analyzer (HP4145B) was found at the Electroscience’s (ES) radiolab, the analysator is intended for small signal transistors only. The next idea to get I-V curves is to use National Instruments (NI) LabVIEW and with external power supplies get the correct currents. With LabVIEW different curve traces is made to get an idea of how the transistor works. The data from the LabVIEW program is analysed with MATLAB to plot the I-V curves, see figure 20.1.

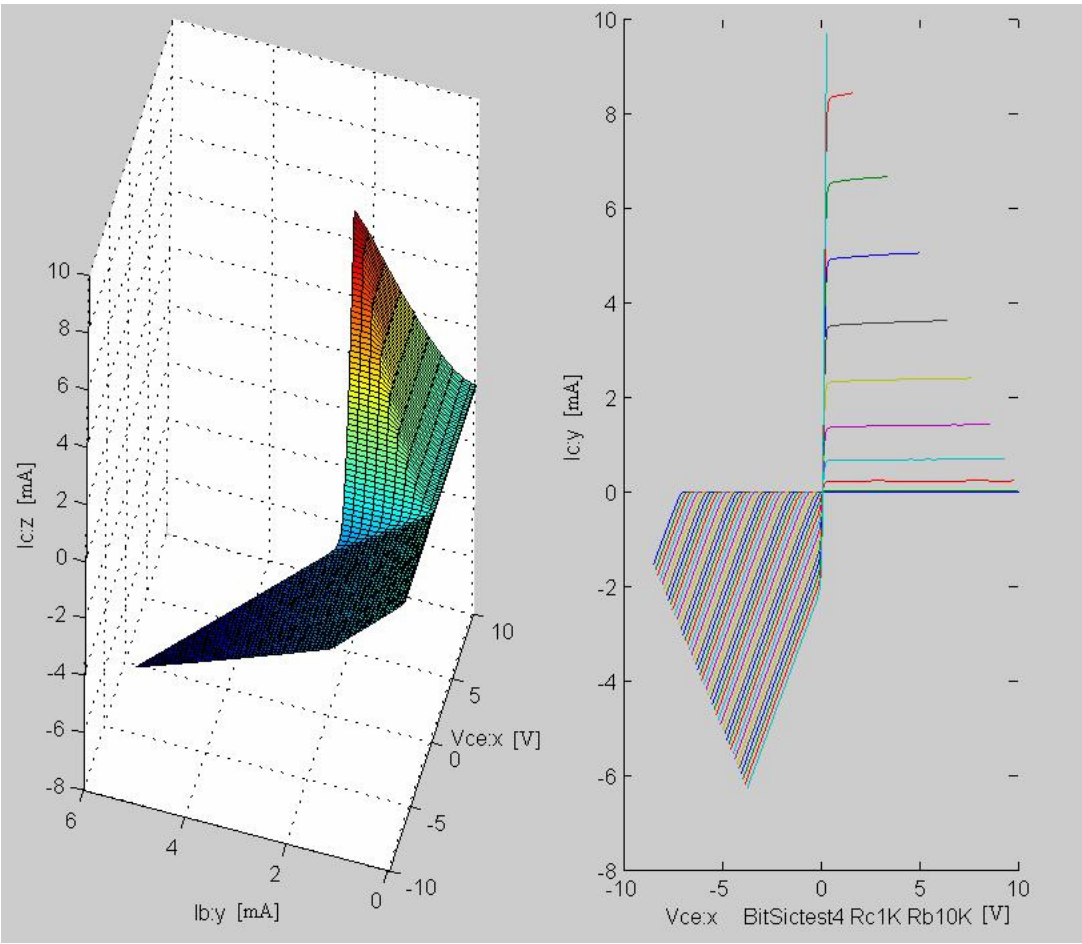


Figure 20.1 I-V curve made with LabVIEW and MATLAB.

20.1.2. The Early effect

With the I-V data the VAF parameter can be calculated using two data points, where the collector currents increases from Ic_1 to Ic_2 and the collector-emitter voltage is increased from Vce1 to Vce2. Looking in figure 19.8, the early effect is the reverse voltage of the base collector junction (Vbc=-Vcb). Practically the difference between Vcb and Vce is Vbe voltage, therefore an increase in Ic with Vce is the same increase in Vcb.

