Rapid Control Prototyping in a 3-phase grid-tie BSNPC inverter

Sensors, PWM and control concepts

Orestis Polychronakis

Delft University of Technology

TU Delft Delft University of Technology

Challenge the future

Rapid Control Prototyping in a 3-phase grid-tie BSNPC inverter

Sensors, PWM and control concepts

By

Orestis Polychronakis

in partial fulfilment of the requirements for the degree of

Master of Science
in Electrical Engineering

at the Delft University of Technology,
to be defended publicly on Friday October 18, 2013 at 12:00 PM.

Supervisor:	Dr. P. Bauer	
Thesis committee:	Dr. P. Bauer,	TU Delft
	Prof. Dr. J.A. Ferreira,	TU Delft
	Dr. O. Isabella,	TU Delft

This thesis is confidential and cannot be made public until October 19, 2013.

TU Delft Delft University of Technology

Preface

In these ages of a more and more increasing energy demand and struggle to reduce CO2 emis- sions, the need of robust, analytical and fast ways to introduce new alternative energy concepts into practice is imminent. Nowadays, engineers struggle to incorporate all parts of modern power electronic applications, such as new topologies, control concepts, supportive features, into compact structures and that require the joining of forces between different sectors of the electrical engineering field. Digital technology, new materials and new control ideas should be combined to produce what is envisioned by new engineers as high end technology for renewable energy systems. One such way to simplify the cooperation of different electrical engineering fields is Rapid Control Prototyping (RCP). In RCP applications, a plant controller is implemented using a real-time simulator and is connected to a physical plant having many advantages over implementing an actual controller prototype. A controller prototype developed using a real-time simulator is more flexible, faster to implement and easier to debug. The project considered in this master thesis involves a Rapid Control Prototyping system for a three-phase grid-tie BSNPC IGBT inverter. A Rapid Control Prototyping system, as mentioned before, includes a computer that runs the Simulink software, a Real Time Computer that runs a compiled version of the Simulink model and the actual hardware. The hardware in the existing case comprises a custom board designed and manufactured especially for this master thesis project and already existing inverter hardware. The custom board acts as an intermediate between the controlling computers and the hardware. It receives and sends signals to and from the Real Time Computer as well as the inverter hardware, is capable of fully operating the hardware and providing the controlling computers with full data and information about the hardware. The data transmissions between the systems are done real-time while the inverter hard- ware is in operation and the editing and producing of new data is also done in real-time. The data processing inside the two controlling computers is also done in real-time and permits the testing and operation of the hardware simultaneously with its change in control variables, gains and duty cycles. The ease of operation of such a system and the full controllability and flexibility makes it perfect for testing in power electronic applications.

Orestis Polychronakis
Delft, October 2013

Contents

1

Introduction

1.1. Foreword

The increase of energy demand in the industrial and domestic environment of current society structures is boosting the market and research attention toward distributed power generation systems based on renewable energy sources. More and more educational centres, energy companies and governments are striving to reduce the amount of fossil fuel consumption in order to reduce CO2 emissions, avoid the growth of electrical energy generation based on nuclear energy and lean to alternative and more clean sources of energy. The need of robust, analytical and fast ways to introduce new concepts into practice is imminent. Nowadays, engineers struggle to incorporate all parts of modern power electronic applications, such as new topologies, control concepts, supportive features, into compact structures and that require the joining of forces between different sectors of the electrical engineering field. Digital technology, new materials and new control ideas should be combined to produce what is envisioned by new engineers as high end technology for renewable energy systems.

The urge to digest and interpret the dynamics and instabilities of power transmission networks boosted the development of some early computers like the Massachusetts Institute of Technology's Differential Analyzer from late 1920s [1]. That was an example of an advanced real-time computing device used operationally. Nowadays, the need to understand and control the modern power grid systems with an ever-increasing number of power- electronics enabled devices calls for innovation in high-speed, low-latency, modular, and flexible control and verification technologies. Traditionally, power grids were designed hierarchically and with a one-way power direction. The need to integrate more renewable energy sources on all levels of the power grid is leading to system complexity increase that can be managed only by applying advanced sensor, real-time computation, and communication in the smart grid of the future.

One of the key physical layers of the aforementioned issues are power electronic systems that provide an intelligent and bidirectional power interface between the grid and possible renewable sources, such as wind, solar, fuel cell etc., as well as its intelligent loads, such as variable speed motor drives, hybrid and electric vehicles, uninterruptable power supplies, energy storage systems, electric trains, etc.

1.2. Previous research and purpose of this thesis

Previous research

The design, verification, and system integration of these advanced grid concepts call for the application of new real-time Rapid Control Prototyping (RCP) and Hardware In the Loop (HIL) simulation tools both on the level of the power grid [2], [3] as well as on the power electronic system and device level [4]–[7]. Multiple examples of grid-level digital real-time simulator tools exist based on the off-the-shelf high-

performance processors and other software and hardware components [3]–[6]. That systems can simulate in real time the transient and subtransient behaviour of the power grid, fault conditions, issues concerning the islanding behaviour of the parts of the grid, and more.

Unlike in power systems, where the switching events occur asynchronously, high frequency switching is the normal operation mode for power electronic converters. The speed of fast signals and the tight coupling between control and protection and power hardware combined, make it extremely difficult and time consuming to test power electronic control hardware, firmware, and software. This is one of the main reasons why real-time simulation tools tend to become an important tool for power electronics control design and testing. To satisfy the requirements for real-time digital simulation of converters, developers have a choice to use low-voltage versions of real converters or hard-coded converter models on field-programmable gate array (FPGA) platforms. For example, in [8], the reduced scale model of the system is used, which is the standard practice in industry for high-power systems. In [9]–[11], developing a power electronic converter that can simulate an electric motor demands a lot of work, while in [12], multi MW power amplifiers are used. The common element of these approaches is that they are time consuming to set up, they are inflexible, labour intensive to use, and expensive, which makes Rapid Control Prototyping and Hardware In the Loop a saving tool.

Due to the demanding requirements regarding the simulator speed and latency, it has already been admitted that only FPGA technology provides the combination of very low latency and massively parallel processing required for Rapid Control Prototyping and Hardware In the Loop for power electronic designs. However, the main problem is that developing simulation models on a FPGA requires extensive knowledge of the digital design and computer architecture design, FPGA design, and verification tool chain as well as detailed understanding of the power electronic modelling techniques. Still, the potential of FPGA technology has been recognized, and the solutions based on this technology are actively studied. In [13], a very fast FPGA-based model of a permanent magnet motor drive is presented, and in [14] and [15], the authors use FPGA technology in order to model two and three level motor drives with the behavioural models of IGBT devices. In [16], FPGA technology is used for the simulation of multicell multilevel converters, while in [17], a flexible programmable FPGA environment for the emulation of power electronic hardware with a 1-μs time step is introduced. Power electronic systems can be well modelled as switched hybrid systems described with a set of discrete states, each with its own continuous dynamics. There are two main approaches to simulating power electronic circuits, namely, “constant topology” and “varying topology” approaches [18]–[20]. Either way, firing pulses coming from the real-life controller are rarely in synchronism with the time step of the simulator, and the delay can be up to one sampling cycle of the simulator. In addition, multiple switching events can happen within one sampling cycle of the simulator. This problem has been recognized, and a rich set of techniques to minimize the errors was studied in [21] and further expanded in [22] by introducing the adaptive discretization technique. In [23], after a comprehensive review of the techniques for real time simulation, it is indicated that, from the simulation point of view, the synchronous oversampling is desirable but requires a $< 5\mu s$ sampling time for systems with a carrier frequency on the order of 20 kHz. They also comment that such a low latency is out of reach of today’s real-time processors and requires FPGA technology even though its tool chain is not adequate for power electronic applications.

Commercially available RTS hardware platforms rely on the existing commercially available processors (usually ×86 with real-time operating system) for calculation and on FPGAs for input/output interfaces. They are configured as hardware accelerators for online simulation tools, e.g., Matlab, Simulink, and as such, they offer impressive performance (compared to offline simulators) as well as ease of use and scalability. However, this approach works well until about the 10-μs model computation cycle time [24]. For shorter computation cycles, the latency of standard processors becomes an insurmountable barrier because of the latencies inherent in their deeply layered memory and I/O organization. In order to “break” the 10-μs barrier, an FPGA becomes the computational platform of choice. Today’s FPGA devices offer the peak computing power comparable to the state-of-the-art general purpose processors which, together with abundant on- chip memory and I/O resources, results in a superb sustained performance [25], [26] and

potentially very low latency. Still, the application of FPGA devices to PE is in its infancy. The approach most documented in the literatures is based on mapping the PE models to FPGA reconfigurable hardware either on hardware description language (HDL) [15], [16] or higher abstraction level [13], resulting in a hardware architecture which is customized for each instance of the problem (PE topology) which results in a highly optimized digital design impractical for wide industrial application.

The issues explained above as well as the comments on the easiness or the difficulty of each implementation and design can be thought of and applied to the direct analogy of Hardware in the Loop system which is a Rapid Control Prototyping system. The direct replacement of the real controller with its Simulink model counterpart and the replacement of the simulated hardware with the real hardware device is what constitute Rapid Control Prototyping systems. In the following text a solution to the aforementioned problems will be presented as well as an extension to whatever solutions are already tried.

Purpose of this thesis

The aim of this master thesis is to extend already existing hybrid approaches that combine the best of both worlds, user the user friendliness and scalability of processor-based simulators and the very low latency performance of FPGAs. The architecture of the system presented in this text, which will be described and explained in detail later on, comprises a Rapid Control Prototyping system which has an intermediate stage in between the Control Simulator (Simulink acting as the controlling hardware) and the real hardware (power electronic device). This intermediate stage is absolutely based on FPGA technology within which a highly modular programmable application is developed presenting the aforementioned advantages. Inside the programmable application floating point operations are made [17], amongst many other implementations, which also presents the advantages of DSPs. Interconnectivity between the setup parts is also done ia optic fibers [29] and most of the FPGA code developed is based on Finite State Machines (FSM) [30-33].

An industrial grade power electronic RCP test environment has to bring decisive advantages in the area of power electronic hardware, firmware, and software design and test automation. In practical terms this implies a short simulation setup time, graphical schematic editor with the look and feel of the state-of-the-art non-real-time simulation tools, very high speed computation, very low latency configurable and scalable I/O interfaces, rich library of power electronic FPGA code modules for Pulse Width Modulation, Analog-to-Digital conversion and communication interfaces, support for multiple converter systems, tools for the automated testing of power electronic designs with fast Fourier transform, junction temperature, RMS, and device loss calculation functions, easy integration with large system-level simulators.

The easiness of development and short concept-to-practise time for power electronic devices was the main reason that triggered the interest of the author of this master thesis work.

1.3. Real Time Simulation

Over the last two decades, commercially available computer has become both increasingly powerful and affordable. This, in turn, has led to the emergence of highly sophisticated simulation software applications that not only enable high-fidelity simulation of dynamic systems and related controls, but also automatic code generation for implementation in industrial controllers.

Used in combination with the generation of real-time simulators, these simulation software applications form the ground basis for a control design methodology that is based on the use of reference system models. The approach has the objective of accelerating the design cycle through the early detection of design flaws and other problems. Engineering applications such as: aircraft flight control design & validation, industrial motor

drive design, complex robotic controller design and power grid statistical protection tests benefit from the use of real-time simulators in a number of ways. First, real-time simulation produces a set of requirements and specifications that can be used by all disparate teams/subcontractors involved in a project. Secondly, it enables testing of simulated devices at or beyond their normal operating limits without the risks involved with testing of real devices, especially when high power levels are present. Third, it is easier and less risky to test fault responses on a simulated model. Finally, the simulation acceleration factor obtained by the use of compiled code (instead of the interpreted code used by most simulation tools) enables the realization of rapid batch simulations.

Real time simulation and its purpose

A simulation is a representation of the operation or features of a system through the use or operation of another. The types of digital simulation mentioned, include simulation with discrete-time and constant step duration.

During discrete-time simulation, time moves forward in steps of equal duration. This is commonly known as fixed time-step simulation. To solve mathematical functions and equations at a given time-step, each variable or system state is solved successively as a function of variables and states at the end of the preceding time-step.

During a discrete-time simulation, the amount of real time required to compute all equations and functions representing a system during a given time-step may be shorter or longer than the duration of the simulation time step. These two situations are referred to as offline simulation. In both cases, the moment at which a result becomes available is irrelevant. Typically, when performing offline simulation, the objective is to obtain results as fast as possible. The system solving speed depends on available computation power and the system's mathematical model complexity.

Conversely, during real-time simulation, the accuracy of computations not only depends upon precise dynamic representation of the system, but also on the length of time used to produce results. For a real-time simulation to be valid, the real-time simulator used must accurately produce the internal variables and outputs of the simulation within the same length of time that its physical counterpart would. In fact, the time required to compute the solution at a given time-step must be shorter than the ”wallclock” duration of the time-step. This permits the real-time simulator to perform all operations necessary to make a real-time simulation relevant, including driving inputs and outputs (I/O) to and from externally connected devices.

For a given time-step, any idle-time preceding or following simulator operations is lost. In such a case, the simulator waits until the clock ticks to the next time step. However, if simulator operations are not all achieved within the required fixed time-step, the real-time simulation is considered erroneous. This is commonly known as an ”overrun”. The overrun and accurate occasions are illustrated in Figure 1.1 and Figure 1.2 respectively.

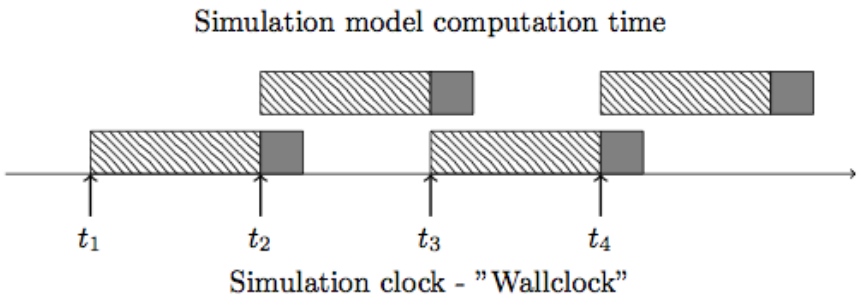


Figure 1.1

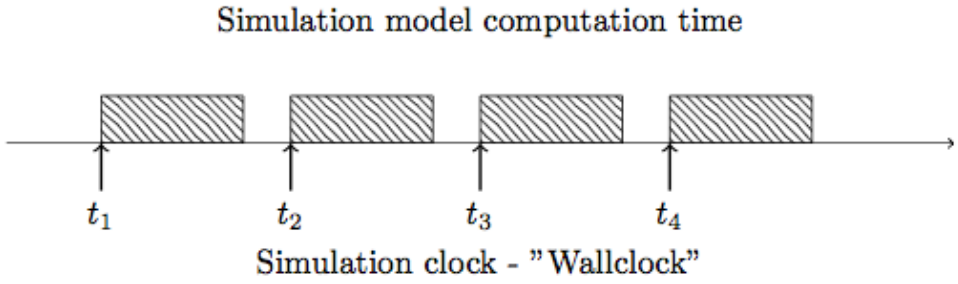


Figure 1.2

Based on these basic definitions, it can be concluded that a real-time simulator is performing as expected if the equations and states of the simulated system are solved accurately, with an acceptable resemblance to its physical counterpart, without the occurrence of overruns.

As previously discussed, real-time digital simulation is based on discrete time-steps where the simulator solves model equations successively. Proper time-step duration must be determined to accurately represent system frequency response up to the fastest transient of interest. Simulation results can be validated when the simulator achieves real-time without overruns.

For each time-step, the simulator executes the same series of tasks: • read inputs and generate outputs

- solve model equations
- exchange results with other simulation nodes
- wait for the start of the next step

A simplified explanation of this routine suggests that the state(s) of any externally connected device is/are sampled once at the beginning of each simulation time-step. Consequently, the state(s) of the simulated system is/are communicated to external devices only once per time-step. If not all real-time simulation-timing conditions are met, overruns occur and discrepancies between the simulator results and its physical counterpart's responses are observed.

Real-time simulators form the basis of the "Model-Based Design" (MBD) paradigm; a control design methodology that is centred on the use of reference system models. MBD is a mathematical and graphical method of addressing problems associated with the design of complex systems. It allows multiple engineers involved in a design and modelling project to use models to communicate knowledge of the system under development, in an efficient and organized manner. Four basic steps are necessary in the process:

- build the plant model
- analyse the plant model and synthesize a controller for it • simulate the plant and controller together
- deploy the controller.

MBD offers many advantages. By using models, a common design environment is available to every engineer involved in creating a system from beginning to end. Indeed, the use of a common set of tools facilitates communication and data exchange. Reusing older designs is also easier since the design environment can remain homogeneous through different projects.

Most commercial simulation tools provide an Automatic Code Generator that facilitates the transition from controller model to controller implementation. The added value of real-time simulation in MBD emerges from the use of an Automatic Code Generator. By using an Automatic Code Generator with a real-time simulator, an RCP can be implemented from a model with minimal effort. The prototype can then be used to accelerate integration and verification testing, something that cannot be done using offline simulation. The same holds true for HIL testing. By using an HIL test bench, test engineers become part of the design workflow earlier in the process, sometimes before an actual plant becomes available. For example, by using the HIL methodology, automotive test engineers can start early testing of a car controller before a physical test bench is available. Combining RCP and HIL, while using the MBD approach, has many advantages:

- Design issues can be discovered earlier in the process, enabling required trade-offs to be deter- mined and applied, thereby reducing development costs.
- Development cycle duration is reduced due to parallelization in the workflow.
- Testing costs can be reduced in the medium- to long-term since HIL test setups often cost less then physical setups and the real-time simulator employed can be typically used for multiple applications and projects.
- Testing results are more repeatable since real-time simulator dynamics do not change through time the way physical systems do.
- Can replace risky or expensive tests using physical test benches.

Interaction with the model has many advantages. These interactions can be with a system user, with physical equipment or with both at the same time. When a user or physical equipment interacts with a real-time model, they can provide model inputs and get model outputs, as it would with a real plant. A model executed on a real-time simulator can also be modified online, which is not possible with a real plant. In addition, any model parameter can be read and updated continuously. For example, in a power plant simulation, the shaft inertia of a turbine can be modified during simulation to determine its effect on stability, something impossible on a real power plant. Furthermore, with a real-time simulator, any model quantity is accessible during execution. For example, in a wind turbine application, the torque imposed on the generator from the gearbox is available, since it is a modelled quantity. In a real wind turbine, getting a precise torque value in real-time is near impossible due to the prohibitive cost of a torque meter.

Online model configuration and full data availability make previously unthinkable applications possible. For example, verifying if a controller can compensate for changes in plant dynamics caused by component aging.

Applications and configurations of real time computing

Various applications of real-time simulation exist and can be broken into three important categories:

- Rapid Control Prototyping (RCP)
- Hardware-in-the-Loop testing (HIL)
- Rapid Batch Simulation (RBS)

Consider a controlled process, which is composed of a plant with a controller acting upon it.

In RCP applications, a plant controller is implemented using a real-time simulator and is connected to a physical plant. RCP offers many advantages over implementing an actual controller prototype. A controller prototype developed using a real-time simulator is more flexible, faster to implement and easier to debug. The controller prototype can be tuned on the fly or completely modified with just a few mouse clicks. In addition, since every internal controller state is available, an RCP can be debugged faster without having to take its cover off. In RCP applications, an engineer will use a real-time simulator to quickly implement a controller and connect it to the real plant. A typical RCP configuration is shown in Figure 1.3.

HIL acts in an opposite manner. For HIL applications, a physical controller is connected to a virtual plant executed on a real-time simulator, instead of to a physical plant. In addition to the advantages of RCP, HIL allows for early testing of controllers when physical test benches are not available. Virtual plants also usually cost less and are more constant. This allows for more repeatable results and provides for testing conditions that are unavailable on real hardware, such as extreme events testing. Its main purpose is to test actual controllers connected to a simulated plant. Of course, it is very important not forget that human beings are included in the category of controllers as for example in the case of an aircraft flight simulator, which can be considered as a form of HIL simulation. A typical HIL configuration is shown in Figure 1.4.

RBS is typically used to accelerate simulation in massive batch run tests.

Figure 1.3

Figure 1.4

Speedgoat real-time machine and IO cards and Simulink xPC target

For the purposes of this text, all test setups and designs concern RCP systems. This means that the user's computer running a simulation program, such as Simulink, is connected through Gigabit Ethernet to a real-time computer, it compiles and downloads the control code to the target machine and it updates some of its variables in real-time. The target machine is also connected to the real plant (hardware under test) and it communicates with it through I/O ports.

A representation of the system is shown in Figure 1.5.

Figure 1.5

1.4. Thesis structure

In the following chapters it is explained how an engineer can achieve to incorporate various aspects of modern power electronic applications into a easily and swiftly developed system. The structure taken here applies for a grid-tie three-phase BSNPC IGBT inverter.

In Chapter 2 the system is described, containing an overview of the system, some Simulink connectivity issues and the hardware specification. Chapter 3 deals with one of the main parts of the system, the custom board and the FPGA, and explains its various parts and components. Chapter 4 presents the VHDL code developed and used in this work and fully analysed. Chapter 5 presents part of the testing down to verify the system operation and Chapter 6 makes a small summary and a modest approach on future applications. At the end of these work, all schematics and layouts of the custom board are presented as well as the FPGA code in its entirety.

2

System description

2.1. Overview

The project considered in this master thesis involves a Rapid Control Prototyping system for a three-phase grid-tie BSNPC IGBT inverter. A Rapid Control Prototyping system, as mentioned before, includes a computer that runs the Simulink software (has a loaded Simulink model and specific data inputs can be changed), a Real Time Computer that runs a compiled version of the Simulink model (already loaded on the computer) and the actual hardware. The hardware in the existing case comprises a custom board designed and manufactured especially for this master thesis project and already existing inverter hardware. The custom board acts as an intermediate between the controlling computers and the hardware. It receives and sends signals to and from the Real Time Computer as well as the inverter hardware, is capable of fully operating the hardware and providing the controlling computers with full data and information about the hardware. The data transmissions between the systems are done real-time while the inverter hardware is in operation and the editing and producing of new data is also done in real-time. The data processing inside the two controlling computers (Real Time Computer and Simulink computer) is also done in real-time and permits the testing and operation of the hardware simultaneously with its change in control variables, gains and duty cycles. The ease of operation of such a system (once it is developed) and the full controllability and flexibility makes it perfect for testing in power electronic applications.

In this master thesis project the hardware is the actual inverter which doesn't include any operating logic on it (DSP, FPGA) but only the supply and error signalling circuitry that couldn't be implemented into software. Thus, the custom board (mounted on the inverter hardware) acts as the intermediate stage between the Real Time Computer and the inverter hardware in order to play that its part. In Rapid Control Prototyping applications the control logic is implemented in the Simulink computer models and acts upon the hardware in real-time. In power electronic applications, though, where fast data transmission, fast control loops, high resolution and high switching frequency Pulse Width Modulation, special interaction with digital devices (analog-to-digital converters, CAN devices) safety watchers and swift tripping mechanisms are indispensable and very important parts of the system, a intermediate device is essential. The role of this device is to quickly and safely operate on the hardware, whether this means creating the PWM for the switching devices, implementing basic but fast control loops, translating data from peripheral hardware devices and configuring them, operating the tripping mechanisms of the hardware (relays), etc, and acquire data, filter them and transmit them back to the Real Time Computer for further editing. Additionally, the board can act as a Master for the system as it can control both the inverter hardware and the execution triggering of the Real Time Computer. In this way the gain is double because, at first, the system is macroscopically controlled by a slower Real Time Computer that can form the overall operation of the inverter hardware (e.g. on-the-fly setting of the controlling references and gains, feeding of changing data inputs, etc) and secondly microscopically provide fast, secure and basic operational control of the inverter hardware.

An overview of the system in the form of a block diagram is illustrated in Figure 2.1.

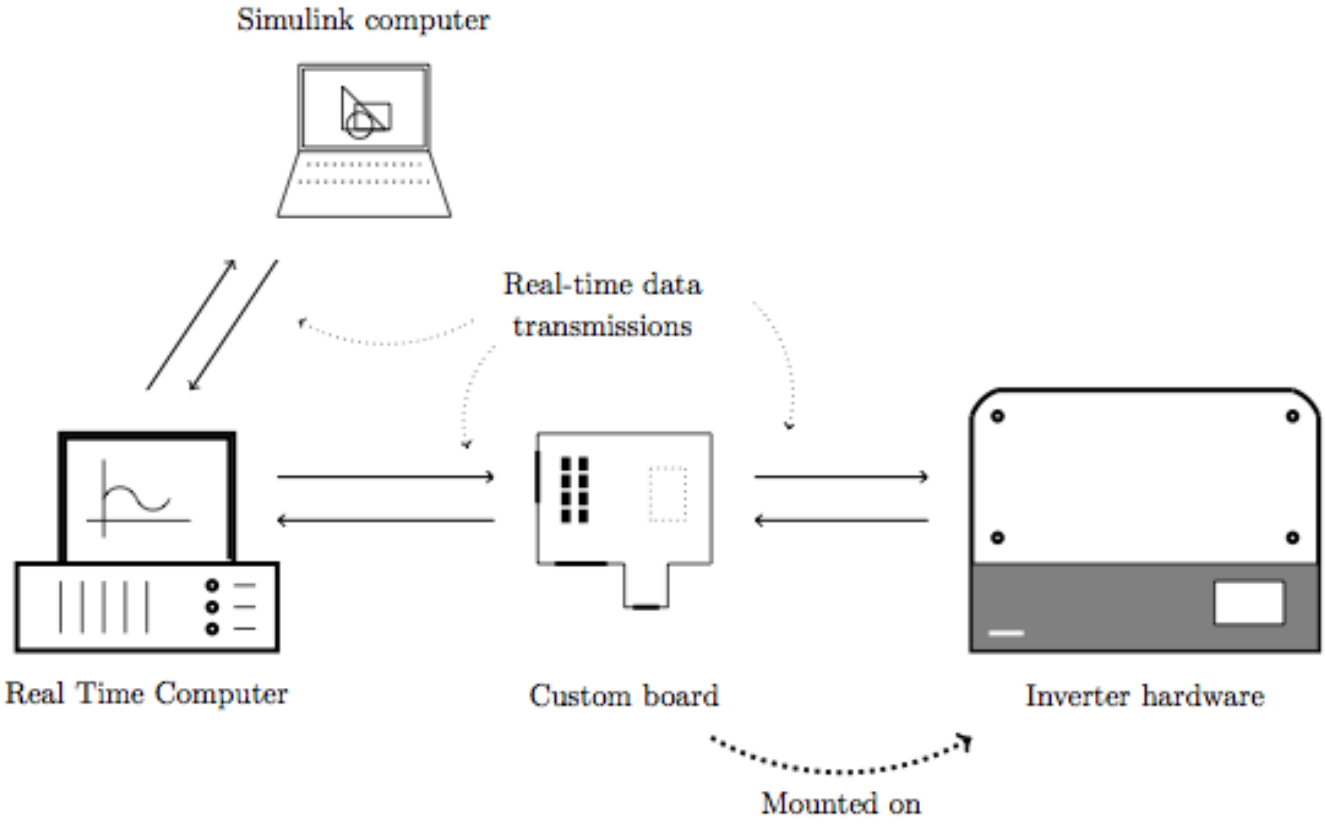


Figure 2.1

The Simulink computer is connected with the Real Time Computer through an Ethernet cable and can compile and download the compiled Simulink model to the Real Time Computer. It inter- acts with the Real Time Computer by firing the execution process as well as stop it, send values in real-time (like constants, gains, etc), log execution data and analyse execution statistics after every operation.

The Real Time Computer is responsible for executing the compiled Simulink model and with the help of existing extension input/output cards to interact with any hardware that is connected to. For this specific application there is a custom interface provided by the Real Time Computer manufacturer that introduces a new Simulink model block to add to an already existing system and communicate with the hardware in a more meaningful way appropriate for the hardware. The custom Simulink model block will be explained in a later section. The Real Time Computer also has the possibility to connect to a monitor and display data in numerical or graphical form in real-time.

The custom board, the detailed description of which resides on another chapter, is mounted on the inverter hardware and connects to both the inverter hardware and the Real Time Computer and exchanges data with both of them. It includes a Field Programmable Gate Array (FPGA) device on it and some peripheral hardware as well as the connecting adaptor ports. It interacts in a fast way with both systems and can fully control the operation of the inverter hardware depending on the FPGA software code and the received data from the Real Time Computer (that has the overall macroscopic control), as well as, partially control the operation of the Real Time Computer by triggering the execution of the compiled Simulink model code.

The inverter hardware includes the basic three-phase BSNPC inverter topology, bridge output and EMI filtering and some peripheral circuitry like voltage and current sensors, analog-to-digital converters, relays, low voltage supplies, etc. The inverter hardware can be connected to the grid or to a load.

2.2. Real Time Computer model specification

The connectivity of the Real Time Computer with the custom board is done through a custom protocol designed only for this purpose. This protocol is represented by a Simulink model block which acts as the input/output port of the Real Time Computer to the custom board and thus to the inverter hardware. The model block is used as any other block in Simulink and can exchange data in real-time with the custom board and provide a great degree of controllability over the hardware.

2.2.1 Core Features

To provide the functionality needed for the operational management of the custom board and in- verter hardware, bidirectional transmission of data is necessary. The data are represented in integer (with a word length of 16 bits) and binary (single signal) form. The list of the input and output ports is shown in the list below:

- 16-channel digital output.
- 5-channel input.
- High speed output of 32 x 16 bit words via the parallel bus.
- High speed input of 32 x 16 bit words via the parallel bus.

2.2.2 Simulink model block

The input/output ports of the custom Simulink model block are shown in Figure 2.2.

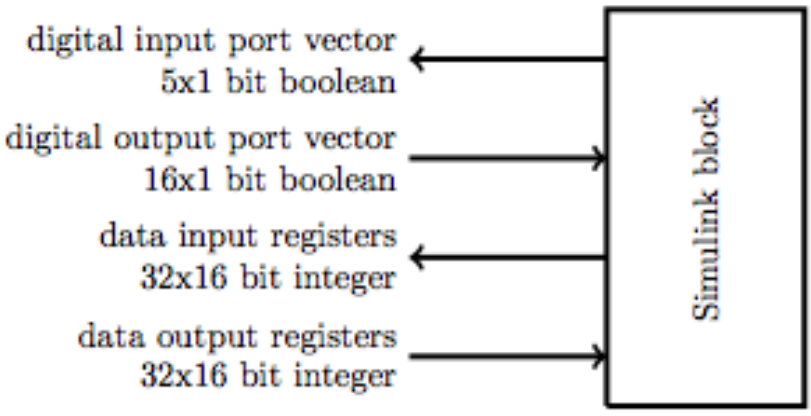


Figure 2.2

2.2.3 Interface between Real Time Computer and custom board

The connection channels (inputs/outputs) between the Real Time Computer and the custom board are shown in Figure 2.3.

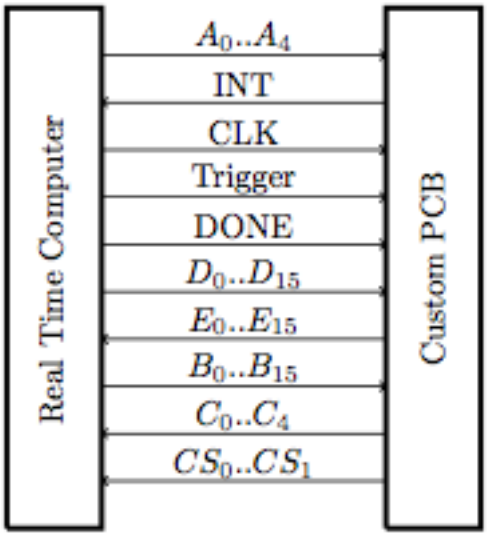


Figure 2.3

Figure 2.4 shows the output pins for the high speed 32x16 bit parallel output and 32x16 bit parallel input bus. The input and output integer values are signed integers and are represented by a two's complement. Thus, the valid range for the output data is $-32768 \leq X16 \leq 32767$.

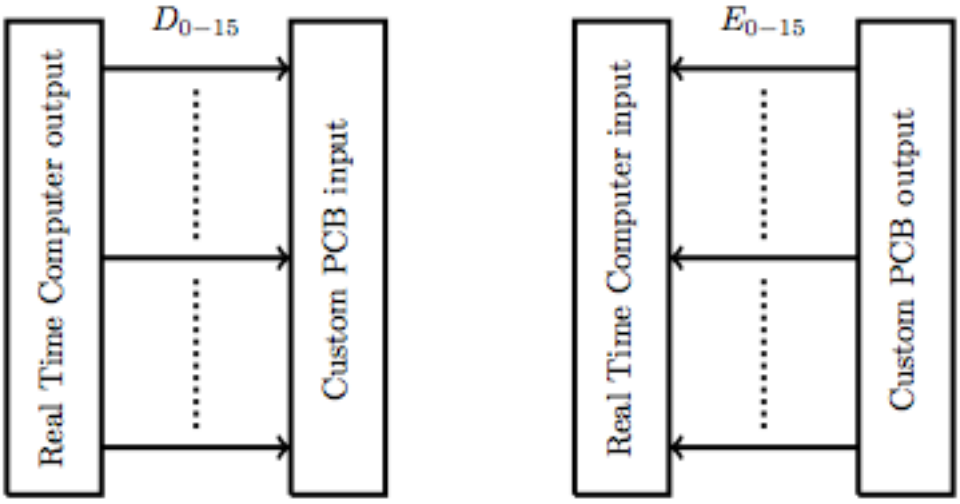


Figure 2.4

2.2.4 I/O configuration - Pin assignment

The Real Time Computer is connected to the custom board through an extension card via a 68-pin connector. Some of the connector's pins are used for grounding and shielding for noise. The following tables 2.1 and 2.2 show the pin assignment and the description for the Real Time Computer connection pins.

Table 2.1

Signal	Description	Real Time Computer Pin	IO Direction
D_0	Data bit 0	19	Output
D_1	Data bit 1	20	Output
D_2	Data bit 2	21	Output
D_3	Data bit 3	22	Output
D_4	Data bit 4	23	Output
D_5	Data bit 5	24	Output
D_6	Data bit 6	25	Output
D_7	Data bit 7	26	Output
D_8	Data bit 8	27	Output
D_9	Data bit 9	28	Output
D_{10}	Data bit 10	29	Output
D_{11}	Data bit 11	30	Output
D_{12}	Data bit 12	31	Output
D_{13}	Data bit 13	32	Output
D_{14}	Data bit 14	33	Output
D_{15}	Data bit 15	34	Output
E_0	Data bit 0	53	Input
E_1	Data bit 1	54	Input
E_2	Data bit 2	55	Input
E_3	Data bit 3	56	Input
E_4	Data bit 4	57	Input
E_5	Data bit 5	58	Input
E_6	Data bit 6	59	Input
E_7	Data bit 7	60	Input
E_8	Data bit 8	61	Input
E_9	Data bit 9	62	Input
E_{10}	Data bit 10	63	Input
E_{11}	Data bit 11	64	Input
E_{12}	Data bit 12	65	Input
E_{13}	Data bit 13	66	Input
E_{14}	Data bit 14	67	Input
E_{15}	Data bit 15	68	Input

Table 2.2

Signal	Description	Real Time Computer Pin	IO Direction
CLK	I/O clock	14	Output
DONE	Matlab model flag	15	Output
$\overline{CONF}$	xPC target ready	16	Output
A_0	Address bit 0	9	Output
A_1	Address bit 1	10	Output
A_2	Address bit 2	11	Output
A_3	Address bit 3	12	Output
A_4	Address bit 4	13	Output
B_0	Digital bit 0	1	Output
B_1	Digital bit 1	2	Output
B_2	Digital bit 2	3	Output
B_3	Digital bit 3	4	Output
B_4	Digital bit 4	5	Output
B_5	Digital bit 5	6	Output
B_6	Digital bit 6	7	Output
B_7	Digital bit 7	8	Output
B_8	Digital bit 8	37	Output
B_9	Digital bit 9	38	Output
B_{10}	Digital bit 10	39	Output
B_{11}	Digital bit 11	40	Output
B_{12}	Digital bit 12	41	Output
B_{13}	Digital bit 13	42	Output
B_{14}	Digital bit 14	43	Output
B_{15}	Digital bit 15	44	Output
INT	Interrupt	45	Input
CS_0	Clock select bit 0	46	Input
CS_1	Clock select bit 1	47	Input
C_0	Digital bit 0	48	Input
C_1	Digital bit 1	49	Input
C_2	Digital bit 2	50	Input
C_3	Digital bit 3	51	Input
C_4	Digital bit 4	52	Input

2.2.4 Data transmission speed

The input signals CS0 and CS1 are used to select the Real Time Computer clock output frequency. The functional principle of CS0 and CS1 is set by the custom board and declares the frequency by which the parallel data will be transmitted to and from the Real Time Computer through the cable. The correlation between the Real Time Computer output clock frequency and the CS0 and CS1 combination can be found in table 2.3.