Two equations:

$$Ic_1 = Ic_{(0)}\left(1+\frac{|V_{CB1}|}{V_A}\right) \tag{20.1}$$

$$Ic_2 = Ic_{(0)}\left(1+\frac{|V_{CB2}|}{V_A}\right) \tag{20.2}$$

Where the Ic(0) is the starting current and $V_{CB} = V_{CE} - V_{BE}$, (in the calculations just use V_{CE}).

Then the solution can be written as:

$$V_{AF} = \frac{\left(V_{CB2} - V_{CB1}\frac{I_{C2}}{I_{C1}}\right)}{\left(\frac{I_{C2}}{I_{C1}} - 1\right)} \tag{20.3}$$

With data from the I-V measurement the early voltage will be:

$$V_{AF} = \frac{\left(6.201 - 1.06\frac{2.399}{2.34}\right)}{\left(\frac{2.399}{2.34} - 1\right)} \approx 202.84\ \text{V} \tag{20.4}$$

The Is parameter can be calculated from the equation:

$$I_C = I_S e^{V_{BR}/V_t} \tag{20.5}$$

See the book "Principles of semiconductor devices" by Sima Dimitrijev [30] for more detailed information. From this equation the Is parameter can be calculated as:

$$I_S = e^{\ln(I_C) - V_{BE}/V_t} \tag{20.6}$$

Where $V_t = k * \frac{T}{q}$ and is constant at constant temperature.
With the numbers inserted the Is parameter is:

$$I_S = e^{\ln(2.399*10^{-3}) - 2.754/0.02585} \approx -1.29 * 10^{-49} \tag{20.7}$$

At first this value seems to be low, however according to Martin Domeij [20] at TranSiC it is reasonable.

20.1.3. Bf and Br.

To calculate the static gain of the BitSiC is one of the easiest parameter to derive.
The data collected is shown in table 20.1.

Table 20.1. Showing collected data for BitSiC 1.

Vbe	Vbc	Ib	Ic
2.841 V	-6.15 V	1.565 mA	9.08 mA
-9.22 V	2.578 V	0.155 mA	0.22 mA

The Bf parameter can be calculated from the equation 20.8:

$$\beta_F = \frac{I_C}{I_B} = \frac{9.08}{1.565} \approx 5.80 \tag{20.8}$$

Br.

$$\beta_r = \frac{I_C - I_B}{I_B} = \frac{0.22 - 0.155}{0.155} \approx 0.419 \tag{20.9}$$

The low gain has to do to the contact resistance of the first sample of the BitSiC.

See the book **"Principles of semiconductor devices" by Sima Dimitrijev** [30] for more detailed information.

20.1.4. Junction capacitance

With a multimeter the different capacitances for base-emitter and base-collector are measured. When measuring the base-emitter capacitance the collector is left open, see figure 20.2.

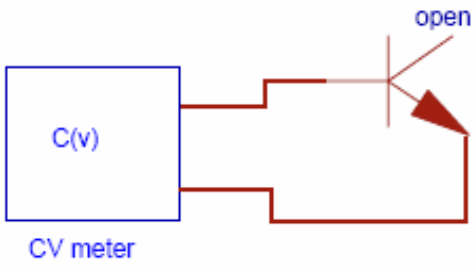


Figure 20.2 The way of measure the base emitter capacitance [29].

Although just measuring on the base and emitter connections there are parasitic effects has to be taken in consideration. When measuring the base – emitter capacitance the measurement will be an overlay of the other capacitances also, see figure 20.3.

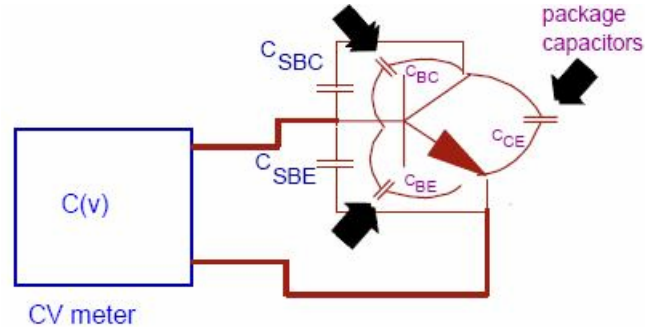


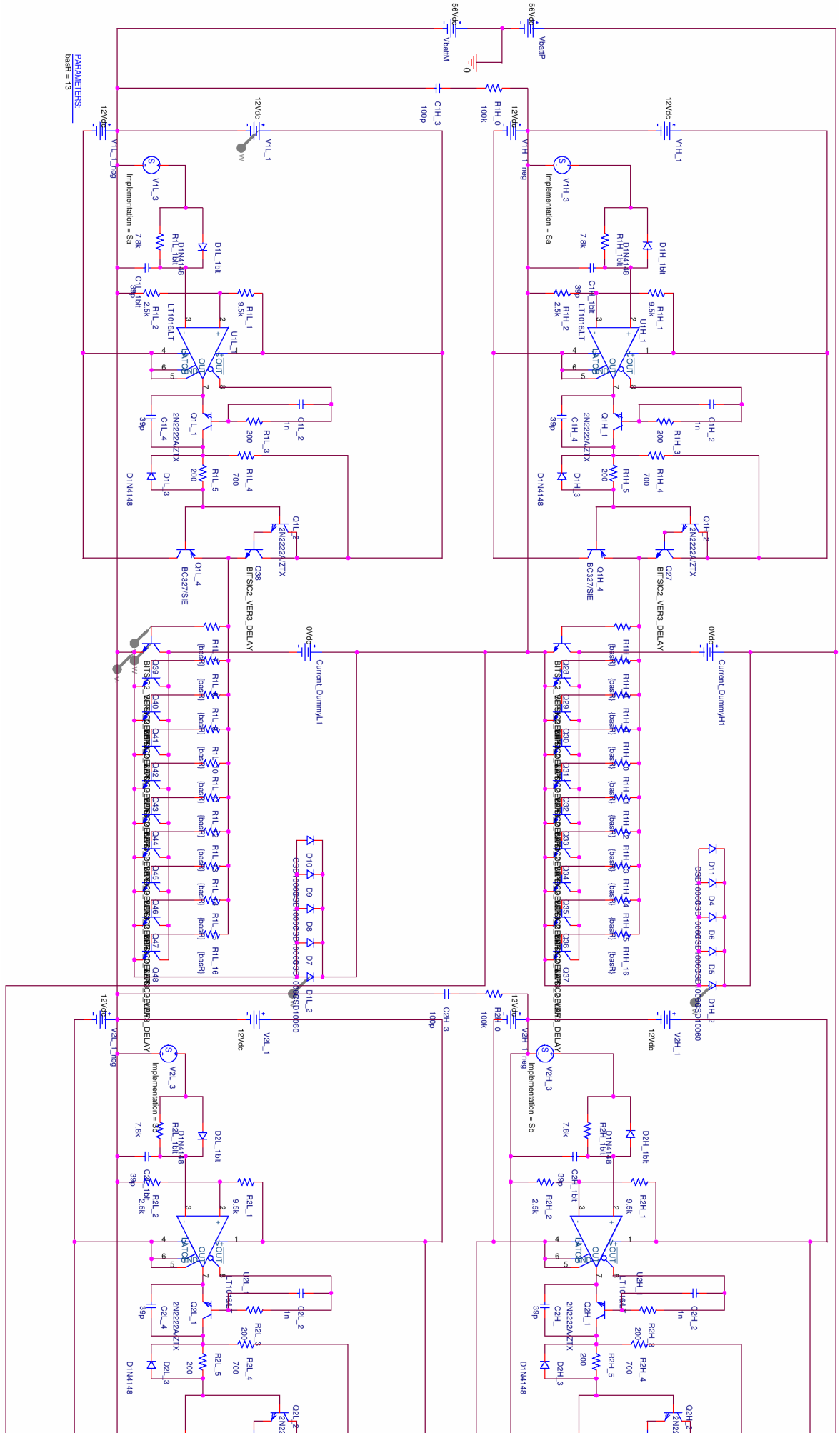
Figure 20.3. The way to measure the parasitic capacitance [29].