Table 2.3

CS_0	CS_1	CLK frequency
0	0	1,03125MHz
1	0	2,0625MHz
0	1	4,125MHz
1	1	8,25MHz

2.2.4 Simulink model block parameters

The MATLAB block parameters that the user can use to configure the operation of the Rapid Control Prototyping system are show in the list below:

- Digital output channel - reset vector: The reset vector controls the behaviour of the output channels at model termination (the elements of the vector which correspond to channels which are defined as inputs have no effect). A value of 0 specifies that the corresponding channel holds its current value at termination. A value of 1 specifies that the corresponding channel is reset to the level specified in the Initial value vector (see below) at termination. If just a scalar value is provided, scalar-expansion applies. This means that a scalar of 0 leads to zeros(1,16) which defines that all 16 channels of the lower group should hold their levels. A scalar of 1 leads to ones(1,16) which defines that all 16 channels of the lower group should reset their levels. If a vector (more than one element) is provided then the length has to be 16. Each element then defines the Reset behaviour of the corresponding channel.
- Digital output channel - initial value vector: The initial value vector defines the initial level and the reset level at termination if the corresponding channel is set to reset. A value of 0 specifies that the corresponding channel should be reset to level Low. A value of 1 specifies that the corresponding channel should be reset to level High. If just a scalar value is provided, scalar-expansion applies. This means that a scalar of 0 leads to zeros(1,16) which defines that all 16 channels of the lower group should reset to level Low. A scalar of 1 leads to ones(1,16) which defines that all 16 channels of the lower group should reset to level High. If a vector (more than one element) is provided then the length has to be 16. Each element then defines the reset level of the corresponding channel.
- Digital input channel - reset vector: Reset vector for the digital input channels analogously to the digital output reset vector.
- Digital input channel - initial value vector: Initial value vector for the digital input channels analogously to the digital output initial value vector.
- Data output register - reset vector: Reset vector for the input registers also analogously to the other reset vectors.
- Data output register - initial value vector: Initial value vector for the input registers analogously to the other initial value vectors.

- Data input register - reset vector: Reset vector for the output registers analogously to the other reset vectors.
- Data input register - initial value vector: Initial value vector for the output registers analogously to the other initial value vectors.

2.2.4 High speed parallel interface protocol

At first the Real Time Computer initializes its system so that it can start running the overall

Simulink model. During that time the CONF signal is set to LOW so as to declare that it is not ready for interaction. After the Simulink model initialization the Real Time Computer designates

its readiness by pulling up the CONF signal. Then the custom board controls the Real Time Computer through the interface and the first calculation starts by pulling up the INT signal. The system operates normally from now on by the alternation of the INT signal in times defined by the custom board's FPGA. The start-up sequence of the Real Time Computer system is illustrated in Figure 2.5.

The signals involved in the start-up sequence and the normal operation are:

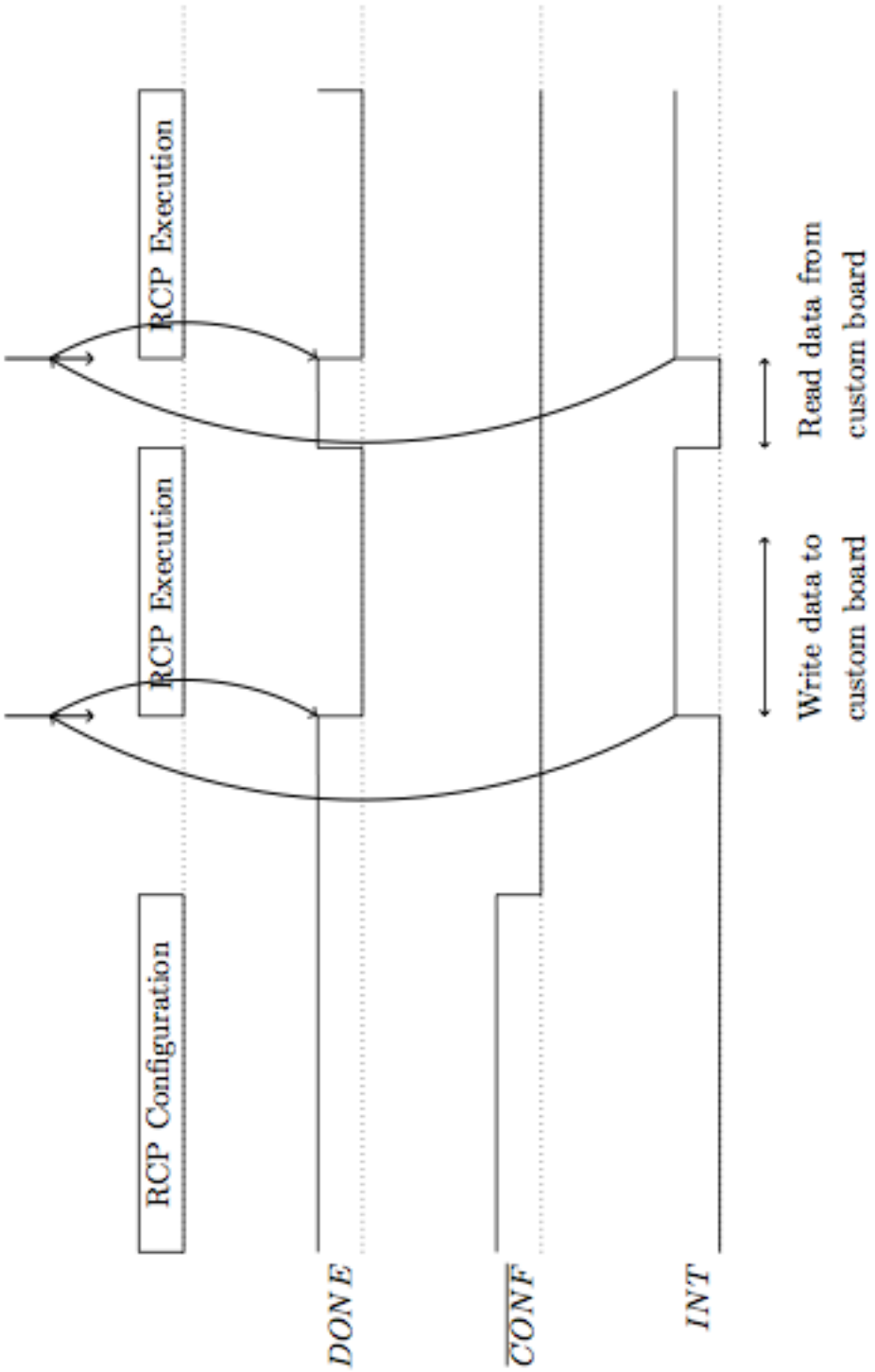


Figure 2.5

- CONF:

  When the Real Time Computer is not configured or during configuration, the signal CONF is HIGH or in high impedance mode. After the configuration is finished, the signal CONF is set to LOW. The first data write or read cycle starts with the first edge of the signal INT after CONF is set to low level.

- INT:

  The rising edge of INT creates a signal for the Simulink model execution triggering and sending of parallel data from the Real Time Computer to the custom board. The falling edge of INT designates the reading of data from the custom board to the Real Time Computer.

- DONE:

  The signal DONE shows the Simulink model execution time. Every time the Simulink model is triggered through the INT signal, the signal DONE is set to LOW and designates the start of the Simulink model execution. When the Simulink model execution is finished and the computed values are available the signal DONE is set to HIGH.

In Figure 2.6 the falling edge of INT starts the data read cycle from the custom board. With the rising edge of CLK the address is sent to the custom board. After some time (due to propagation deaf caused by the isolation hardware buffers) the requested data is available at the input ports of the port of the Real Time Computer. With the next falling edge of CLK the data from the custom board is latched.

The rising edge of INT starts the data write cycle to the custom board. With the rising edge of CLK the address and the data are sent to the custom board.

The clock selection input signals are latched with the falling and rising edge of the INT signal. Hence the CLK frequency can be changed for each read or write cycle independently by the custom board's FPGA.

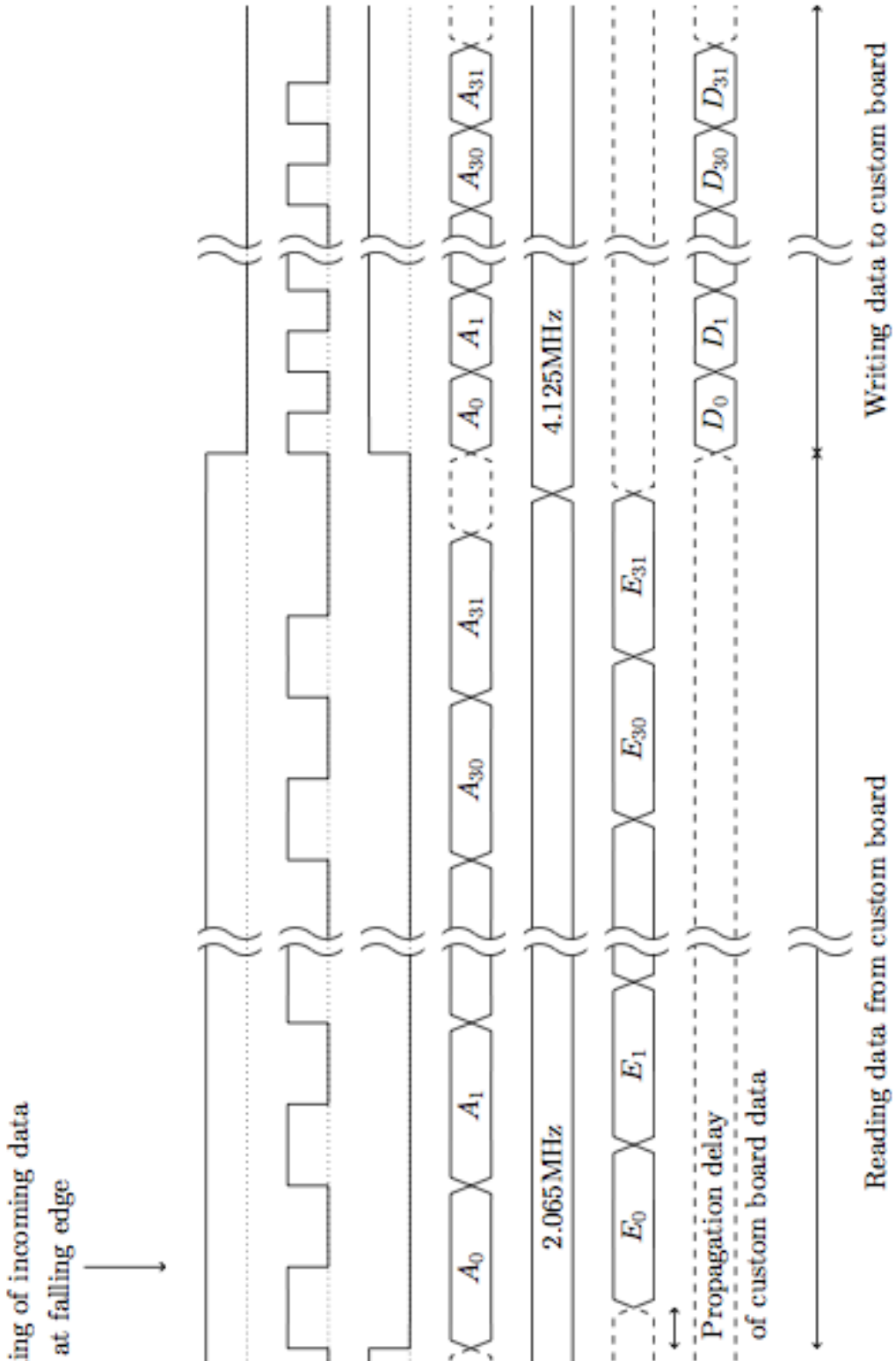

Latching of incoming data
at falling edge
DONE
CLK
INT
A_{0-5}
CS_{0-1}
E_{0-15}
D_{0-15}
A_0
A_1
A_{30}
A_{31}
2.065MHz
E_0
E_1
E_{30}
E_{31}
Propagation delay
of custom board data
Reading data from custom board
4.125MHz
D_0
D_1
D_{30}
D_{31}
Writing data to custom board

Figure 2.6

2.3. Inverter hardware

2.2.4 Overview

The Rapid Control Prototyping system requires the existence of the real hardware in order to op- erate it. The hardware used in this project involves a three phase BSNPC bridge inverter fed by a capacitor bank (called the DC link) split in two parts for access to a middle point and with an LCL filter at the output of each leg of the bridge. The output of the three phase LC filter can be connected to an external load or the grid through three relays. The voltages and currents of the DC link capacitors are measured used dedicated circuitry and analog-to-digital converters. The voltages of the three AC filter capacitors are also measured and converted using analog-to-digital- converters as well as the currents flowing to the three output phases that are connected to the grid or an external load are measured and converted. The overall hardware includes also some other components (such as boost converters, 5V power supply, power supply checks, etc) that will not concern the present project and will not be considered. Although, some setup and configuration of these parts of the hardware will be done in order to allow for the interesting part of the hardware system to become operational and safe.

The overview of the hardware into consideration is illustrated in Figure 2.7.

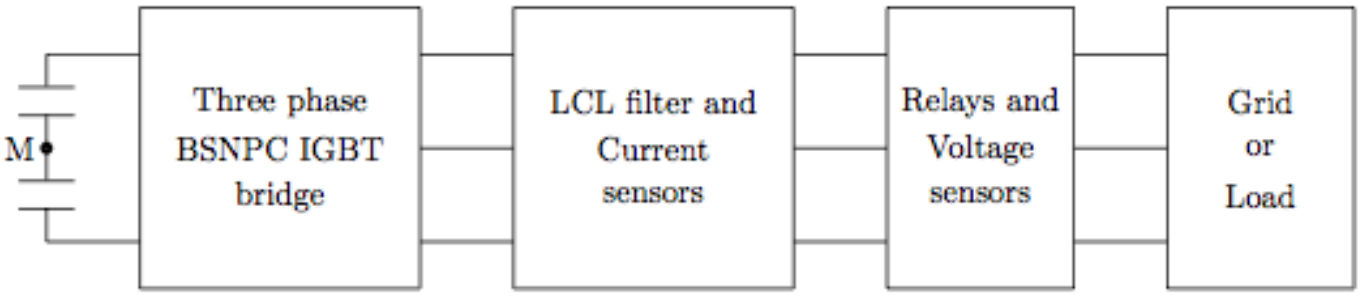


Figure 2.7

2.2.4 Overview

The Rapid Control Prototyping system needs to communicate with the real hardware in order to successfully operate it in real time. This is achieved through the use of some signals that get trans- mitted in and out of the inverter hardware. The list of these signals is demonstrated in table 2.4.

Table 2.4

Signal	Description
$\overline{RESET\_BR}$	Turns off all bridge IGBTs
$GATE\_A1$	Controls the switching operation of IGBT A_1
$GATE\_A2$	Controls the switching operation of IGBT A_2
$GATE\_A3$	Controls the switching operation of IGBT A_3
$GATE\_A4$	Controls the switching operation of IGBT A_4
$GATE\_B1$	Controls the switching operation of IGBT B_1
$GATE\_B2$	Controls the switching operation of IGBT B_2
$GATE\_B3$	Controls the switching operation of IGBT B_3
$GATE\_B4$	Controls the switching operation of IGBT B_4
$GATE\_C1$	Controls the switching operation of IGBT C_1
$GATE\_C2$	Controls the switching operation of IGBT C_2
$GATE\_C3$	Controls the switching operation of IGBT C_3
$GATE\_C4$	Controls the switching operation of IGBT C_4
$\overline{SPI1\_CS}$	Chip select signal of all single-channel analog-to-digital converters
$SPI1\_CLK$	Clock signal of all single-channel analog-to-digital converters
$SPI1\_A\_MISO$	Data bit sent by the single-channel analog-to-digital converter of phase A
$SPI1\_B\_MISO$	Data bit sent by the single-channel analog-to-digital converter of phase B
$SPI1\_C\_MISO$	Data bit sent by the single-channel analog-to-digital converter of phase C
$\overline{SPI2\_CS}$	Chip select signal of the first 16-channel analog-to-digital converter
$\overline{SPI3\_CS}$	Chip select signal of the second 16-channel analog-to-digital converter
$ADC23\_CLK$	Clock signal sent to the two 16-channel analog-to-digital converters
$SPI2\_MOSI$	Control bit sent to the first 16-channel analog-to-digital converter
$SPI3\_MOSI$	Control bit sent to the second 16-channel analog-to-digital converter
$ADC2\_MISO$	Data bit sent by the first 16-channel analog-to-digital converter
$ADC3\_MISO$	Data bit sent by the second 16-channel analog-to-digital converter
$RLYABCTL$	Control signal for the AC double-contact relay of phases A and B
$RLYBCCTL$	Control signal for the AC double-contact relay of phases B and C
$RLYCACTL$	Control signal for the AC double-contact relay of phases C and A
$\overline{SAFEBREAK}$	Control signal to control the group of relays
+5V	Power supply to the custom board

The signals under consideration are those who are involved into the control of the inverter switching devices, the current and voltage measuring units, the control of the relays and the safety signals. A summary of the signal groups and a short description of each can be seen below:

- **BSNPC bridge signals:**

This group of signals controls the switching operation of the top, bottom and middle IGBT switches of the three-phase BSNPC bridge and also ensures its safe group turning off through RESET BR in case of malfunction.

- **Analog-to-digital converter signals:**

  This group of signals controls the operation of the two 16-channel and three single-channel analog-to-digital converters. The chip select signals SPI1 CS, SPI2 CS and SPI2 CS control whether the converter initiates or not a conversion-transmission. The clock signals SPI1 CLK and ADC23 CLK control the pace by which the transmission of the data is done between the Master (custom board) and the Slave (analog-to-digital converter). The Master-Input-Slave-Output (MISO) signals SPI1 A MISO, SPI1 B MISO, SPI1 C MISO, ADC2 MISO, ADC3 MISO, are the bits that represent the actual converted measured values that the Master receives by the Slave. For the 16-channel analog-to-digital converter only there is also the possibility to configure the chip so as to retrieve specific channel in specific order and that can be done by sending data serially in the form of bits through the Master-Output-Slave-Input signals SPI2 MOSI and SPI3 MOSI.

- **Relay signals:**

  This group of signals controls the triggering of the double-contact AC relays separately and altogether. The signals RLY ABCT L, RLY BCCT L, RLY CACT L can control the switching operation of the respective relays independently, but they are all connected to AND gates with the signal SAFEBREAK that controls the overall relay switching operation.

The actual hardware involves many other signals, which are not relevant for our application but can be used for future extensions.

2.2.4 Inverter topology

The inverter hardware includes a three phase Bipolar Switch Neutral Point Clamped (BSNPC) bridge with IGBTs and antiparallel diodes. Thus, each leg has 4 switching devices and 4 antiparallel diodes and this allows for unipolar switching and so less harmonics at the output of the bridge.

The inverter topology is illustrated in Figure 2.8.

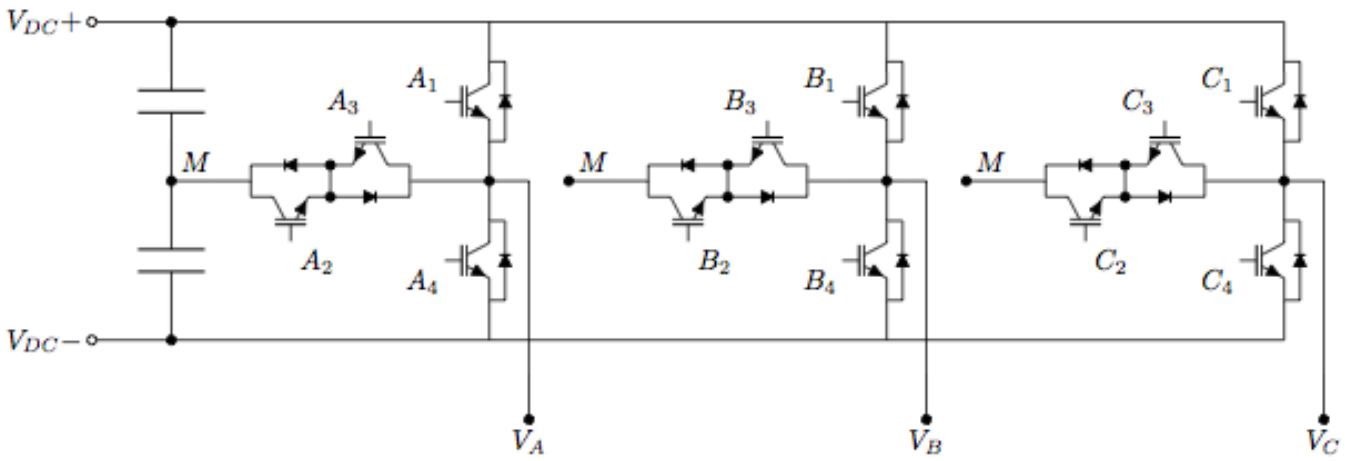


Figure 2.8

The restrictions of this topology is that for each leg switches S1 and S4 cannot be on simultaneously as well as S1 with S2 and S4 with S3. This limitation protects the switching devices from short circuiting the DC link capacitors with destructive results.

Due to the fact that the components are not ideal switching devices and switching off and on requires some time, dead time has to be accounted for in order to avoid the aforementioned case where two non-permitted switching devices are on simultaneously. The dead time accounted for is 1000ns for devices that could cause a short circuit and 300ns for short ON pulses and 1000ns for short OFF pulses. In case the short ON pulses are shorter than 300ns then the switching action is not performed and the switch stays at its current state (pulse not outputted). The same applies for the case of the short OFF pulses where if the duration of the pulse is shorter than 1000ns then the switch stays at its current state (pulse not outputted). The dead times and minimum and maximum pulse widths are illustrated in Figure 2.9.

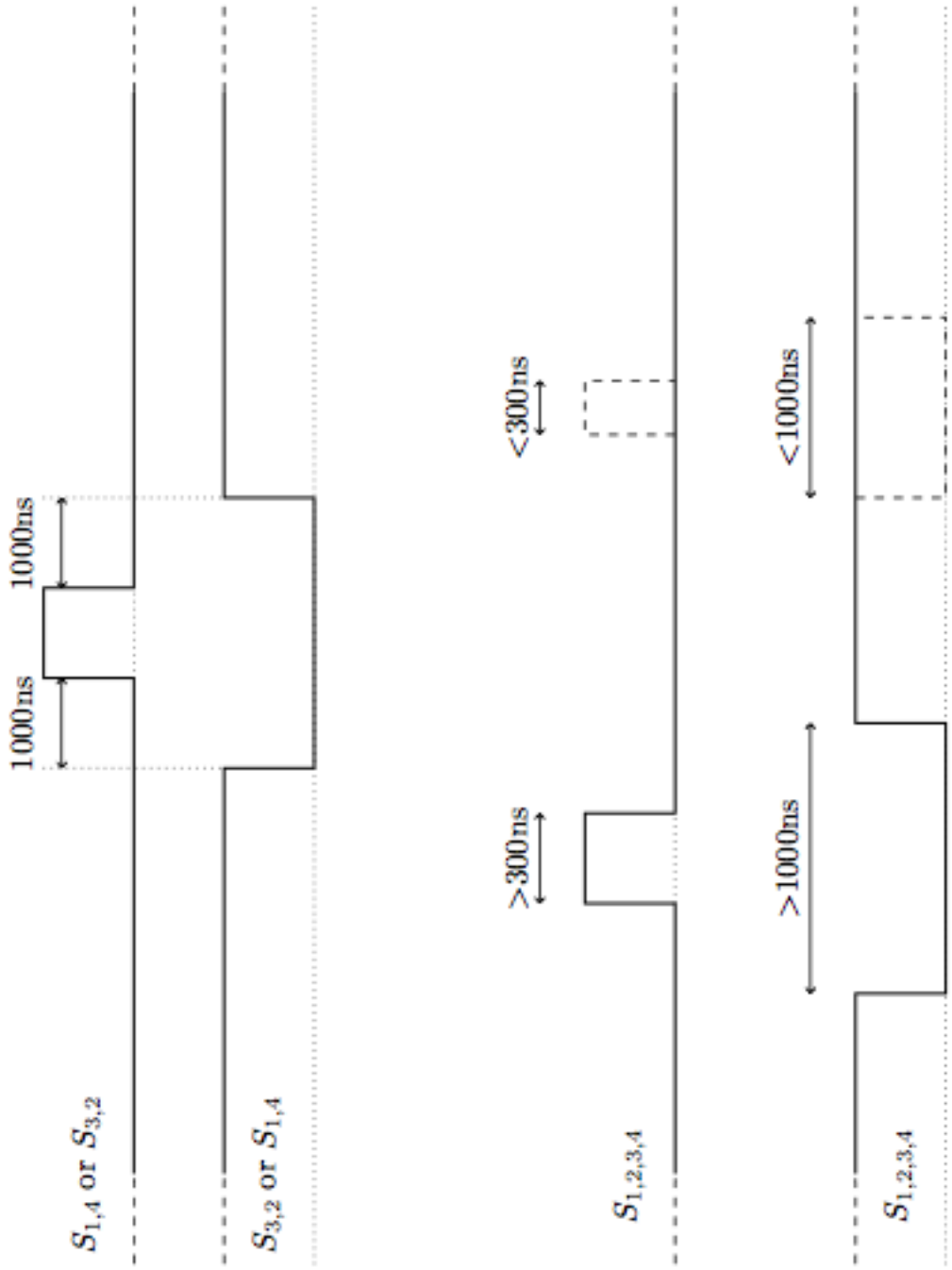


Figure 2.9

The DC link input to the BSNPC bridge is designed for a voltage maximum of 1000V and it produces at the output 230Vac nominal output phase-to-neutral voltage.

2.2.4 Measuring units

In order for the inverter to operate successfully a feedback of many quantities has to be acquired. Whether that would be current, voltage or a device status measurement that is necessary for the real-time state recognition of the real hardware.

In the hardware into consideration, 9 voltage and 3 current measurements circuits are used. The quantities measured are the three phase voltages and currents, the DC link voltages and the relay voltages. The measured values are fed to the analog-to-digital converters to be converted to digital values. The measured signals are normalized into some analog values that are recognizable and permitted by the analog-to-digital converters. The inverter hardware system has two 16-channel, 12-bit analog-to-digital converters and three faster 1-channel, 12-bit analog-to-digital converters. The models used are the ADS7953 and the ADS7883 from Texas Instruments. The external reference provided to the analog-to-digital convert- ers is 2V and the range of input is 0-4V.

Voltage measurement

The voltage measuring circuit connected to the analog-to-digital converters involves (for every measured voltage) a resistance array to drop the voltage to a specific value and keep the creepage distances adequate and an operational amplifier with the appropriate negative feedback in order to cut out high harmonics and normalize the voltage level to the one that the analog-to-digital converter permits. The output range of this circuit is 0-4V. One can see that there is a possibility to add decoupling capacitors at the input of the operational amplifiers in order to suppress the dc offset of the measured signal. Also the measured value is compared to a reference value which is also the reference for the analog-to-digital converter. This applies for measuring AC values that os- cillate around the reference value. In the case of DC values we take as reference a local ground point.

The typical voltage measurement circuit is shown in Figure 2.10.

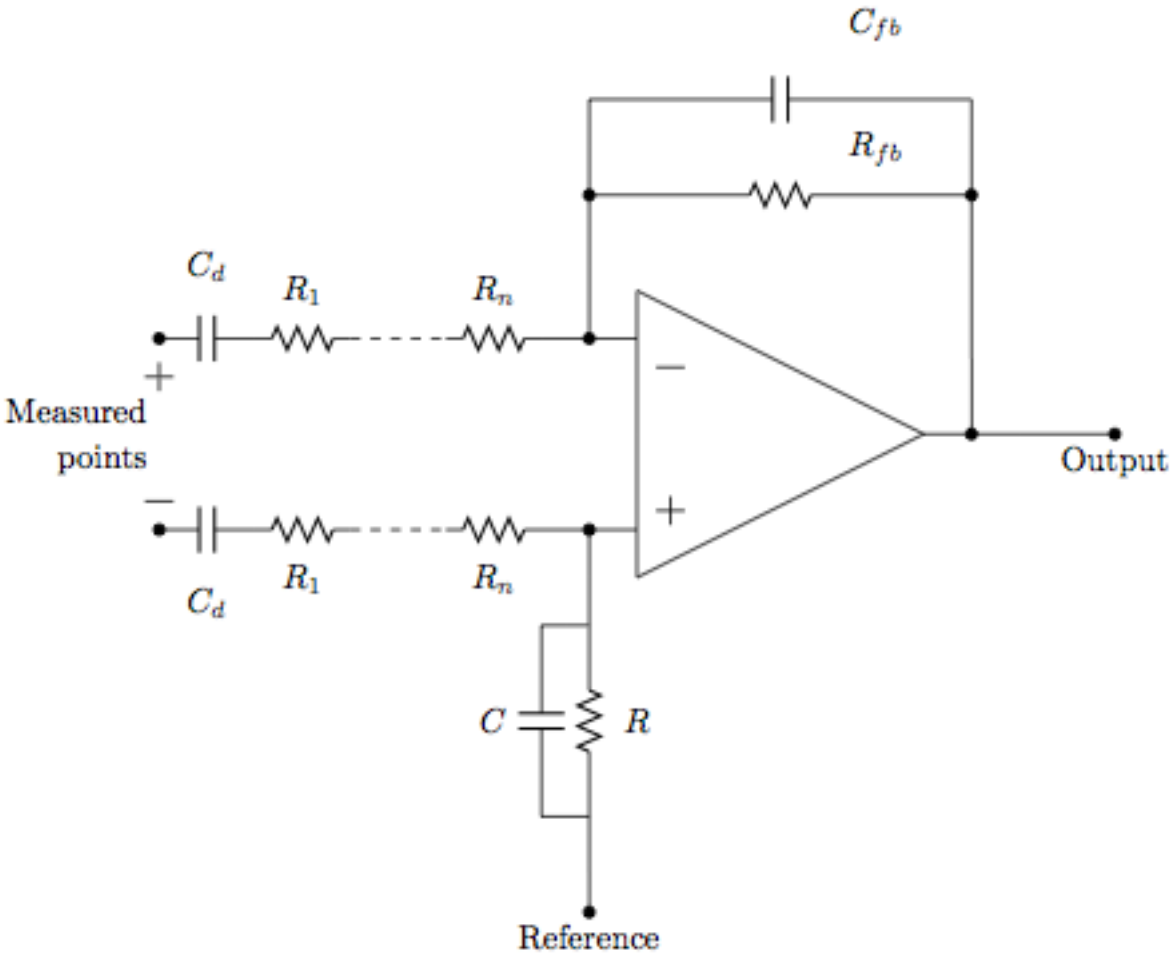


Figure 2.10

The position of measurement on the inverter hardware is illustrated in Figure 2.11.

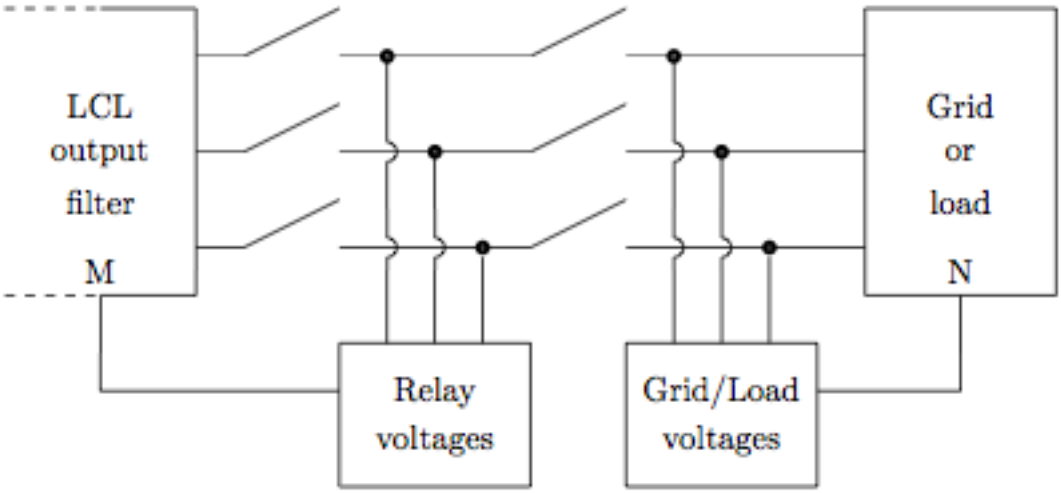


Figure 2.11

Current measurement

The current measuring circuit connected to the analog-to-digital converters involves (for every mea- sured voltage) a closed loop hall-effect transceiver with a 2V reference and a 0-4V output range. The output of the hall-effect current sensor is passed through a RC filter to cut out the high fre- quency components of the measured value and reduce noise.

The typical current measurement circuit is shown in Figure 2.12.

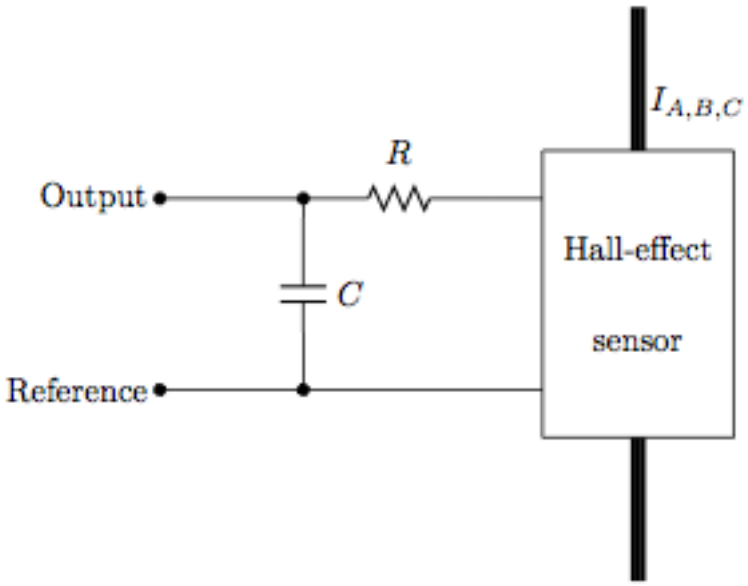


Figure 2.12

For control issues the current measurement is taken from the output branch of each leg of the inverter bridge. Due to the fact that the potential of the node before the output filter inductance is pulsating with a high ΔV and that creates common mode voltages on the measurement device which introduces noise to the measurement output, the position of the current measuring sensor is chosen to be at the node of the output filter capacitance. In that way the common mode voltage is greatly reduced and the measurement is done with less noise. The position of the current sensors on the inverter hardware is illustrated in Figure 2.13.