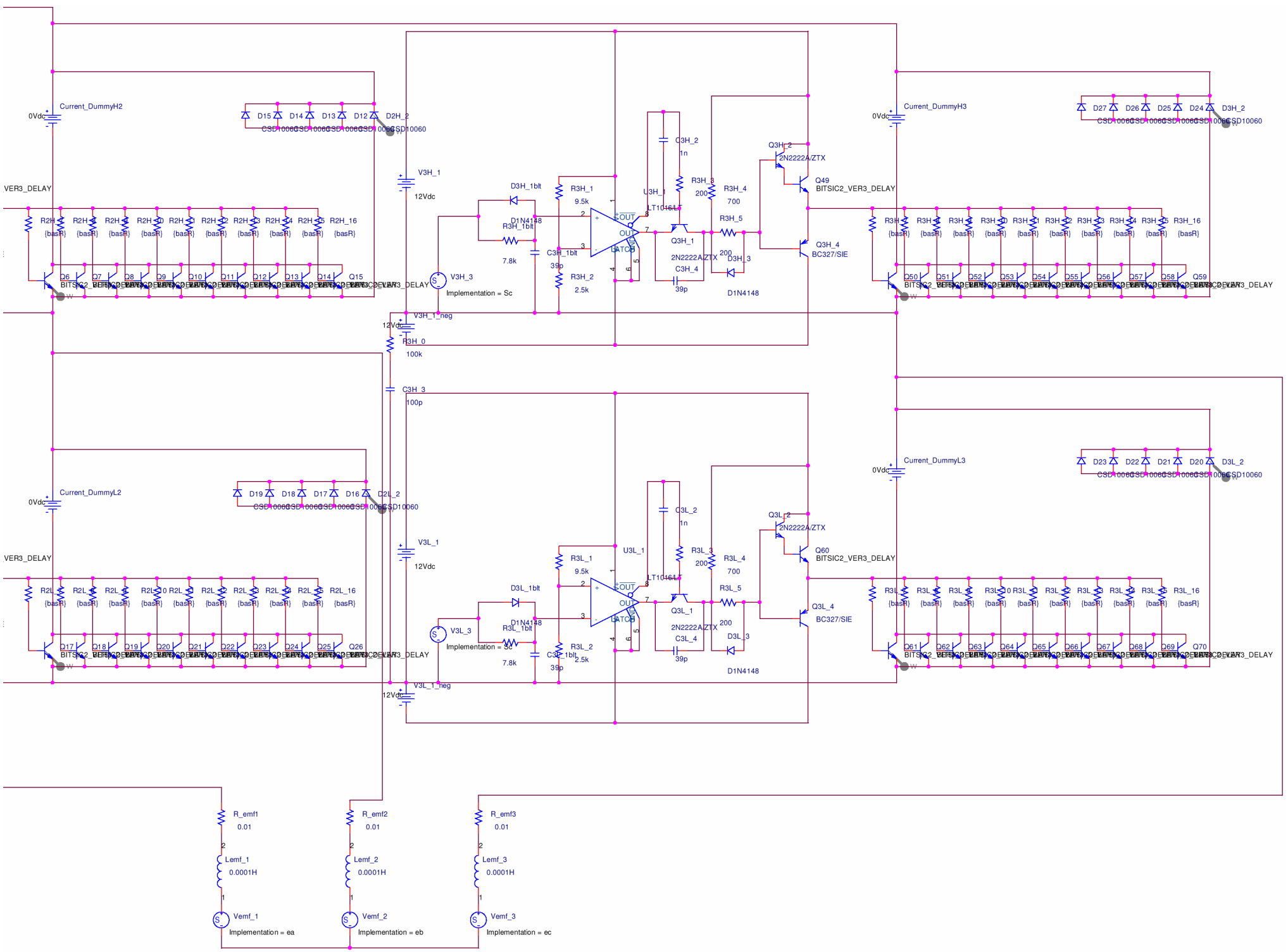
This means that the measurement will always be too big, this effect is limited a bit by connecting the open BJT connection to ground.

Measurement result with a FLUKE 189

- Base-emitter capacitance is 1.5nF
- Base-collector capacitance is 0.555nF
- Collector-emitter capacitance (package capacitance) is 0.72nF