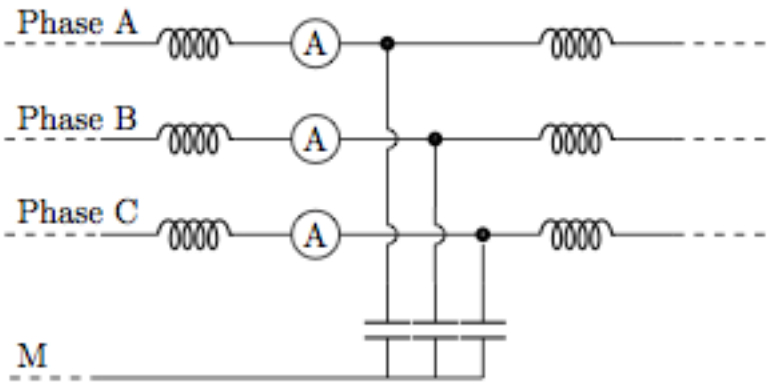


Figure 2.13

2.2.4 Relays

The relays used in this project are three double pole relays controlled independently and each one of them is capable of cutting two phases simultaneously for security reasons. The goal of this is in case of one relay failing to open, the other two will successfully isolate the inverter from the grid and avoid any case of voltage pollution on the grid lines in case of inverter failure. The position of the relays can be seen in Figure 2.14.

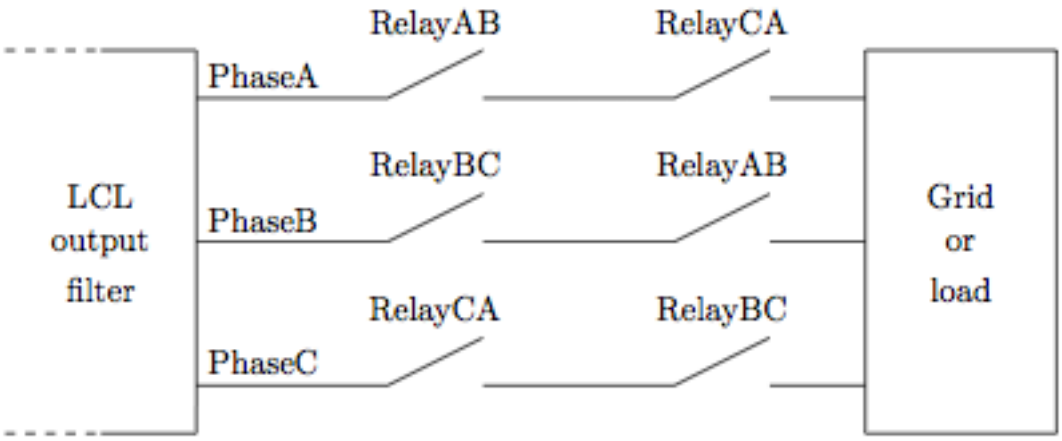


Figure 2.14

The relays are control through three signals. The relay system also has another signal (SAF EBREAK) that can handle all three in case of some failure. The operation is illustrated in Figure 2.15.

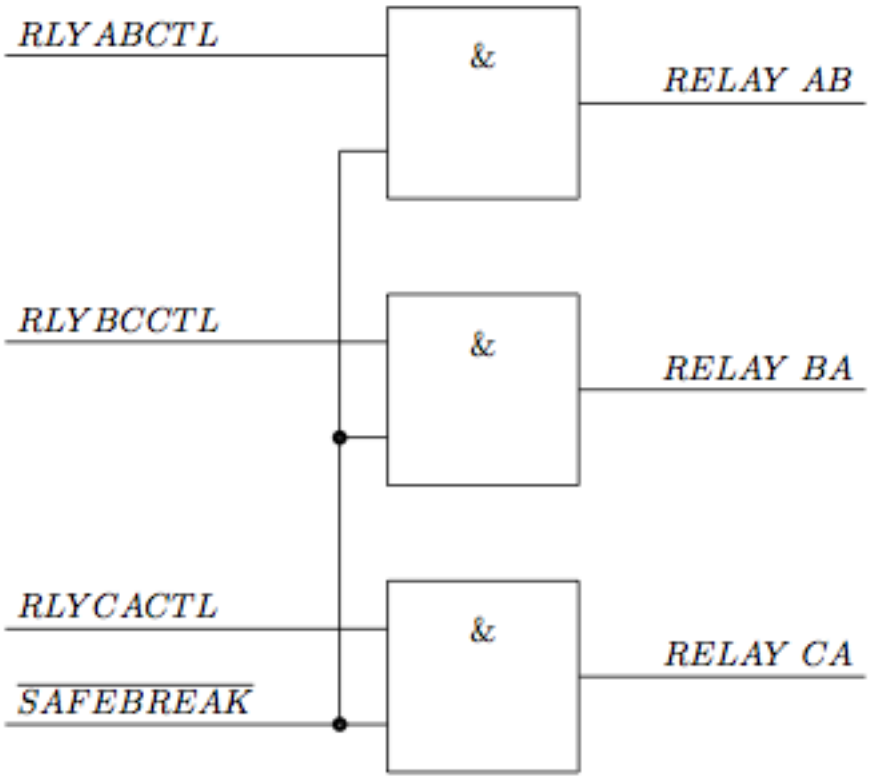

RLYABCTL
&
RELAY AB
RLYBCCTL
&
RELAY BA
RLYCACTL
&
RELAY CA
SAFEBREAK

Figure 2.15

3

Custom board description

3.1. Overview

For the purpose of the this Master Thesis a 4-layer Printed Circuit Board was designed and manufactured using the free version of Eagle software from CADSoft. The main purpose of the board is to connect the Real Time Computer to the inverter hardware. In that way the board will act as an intermediate that will drive and receive signals from and to the Real Time Computer (Rapid Control Prototyping System) and from and to the inverter hardware. This procedure is possible through particular connections on the board, protective and configurative hardware and with the aid of a Field Programmable Logic Array (FPGA) block. The board also has three supply circuits on it to provide for some of the special chips. In addition to the above, there are two isolation stages to protect the Real Time Computer from the high voltage components of the inverter hard- ware as well as to isolate the Real Time Computer ground from the inverter ground. The isolation stages are achieved with special chips that allow for bidirectional data transmission. The board also has some optional optical transceivers and some jumpers for different configuration possibilities.

This custom board has three ports to connect with the Rapid Control Prototyping System and the inverter hardware. The connection with the Rapid Control Prototyping System is done through a 68 pin SCSI cable and the connection with the inverter is done with one 80 pin and one 50 pin connector.

One can see the block diagram of the custom board with the most important parts as well as its physical dimensions illustrated in Figure 3.1

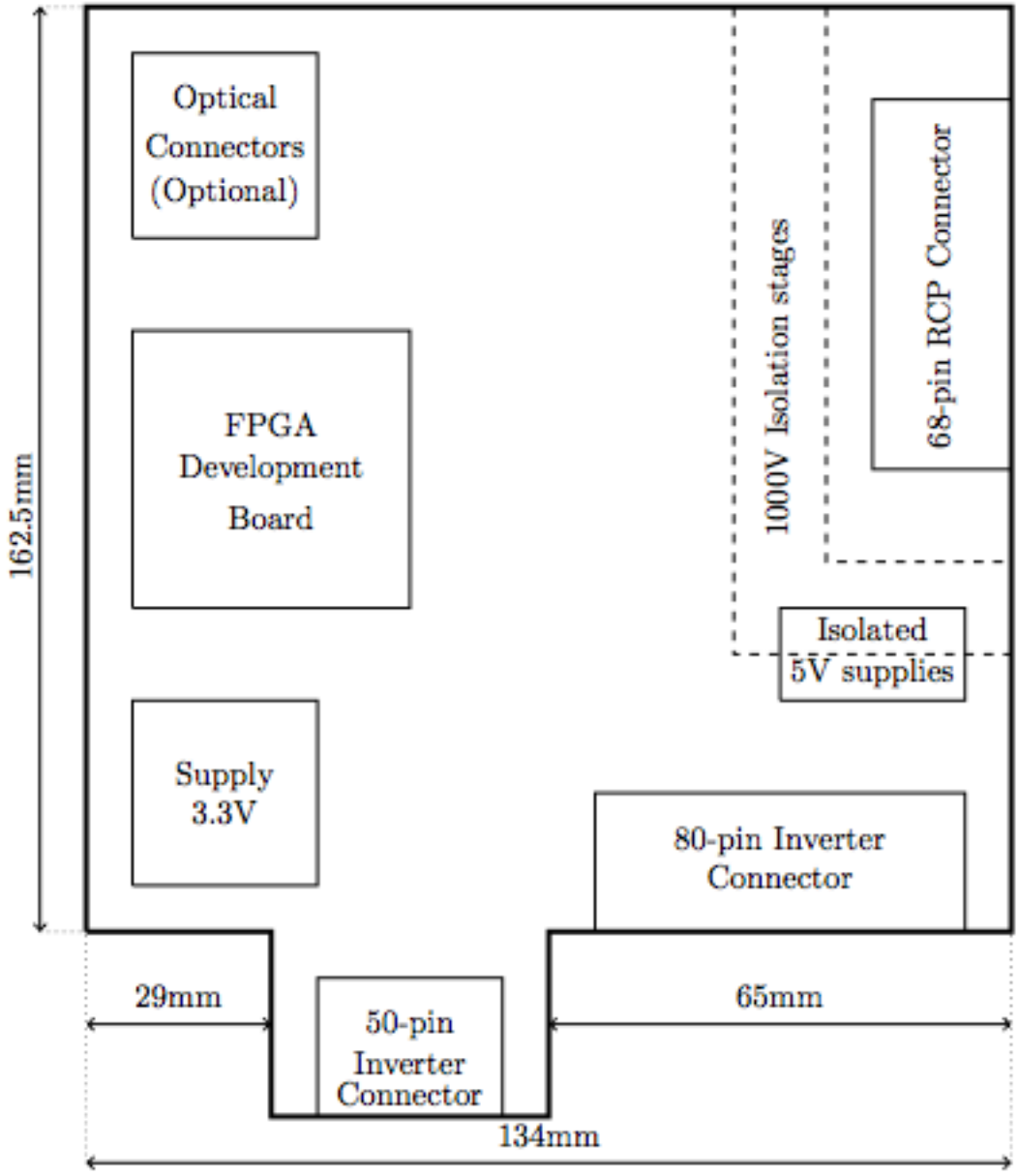


Figure 3.1

Some of the auxiliary components were not shown in the above block diagram for simplification but they are explained thoroughly in the sections of this chapter.

3.2. Isolation

The voltage levels between the Real Time Computer ground and the inverter ground can reach 1000V. In order to prevent destruction of the Rapid Control Prototyping system a 1000V isolation should be designed.

Through this isolation though, signals to and from the Real Time computer should be transmitted and received. For that purpose the Quad-Channel Digital Isolators ADUM2400 from Analog Devices were used. Due to the fact that one such device can provide sufficient isolation only for 500V, two devices in series had to be used in order to provide isolation for 1000V that was required. For this reason three isolated planes

were created each with its own supply. The supply of all sides of the isolating chips was 5V except for the main supply-ground plane where it was 3.3V.

In order to comply with the regulation DIN EN 62109-1 about photovoltaic inverters, taking into consideration that the voltage level is 1000V in total and 500V partial between the isolating chips then a creepage distance of 2.5mm for 500V and of 5mm for 1000V is required. This distance will be applied to the ground and supply planes between the isolating chips so as to prevent voltage breakdown between the planes. For the 500V isolation 2.5mm distance could be achieved and is illustrated in the figures below.

On the other hand, due to the fact that the creepage distance between some pads of the 5V supply unit on the 3.3V plane and some pads of the same 5V supply unit on the higher voltage 5V plane (Real Time Computer connector plane) wasn't big enough, an air gap was introduced on the custom board under one of the two TRACO isolated power supplies to increase the creepage distance and prevent voltage breakdown. According to the same regulation (DIN EN 62109-1) the width of the air gap is dictated by the required clearance for 1000V and for an altitude of 2000m or less (for this testing application). The clearance distance is 1.5mm and it can be shown in Figure 3.2.

The number of signals to be transferred is 64, as it will be described at a later section so due to the fact that each isolated chip can carry 4 signals, the number of chips used is 16x2 isolated stages = 32 chips. For space economy both sides of the board were used, so 16 chips were placed per side.

A block diagram of actual layout of the isolating buffer chips is shown in Figure 3.3. One can see the isolating distances between the planes.

Another important issue is the equal distribution of the voltage difference between the planes. Due to the fact that there are three planes where the FPGA side plane is attached to the inverter ground and the Real Time Computer connector side plane which is attached to the Real Time Computer ground, there is the middle side plane left not being attached to any known potential. Since a floating potential for this middle side plane is not wanted nor permitted due to possible inequality in the voltage distribution between the planes (the middle side plane may display 700V difference over the FPGA side plane and 300V difference over the Real Time Computer connector plane which would violate the creepage and clearance distances and cause a voltage breakdown) an array of resistances to distribute the voltage equally. For that reason 12 x 1 MΩ resistances are used, 6 for each plane gap and the voltage drop per resistance is 83.3V. The number was chosen like this and not 1 per gap so as to avoid also voltage breakdown between the resistances due to inadequate creepage distance. The resistances are aligned in a straight line in order to preserve the creepage distances and so the security standard. This specific layout of the resistances on the different planes can be seen in Figure 3.4 with the connection to the ground planes done with vias that are easily recognizable.

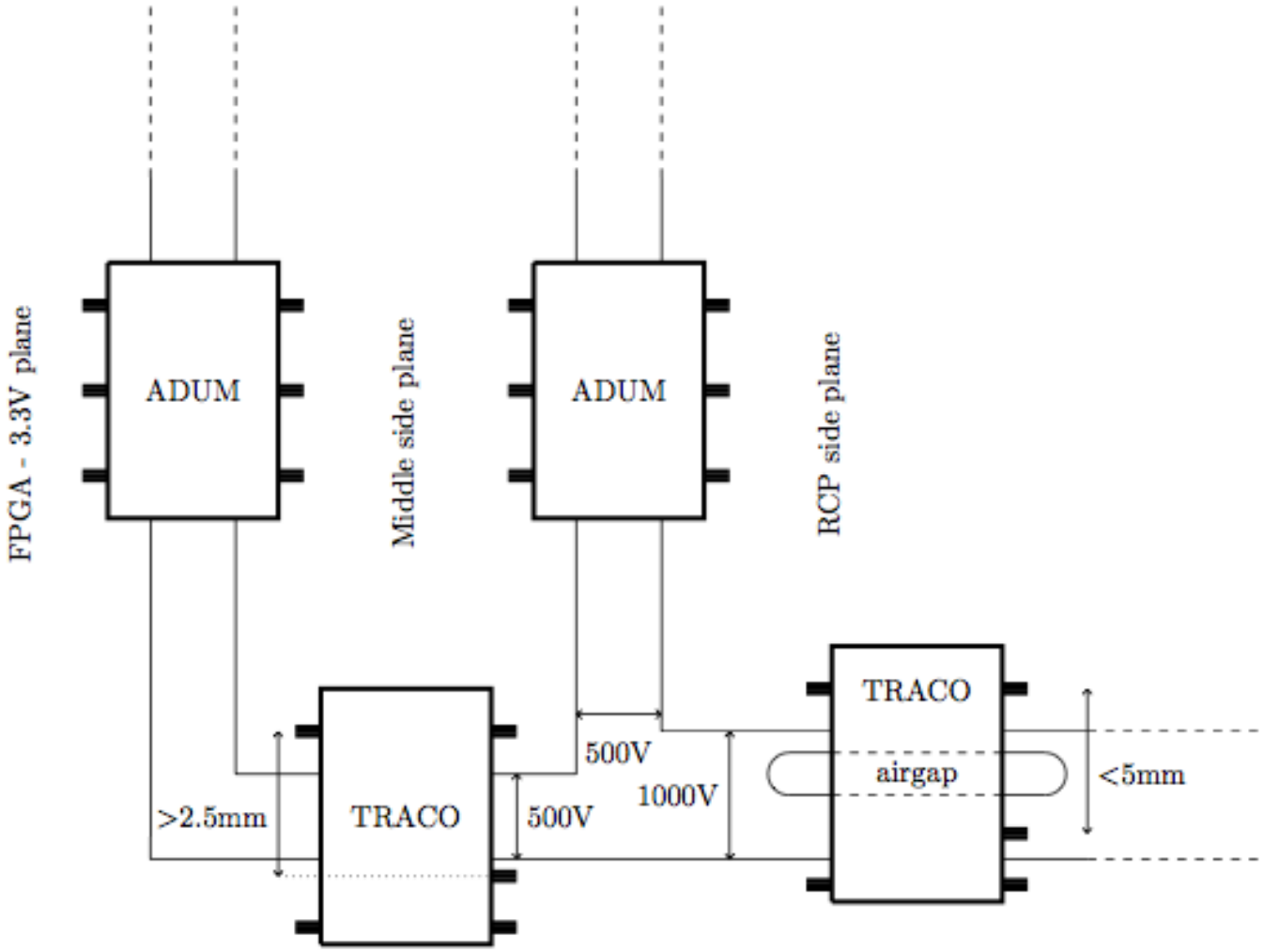

FPGA - 3.3V plane
ADUM
Middle side plane
ADUM
RCP side plane
TRACO
airgap
<5mm
500V
1000V
>2.5mm
TRACO
500V

Figure 3.2

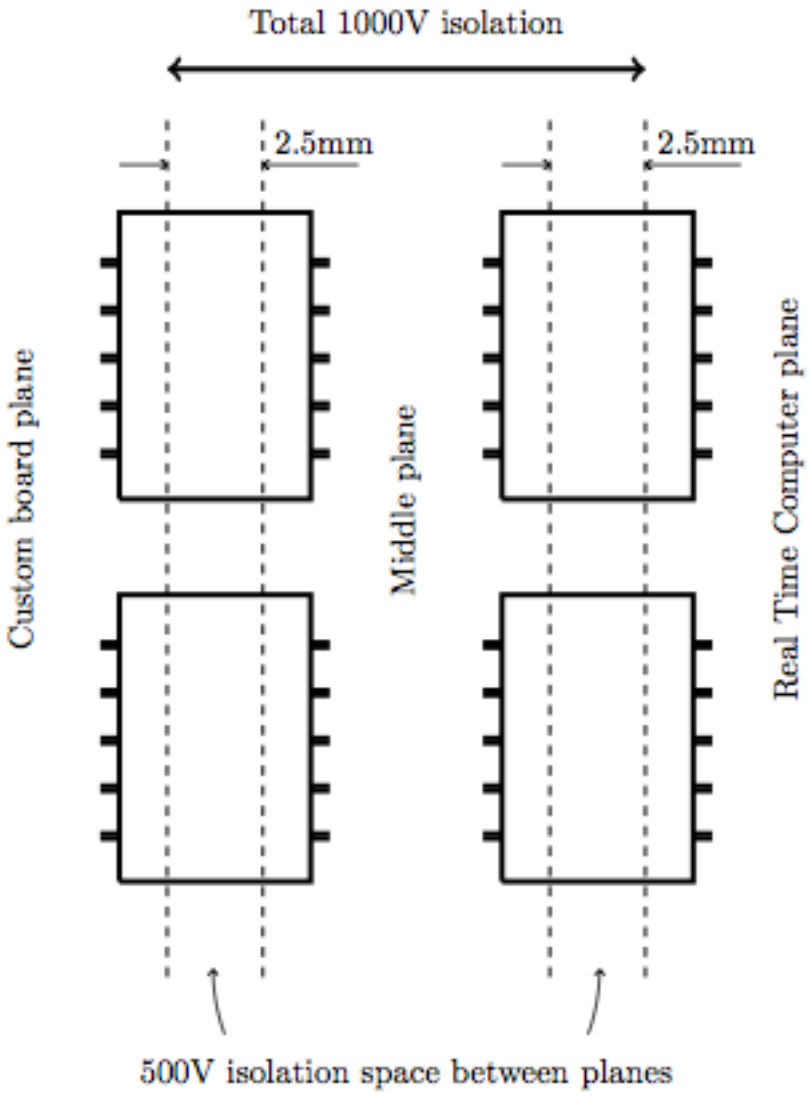

Total 1000V isolation
2.5mm
2.5mm
Custom board plane
Middle plane
Real Time Computer plane
500V isolation space between planes

Figure 3.3

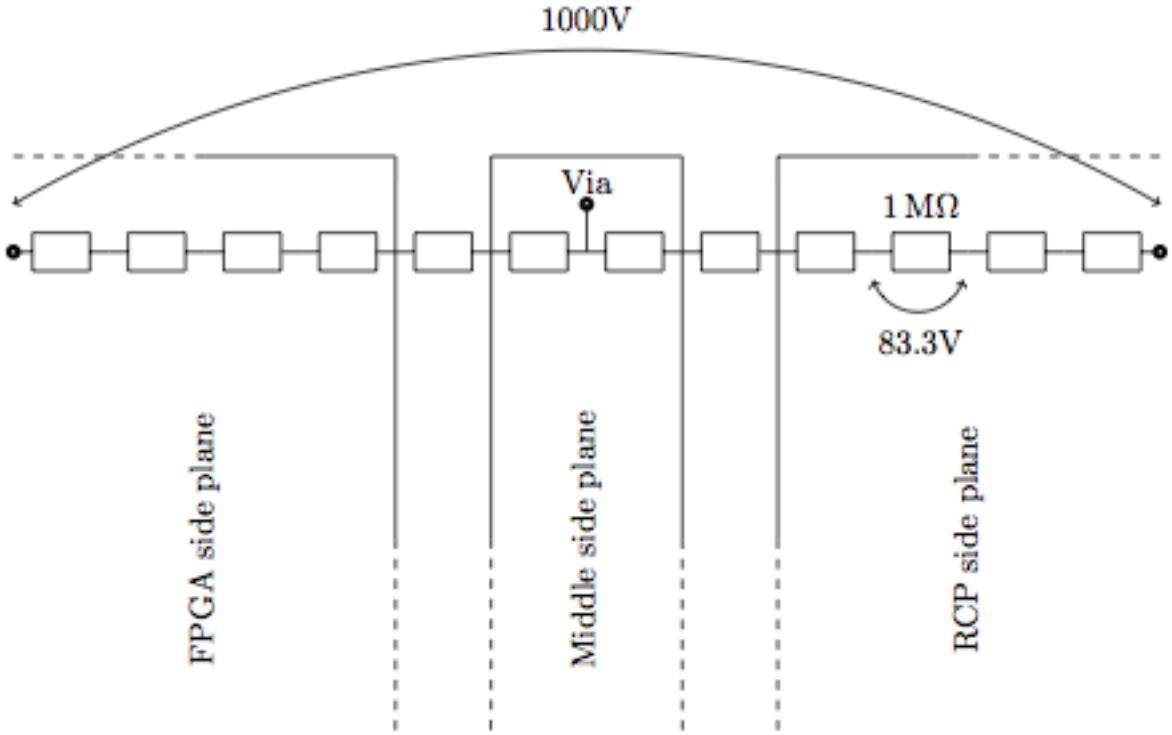


Figure 3.4

3.3. Levelshifting

The inverter interface with the board exchanges 74 signals with the inverter system through the two ports (80-pin and 50-pin minus the ground pins) that it has. Some of the signals require a voltage level of 5V and some of them 3.3V. For this reason voltage levelshifter chips are required to bring the voltage level of the signals to the one that is dictated by the inverter hardware. Some of the voltage levelshifters are shifting the voltage level of the signals from 3.3V to 3.3V and for that case the level shifters are used only for modularity, in case the inverter system changes.

The voltage level shifter chips used are the 74ALVC164245 16-bit dual supply translating 3-state transceiver from NXP semiconductors. The primary supply is 3.3V for all and for some of them the secondary supply voltage is either 3.3V or 5V. Due to the fact that the output of the voltage level shifter chips can also be tristate this means that the output pins can float. In order to avoid fuzzy outputs pull-up or pull-down resistors are used at the output just before the inverter ports.

The tristate capability of the voltage levelshifter chips is used for one voltage levelshifter that acts as a data line arbiter for between the Real Time Computer interface and the FPGA on the board. As it was explained in a previous chapter, the Rapid Control Prototyping system used 32 data lines, half of them as input and half of them as outputs. The fact that the FPGA does not have enough I/O pins means that the same pins will be used as outputs and as inputs depending on the case. When the FPGA pins are used as inputs then the signals coming in the FPGA board are taken from the outputs of the voltage levelshifter, which are driven in the primary from the Real Time Computer interface. When the FPGA pins are used as outputs then the same

lines are used from the FPGA to drive signals directly to the Real Time Computer overriding the voltage levelshifter. Because the lines are hardwired this means that the output of the voltage levelshifter should be tristated so as not to react with the rest of the circuit.

The output enable pins of the voltage levelshifters is controlled by the CONF pin of the Rapid Control Prototyping System as well as one of the pins of the FPGA board with an NAND boolean operator achieved by the use of MOSFETS. This actually means that the signal output to the inverter is blocked if the Rapid Control Prototyping system is not ready or the FPGA doesn't allow it. This can be seen in Figure 3.5

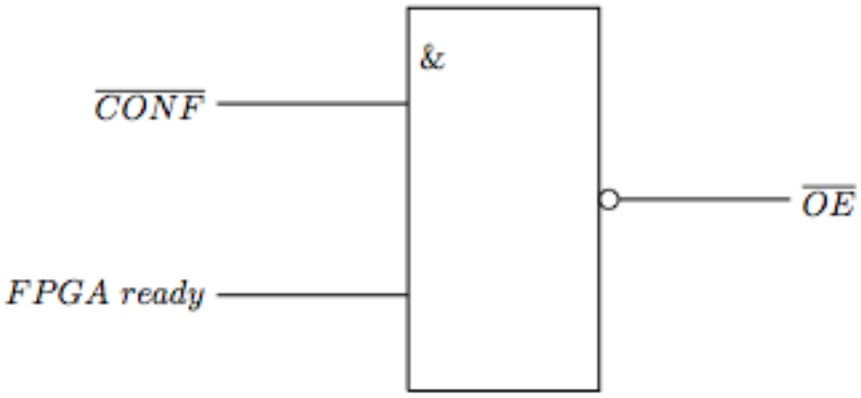


Figure 3.5

3.4. FPGA

The FPGA is a semiconductor device that can be programmed after manufacturing. Instead of being restricted to any predetermined hardware function, an FPGA allows the programming of product features and functions, adaptation to new standards, and reconfiguration of hardware for specific applications. An FPGA can be used to implement any logical function that an application- specific integrated circuit (ASIC) could perform, but the ability to update the functionality at any time without any change in hardware offers advantages for many applications.

Unlike previous generation FPGAs using I/Os with programmable logic and interconnects, to- day's FPGAs consist of various mixes of configurable embedded SRAM, high-speed transceivers, high-speed I/Os, logic blocks and routing. Specifically, an FPGA contains programmable logic components called logic elements (LEs) and a hierarchy of reconfigurable interconnects that allow the LEs to be physically connected. The LEs can be configured to perform complex combinational functions, or merely simple logic gates like AND and XOR. In most FPGAs, the logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory.

As FPGAs continue to evolve, the devices have become more integrated. Hard intellectual prop- erty (IP) blocks built into the FPGA fabric provide rich functions while lowering power and cost and freeing up logic resources for product differentiation. Newer FPGA families are being developed with hard embedded processors, transforming the devices into systems on a chip (SoC).

The FPGA board used for this application is the DE0-Nano experimental board made by TERA- SIC and the manual.

The experimental board is connected on a Trough-Hole-Technology adaptor mounted on the board so as to easily extract and replace in case of failure. The experimental board has 106 pins from which 88 are used to interface through signals with the custom board. Some of the pins are configured as inputs and some as outputs. 16 pins are used as dual purpose due to lack of adequate number of pins, as stated earlier, so depending on the operation, the pins are configured either as inputs or outputs, interacting with the Rapid Control Prototyping system.

The experimental board has the Cyclone¨ IV EP4CE22F17C6N FPGA from ALTERA which has 22,320 Logic elements (LEs), 594 Embedded memory (Kbits), 66 Embedded 18 x 18 multipliers, 4 General-purpose PLLs. It also has an On-board USB-Blaster circuit for programming and an FPGA Serial Configuration Device (EPCS), which is used to store a permanent firmware that is loaded automatically when the board is power up.

Additionally, it has two 40-pin Headers (GPIOs) provides 72 I/O pins, two 5V power pins, two 3.3V power pins (which in our case are not connected) and four ground pins, one 26-pin header which provides 16 digital I/O pins and 8 analog input pins to connect to analog sensors (which in our case are not used).

As for memory devices, it has 32MB SDRAM and 2Kb I2C EEPROM. The general user in- put/output includes 8 green LEDs, 2 debounced push-buttons, 4 dip switches, a G-Sensor ADI ADXL345 3-axis accelerometer with high resolution (13-bit), a NS ADC128S022 8-Channel 12-bit A/D converter and an on-board 50MHz clock oscillator. The power supply of the board is done by supplying 5V to the board and the on-board supply of the experimental board feeds 3.3V to the experimental board. For this reason, the 3.3V supply of the custom board is not mixed with the 3.3V supply of the experimental board by not connecting the pins.

For the specific application the FPGA, the EPCS and the 50MHz clock oscillator are used. All the other hardware used is included in the inverter hardware that the custom board is connected to.

3.5. Supply

The supply of the custom board has a specific strategy. The inverter hardware provides, through its own supply circuitry, 5V to the custom board and the supply circuit of the board uses that to distribute its own supply system.

The board has three compulsory supply voltages used for achieving 1000V isolation from the inverter hardware. The supply system of the custom board uses three supply and ground planes and the voltages used are 5V, 5V and 3.3V.

In order to be able to have three supply and ground planes, three supply units had to be used. So, two 5V-to-5V TES 1-0511 isolated power supplies from TRACO and one 5V-to-3.3V SC4524C switching regulator from SEMTECH were used.

The 5V supply planes are used in order to feed the isolating chips laying on the two isolating planes and no voltage output regulation was needed. For the 5V supplies it can be noticed from the datasheet that each supply chip can provide 200mA to its load. For the Rapid Control prototyping side plane the 5V supply has to feed 16 ADUM chips which it can be extracted from the data sheets that they consume 2.2mA per chip so 35.2mA in total which is acceptable. For the middle side 32 ADUM chips have to be supplied so the total load is 70.4mA which is also within the supply chip range.

The 3.3V supply plane is used to feed the auxiliary circuitry on the 3.3V plane and the voltage output of this supply is regulated. Due to the fact that the input voltage specifications of the chip is a voltage range, the user of the chip can choose the output voltage of the chip through a regulating voltage divider with precision resistors. The specific regulator chip includes only the switching component, so the buck converter design had to be made. During the design of the layout of the buck converter special placing considerations were made in order to reduce the EMI as much as possible so that it wouldn't affect the signals riding through the nearby lines. For the layout of the components, the switching current loop was minimized, no regulating components were placed on the bottom side of the power inductor so that the regulating of the supply wasn't affected,the nodes with high voltage alternating were made big to avoid high dv and thermal bias were placed dt right below the supply chip to achieve higher heat dissipation.

The load calculated for the 3.3V supply is done by considering the already existing chips on this plane side and possible circuit extensions for future applications. The already existing chips include 6 voltage levelshifters which consume maximum 500uA per channel so in total (for 6 voltage levelshifters and 16channels each) 48mA. Also the 3.3V supply has to feed 16 ADUM chips which consume 2.2mA each so 35.2mA. So the total consumption rises to 83.2mA. The chip limit is 2A but that is not considered as oversizing due to the fact that this side plane is very easily extended with new components in case of change of application, so temporary oversizing will avoid the burden of resizing the supply unit in later applications.

3.6. Real Time Computer port

The custom board interfaces with the Rapid Control prototyping system through a 68 pin connector from which the 64 are used as signals and the rest are ground pins. The pin specification includes two groups of parallel digital pins B0 −B15 and C0 −C4, two groups of serial-parallel interface pins D0 − D15 and E0 − E15 with their address pins A0 − A4 and the control pins CS0, CS1, INT, CONF, DONE and CLK.

The Rapid Control Prototype interface has input and output signals. The output signals to the Real Time Computer don't require filtering but only over current protection. So for most signals 470Ω of series resistance is enough. For the INT signal this resistance is too big considering the parasitic inductance and capacitance of the cable connecting the custom board with the Real Time computer creating a low-pass filter that attenuates the signal. The problem created is that the INT signal which is a clock signal given to the Rapid Control Prototyping System to trigger model execution gets distorted and the triggering becomes untrustworthy. So after experiments a zero resistance was chosen to be used for that particular signal.

For the fast input signals from the Real Time Computer such as the D and E as well as the control signals used for the transmission a simple RC low pass filter is used in combination with a big pull-down resistance for security reasons in case the Real Time Computer is not connected. The RC filter is calculated for transmission frequency of 4MHz maximum and the values of the components are C = 100 nF and R = 100 Ω.

3.7. Inverter ports

The interface of the inverter and the custom board includes a 80-pin and a 50-pin connector from which 74 pins are used for signal transmission. The signal output to the inverter are connected directly to the voltage levelshifters. The signal inputs from the inverter are first passed through an RC low pass filter to attenuate the higher harmonics and distortions especially for the fast signals such as the Analog-Digital-Converter

signals. The fast signals are passed through a C = 22 pF and R = 470 Ω low-pass RC filter and the slow signals are passed through a C = 220 pF and R = 470 Ω low-pass RC filter.

An example of fast signals are the Master Input Slave Output bits transmitted from the Analog- Digital-Converters at the inverter hardware. Due to the PCB trace distance covered to reach from the Analog-Digital-Converter chip to the FPGA a high possibility of distortion has to be faced. So use of a low-pass RC filter is obligatory.

3.8. Jumper strategy

The ideal configuration would be all of the signals that come from the Rapid Control Prototyping system to be routed through the FPGA, the FPGA would operate on the data and then output the correct control signals to the inverter and the correct data back to the Real Time Computer. Due to limited number of pins of the FPGA some of the signals going to the inverter can be routed either from the FPGA or the RCP directly without passing through the FPGA. This can be achieved with jumpers.

The specific jumpers used in this application are soldering jumpers than can connect the a specific inverter signal with a Rapid Control Prototyping system pin or a FPGA pin or both.

An example of this is control of the AC relays of the inverter directly by the Real Time Computer but the protective VHDL module inside the FPGA would flag the SAFEBREAK signal, which would trip the relays instantly. This means that the jumpers related to the AC relays will be soldered appropriately to connect the three signals to the Rapid Control Prototyping system directly and the jumper related to the SAFEBREAK signal will be soldered so as to connect the output pin of the FPGA to the inverter system directly.

The table with the signal-jumper correlation is displayed in table 3.1.

Table 3.1

Inverter signal	FPGA port (header)pin	RCP signal
Reset bridge	GPIO.(0)21	B_{11}
Reset boost converter	GPIO.(0)20	B_{10}
$\overline{ERROR}$	GPIO.(0)18	C_0
$\overline{SAFEBREAK}$	GPIO.(2)6	B_0
Relay AB	GPIO.(0)32	B_3
Relay BC	GPIO.(0)28	B_1
Relay CA	GPIO.(0)30	B_2
Relay Test OK	GPIO.(0)24	C_1
Fan Control 1	GPIO.(2)4	B_{12}
Fan Control 2	GPIO.(2)5	B_{13}
AFCI Clock	GPIO.(0)26	B_{14}
Varystor error	GPIO.(0)25	C_2
Supply voltage fail	GPIO.(0)23	C_4
5V Power good	GPIO.(0)22	C_3
Isolation Test 1	GPIO.(2)7	B_4
Isolation Test 2	GPIO.(0)31	B_6
Isolation Test 3	GPIO.(0)33	B_5
Misc signal 1	GPIO.(2)9	B_9
Misc signal 2	GPIO.(0)27	B_8
Misc signal 3	GPIO.(0)29	B_7

3.9. Schematic and board layout

The top and bottom side of the custom board layout with the whole schematic are illustrated in the appendix A.

4

FPGA VHDL modules

4.1. General system architecture

The present chapter introduces the software developed for the Field Programmable Gate Array (FPGA) chip mounted on the custom board in order to implement fast controllability of the inverter hardware. The design suite used to implement the code was Altera Quartus II and the coding language was VHSIC Hardware Description Language (VHDL). This design software allows for actual code implementation through code files as well as graphical representations of systems for overall view of the system. The procedure followed in this master thesis project was to create discrete blocks of software code and join them together through one block design graphical file. In this section the overall system is presented as a block diagram and a short decryption of each block will be presented. The core blocks of the project will be discussed in detail in each section and the development procedure will be analysed.

The overall FPGA system architecture block diagram is illustrated in Figure 4.1.

From the system block diagram it is evident that the 50MHz external crystal cannot provide with a adequate clock frequency for the system so a Phase Locked Loop is necessary. Using an embedded PLL dedicated hardware inside the chip the frequency was elevated to 150MHz. The same clock is distributed throughout the whole system so that there is no cross-clocking domains and no possibility of violating the setup and hold times of any registers. In that way all data are available to the interconnected system with no obscurity..

The main blocks of the FPGA hardware system are the Rapid Control Prototyping controller, the analog-to-digital controller, the pulse width modulation generator, the hardware protection system and the Read Only Memory lookup table. Except for the block there is also a flexible bus system that receives data from some blocks and delivers them into some others.

The RCP controller is responsible for the communication between the custom board and the Real Time Computer which is actually the custom Simulink model block running on the Real Time Computer. The RCP controller in the front end interfaces with the Real Time Computer through some FPGA pins that are responsible for the configuration and data transmission and in the back end it connects to the 512bit RCP write bus where it writes the data coming from the Real Time Computer and to the 512bit read bus from where it reads data ready to be transmitted to the Real Time Computer.

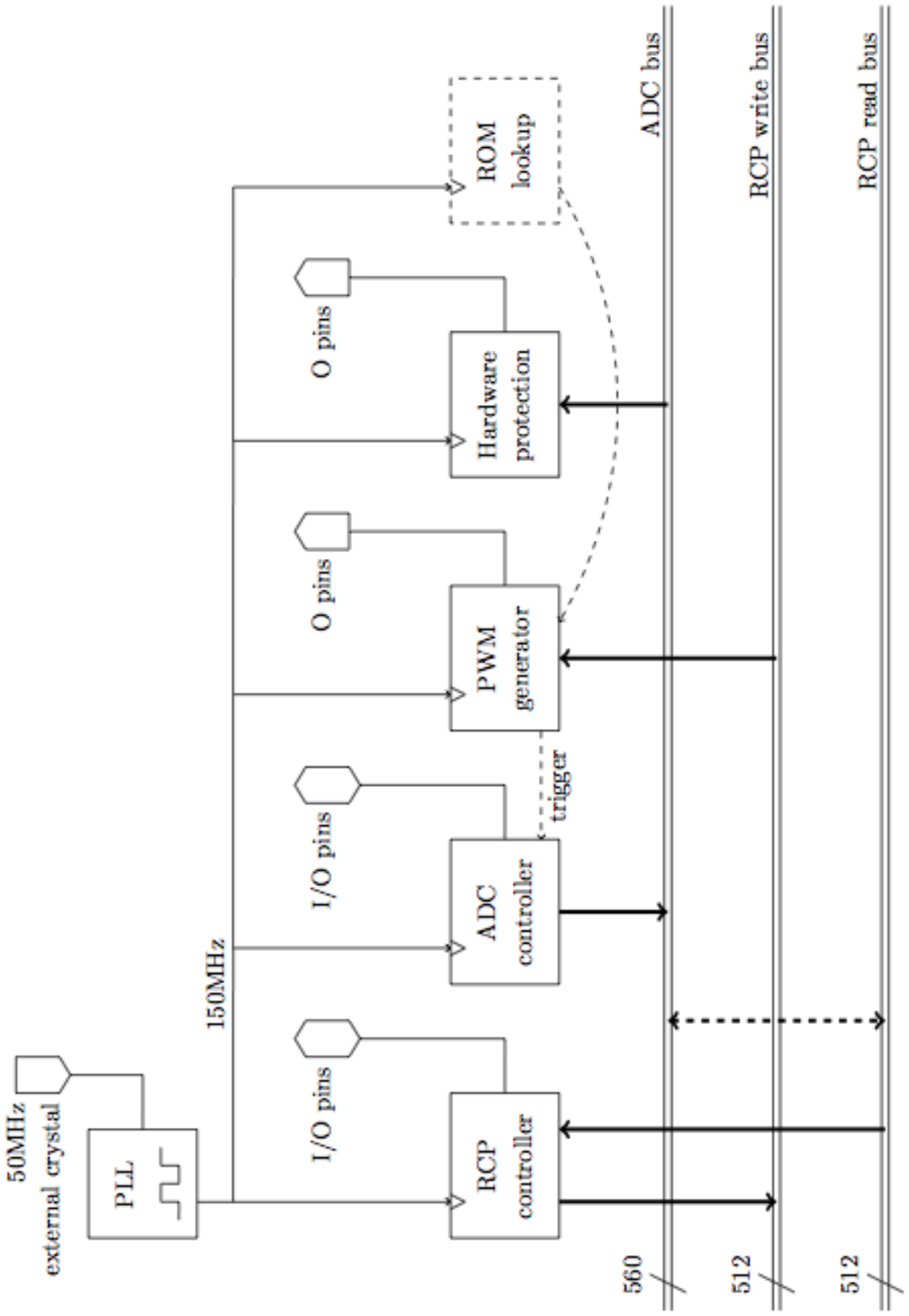

50MHz
external crystal
PLL
150MHz
I/O pins
RCP controller
I/O pins
ADC controller
trigger
O pins
PWM generator
O pins
Hardware protection
ROM lookup
ADC bus
RCP write bus
RCP read bus
560
512
512

Figure 4.1

A comment here is that the bus width is 512bit because the actual data being transmitted in or out of the Real Time Computer are 32x16bit=512bit. The addressing of the RCP bus follows the sequential order of the data in the Simulink model block. For example for the input registers, the first 16bit data value being received at the RCP (16bit integer value #0 out of 32) is stored at address 0x0000 of the RCP read bus (thus from line #0 to line #15, with line #15 acting as the Most Significant Bit). Also, the 10th data value that the RCP sends to the custom board gets stored at address 0x000A of the RCP write bus which means from line #144 to #159 with line #159 acting as the MSB.

The ADC controller is responsible for configuring the 2 16-channel and the three single-channel analog-to-digital converters, clocking them and fetching the data transmitted by them. The front end of the ADC controller comprises of the control and data signals that get sent through the FPGA input and output pins. In the back end, the ADC controller interfaces with the rest of the system though the 560bit ADC bus where it constantly renews the values so that they are fresh for use by the rest of the system. The bus width is 560bit because the two 16-channel analog-to- digital converters provide 16x16bit values each, thus 512bits of values, and the three single-channel analog-to-digital converters provide 3x16bit values, so 48bit. The data lines sum up to 560bit and the ADC bus follows the same addressing philosophy. The first 256 lines represent the 16x16bit data fetched but the first 16-channel analog-to-digital converter, the next 256 lines from the second 16-channel converter and the rest represent the 3x16bit values acquired from the single-channel analog-to-digital converters. We can see that in order for the Real Time Computer to be aware of the ADC values a partial connection between the two buses is possible so that the measured values are connected to the appropriate data lines for the Real Time Computer to use. Another important thing to notice is that the ADC controller gets triggered from the PWM generator. This means that the analog-to-digital conversion and the fetching of data happens at predefined instants depending on the switching frequency of the PWM generator as explained later.

The PWM generator is responsible for the creating of the carrier triangle, trigger the analog-to- digital converter FPGA control block, create an innovative PWM comparison method with real-time flexible deadtime creation and pulse suppression, self-checking for short-circuiting situations and blanking and outputting the switching pulses to the FPGA pins. Thus, in the front end, the PWM generator uses only FPGA output pins to output the safe switching pulses and in the back end it interface with the rest of the system just by reading PWM reference values from the buses. De- pending on the controlling scheme, the PWM block can read PWM reference values from the RCP write bus, meaning that the Real Time Computer defines the PWM reference that gets passed on to be compared with the triangle. It can also read from a Read Only Memory, where predefined sinusoidal shapes are saved, recalled and rescaled using embedded multipliers to create a reference to compare with the carrier triangle.

The Hardware protection block has a constant connection to the ADC bus and reads current and voltage values measured from the analog-to-digital converters at important points of the inverter hardware. Depending on the values, the predefined limits set in the code and some recalculations

done inside the block, the hardware protection block can set off the SAFEBREAK signal and open the AC relays through the FPGA output port connected to them. This operation is necessary in grid connected or sensitive hardware systems so as to diminish the possibility of hardware damage in cases of operation or control failure. The fact that the protection block is inside the FPGA and not inside the Simulink model blocks affirms the necessity of an intermediate much faster controlling board between the Real Time Computer and the actual hardware.

The Read Only Memory block is a lookup table that stores pre-calculated sinusoidal (not necessarily pure sinusoid) values in order to set as reference to the PWM block. This block is not a necessity for the operation of the grid tie inverter because the sinusoidal reference values in the case of a grid tie inverter come from locking to the grid sinusoidal waveforms. Though, when the output of the inverter is connected with a load then there is no sinusoidal waveform to imitate so the sinusoidal references have to be existent. Due to the

fact that a small scale FPGA is not capable of multiple floating-point fast calculations, having a look up table is the only solution in that case. A convenient clue is that there is no need to save the whole sinusoidal waveform from 0 to 2π as there is quarter-wave symmetry. So only a quarter of the sinusoidal waveform is needed to be recalculated and saved.

4.2. RCP interface block

The Rapid Control prototyping block mainly represents the interface of the custom board side to the Real Time Computer. This is the block that controls the Simulink model block execution, the parallel data transmission sequence and speed, as well as the digital data lines. During each parallel data read transmission this block has to receive the data, assemble them and push them to the RCP write bus. Also, during a write sequence the block read the data from the RCP read bus and sends them to the Real Time Computer. The signal sequences were presented in a previous chapter and illustrated again in Figure 4.2.

The operations of this block is based on a Mealy Finite State Machine that creates a basic programming sequence. In the theory of computation, a Mealy machine is a finite-state machine whose output values are determined both by its current state and the current inputs. The state machine for the Rapid Control Prototyping block is illustrated in Figure 4.3.

In this state machine 7 states are introduced the interconnection of which is dictated by two inputs. The inputs of the state machine is the INT signal that controls the Simulink model block execution and thus the direction of the parallel data transmission and the clock signal CLK received by the Real Time Computer system. The states used in this finite state machine are: START(initial), PREPAREREAD, READHIGH, READLOW, PREPARERWRITE, WRITEHIGH, WRITEHIGH. The INT signal is produced by a small timer generator that creates a pulsating signal with defined frequency. The frequency of this signal is defined as a con- stant inside can the VHDL code and can be changed during compilation time. The current INT frequency is set to 8kHz (a counter value of 18750 @150MHz), which means that the Real Time Computer communicates with the custom board every 8kHz and exchanges data.

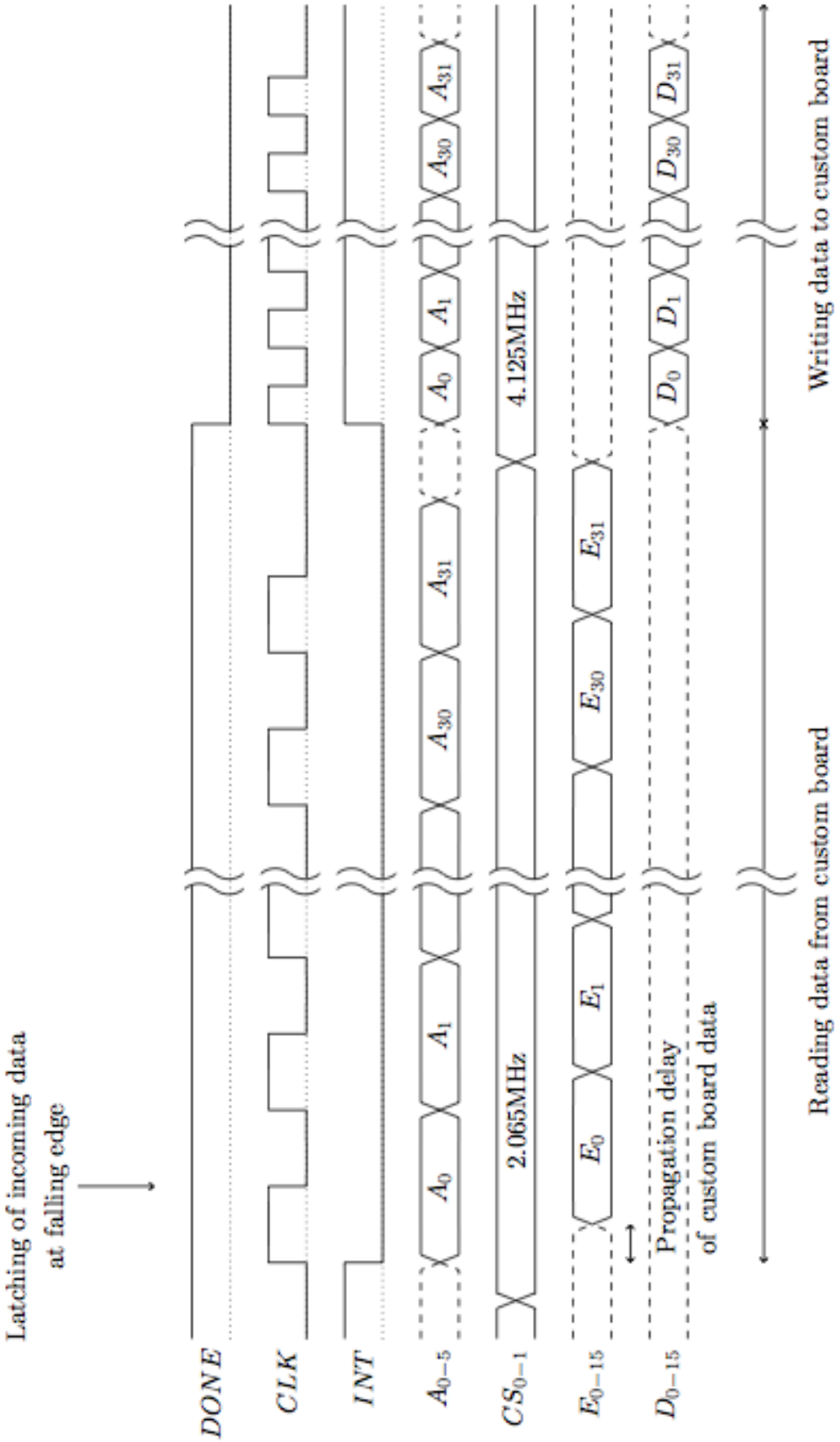

Latching of incoming data
at falling edge
DONE
CLK
INT
A_{0-5}
A_0
A_1
A_{30}
A_{31}
CS_{0-1}
2.065MHz
4.125MHz
E_{0-15}
E_0
E_1
E_{30}
E_{31}
D_{0-15}
Propagation delay
of custom board data
D_0
D_1
D_{30}
D_{31}
Reading data from custom board
Writing data to custom board

Figure 4.2

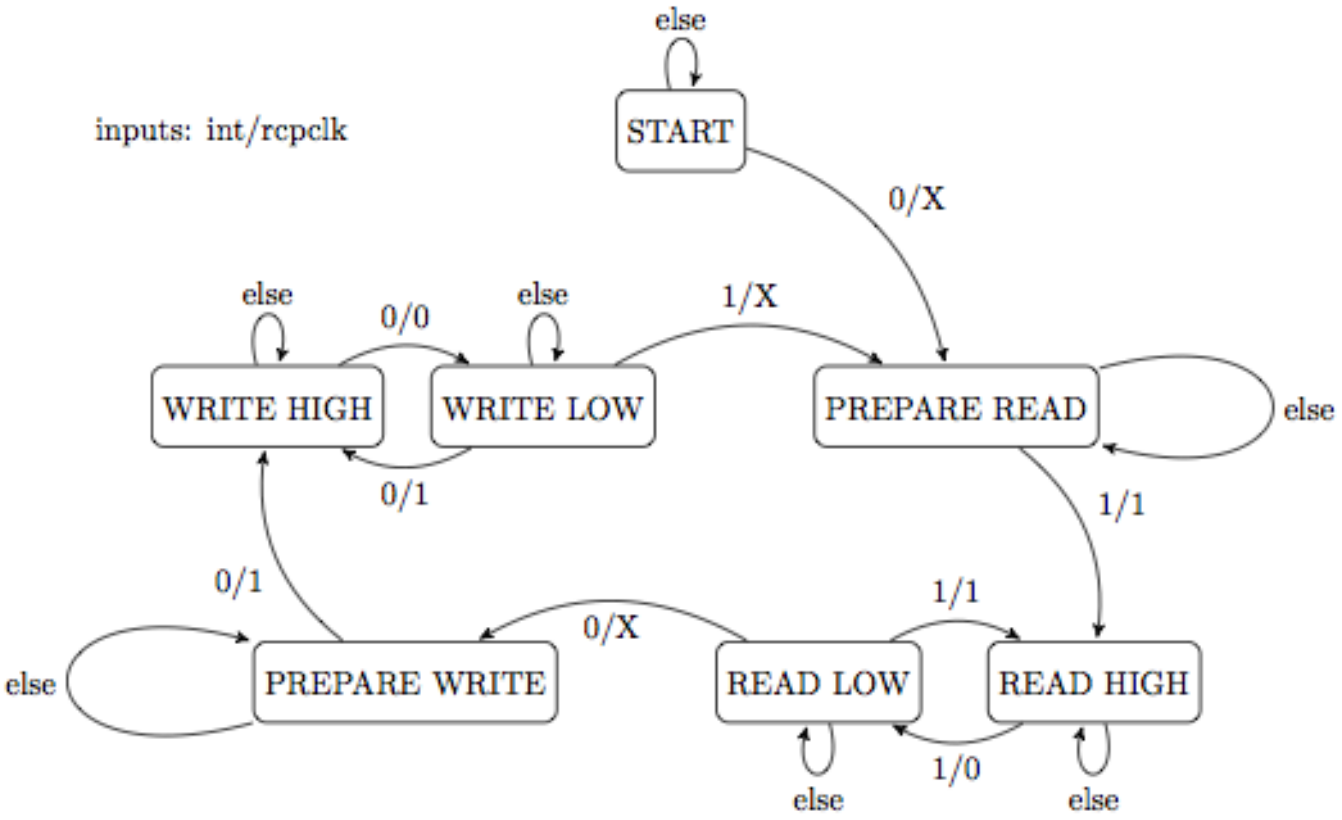


Figure 4.3

During a write data sequence, the block has already set the clocking frequency via the CS0,1 signals and awaits for the INT signal to be brought down. As soon as that happens the Real Time Computer places the address data on the A0−4 lines and starts clocking through the CLK signal. The block is responsible in fetching the corresponding data from the correct address at the rising edge of the CLK signal and place the data on the output lines E0−15 so that they are available at the Real Time Computer (also accounting for some delay time due to the propagation through the isolation level chips) at the falling edge of the CLK signal. The same procedure continues until all 32 data are transmitted and the system awaits for the read sequence while having already put the desired clock selection for the next sequence at the CS0,1. As soon as the INT signal is brought up the Real Time Computer puts the address signals at the A0−4 pins as well as the data to the D0−15 pins and starts clocking brought the CLK signal. After each 16-bit packet arrives the block stores it to the appropriate bus lines dictated but the address pins. The reading sequence ends when all 32 packets have been received. The output lines of the RCP write bus are registered with the system clock. The values of the RCP read bus are only dependent on the connection of these lines to the other blocks of the system. For example the analog-to-digital block stores its data to the RCP read bus and keeps them registered with the system clock.

The latching and sending of data is not limited to rising or falling edges of some critical signals. The finite state machines used in this project are sophisticated and fully flexible and configurable. There is a counter that counts the clock cycles for staying at each state and there is also a trick to avoid cycle delays. This creates full flexibility to fetch and send data at any time or multiple defined times during every state.

4.3. ADC interface block

The analog-to-digital converter block is developed based on the ADS7953 and ADS7883 chips from Texas Instruments. The ADS7953 analog-to-digital is 16-channel converter and two of them are used in this project and the ADS7883 is a single-channel converter and three of them are used in this master thesis project.

The signals used to operate the 16-channel analog-to-digital converters are the chip select signal CS, a clocking signals SCLK, the Master-Input-Slave-Output signal MISO and the Master- Output-Slave-Input signal MOSI. The two 16-channel analog-to-digital converters have different CS signals (SPI2 CS and SPI3 CS), different MISO and MOSI signals (SPI2 MOSI, ADC2 MISO, SPI3 MOSI, ADC3 MISO) but are clocked with the same SCLK signal (ADC23 CLK). The signals used to interface the single-channel analog-to-digital converters are again a chip select signal CS, a clocking signal SCLK and a Master-Input-Slave-Output signal MISO. The three single-channel analog-to-digital converters share the same chip select signal (SPI1 CS) and clock signal (SPI CLK) but have different MISO signals (SPI1 A MISO, SPI1 B MISO, SPI1 C MISO).

The signal sequence of the two 16-channel analog-to-digital converters are illustrated in Figure 4.4. The latching of the MISO bit is done in every falling edge of the clock signal and the analog- to-digital chips latch their counterpart of the data, the MOSI bits, at every rising edge of the clock signal. That is why the clock signal is centred between the edges of all MISO and MOSI pulses, so that the data is stable in order to be latched.

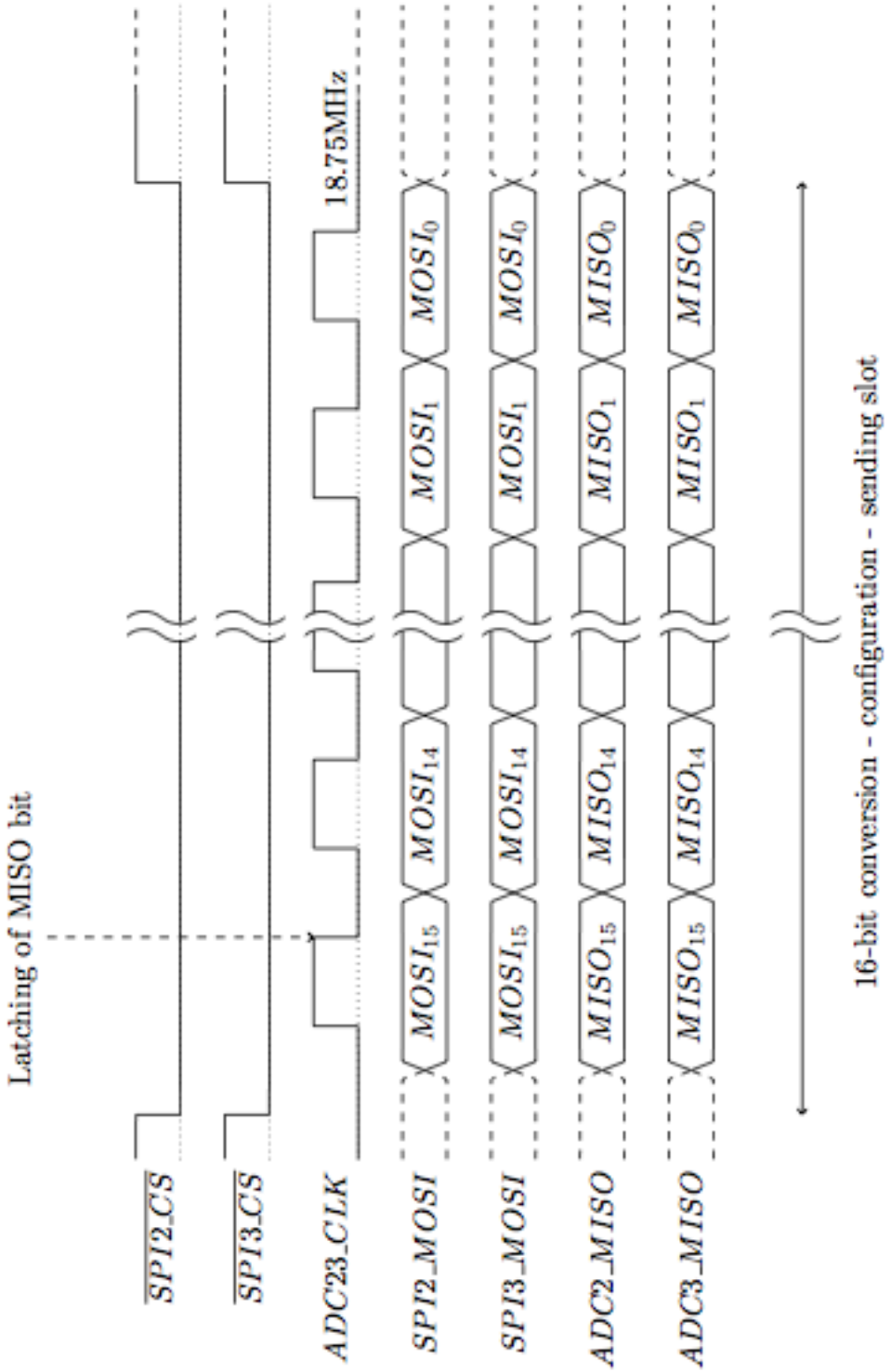


Figure 4.4

The finite state machine used to operate the signal sequence of the 16-channel analog-to-digital converters is illustrated in Figure 4.5.

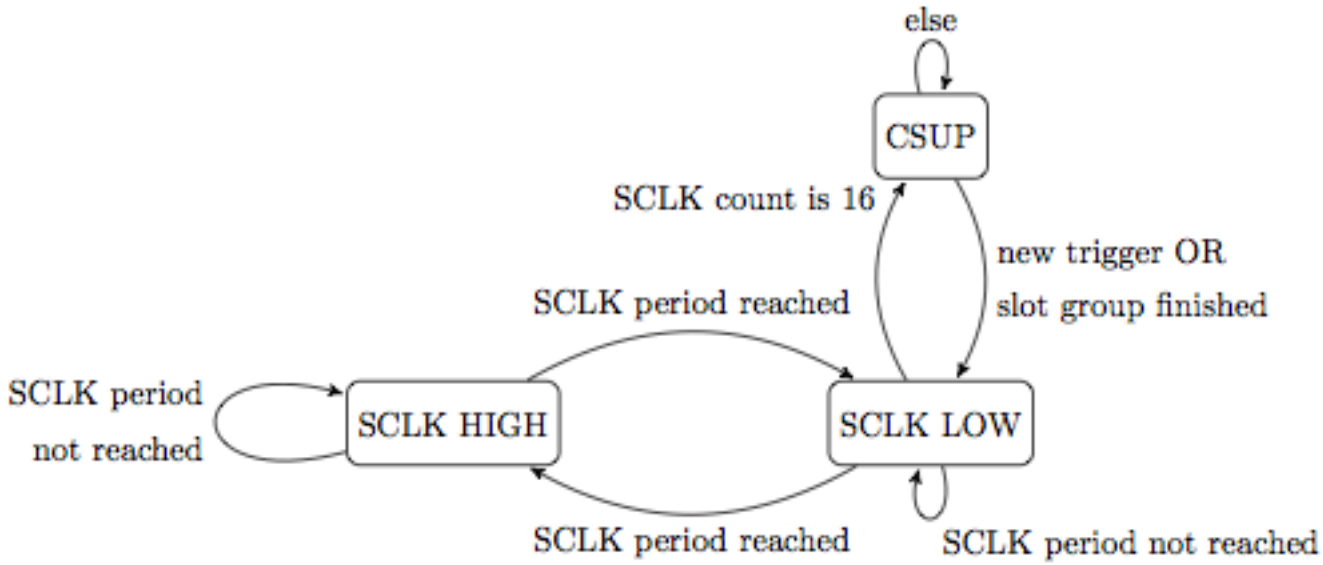


Figure 4.5

The states used in this finite state machine are CSUP, SCLKLOW and SCLKHIGH. The inputs used to define the transitions between the states are the trigger input from the Pulse Width Modulation block described later, but after edge recognition, a counter used to count the stay-at- state cycles, a counter used to count the slots and a counter used to count the slot groups. The finite state machine starts at the CSUP state and awaits either for the trigger signals from the PWM block or the state counter to reach a specified delay number and start a new slot. When inside the slot, the sequence of the states is from SCLKLOW to SCLKHIGH until all 16 bits are sent and received and then the state goes back to CSUP. The clock frequency used to clock the 16-channel analog-to-digital converter chips is 18.75MHz.

The operation of this type of analog-to-digital converters is based in 16-bit slots during each one of which the next measuring channel is configured through the MOSI bits, a new measurement is taken inside the chip and the converted measurement from the previous slot is sent back to the Master through the MISO bits. By configuring the analog-to-digital converters different measure- ments (from different channels) in the desired sequence can be taken. The number of the slots has to be enough so as to account for all measurements to be configured, converted and sent back to the Master. As already mentioned, the operation of this type of analog-to-digital converter postulates that the measurement - conversion of slot N has to be configured at slot $N - 1$ and sent back to the Master bit by bit at slot $N + 1$. So for example, for 9 measured channels in one ADC triggering cycle (triggered by the PWM block) the minimum number of slots needed is 11. This example is illustrated in Figure 4.6.

In the case of the single-channel analog-to-digital converters, the operation is similar but there is no need for chip configuration due to the fact that there is only one channel being measured per converter. Thus, the only slot delay accounted for the single-channel analog-to-digital converters is the slot delay between measuring - converting and sending back the data to the Master. The signal sequence for the single-channel analog-to-digital converters is shown in Figure 4.7.

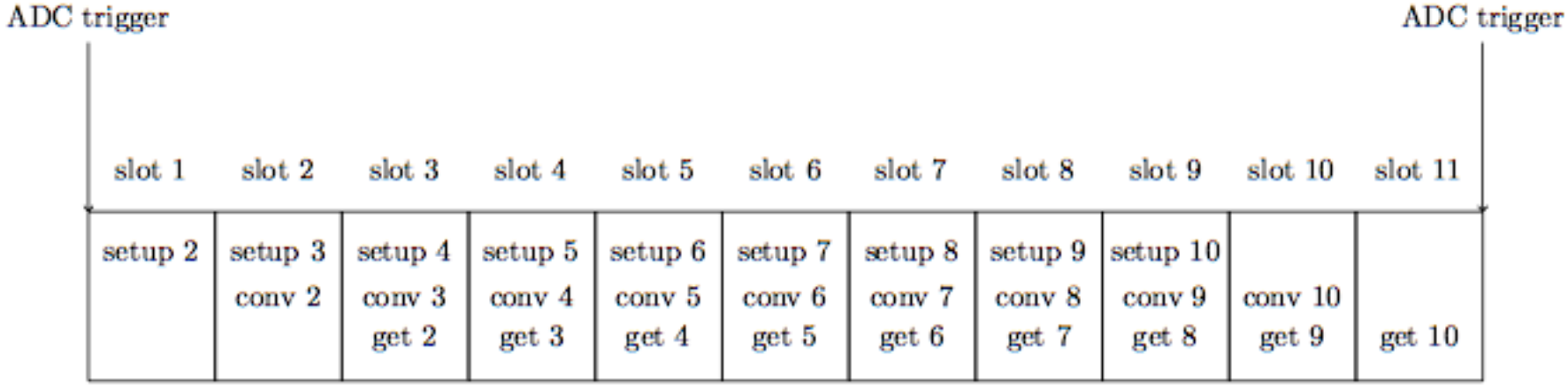

ADC trigger
ADC trigger
slot 1
slot 2
slot 3
slot 4
slot 5
slot 6
slot 7
slot 8
slot 9
slot 10
slot 11
setup 2
setup 3
conv 2
setup 4
conv 3
get 2
setup 5
conv 4
get 3
setup 6
conv 5
get 4
setup 7
conv 6
get 5
setup 8
conv 7
get 6
setup 9
conv 8
get 7
setup 10
conv 9
get 8
conv 10
get 9
get 10

Figure 4.6

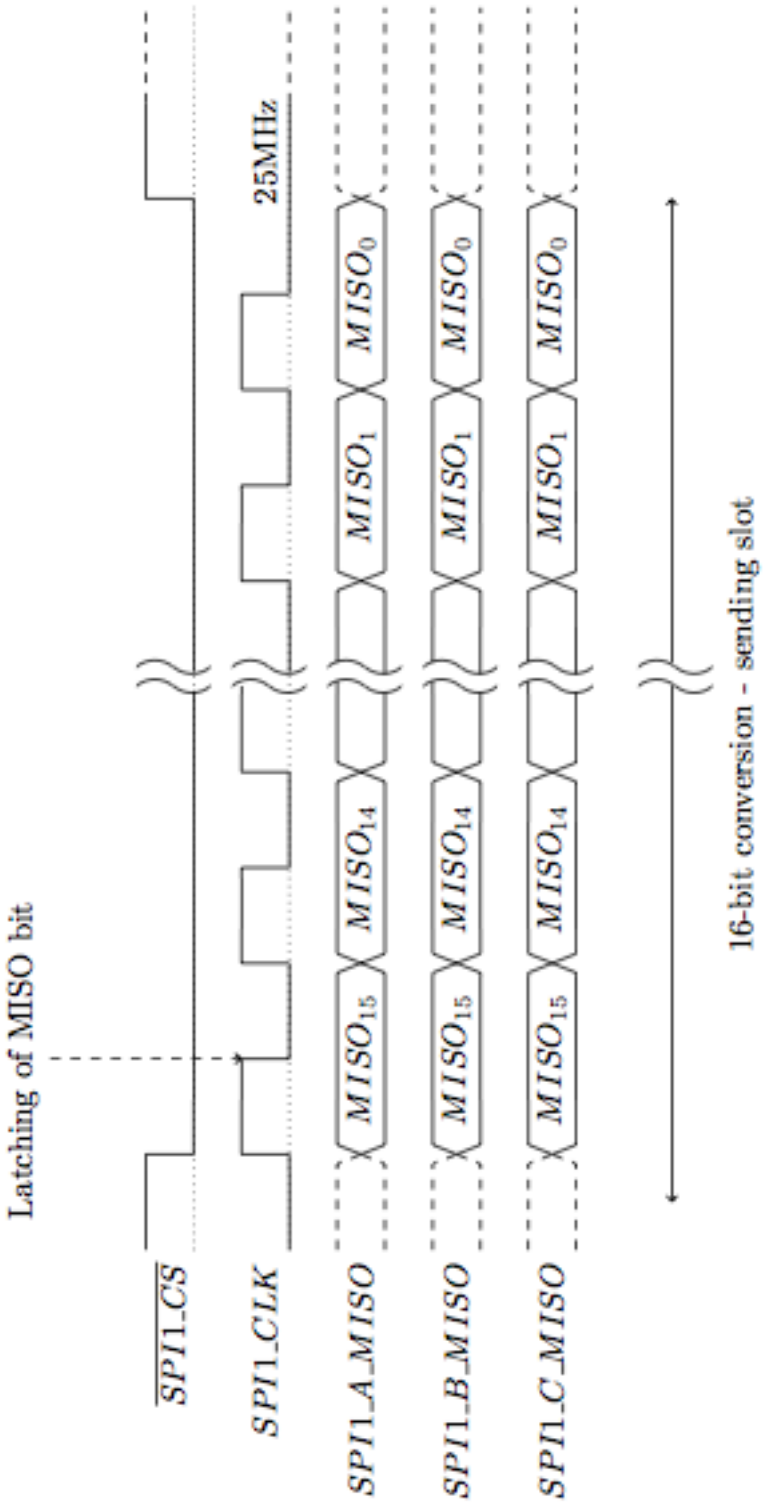


Figure 4.7

The finite state machine has the same states, it starts with CSUP but it continues with SCLKHIGH due to a minor modification of the operation of the single-channel analog-to-digital converters. The clock frequency used to clock the single-channel analog-to-digital converter chips is 25MHz. The finite state machine diagram of the single-channel analog-to-digital converter is presented in Figure 4.8.