21. Appendix H - The full three-phase inverter




Current_DummyH2
0Vdc
D15 D14 D13 D12 D2H_2
CSD1006G
V3H_1
12Vdc
D3H_1blt
R3H_1
9.5k
R3H_1blt
7.8k
D1N4148
C3H_1blt
39p
R3H_2
2.5k
U3H_1
LT1016LT
C3H_2
1n
R3H_3
200
R3H_4
700
R3H_5
200
Q3H_1
2N2222A/ZTX
C3H_4
39p
Q3H_2
2N2222A/ZTX
Q49
BITSIC2_VER3_DELAY
Q3H_4
BC327/SIE
D3H_3
D1N4148
R2H_1 R2H_2 R2H_3 R2H_4 R2H_5 R2H_6 R2H_7 R2H_8 R2H_9 R2H_10 R2H_11 R2H_12 R2H_13 R2H_14 R2H_15 R2H_16
{basR}
Q6 Q7 Q8 Q9 Q10 Q11 Q12 Q13 Q14 Q15
V3H_3
Implementation = Sc
V3H_1_neg
12Vdc
R3H_0
100k
C3H_3
100p
Current_DummyH3
0Vdc
D27 D26 D25 D24 D3H_2
R3H_1 R3H_2 R3H_3 R3H_4 R3H_5 R3H_6 R3H_7 R3H_8 R3H_9 R3H_10 R3H_11 R3H_12 R3H_13 R3H_14 R3H_15 R3H_16
Q50 Q51 Q52 Q53 Q54 Q55 Q56 Q57 Q58 Q59
Current_DummyL2
0Vdc
D19 D18 D17 D16 D2L_2
V3L_1
12Vdc
D3L_1blt
R3L_1
9.5k
R3L_1blt
7.8k
D1N4148
C3L_1blt
39p
R3L_2
2.5k
U3L_1
LT1016LT
C3L_2
1n
R3L_3
200
R3L_4
700
R3L_5
200
Q3L_1
2N2222A/ZTX
C3L_4
39p
Q3L_2
2N2222A/ZTX
Q60
BITSIC2_VER3_DELAY
Q3L_4
BC327/SIE
D3L_3
D1N4148
R2L_1 R2L_2 R2L_3 R2L_4 R2L_5 R2L_6 R2L_7 R2L_8 R2L_9 R2L_10 R2L_11 R2L_12 R2L_13 R2L_14 R2L_15 R2L_16
Q17 Q18 Q19 Q20 Q21 Q22 Q23 Q24 Q25 Q26
V3L_3
Implementation = Sc
V3L_1_neg
12Vdc
Current_DummyL3
0Vdc
D23 D22 D21 D20 D3L_2
R3L_1 R3L_2 R3L_3 R3L_4 R3L_5 R3L_6 R3L_7 R3L_8 R3L_9 R3L_10 R3L_11 R3L_12 R3L_13 R3L_14 R3L_15 R3L_16
Q61 Q62 Q63 Q64 Q65 Q66 Q67 Q68 Q69 Q70
R_emf1
0.01
Lemf_1
0.0001H
Vemf_1
Implementation = ea
R_emf2
0.01
Lemf_2
0.0001H
Vemf_2
Implementation = eb
R_emf3
0.01
Lemf_3
0.0001H
Vemf_3
Implementation = ec

22. Appendix I - Power and efficiency

Power [W]	5 kHz	25 kHz	100kHz
** Battery Power **			
Power_WBatt_m	-1603.5	-2457.1	-2343.0
Power_WBatt_p	-1615	-2468.8	-2355.1
Power_batt_tot	-3218.5	-4925.9	-4698.1

** Transistor Power High **			
Power_wQ1H_5	14.533	30.103	65.157
Power_wQ2H_5	25.182	28.687	44.946
Power_wQ3H_5	7.8495	27.953	43.265

** Transistor Power Low **			
Power_wQ1L_5	13.01	26.251	33.49
Power_wQ2L5	8.6796	28.638	59.234
Power_wQ3L_5	22.894	29.978	60.074

** Diode Power **			
Power_wDH1	1.4231	6.6968	1.9408
Power_wDL1	1.3536	8.3623	4.2146
Power_wDH2	1.4231	6.6968	1.9408
Power_wDL2	0.45556	8.142	2.8868
Power_wDH3	3.041	10.635	3.8188
Power_wDL3	0.31452	6.7729	2.6218

** Driver Power **			
Power_wvh1	-21.572	-21.911	-20.421
Power_wvh	-64.715	-65.734	-61.264
Power_wvl1	-21.924	-22.658	-20.803
Power_wvl	-65.771	-67.975	-62.408

** Load Power **			
Power_Load1	1030.4	1600	1414.7
Power_Load2	1131.5	1570	1484.2
Power_Load3	991.85	1566.8	1508.1
Power_Load_tot	3153.8	4736.7	4407

** Tot Power **			
Power_batt_tot	-3218.5	-4925.9	-4698.1
Power_Trans	92.148	171.61	306.17
Power_Diode	8.0108	47.306	87.118
Power_Load_tot	3153.8	4736.7	4407
Power_driver_tot	-130.49	-133.71	-123.67
Power_netto	29.09	218.91	102.09
Efficiency_with_driver	0.942	0.936	0.914
Efficiency_without_driver	0.980	0.962	0.918