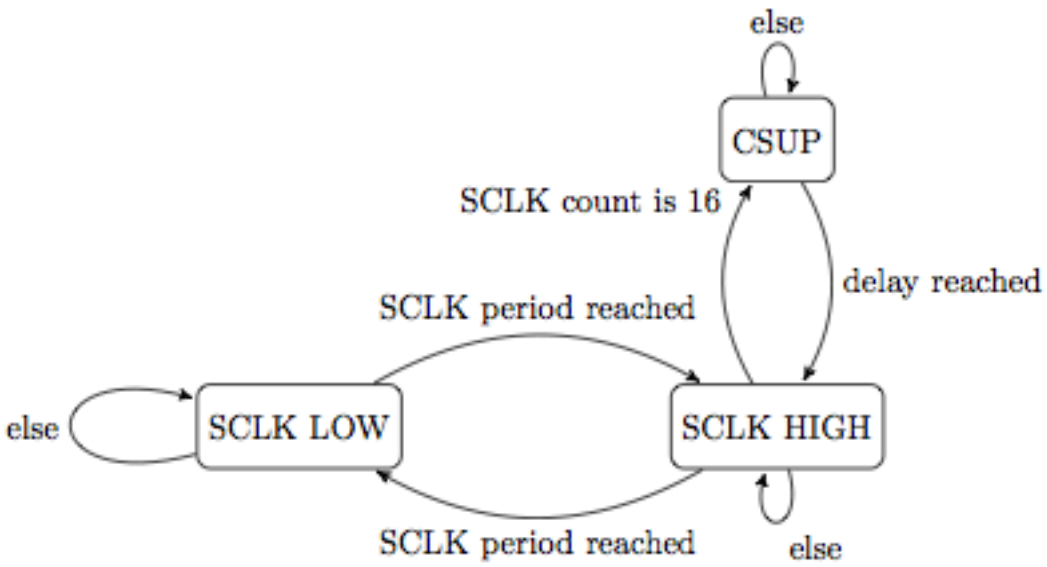


Figure 4.8

The operation of the single-channel analog-to-digital converters is continuous and does not depend on triggering by the PWM module. Thus, the slot sequence is that for every slot there is a conversion and a simultaneous sending of the converted data from the previous measurement. Figure 4.9 displays this operation.

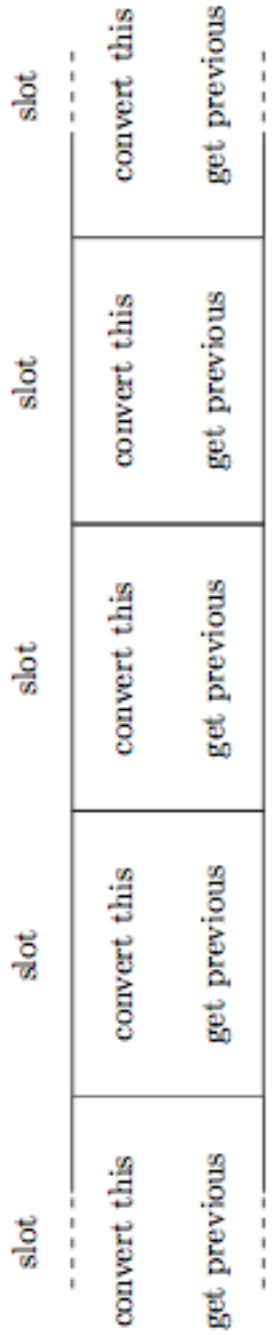


Figure 4.9

4.4. PWM generation block

One of the heart-forming parts of the FPGA system is the Pulse Width Modulation block. Its role is to create a pulse pattern depending on some input variables that drive the inverter bridge switches. The method used is a carrier-based PWM with reference comparison, inherent deadtime calcula- tion, real-time pulse width fixation and total flexibility of desired actions. The module is designed to be used for a three-phase BSNPC bridge, it involves deadtime calculation between the main and middle switches, it has every phase leg independent from each others that the switching pattern of one leg doesn't affect the switching pattern of the other (because the inverter is considered a Voltage Source Inverter), it is specially designed to give good

resolution and also implemented in a way to be FPGA friendly but also to provide the engineer with full flexibility on the desired actions. Thus, each leg has its own state machine which is identical to the other two. The input of the state machines is a reference (one and different for each phase) which is used for the comparison as well as the deadtime calculation as explained in the next sections and after the comparison, deadtime calculation and reference fixation is made, the output driving pulses for the four switches of one leg are created. The input reference is always positive (FPGA friendly) and in order to distinguish between positive and negative reference another 1-bit signal gives that information. The module also distinguishes between input sign, reference height and whether the triangle is going up or down. The PWM triangles receives a new reference value on every edge of the triangle, meaning on every triangle valley and every triangle summit. So, for instance, the triangle frequency in this application is 48kHz and the reference gets updates every 96kHz. In the sections below, the state machine, the innovative real-time deadtime creation and the pulse width normalization and fixation are presented.

4.4.1 Pulse Width Modulation state machine

The operation of the PWM module is also based on a state machine, named by the author of this master thesis as ”centipede” due to the fact that the shape of the state machine looks like a cen- tipede. The number of states is 12+1 (start state) and refer to whether the reference input of the state machine is positive or negative, on which band the reference resides and whether the triangle instant is going up or going down.

A more illustrative list of the inputs to the state machine is shown here:

- **Sign:** The reference height stays to a positive value and a separate 1-bit signal reveals the sign of the reference. So that kind of input is either positive or negative.
- **Reference band:** The reference input has a 16-bit value which states the height of the reference in respect to the triangle height and that value is used to judge which band the reference lies upon.
- **Triangle direction:** The triangle direction stated by an 1-bit signal dictates the direction of the triangle, information which is useful for the state machine to decide upon the way it should turn off or on the switches so that the deadtimes and the pulse width limits are not violated.

The state machine created to form the action group of the PWM module can be seen in Figure 4.10.

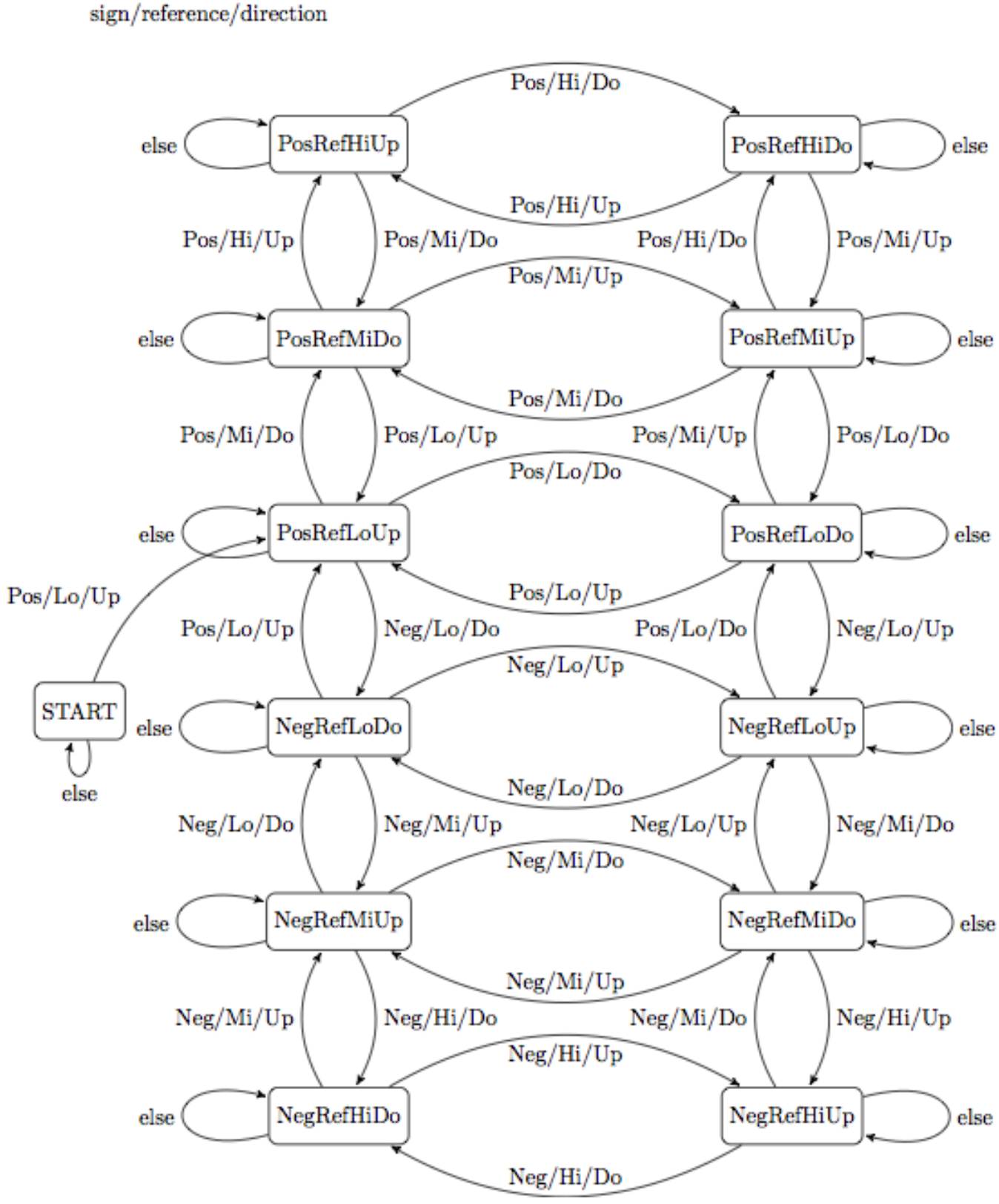


Figure 4.10

The states as shown in the state machine are: A more illustrative list of the inputs to the state machine is shown here:

- **Start:** Start state which waits for the reference to comply with PosRefLoDo.
- **PosRefLoDo:** Positive reference at low band and triangle going down.
- **PosRefLoUp:** Positive reference at low band and triangle going up.
- **PosRefMiDo:** Positive reference at middle band and triangle going down.
- **PosRefMiUp:** Positive reference at middle band and triangle going up.
- **PosRefHiDo:** Positive reference at high band and triangle going down.
- **PosRefHiUp:** Positive reference at high band and triangle going up.
- **NegRefLoDo:** Negative reference at low band and triangle going down.
- **NegRefLoUp:** Negative reference at low band and triangle going up.
- **NegRefMiDo:** Negative reference at middle band and triangle going down.
- **NegRefMiUp:** Negative reference at middle band and triangle going up.
- **NegRefHiDo:** Negative reference at high band and triangle going down.
- **NegRefHiUp:** Negative reference at high band and triangle going up.

It is evident that the state machine requires that the passing through the states is smooth as smooth asa sine wave passes through the values. For example a positive semi-period is formed by beginning from the PosRefLoDo state, then moving to the PosRefLoUp and back to PosRefLoDo and then moving to the PosRefMiUp. After changing states between PosRefMiDo and PosRefMiUp the state machine enters the high region and back again. The state machine does not allow for jumps between non-consecutive states a fact which forces for good utilization of the PWM module.

4.4.2 Comparison and deadtime creation

The triangle carrier is based on an 11-bit counter that counts up to 1562 clocked on 150MHz. The upper counter value was chosen so as to give a switching frequency of 48kHz (counting up and down is 150MHz/(2*1562) = 48,015kHz). The 11-bit counter gives up resolution of 10-11 bits which is very $2*1562$ accurate for a PWM module.

The reference input to the state machine refers to the reference used for the main switches. The comparison done regards the reference as the positive part and the triangle height as the negative part of the comparison. In that way, when a positive reference value is higher than the triangle value, a turning on pulse is formed for the higher main switch and when the positive reference is lower than the triangle value then a turning off pulse is formed for the higher main switch. Respectively when the above happen for a negative reference, the exact same operation is executed for the lower main switch. The higher main switch switches on and off during the positive grid semi-period while the lower main switch remains switched off. The lower main switch takes turn at the negative grid semi-period while the higher main switch stays turned off.

The three-phase BSNPC bridge is shown again in Figure 4.11.

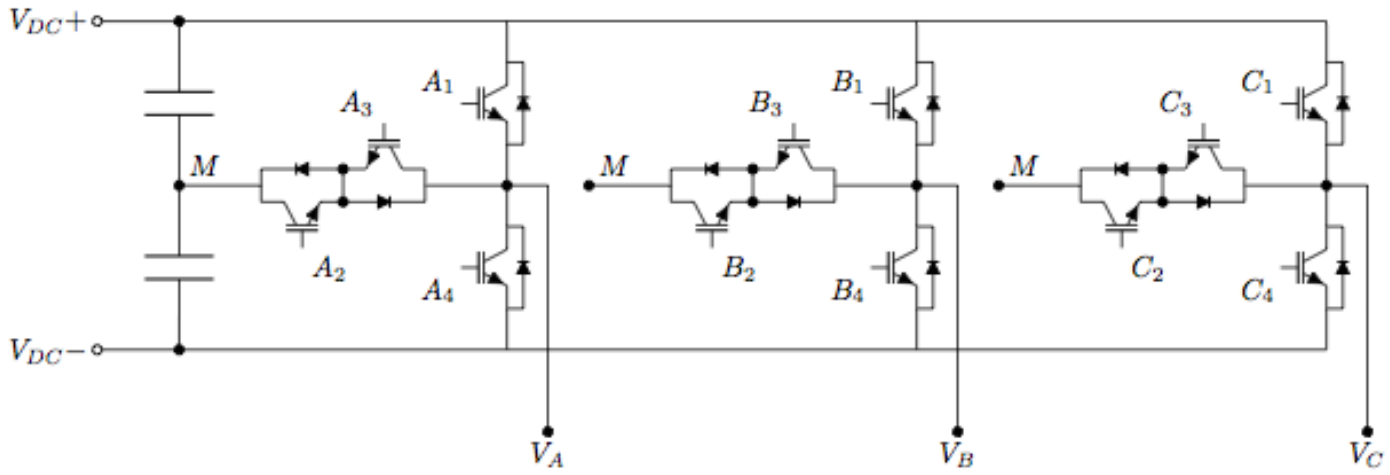


Figure 4.11

In order for the BSNPC bridge to operate in its full advantage, the middle switches have to turn on and off so as to create a unipolar switching pattern. This means that during the positive grid semi-period switches 1 and 3 switch oppositely and with a strict no-both-turn-on limitation. The comparison due to the duality of the swathing pattern the comparison operation of the middle switches is opposite of that of the main switches. So the comparison operation of the middle switches is opposite. Due to inideality of the switches there is a deadtime involved between the switchings. This deadtime is created by inserting an inherent deadtime creator inside the module. This is implemented by creating a second reference out of the input reference (for the main switches) and offsetting it by a precalculated deadtime. In that way, whatever the switching is for the main switches, the deadtime for the middle switches is always equal to the desired value. Respectively, for the negative grid semi-period this operation happens for switches 2 and 4. The deadtime for this application was chose at 1000ns but it is configurable at development time. This operation is illustrated in Figure 4.12.

4.4.3 Minimum pulse width

Due to the non-ideality of the switching devices there is a minimum hollow and summit pulse width that has to be sustained as shown in Figure 4.13.

For this application, the minimum hollow pulse widths are 1000ns and the minimum summit pulse widths are 300ns. This means that if the created pulses are shorter than that, then the pulse should be suppressed. Hollow pulses should remain high and summit pulses should remain low.

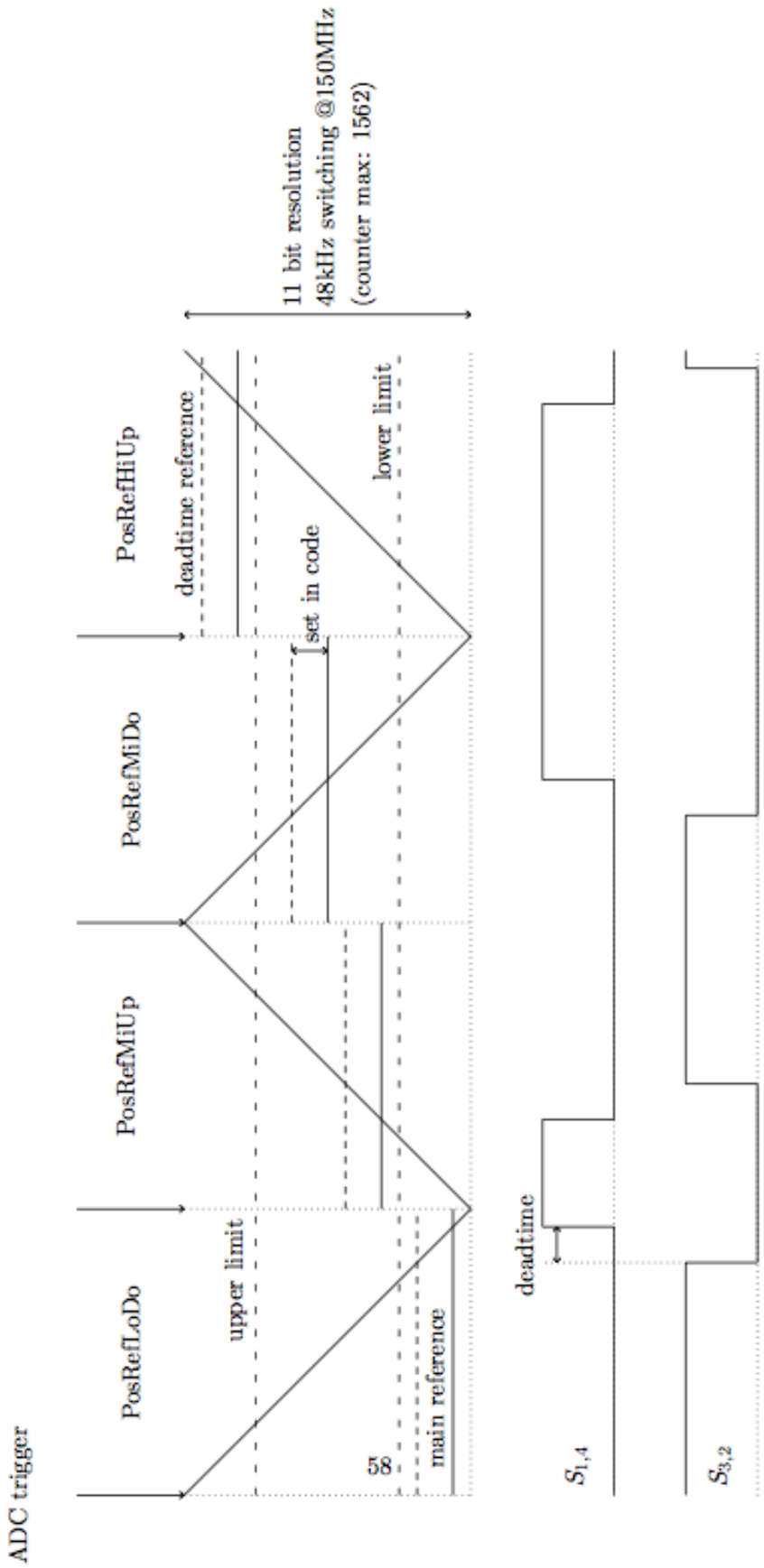

ADC trigger
PosRefLoDo
PosRefMiUp
PosRefMiDo
PosRefHiUp
upper limit
main reference
58
deadtime reference
set in code
lower limit
11 bit resolution
48kHz switching @150MHz
(counter max: 1562)
deadtime
$S_{1,4}$
$S_{3,2}$

Figure 4.12

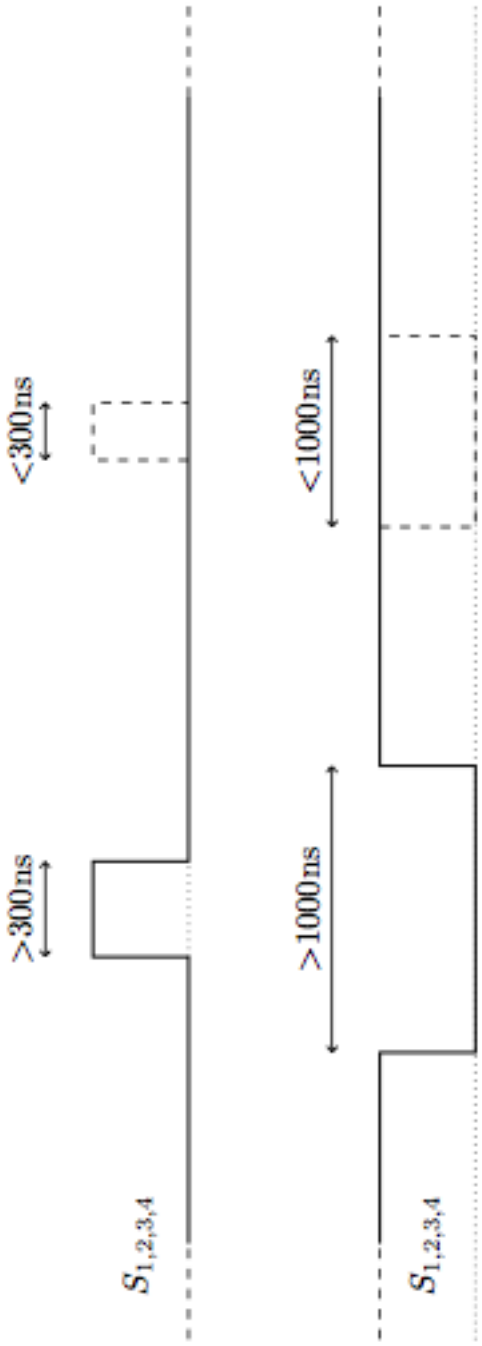


Figure 4.13

This limitation can be applied by introducing two bands, one upper and one lower band. These bands have direct connection with the reference value and the states of the state machine and the operation of the switches when inside these bands is different than that when the reference value is in the middle. These bands, depending on the previous value of the output of the module (whether the switch is on or off) decide upon whether the switch should remain on, remain off or not proceed to a switching action. The bands are different for the middle and main switches due to the duality of the switching pattern. Thus the upper band of the main switches is the lower band of the middle switches and vice versa. The bands are formed so as to account for a pulse created by two switching semi-periods. This means that the band has half the height that

it should have. This can create issues and violations in case of some state transitions but this is fixed with another feature of the state machine, real-time reference fixation. An illustration of these bands is shown in Figure 4.14.

Another very important feature of this Pulse Width Modulation implementation using a state machine is the real-time fixation of illegal pulses. This practically means that when changing states, and the resulted pulse width has an illegal width, then the state machine automatically detects this violation and fixes the pulse so as to comply with the rules. For instance, when the current state is PosRefMiDo with a reference very close to the lower limit band and the next state is PosRefLoUp with a reference really close to 0 then the resulting pulse is shorter than the pulse width limit. Being that the state machine is totally flexible and keeps track of every current and previous variable, it is aware of this violation and can rearrange the next reference so as to be exactly the value that results in an allowed pulse width. This feature makes the module enormously powerful and flexible and most impotently real-time. Two examples of this feature, for the main and middle switches are illustrated in Figures 4.15 and 4.16.

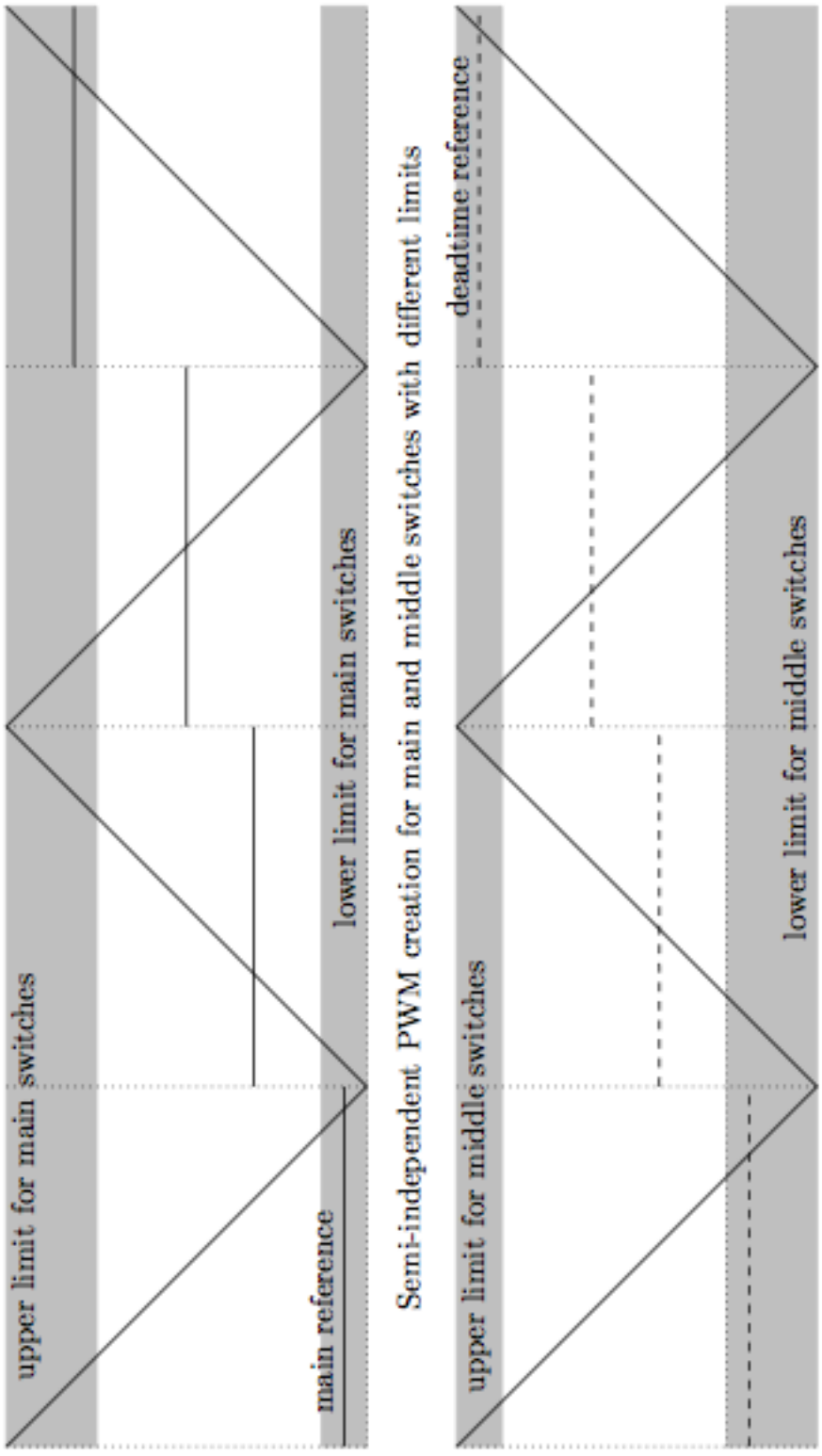

upper limit for main switches
lower limit for main switches
main reference
Semi-independent PWM creation for main and middle switches with different limits
deadtime reference
upper limit for middle switches
lower limit for middle switches

Figure 4.14

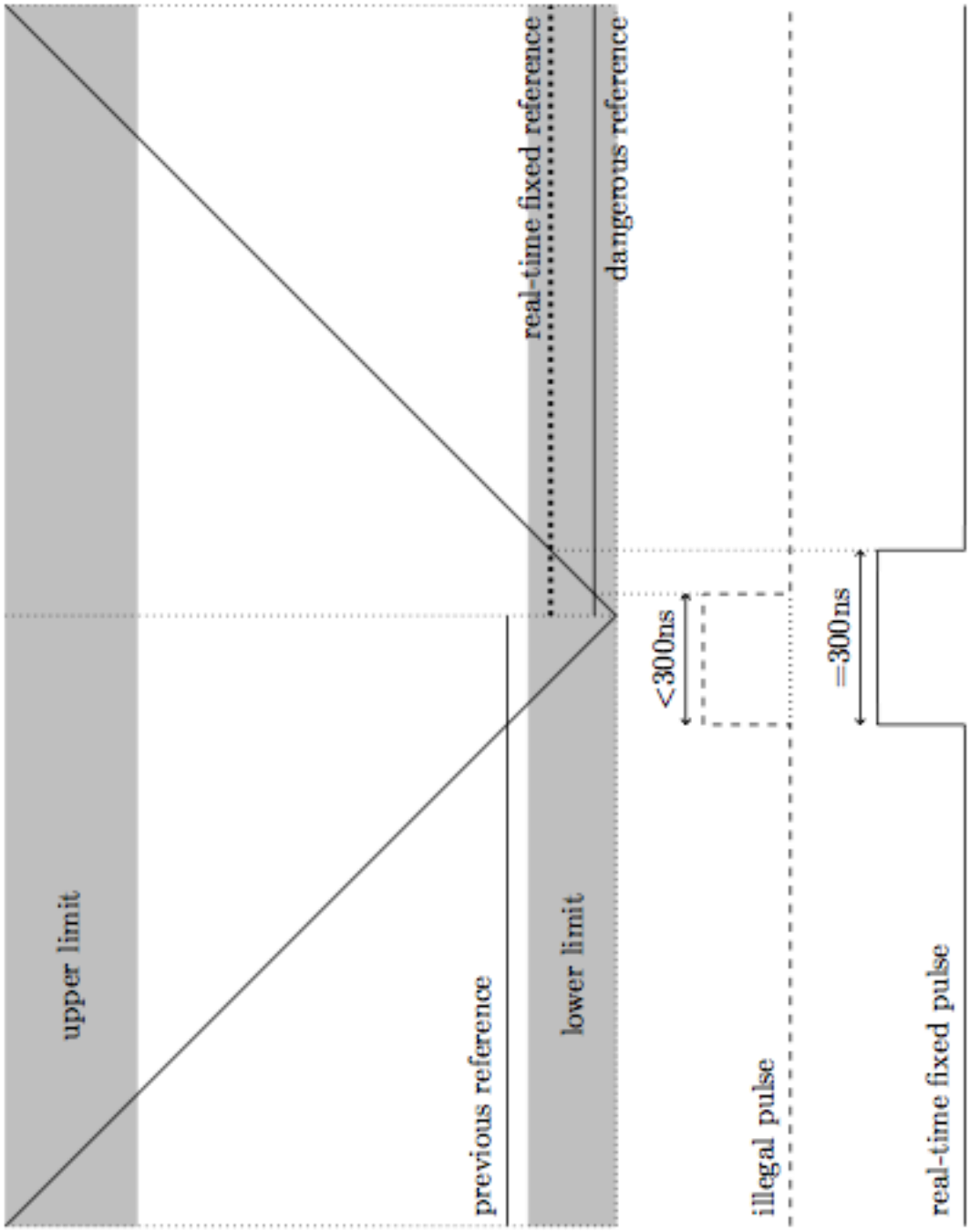

upper limit
previous reference
lower limit
real-time fixed reference
dangerous reference
illegal pulse
<300ns
=300ns
real-time fixed pulse

Figure 4.15

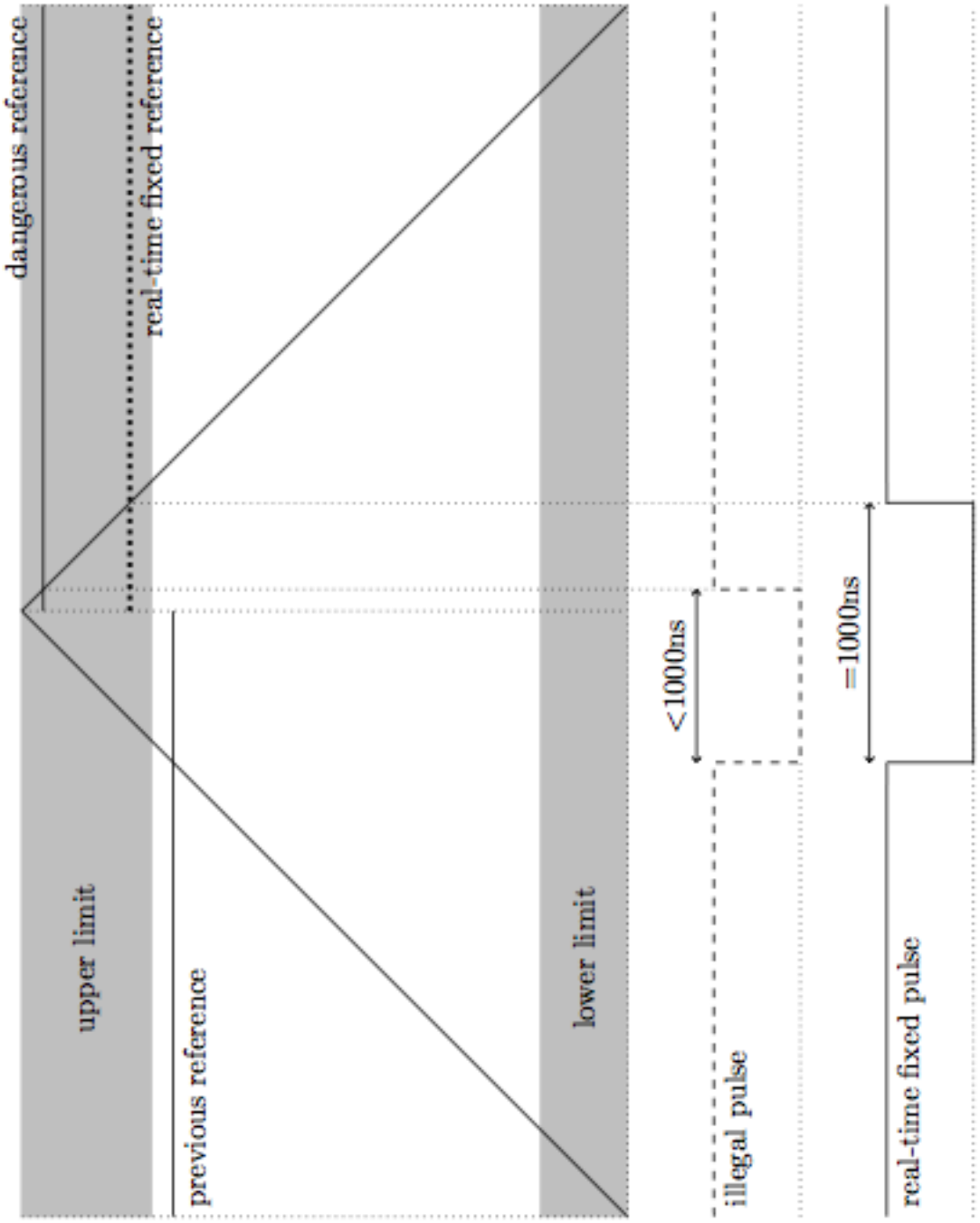

dangerous reference
real-time fixed reference
upper limit
previous reference
lower limit
<1000ns
illegal pulse
=1000ns
real-time fixed pulse

Figure 4.16

5

Testing and comissioning

5.1. Testing configuration

The system setup follows that of the system description. There is a user computer that develops the Simulink block, compiles it and downloads it to the Real Time Computer, which runs the model in real-time being connected to the custom board. The user computer can in real time change the values in the simulink block running. The custom board has a direct connection to the inverter hardware and can receives measurements, act on the relays and form the PWM pulses of the bridge. The test system is shown in Figure 5.1.

The tests in this thesis consist of the analog-to-digital conversion test in combination to the reception in part of the Real Time Computer, a voltage feedforward test to ensure that the grid waveform is received in an appropriate manner, to check that PWM and memory system works, to check that writing operation in part of the Real Time Computer works as well as that floating-point operations inside the FPGA can work. Also, because of the fact that the FPGA was experimen- tal, the available number of combinational logic and register elements wasn't enough to implement a control logic, a grid connection current control loop for the grid connected inverter wasn't possible.

The list of the performed tests is shown below:

- **Analog-to-digital converter and Real Time Computer reception test**
- **Voltage feed-forward, Real Time Computer transmission, floating point test**

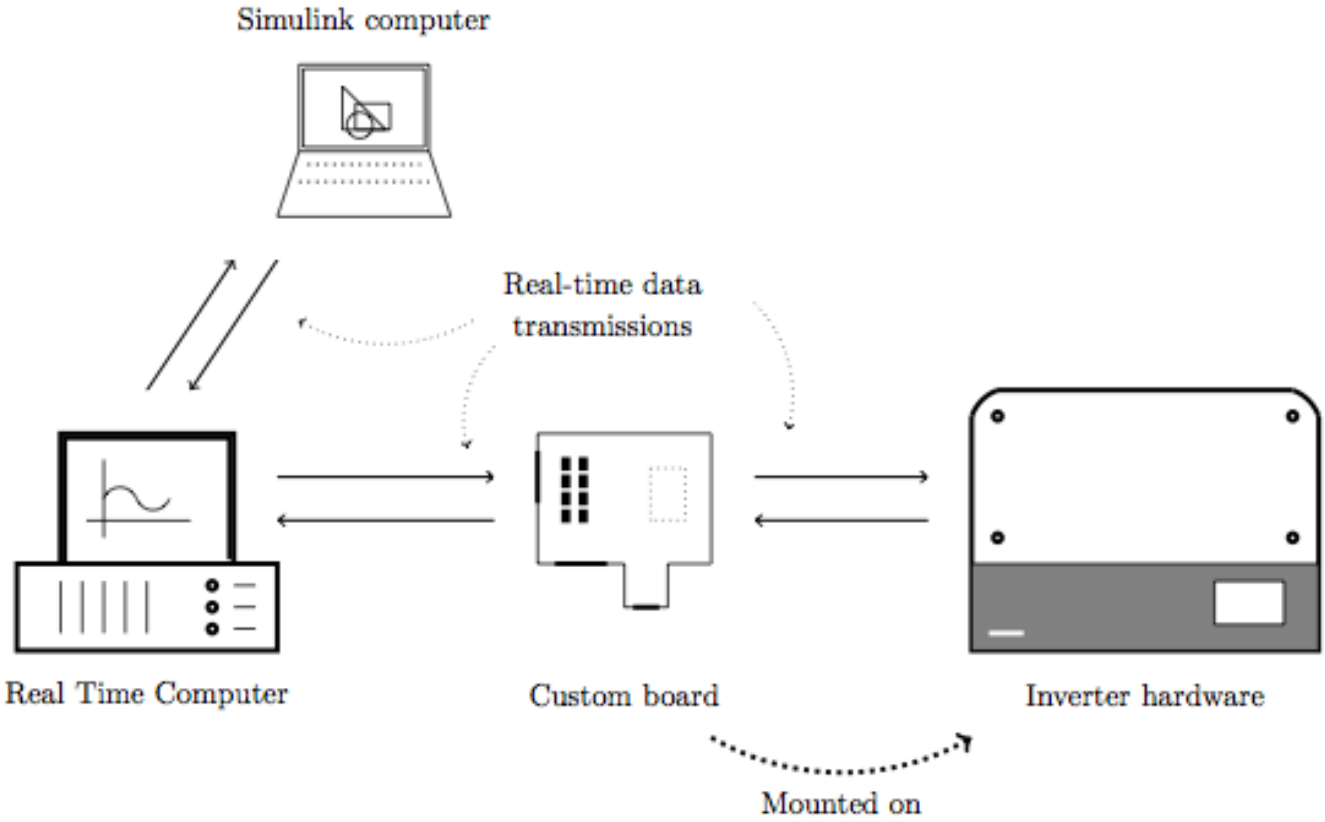


Figure 5.1

5.2. ADC and Real Time Computer reception test

For this test, the inverter was connected to the grid, the relays were kept open and the values of the ADC were taken by the FPGA blocks and transmitted to the Real Time Computer. Then the Real Time Computer in real-time used the xPC screen to demonstrate them in numerical values as well as waveforms.

Below there are screenshots of all the values of the system in a list as numerical values in the range of 0-4096 (12-bit) as well as another screenshot of the waveform of the grid voltage again as shown at the Real Time Computer screen. The values are normalized inside the Simulink program so that they represent real values. The left two tables represent the first 16-channel ADC and the right two tables the second 16-channel ADC. The first three channels of the first 16-channel ADC are multiplexed with the measurements of the three single-channel ADCs. For debugging reasons the first three channels representing the three single-channel ADCs are not normalized values. The two screens of the test are shown in Figures 5.2 and 5.2.


Loaded App: rcpore1
Memory: 2034MB
Mode: RT, single
Logging: tet
StopTime: Inf d
SampleTime: 0.000125
AverageTET: 3.176e-005
Execution: 76.10 s
Scope: 4, signal 19 added
Scope: 4, signal 55 added
Scope: 4, trigger level set to 100.000000
Scope: 4, TriggerSlope set to 'Rising'
Scope: 4, lower y-axis limit set to 0.000000e+000
Scope: 4, upper y-axis limit set to 0.000000e+000
System: initializing application finished
System: model thread stack is 64K bytes
System: execution started (sample time: 0.000125)
F1 SC1 41 39 38 42 43 37 36 35
CH00 Free 2064.000 N/A
CH01 Free 2045.000 N/A
CH02 Free 2046.000 N/A
CH03 Free 2 N/A
CH04 Free 0 N/A
CH05 GridMs_V_N_PE -5.722 V
CH06 DIW_Ref_2V5 61 ?
CH07 AFCI_in 1900 ?
F2 SC3 34 33 32 31 30 28 27 26
CH00 Free 0 N/A
CH01 Free 0 N/A
CH02 Free 0 N/A
CH03 Free 0 N/A
CH04 Temp_PCB 2119 ?
CH05 DIW_Cur -245.949 mA
CH06 InsRis 0 ?
CH07 GridMs_VRly_phsC -502.000 V
F3 SC2 54 53 52 51 50 40 29 Grid Cu
CH08 GridMs_VRly_phsB -12.746 V
CH09 DCMs_V_C 380.100 V
CH10 DCMs_V_A 289.536 V
CH11 DCMs_A_B -54.180 A
CH12 Inv_DclVolLo 315.421 V
CH13 GridMs_V_phsB 55.396 V
CH14 GridMs_A_phsC -1.211 A
CH15 GridMs_A_phsA -1.172 A
F4 SC4 25 24 23 22 21 20 Grid Volta
CH08 GridMs_VRly_phsA -41.670 V
CH09 DCMs_V_B 272.703 V
CH10 DCMs_A_C -58.320 A
CH11 DCMs_A_A -63.398 A
CH12 Inv_DclVolHi 322.686 V
CH13 GridMs_V_phsC 321.594 V
CH14 GridMs_V_phsA -83.095 V
CH15 GridMs_A_phsB -0.156 A

Figure 5.2

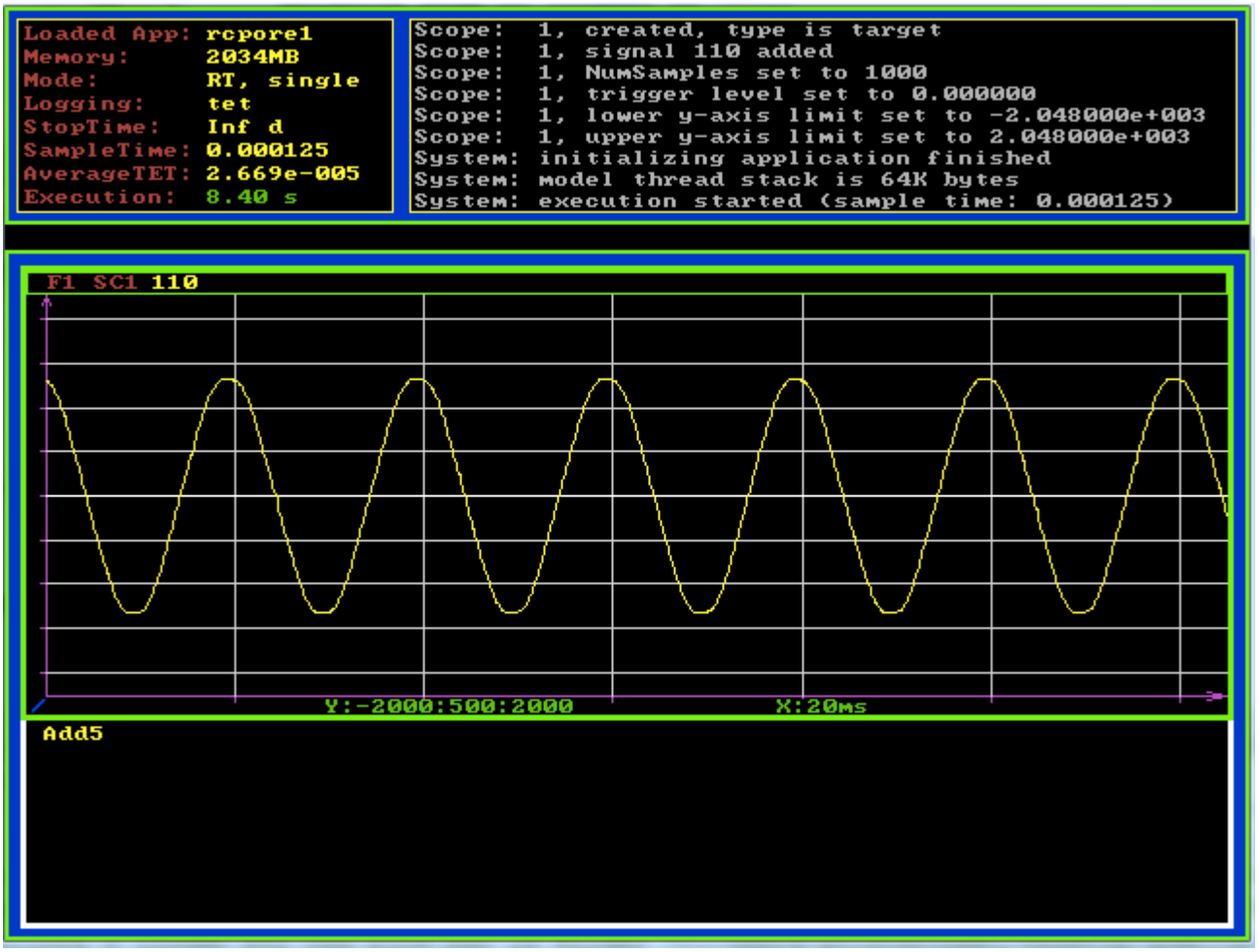


Figure 5.3

5.3. Voltage feed-forward and Real Time Transmission test

In this test, the voltage measurement of the grid was acquired and it was passed to a floating point normalizer inside the FPGA. The result was multiplied by a values passed by the Real Time Computer that could be changed in real-time and the result was normalized to become a reference to the PWM module. The DC link voltage of the capacitor bank was 620V and the resulting waveforms represent duty cycles from 100% to 0%.

The waveforms were captured using an oscilloscope and include the grid waveform (yellow), switching waveform before the filter (magenta) and the output after the filter and before the always- opened output relays (cyan). The Figures are 5.4,5.5,5.6,5.7,5.8,5.9,5.10,5.11,5.12,5.13,5.14

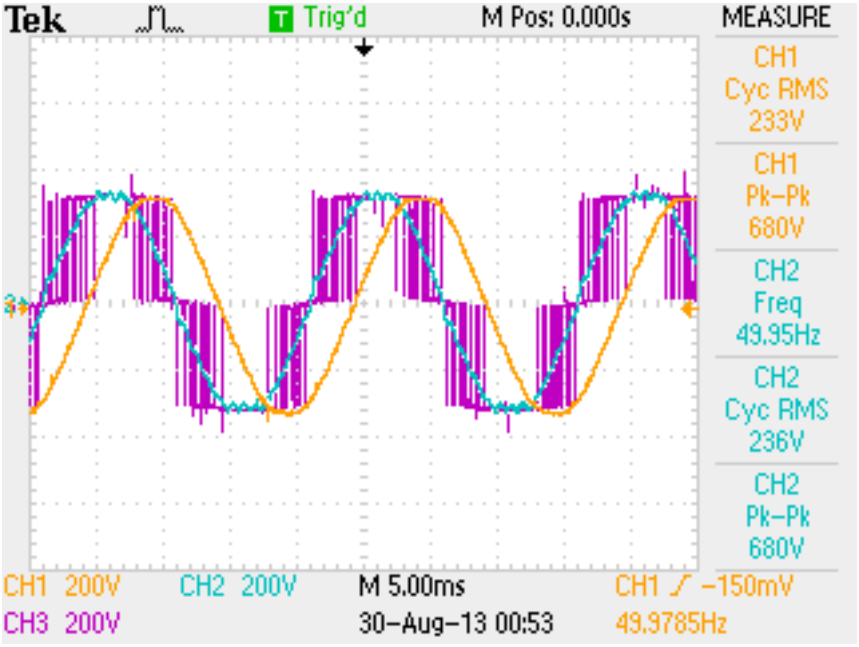


Figure 5.4

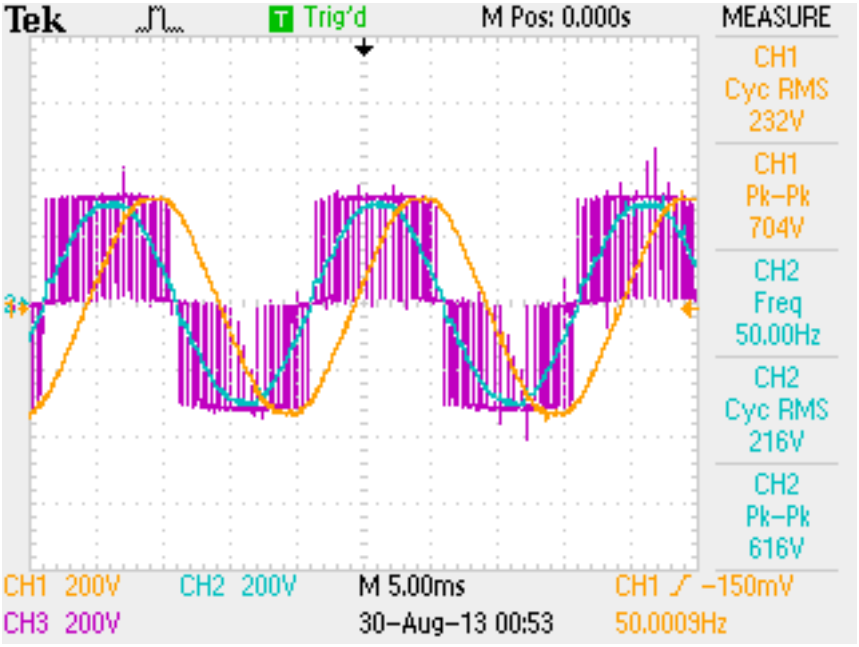


Figure 5.5

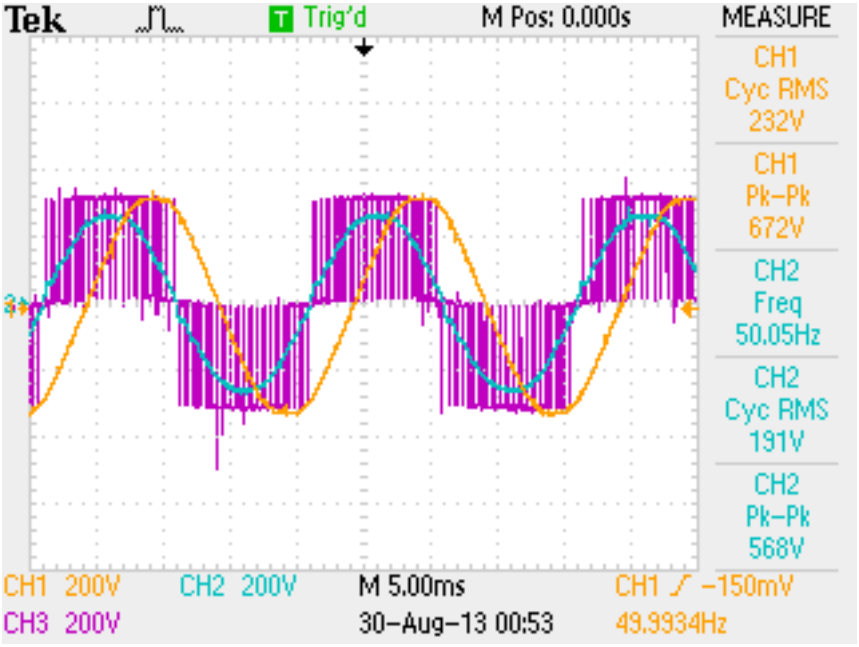


Figure 5.6

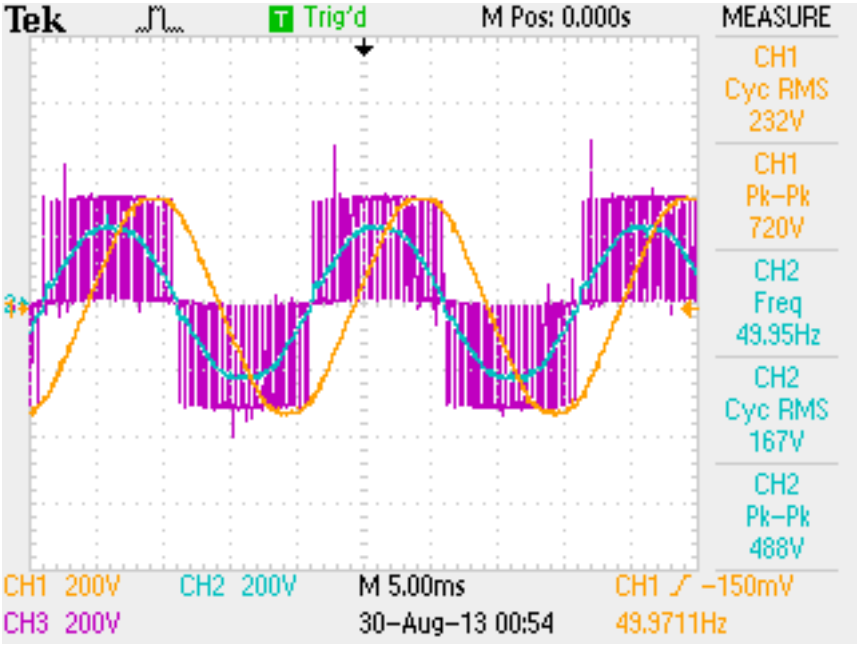


Figure 5.7

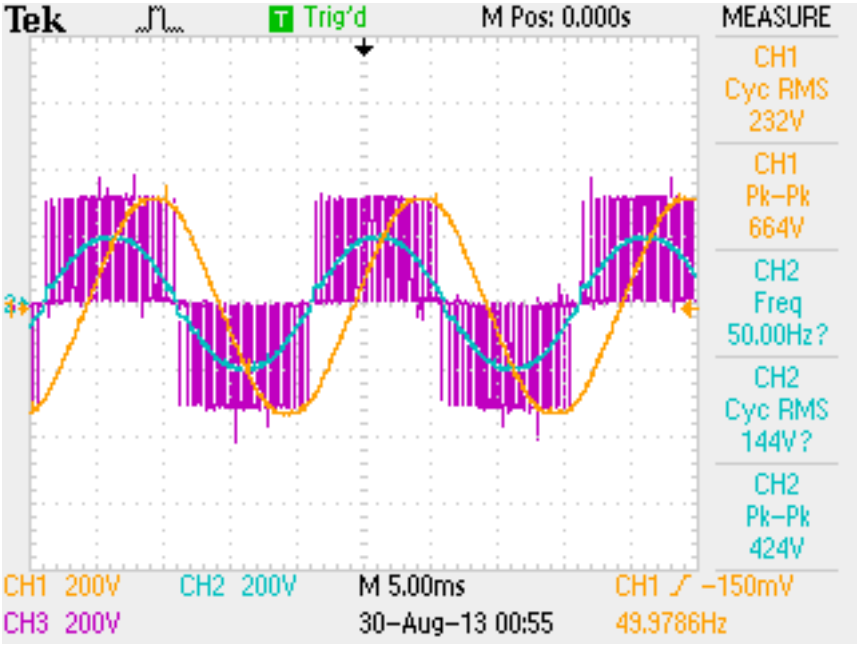


Figure 5.8

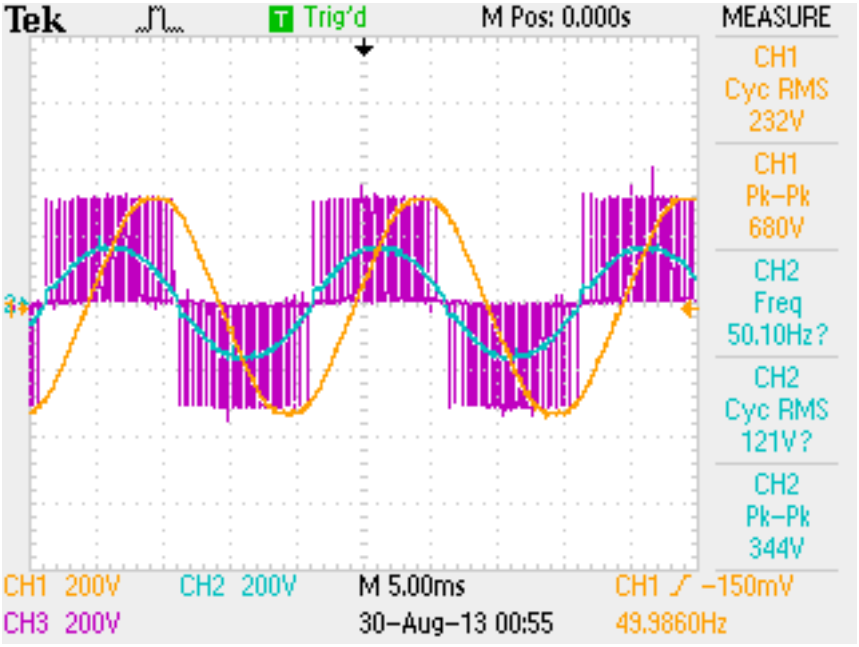


Figure 5.9

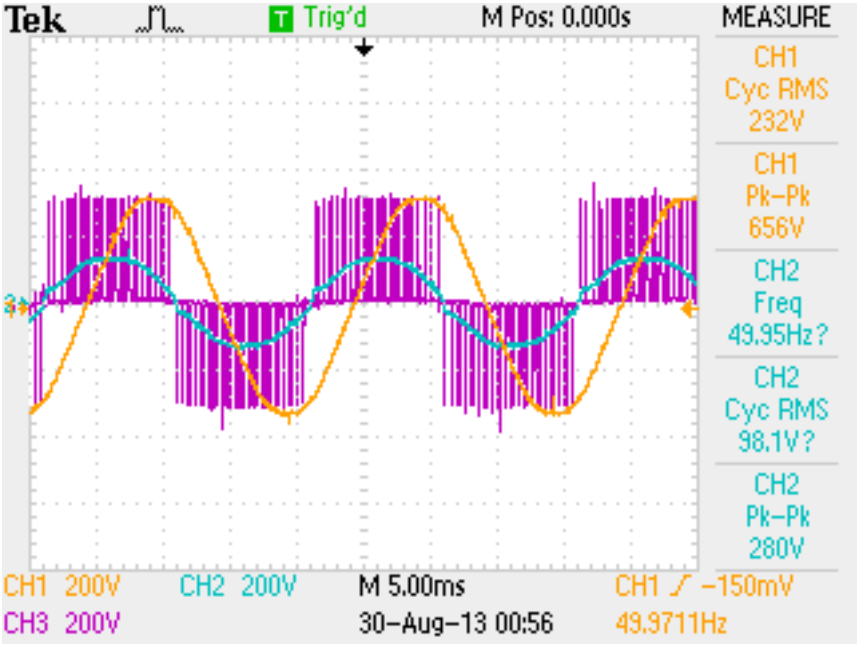


Figure 5.10

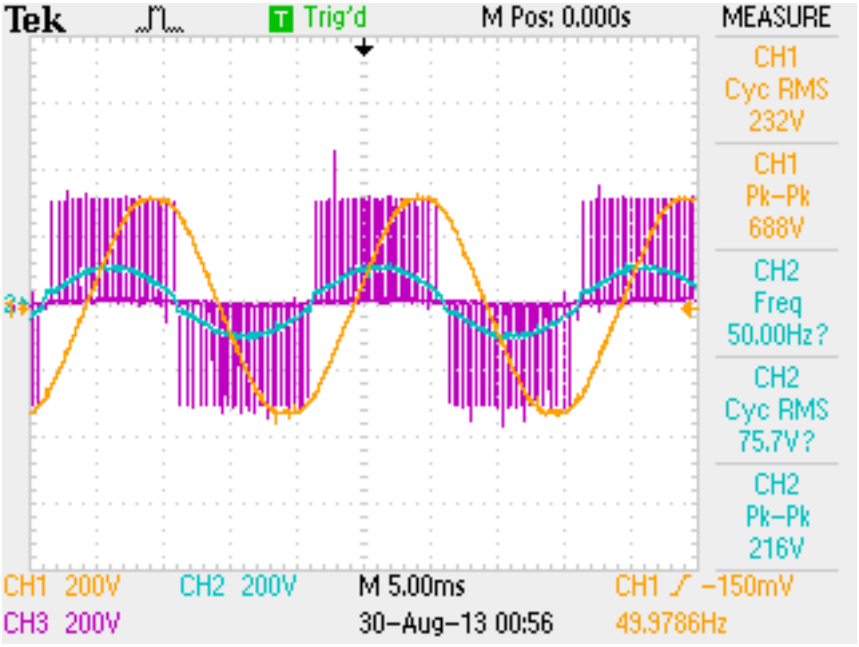


Figure 5.11

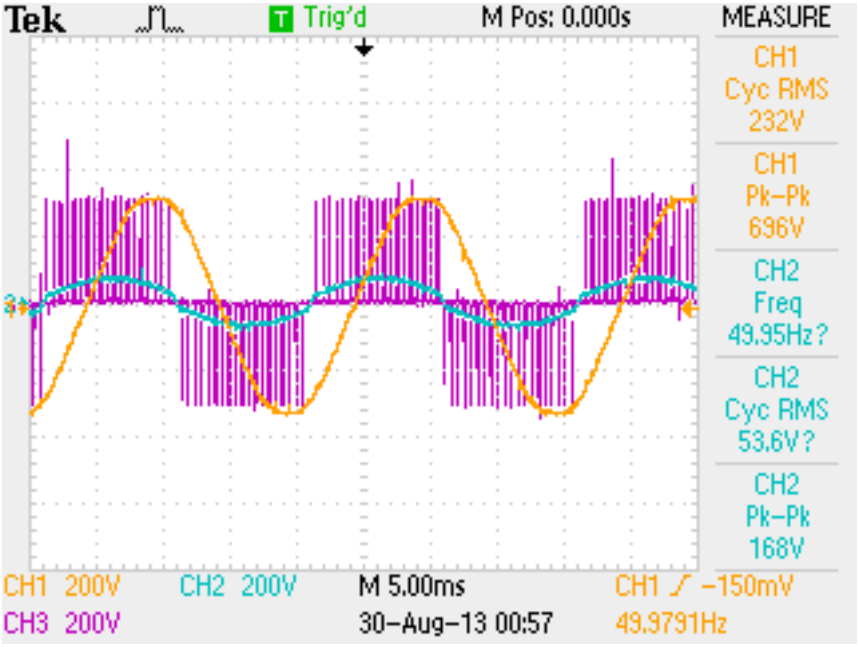


Figure 5.12

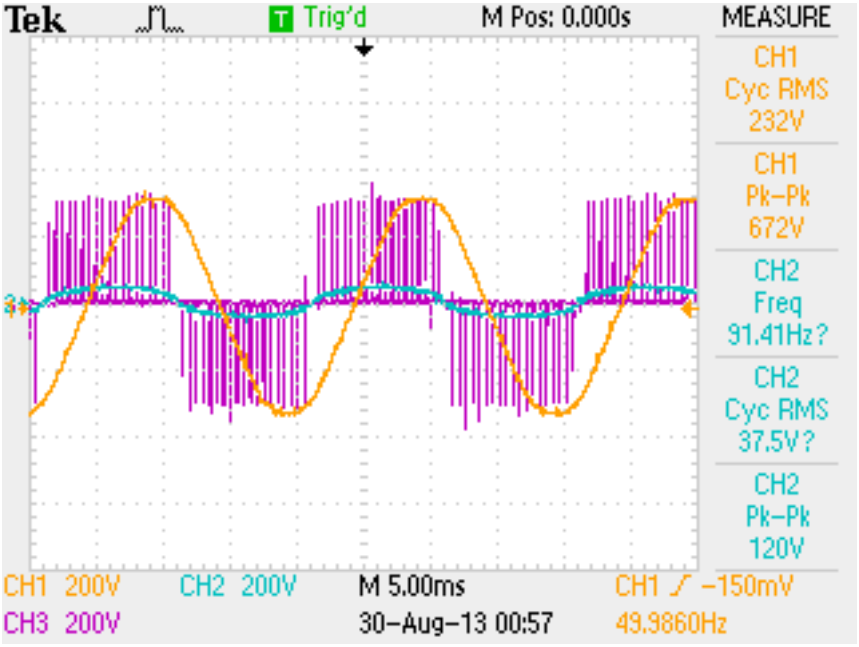


Figure 5.13

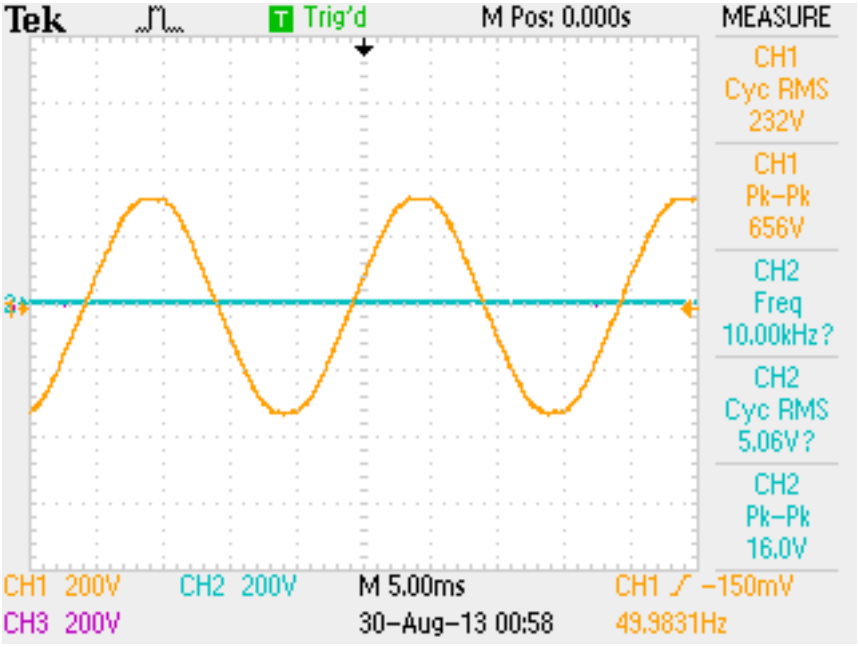


Figure 5.14

One can see at the side of each screenshot the most important values measured like RMS values of the quantity illustrated, frequency etc so as to confirm the results.

The phase difference between the two voltages is caused by the output LCL filter that lies be- tween the bridge output and the grid.

The SIMULINK model used to produce these functionality can be shown below in Figure 5.15.

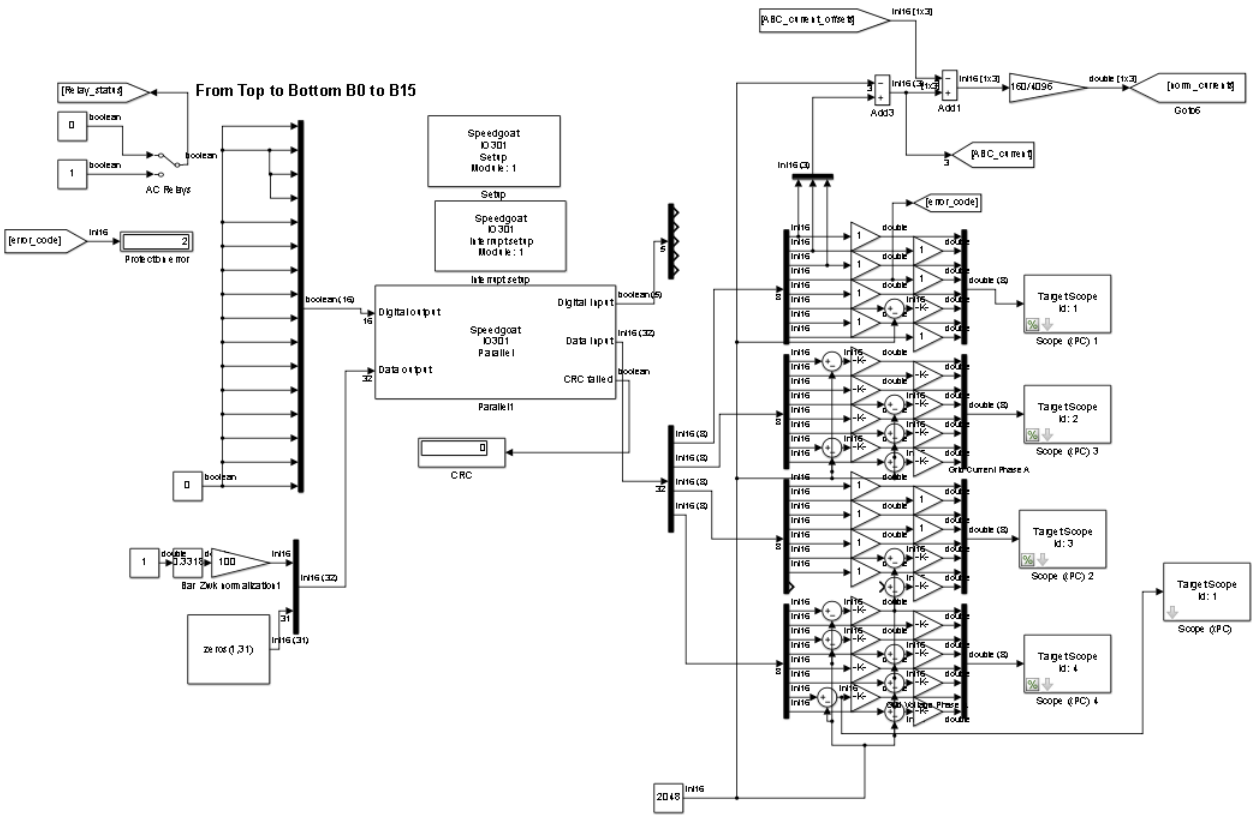


Figure 5.15

5.4. Miscellaneous tests

In the course of development, various other tests and simulations were taken place such as FPGA simulation of each block of FPGA code, single tests of operation of each block with its connecting part (e.g. Real Time Computer data exchange with the custom board, ADC conversion, etc). Additionally, a ROM test was performed so as to hard test the PWM FPGA module. Those test are not presented to avoid unnecessary focus on implementation details.

6

Conclusion

In this master thesis project Real Time Prototyping in grid-tie inverter systems is examined. The advantages and disadvantages of such a system as it is are seen and the possibility of expanding and extending the already existing practices is attempted. It is seen that a need for an intermediate stage is imminent in such systems in cases where the application needs and demands fast inter- action with the hardware. In the boundaries of these project lies the design and manufacturing of an intermediate custom electronic board that acts as a faster step to the system. This board communicates with both the inverter hardware and the Real Time Computer. In such a system this custom board overdoes the speed and data size limitations of the Rapid Control Prototyping system and allows for generic control of a power electronic system. On the custom board an Field Programmable Gate Array is mounted and some logic is developed in it. Basic inverter functionality such as Pulse Width Modulation engines, Analog-to-Digital converter code, current, voltage and differential protection modules as well as code for the interaction with the Real Time Computer is developed. The system is tested under some basic functionality and the results are presented.

6.1. Contribution

This master thesis work has proposed an industrial strength Real Time Simulation/Rapid Control Prototyping platform based on a novel very low latency FPGA design and a supporting FPGA modularised software library that makes it flexible, easy to use, and scalable without compromising its low latency performance. Emphasis was put on the novel fully programmable parallel FPGA architecture developed exclusively for power electronic RCP applications. The experimental results on an FPGA prototype intermediate board demonstrated a performance unmatched by directly connected commercial hardware-based RTS platforms to hardware in terms of both simulation time step and latency. The tight integration of the FPGA with its subsystems (PWM, ADC, COM interface, protection, memory modules), the I/O subsystems and isolation stages brings a power electronic testing tool which is, until now, available only in high-power laboratories, to the office desk. It enables a high-fidelity (with very low latency), safe, and fully realistic study of the most detailed aspects of power electronic development, qualification, commissioning, training, and teaching processes.

6.2. Further research

This project involved analog electronic design, VHDL programming, power electronic knowledge and understanding of inverter topology concepts. The results were at a very high percentage satisfactory and the initiative of designing such systems can help and speed up the development process.

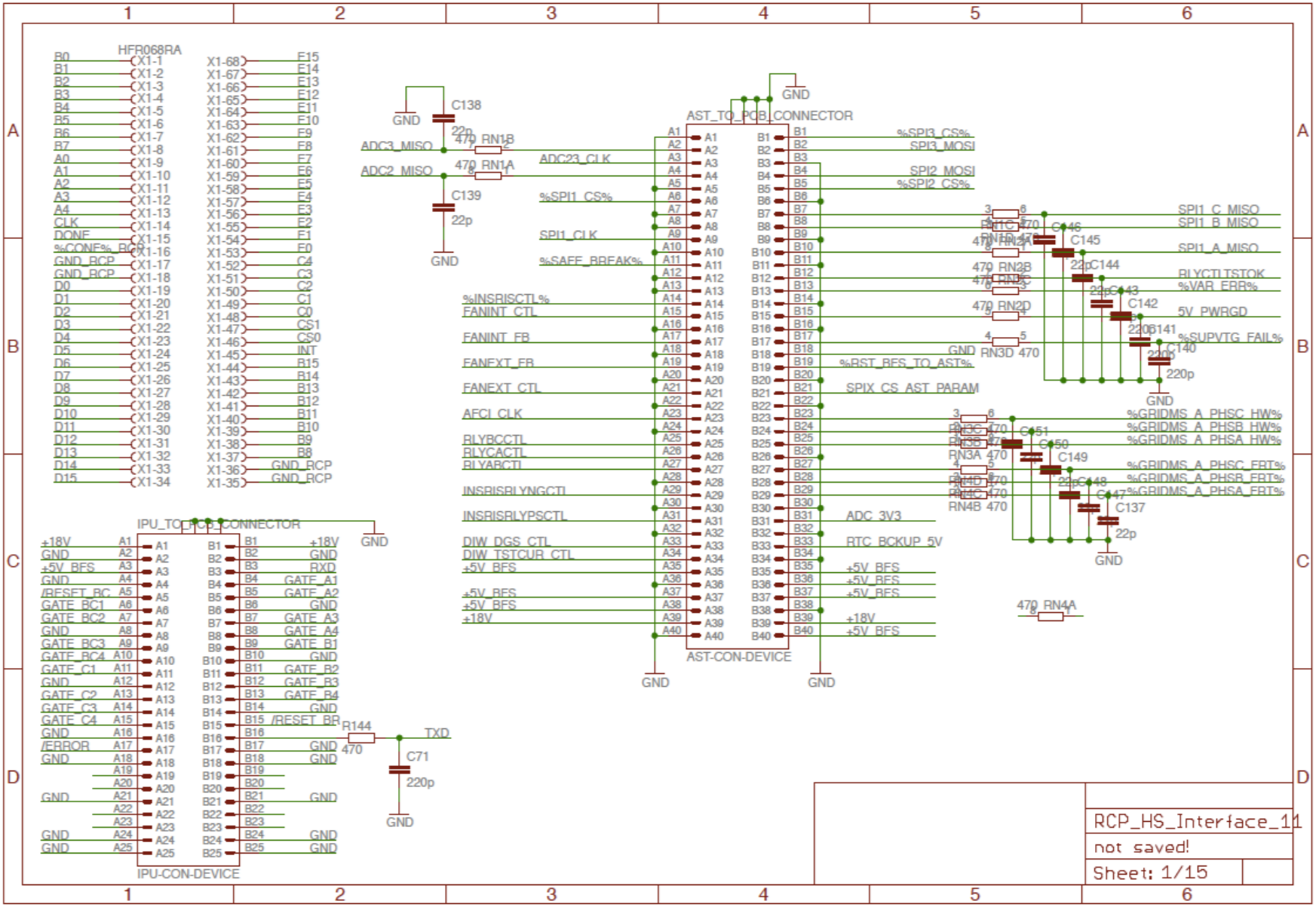
Although, the current system was designed and dimensioned in an experimental manner. In the ages of high electronic concentration, power electronic advancement and more and more combining of digital technology, embedded programming and power designs the need for more advanced and more industrial designs is imminent. An extension to this project would be to use a large scale FPGA, with an adequate amount of logic and combinational elements so that some part of the control (also using floating point arithmetic) is implemented in the FPGA. Additionally, the very fast functionality of a large scale industrial FPGA being programmed with a generic set of blocks used in power electronic designs can be combined and enforced with the use of a higher level programming module like an ARM microcontroller which would directly communicate with the FPGA, the Real Time Computer and the real hardware. Such a system would allow the designer to shift functionality and logic between the parts of the system (Real Time Computer, FPGA, ARM) depending on the complexity of the logic, the speed requirements and the already existing code and achieve the optimum and most productive result.

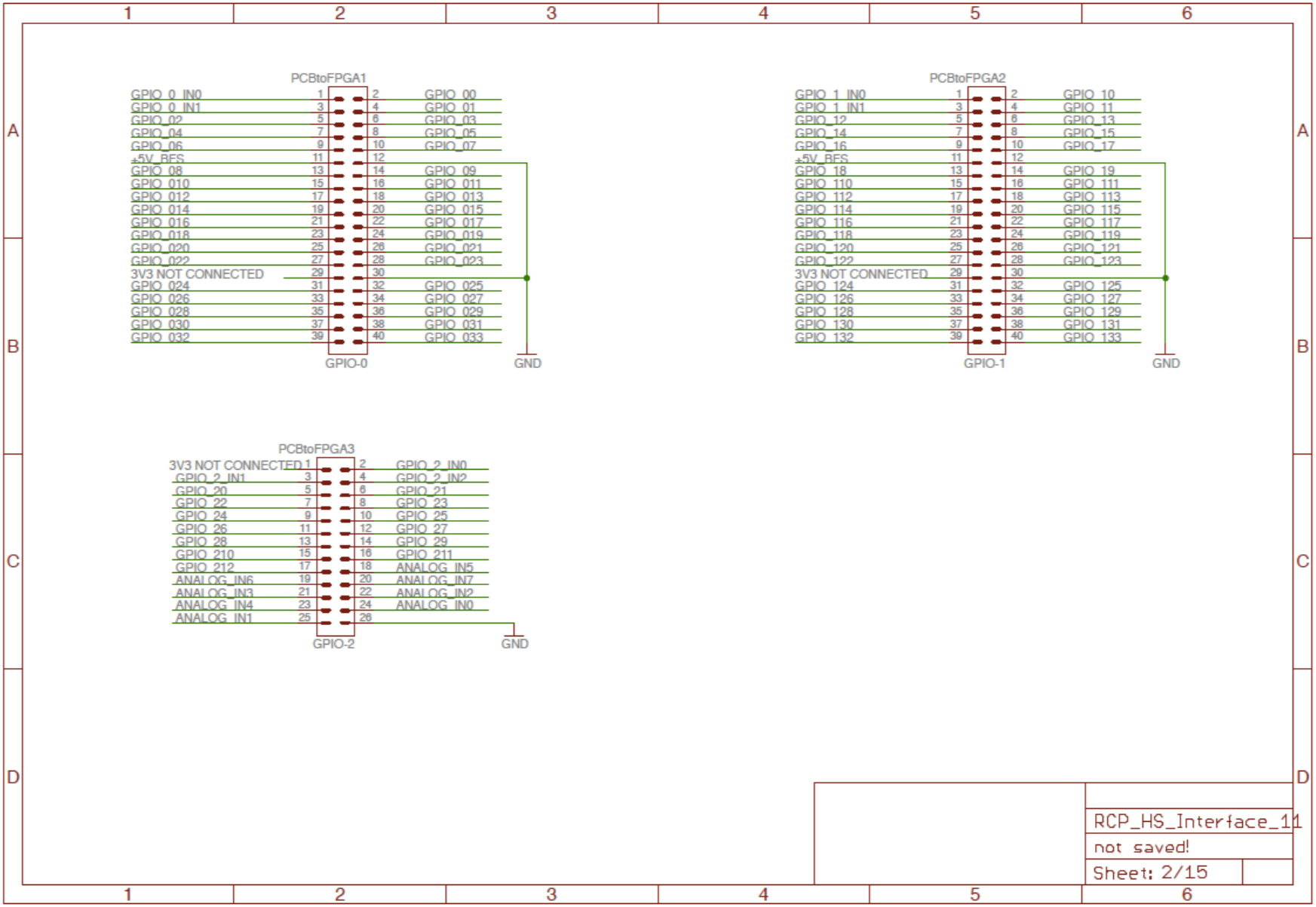
Such systems prove to be robust, easily configurable, generic and modular and can help speed up productivity, development process and make the controlling, commissioning and operation of new power electronic concepts easier. The engineer can conceive new designs, develop simple and ”dummy” hardware, plug an intermediate board like the one developed in this project and easily start operating, testing and optimizing the system at day one. Moreover, in case there is need for extension of a system the modularized nature of this system can help expand such a project into a more complex system.

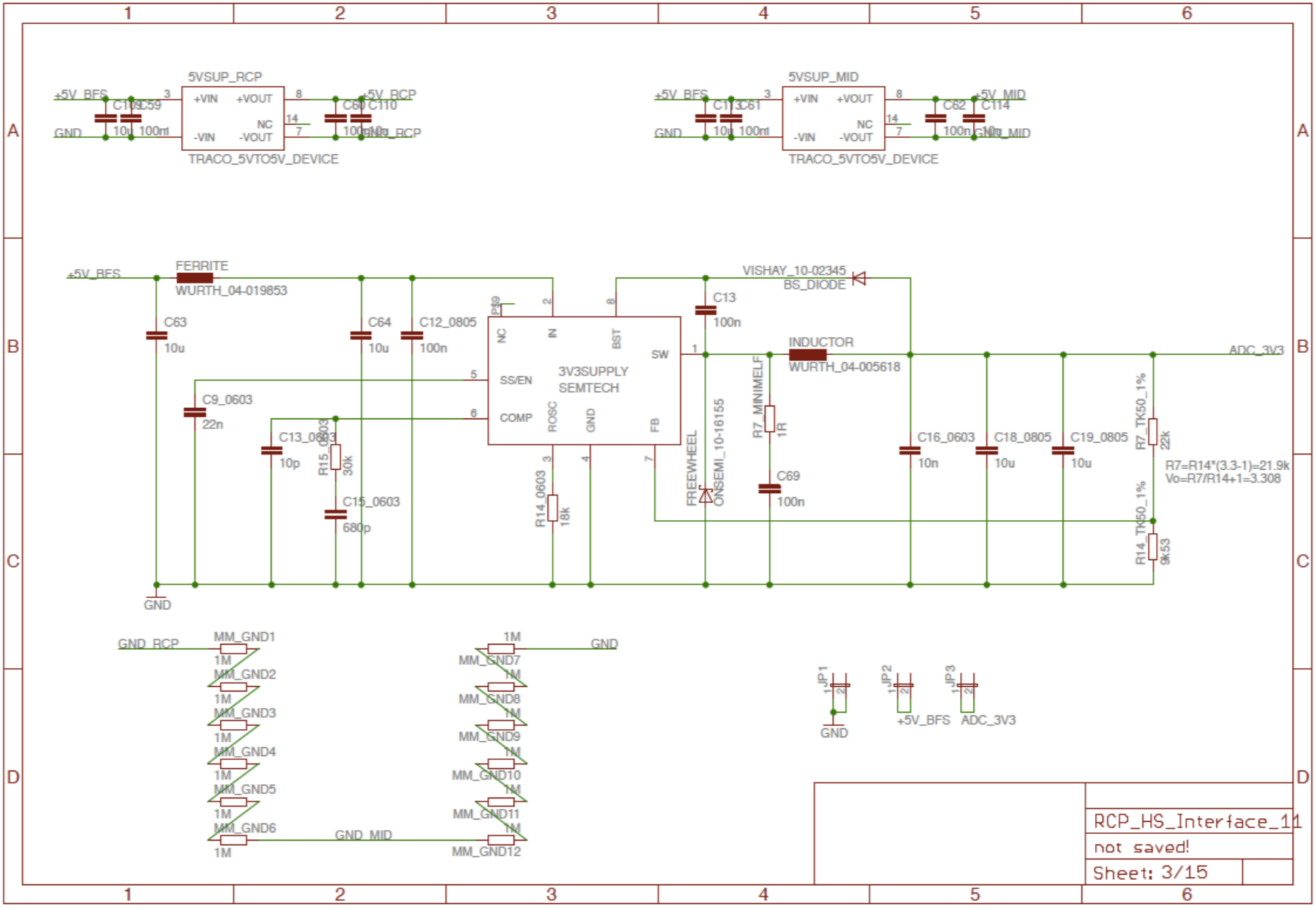
A possible extension of this system would be a development of a generic inverter system, combined with such a board, which would serve as an educational tool for young engineering students or engineers to develop practically concepts like control, PWM schemes, etc that would help in the promotion of the use and spreading of renewable energy devices and machinery.

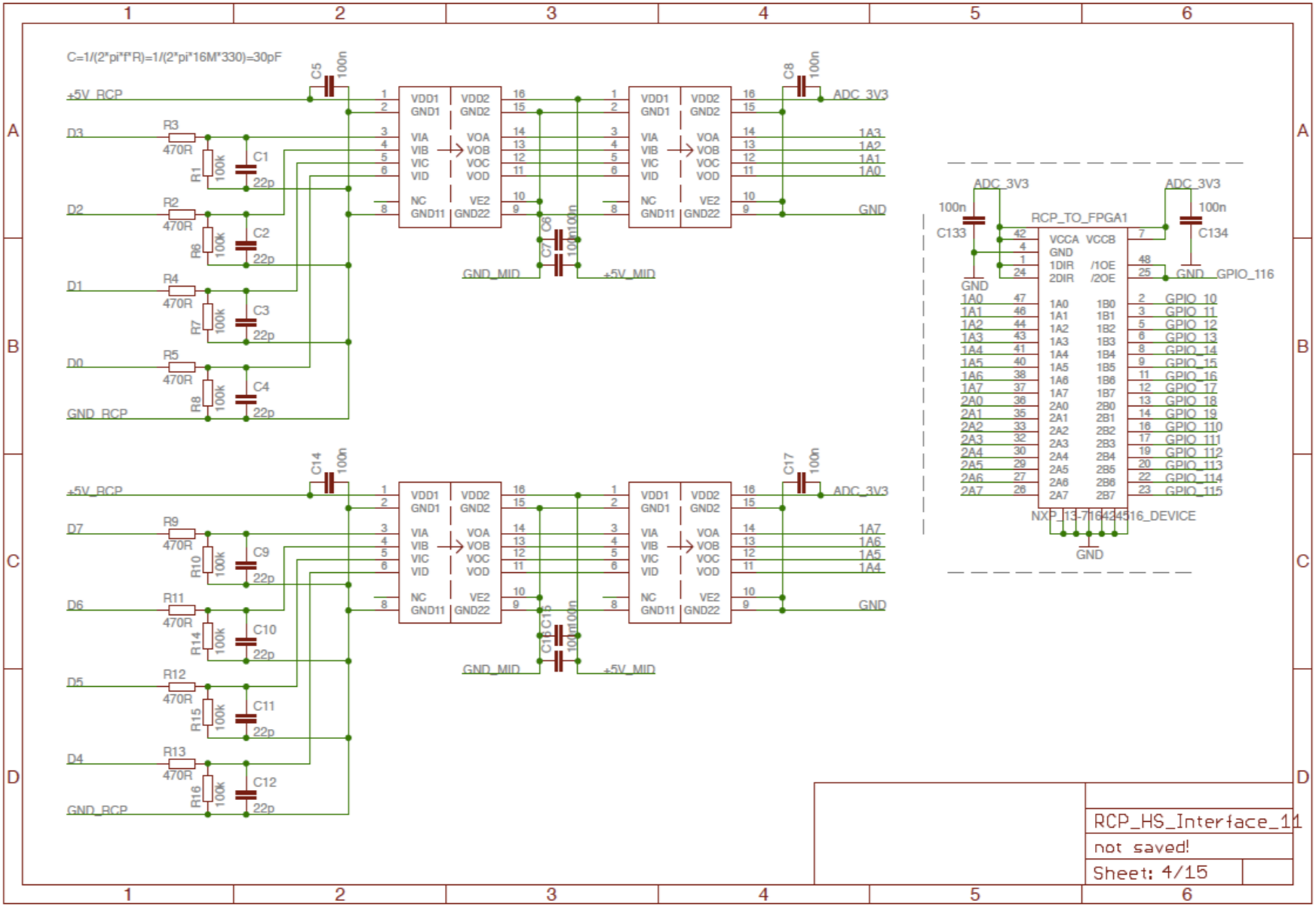
A

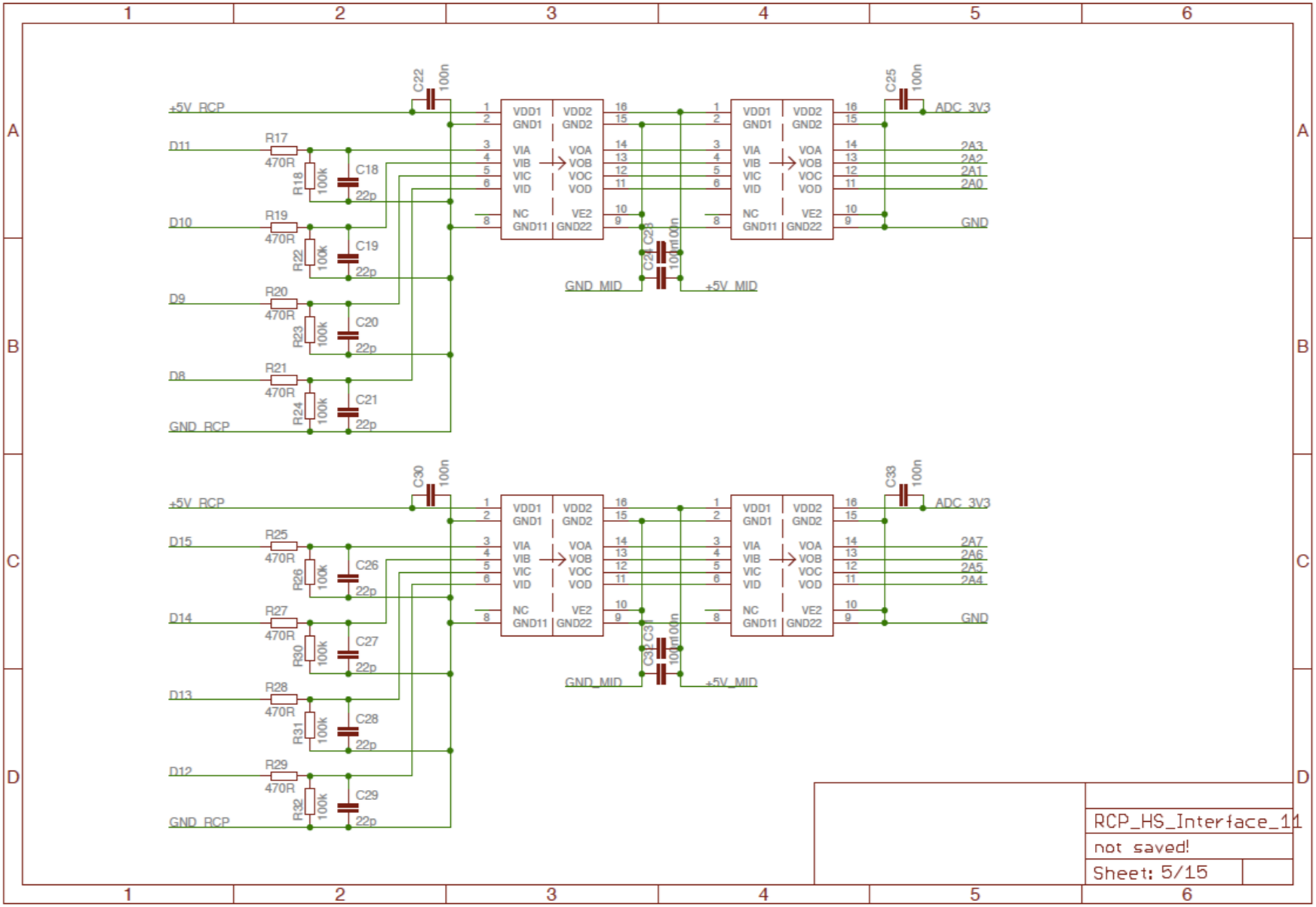
Appendix

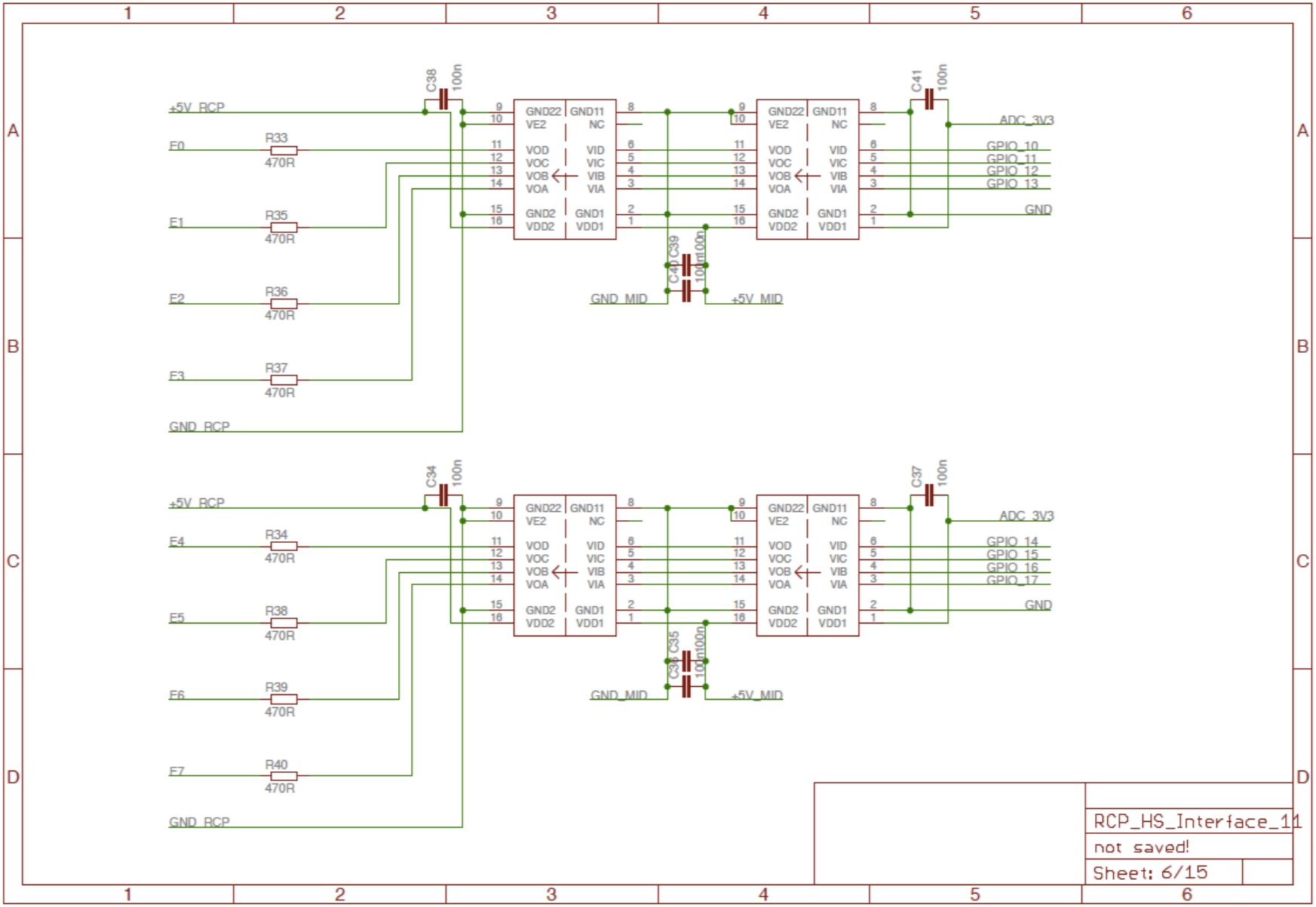

HFR068RA
IPU_TO_PCB_CONNECTOR
IPU-CON-DEVICE
AST_TO_PCB_CONNECTOR
AST-CON-DEVICE
C138
C139
C71
R144
C46
C145
C144
C143
C142
C141
C140
C150
C149
C148
C147
C137
RN1A
RN1B
RN2A
RN2B
RN2C
RN2D
RN3A
RN3B
RN3C
RN3D
RN4A
RN4B
RN4C
RN4D
GND
TXD
ADC3_MISO
ADC2_MISO
ADC23_CLK
%SPI1_CS%
SPI1_CLK
%SAFF_BREAK%
%SPI3_CS%
SPI3_MOSI
SPI2_MOSI
%SPI2_CS%
SPI1_C_MISO
SPI1_B_MISO
SPI1_A_MISO
RLYCTLTSTOK
%VAR_ERR%
5V_PWRGD
%SUPVTG_FAIL%
%RST_BFS_TO_AST%
SPIX_CS_AST_PARAM
ADC_3V3
RTC_BCKUP_5V
%GRIDMS_A_PHSC_HW%
%GRIDMS_A_PHSB_HW%
%GRIDMS_A_PHSA_HW%
%GRIDMS_A_PHSC_FRT%
%GRIDMS_A_PHSB_FRT%
%GRIDMS_A_PHSA_FRT%
%INSRISCTL%
FANINT_CTL
FANINT_FB
FANEXT_FB
FANEXT_CTL
AFCI_CLK
RLYBCCTL
RLYCACTL
RLYABCTL
INSRISRLYNGCTL
INSRISRLYPSCTL
DIW_DGS_CTL
DIW_TSTCUR_CTL
+5V_BFS
+18V
RCP_HS_Interface_11
not saved!
Sheet: 1/15

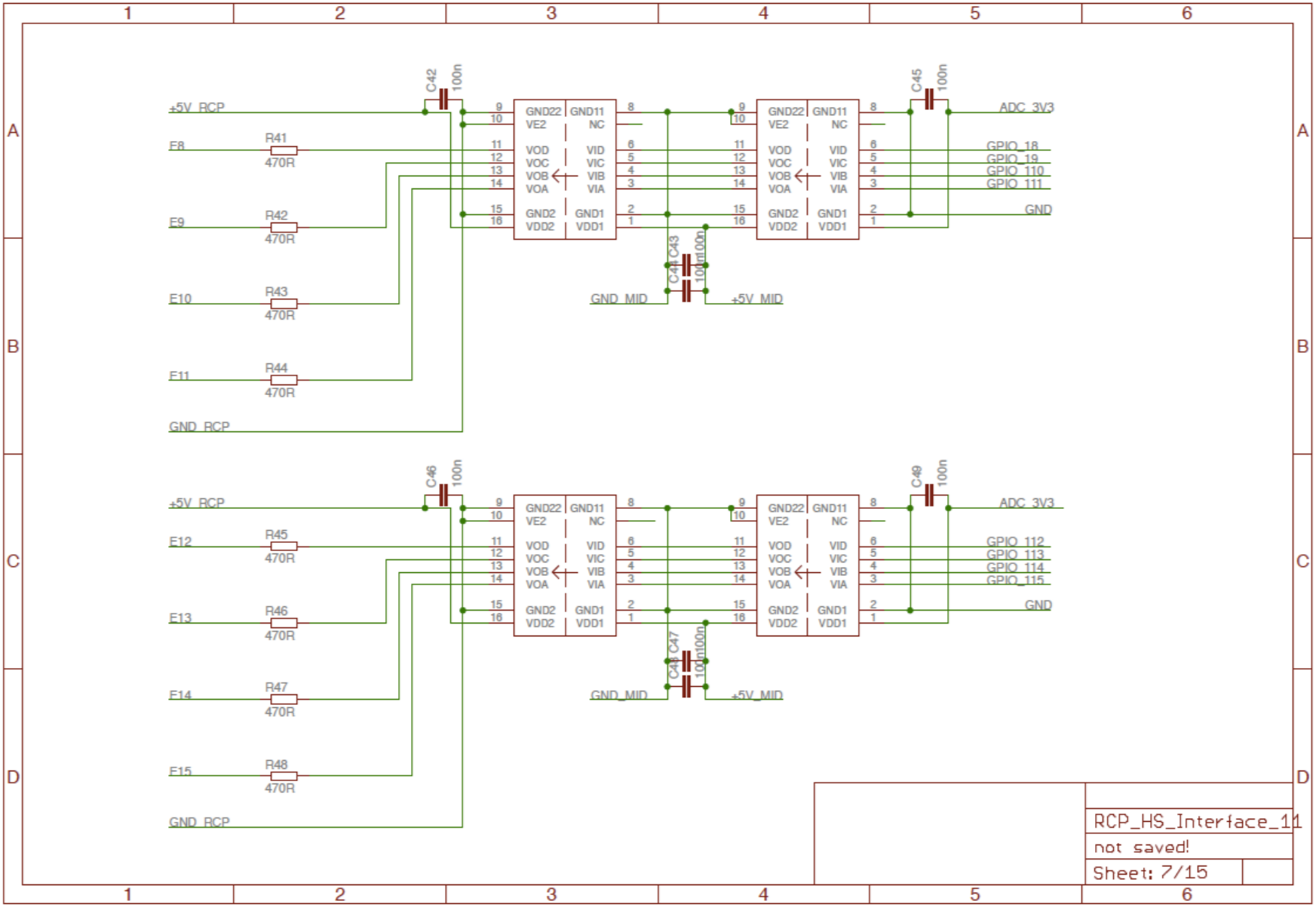

PCBtoFPGA1
GPIO_0_IN0 1 2 GPIO_00
GPIO_0_IN1 3 4 GPIO_01
GPIO_02 5 6 GPIO_03
GPIO_04 7 8 GPIO_05
GPIO_06 9 10 GPIO_07
±5V_RES 11 12
GPIO_08 13 14 GPIO_09
GPIO_010 15 16 GPIO_011
GPIO_012 17 18 GPIO_013
GPIO_014 19 20 GPIO_015
GPIO_016 21 22 GPIO_017
GPIO_018 23 24 GPIO_019
GPIO_020 25 26 GPIO_021
GPIO_022 27 28 GPIO_023
3V3 NOT CONNECTED 29 30
GPIO_024 31 32 GPIO_025
GPIO_026 33 34 GPIO_027
GPIO_028 35 36 GPIO_029
GPIO_030 37 38 GPIO_031
GPIO_032 39 40 GPIO_033
GPIO-0
GND
PCBtoFPGA2
GPIO_1_IN0 1 2 GPIO_10
GPIO_1_IN1 3 4 GPIO_11
GPIO_12 5 6 GPIO_13
GPIO_14 7 8 GPIO_15
GPIO_16 9 10 GPIO_17
±5V_RES 11 12
GPIO_18 13 14 GPIO_19
GPIO_110 15 16 GPIO_111
GPIO_112 17 18 GPIO_113
GPIO_114 19 20 GPIO_115
GPIO_116 21 22 GPIO_117
GPIO_118 23 24 GPIO_119
GPIO_120 25 26 GPIO_121
GPIO_122 27 28 GPIO_123
3V3 NOT CONNECTED 29 30
GPIO_124 31 32 GPIO_125
GPIO_126 33 34 GPIO_127
GPIO_128 35 36 GPIO_129
GPIO_130 37 38 GPIO_131
GPIO_132 39 40 GPIO_133
GPIO-1
GND
PCBtoFPGA3
3V3 NOT CONNECTED 1 2 GPIO_2_IN0
GPIO_2_IN1 3 4 GPIO_2_IN2
GPIO_20 5 6 GPIO_21
GPIO_22 7 8 GPIO_23
GPIO_24 9 10 GPIO_25
GPIO_26 11 12 GPIO_27
GPIO_28 13 14 GPIO_29
GPIO_210 15 16 GPIO_211
GPIO_212 17 18 ANALOG_IN5
ANALOG_IN6 19 20 ANALOG_IN7
ANALOG_IN3 21 22 ANALOG_IN2
ANALOG_IN4 23 24 ANALOG_IN0
ANALOG_IN1 25 26
GPIO-2
GND
RCP_HS_Interface_11
not saved!
Sheet: 2/15

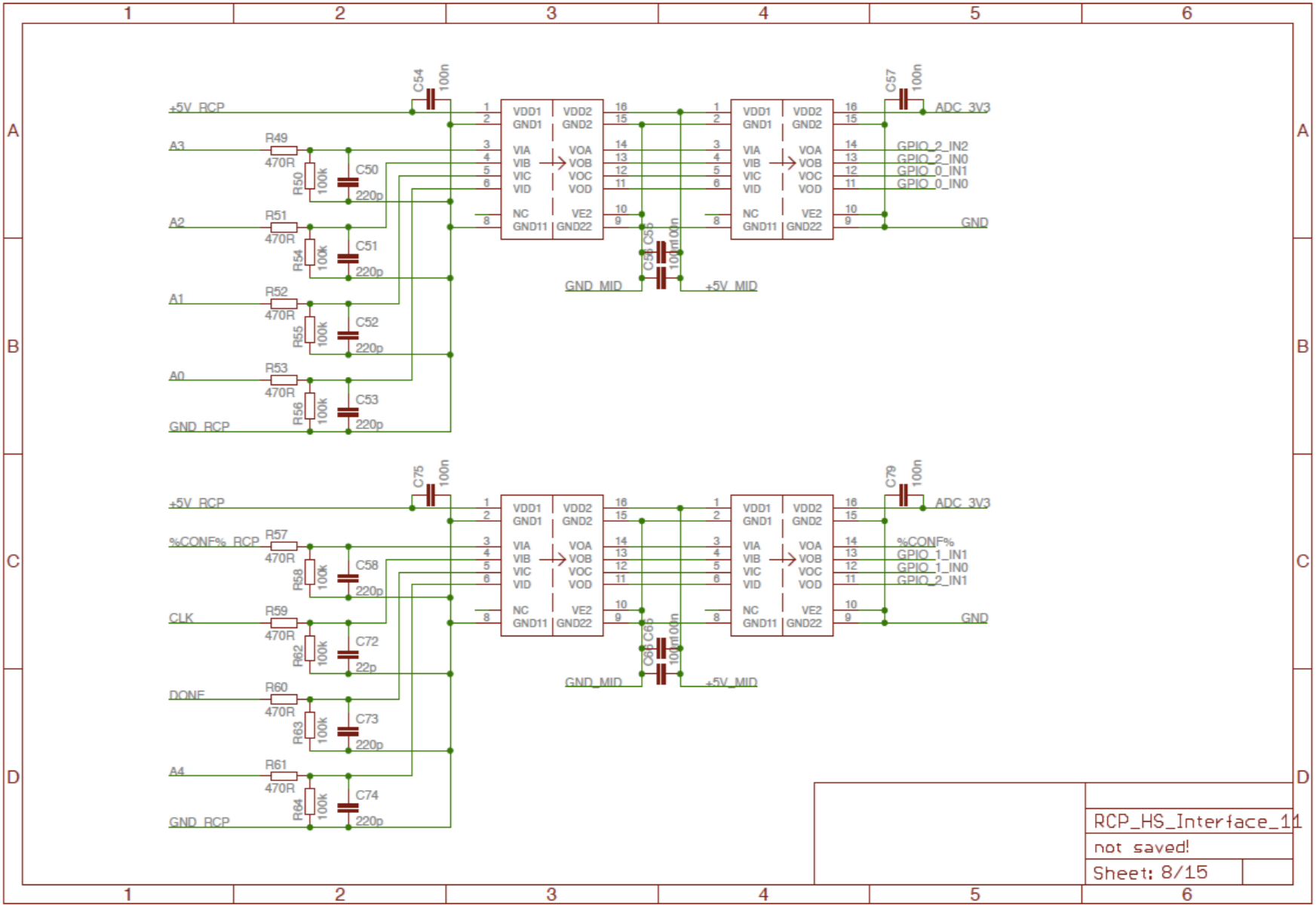

5VSUP_RCP
+5V_BFS
GND
C109
C59
10u
100nF
+VIN
+VOUT
-VIN
-VOUT
NC
TRACO_5VTO5V_DEVICE
C60
C110
100n
+5V_RCP
GND_RCP
5VSUP_MID
C113
C61
C62
C114
+5V_MID
GND_MID
FERRITE
WURTH_04-019853
C63
10u
C64
C12_0805
100n
C9_0603
22n
C13_0603
10p
R15_0603
30k
C15_0603
680p
3V3SUPPLY
SEMTECH
NC
IN
BST
SW
SS/EN
COMP
ROSC
GND
FB
R14_0603
18k
VISHAY_10-02345
BS_DIODE
C13
100n
FREEWHEEL
ONSEMI_10-16155
R7_MINIMELF
1R
C69
100n
INDUCTOR
WURTH_04-005618
ADC_3V3
C16_0603
10n
C18_0805
10u
C19_0805
10u
R7_TK50_1%
22k
R14_TK50_1%
9k53
R7=R14*(3.3-1)=21.9k
Vo=R7/R14+1=3.308
GND_RCP
MM_GND1
MM_GND2
MM_GND3
MM_GND4
MM_GND5
MM_GND6
1M
GND_MID
MM_GND7
MM_GND8
MM_GND9
MM_GND10
MM_GND11
MM_GND12
GND
JP1
JP2
JP3
+5V_BFS
ADC_3V3
RCP_HS_Interface_11
not saved!
Sheet: 3/15

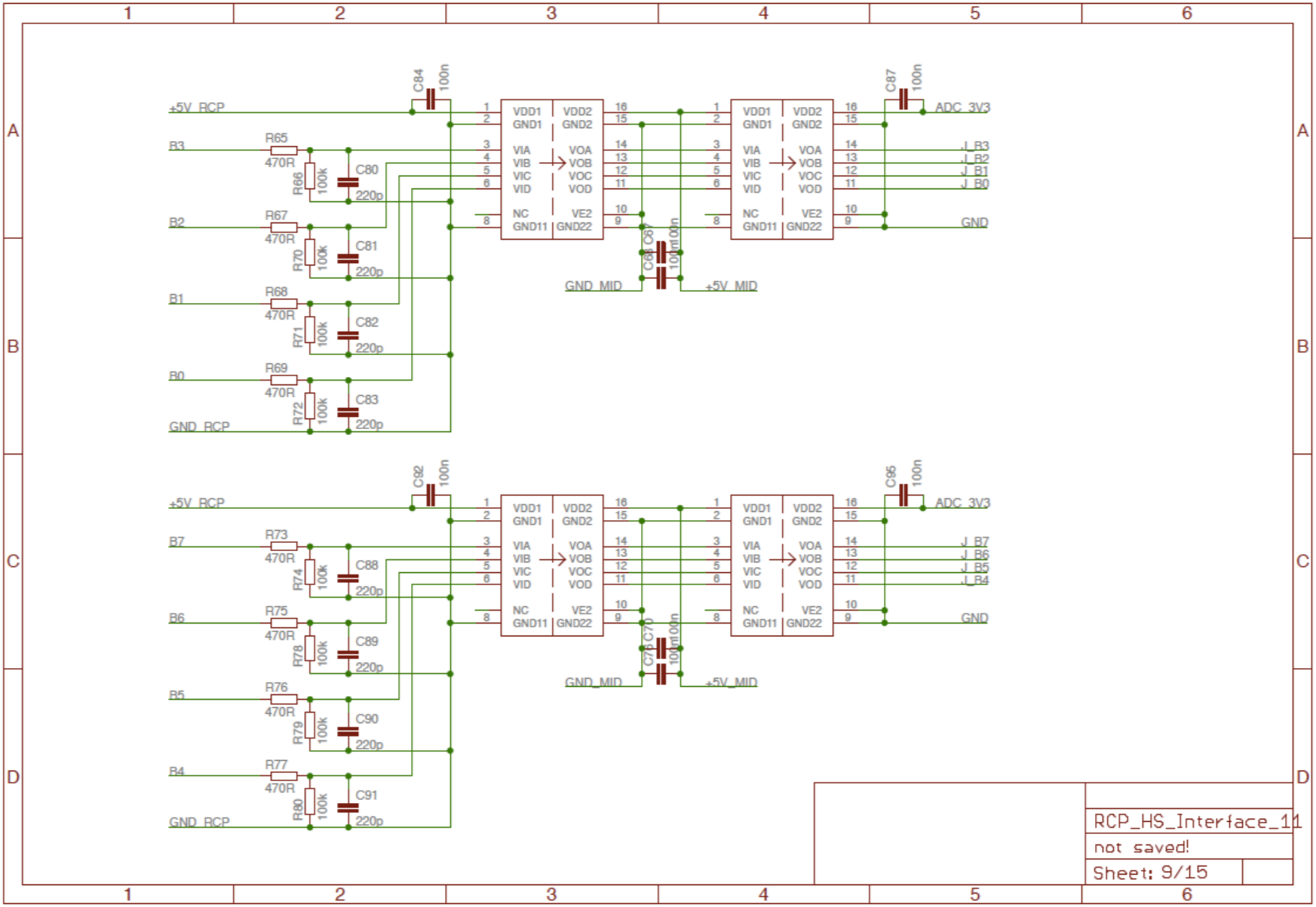

C=1/(2*pi*f*R)=1/(2*pi*16M*330)=30pF
+5V_RCP
D3
D2
D1
D0
GND_RCP
R3
R2
R4
R5
470R
R1
R6
R7
R8
100k
C1
C2
C3
C4
22p
C5
100n
VDD1
GND1
VIA
VIB
VIC
VID
NC
GND11
VDD2
GND2
VOA
VOB
VOC
VOD
VE2
GND22
C6
C7
100n
GND_MID
+5V_MID
C8
ADC_3V3
1A3
1A2
1A1
1A0
GND
D7
D6
D5
D4
R9
R11
R12
R13
R10
R14
R15
R16
C9
C10
C11
C12
C14
C17
1A7
1A6
1A5
1A4
ADC_3V3
100n
C133
C134
RCP_TO_FPGA1
VCCA
VCCB
1DIR
/1OE
2DIR
/2OE
GPIO_116
1A0
1A1
1A2
1A3
1A4
1A5
1A6
1A7
2A0
2A1
2A2
2A3
2A4
2A5
2A6
2A7
1B0
1B1
1B2
1B3
1B4
1B5
1B6
1B7
2B0
2B1
2B2
2B3
2B4
2B5
2B6
2B7
GPIO_10
GPIO_11
GPIO_12
GPIO_13
GPIO_14
GPIO_15
GPIO_16
GPIO_17
GPIO_18
GPIO_19
GPIO_110
GPIO_111
GPIO_112
GPIO_113
GPIO_114
GPIO_115
NXP_13-716424516_DEVICE
RCP_HS_Interface_11
not saved!
Sheet: 4/15

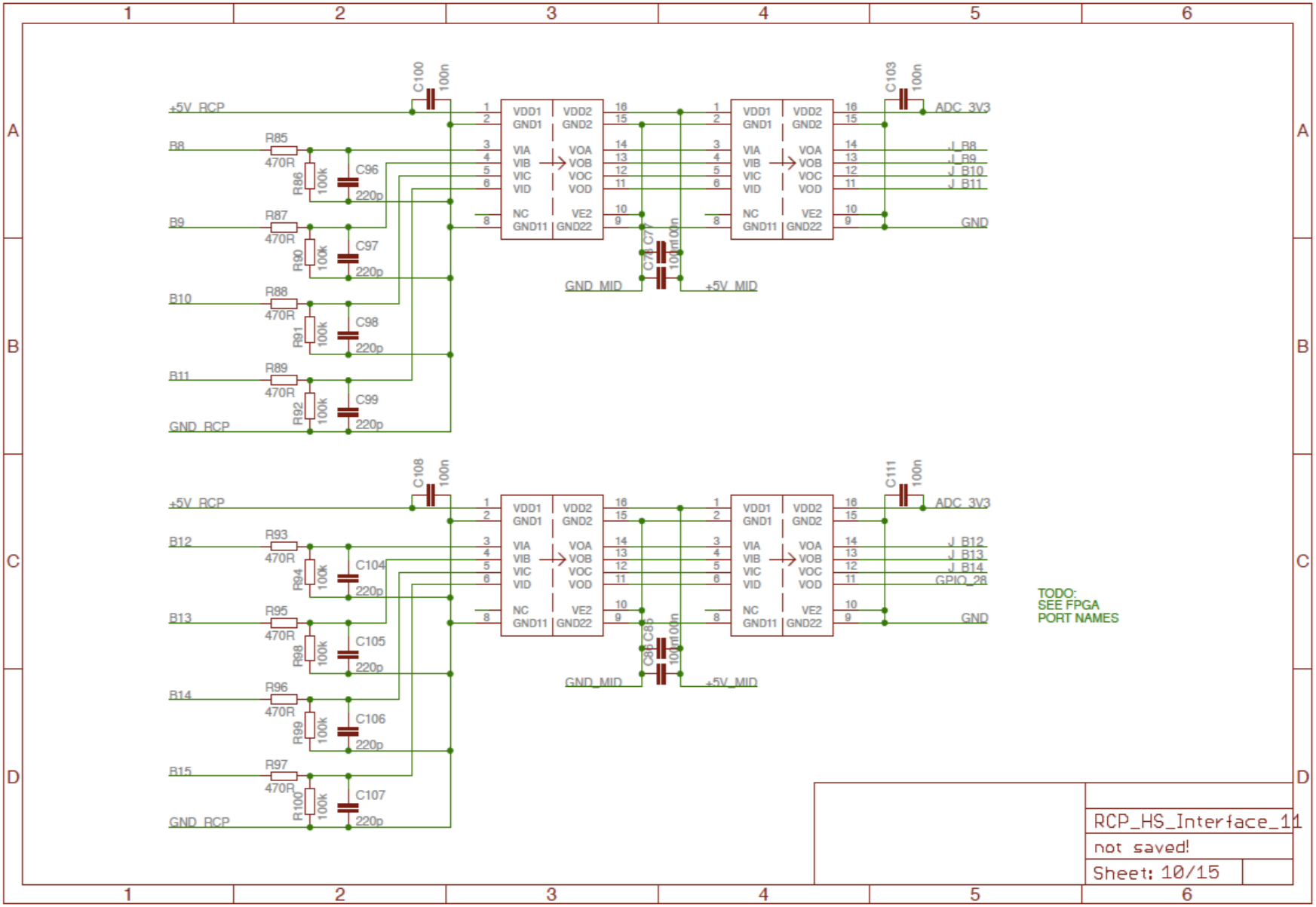

1
2
3
4
5
6
A
B
C
D
+5V_RCP
D11
D10
D9
D8
GND_RCP
R17
R19
R20
R21
470R
R18
R22
R23
R24
100k
C18
C19
C20
C21
22p
C22
100n
VDD1
GND1
VIA
VIB
VIC
VID
NC
GND11
VDD2
GND2
VOA
VOB
VOC
VOD
VE2
GND22
GND_MID
C24
C23
10n
+5V_MID
C25
ADC_3V3
2A3
2A2
2A1
2A0
GND
D15
D14
D13
D12
R25
R27
R28
R29
R26
R30
R31
R32
C26
C27
C28
C29
C30
C31
C33
2A7
2A6
2A5
2A4
RCP_HS_Interface_11
not saved!
Sheet: 5/15

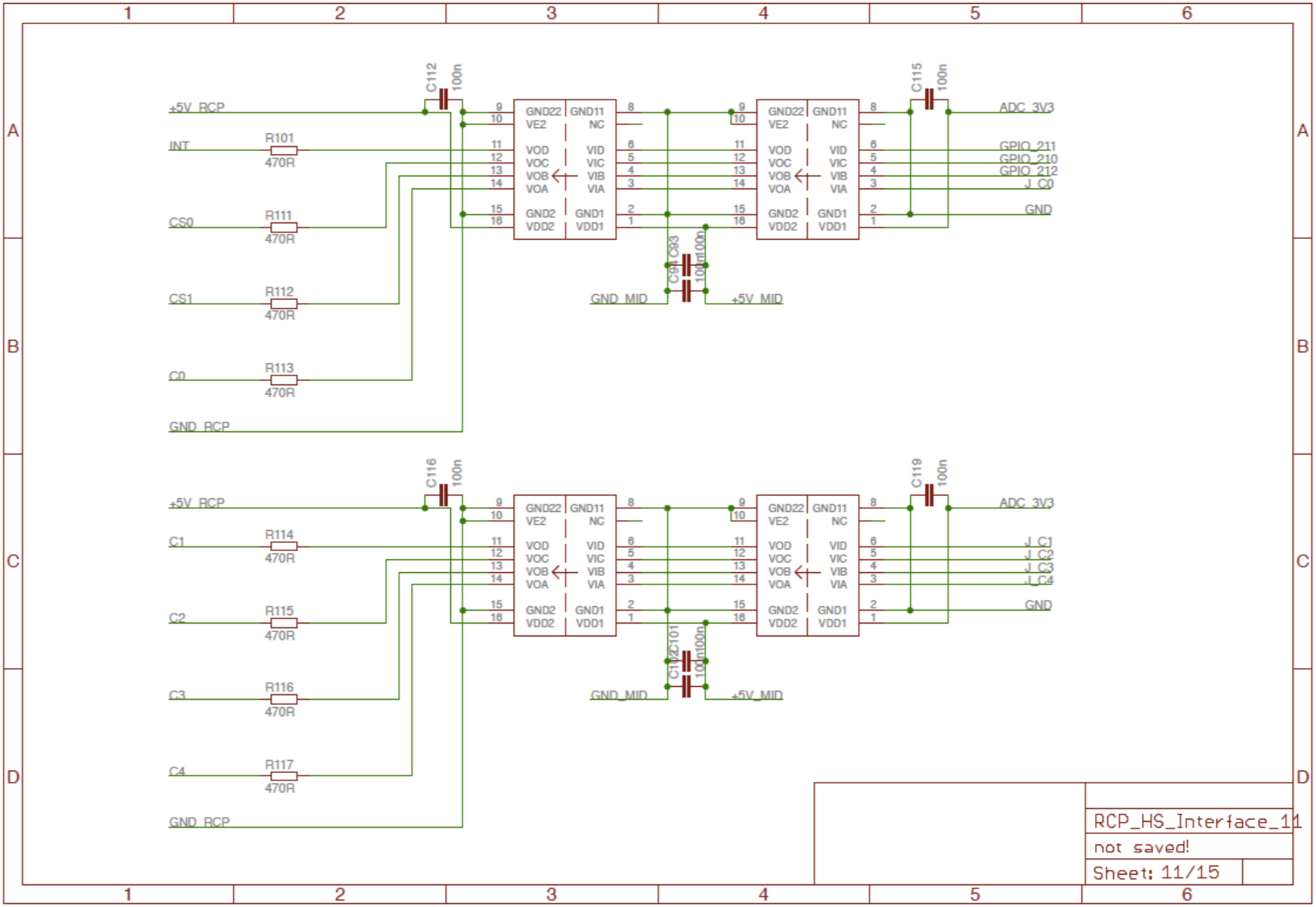

1
2
3
4
5
6
A
B
C
D
+5V_RCP
E0
R33
470R
E1
R35
470R
E2
R36
470R
E3
R37
470R
GND_RCP
C38
100n
C41
100n
C39
100n
C40
100n
GND_MID
+5V_MID
GND22
GND11
VE2
NC
VOD
VOC
VOB
VOA
VID
VIC
VIB
VIA
GND2
VDD2
GND1
VDD1
ADC_3V3
GPIO_10
GPIO_11
GPIO_12
GPIO_13
GND
+5V_RCP
E4
R34
470R
E5
R38
470R
E6
R39
470R
E7
R40
470R
GND_RCP
C34
100n
C37
100n
C35
100n
C36
100n
GND_MID
+5V_MID
ADC_3V3
GPIO_14
GPIO_15
GPIO_16
GPIO_17
GND
RCP_HS_Interface_11
not saved!
Sheet: 6/15

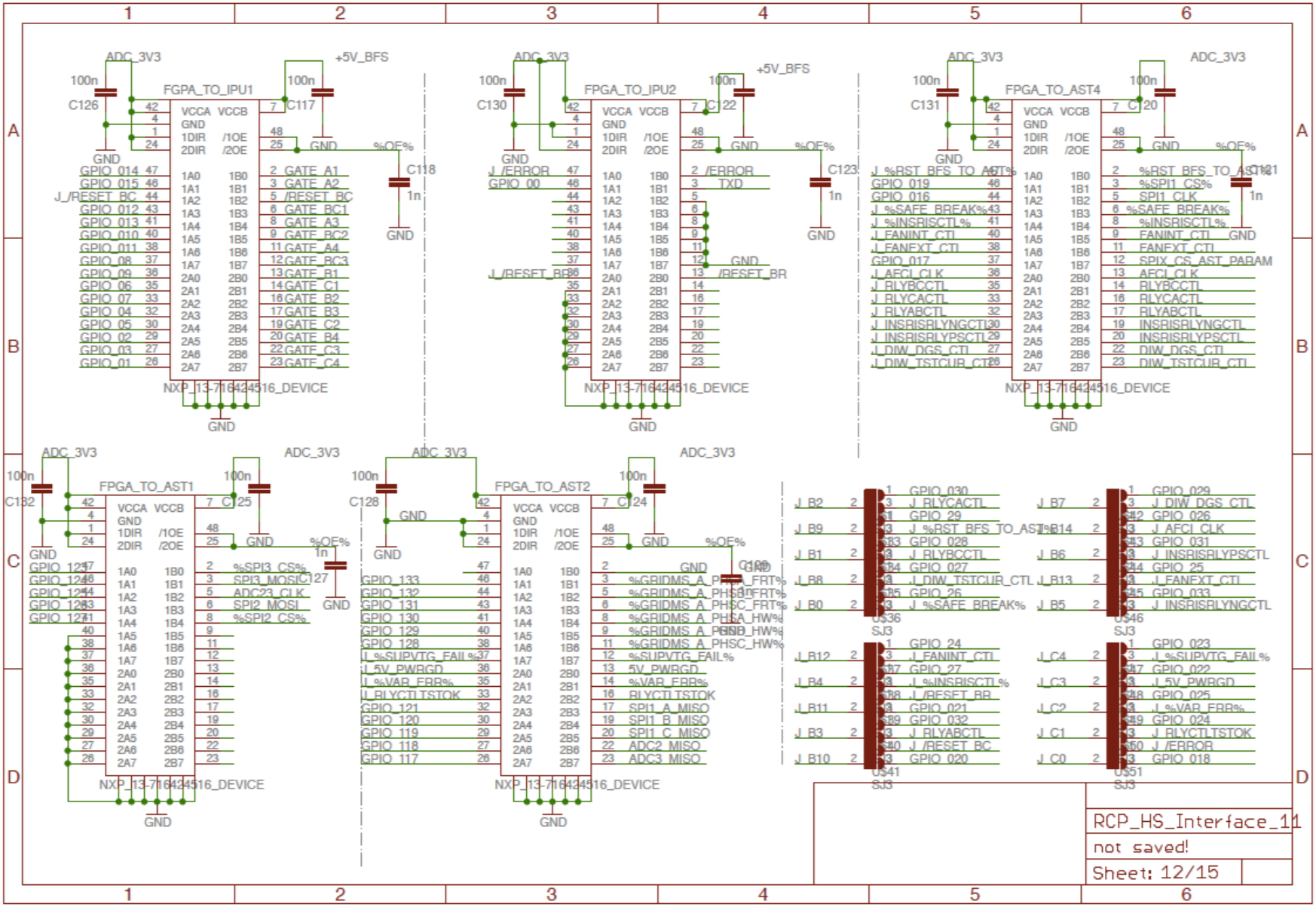

1
2
3
4
5
6
A
B
C
D
+5V_RCP
E8
E9
E10
E11
GND_RCP
R41
R42
R43
R44
470R
C42
100n
C43
100n
C44
100n
C45
100n
GND22
GND11
VE2
NC
VOD
VID
VOC
VIC
VOB
VIB
VOA
VIA
GND2
GND1
VDD2
VDD1
GND_MID
+5V_MID
ADC_3V3
GPIO_18
GPIO_19
GPIO_110
GPIO_111
GND
E12
E13
E14
E15
R45
R46
R47
R48
C46
100n
C47
100n
C48
100n
C49
100n
GPIO_112
GPIO_113
GPIO_114
GPIO_115
RCP_HS_Interface_11
not saved!
Sheet: 7/15

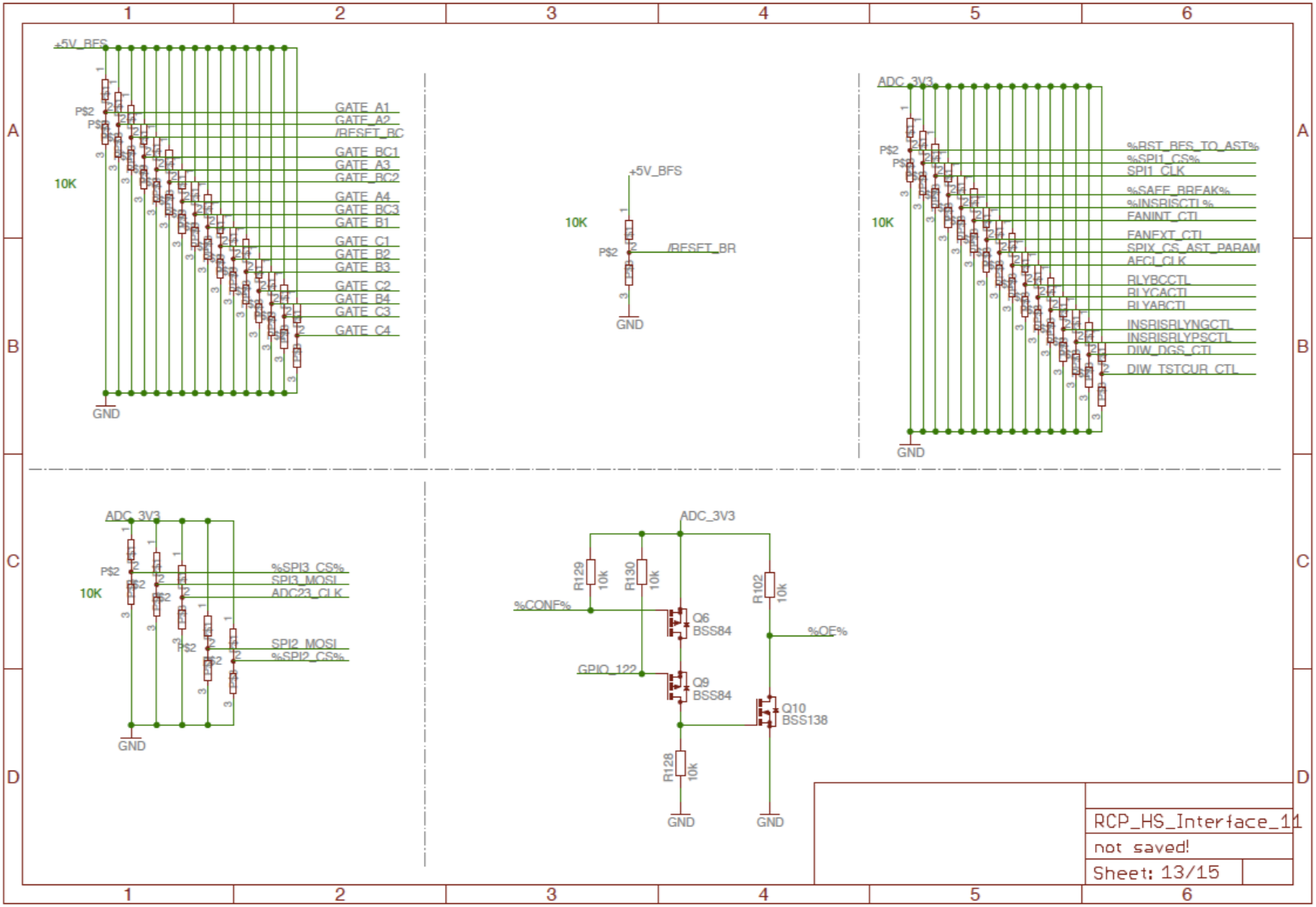

1
2
3
4
5
6
A
B
C
D
+5V_RCP
A3
A2
A1
A0
GND_RCP
R49
R51
R52
R53
470R
R50
R54
R55
R56
100k
C50
C51
C52
C53
220p
C54
100n
VDD1
GND1
VIA
VIB
VIC
VID
NC
GND11
VDD2
GND2
VOA
VOB
VOC
VOD
VE2
GND22
GND_MID
C56
C55
100n
+5V_MID
C57
ADC_3V3
GPIO_2_IN2
GPIO_2_IN0
GPIO_0_IN1
GPIO_0_IN0
GND
%CONF%_RCP
CLK
DONE
A4
R57
R59
R60
R61
R58
R62
R63
R64
C58
C72
C73
C74
22p
C75
C66
C65
C79
%CONF%
GPIO_1_IN1
GPIO_1_IN0
GPIO_2_IN1
RCP_HS_Interface_11
not saved!
Sheet: 8/15

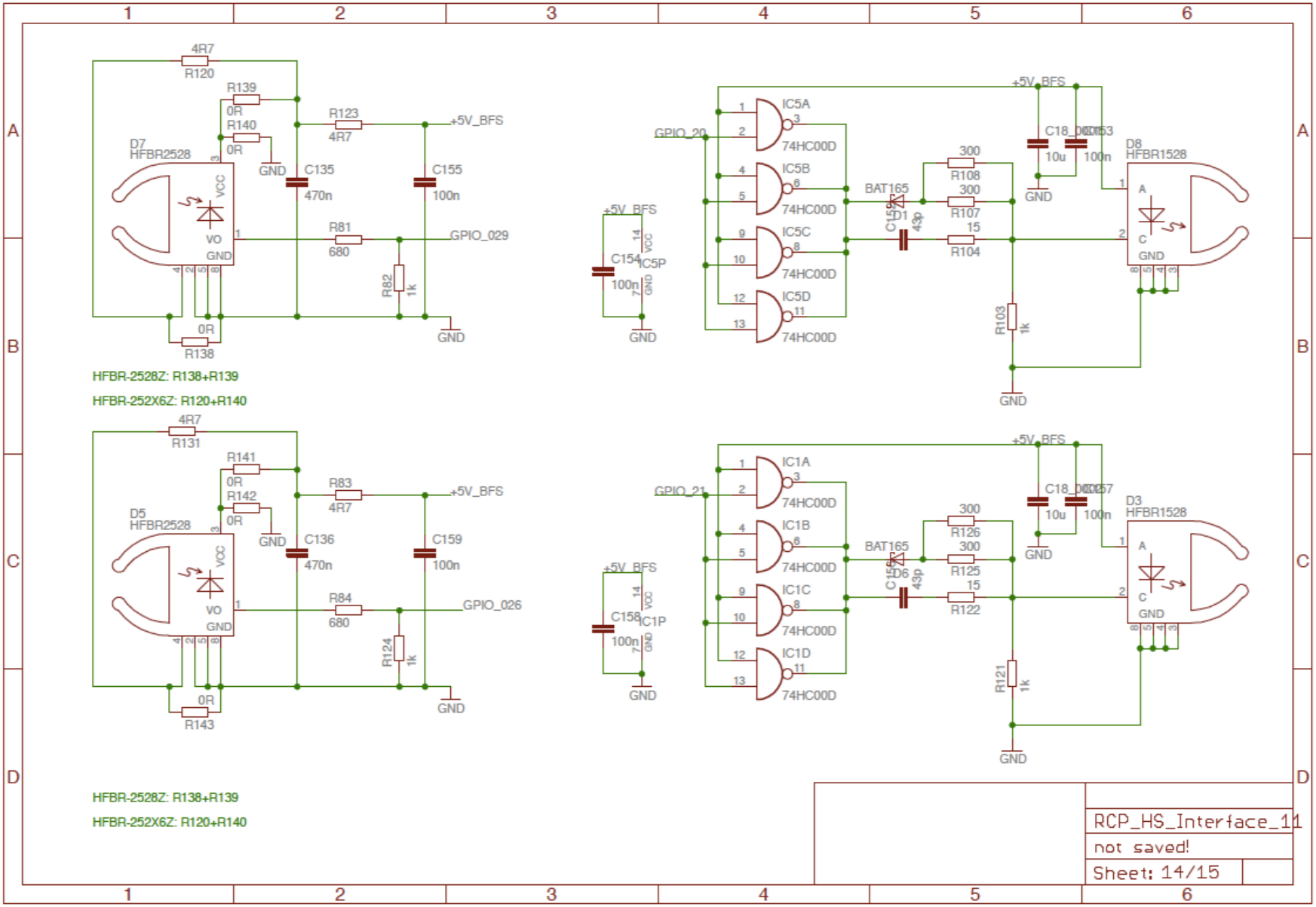

+5V_RCP
B3
B2
B1
B0
GND_RCP
R65
R66
R67
R70
R68
R71
R69
R72
470R
100k
C80
C81
C82
C83
220p
C84
100n
VDD1
GND1
VIA
VIB
VIC
VID
NC
GND11
VDD2
GND2
VOA
VOB
VOC
VOD
VE2
GND22
GND_MID
+5V_MID
C86
C87
100n
ADC_3V3
J_B3
J_B2
J_B1
J_B0
GND
+5V_RCP
B7
B6
B5
B4
GND_RCP
R73
R74
R75
R78
R76
R79
R77
R80
C88
C89
C90
C91
C92
C95
C70
C76
J_B7
J_B6
J_B5
J_B4
RCP_HS_Interface_11
not saved!
Sheet: 9/15

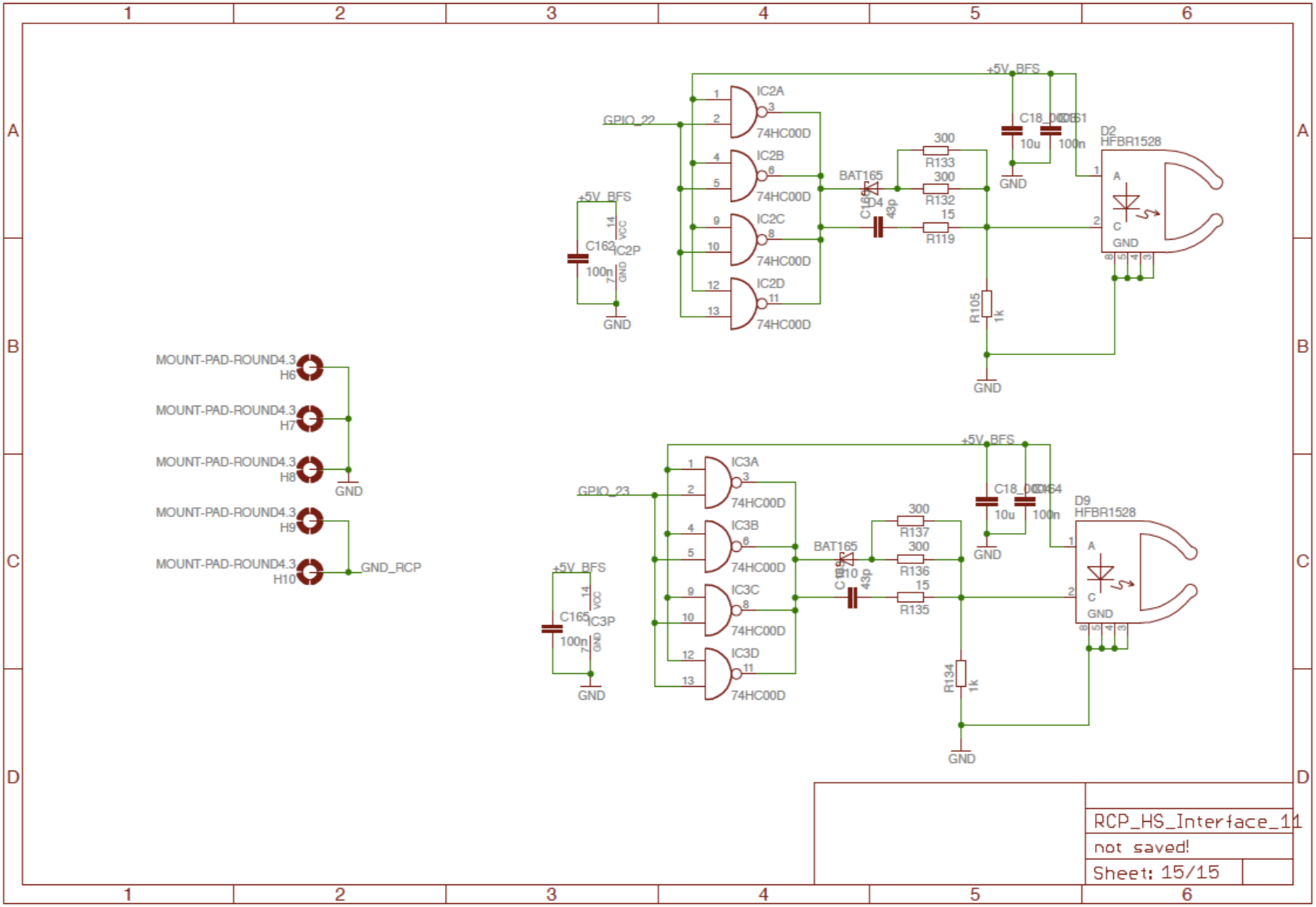

1
2
3
4
5
6
A
B
C
D
+5V_RCP
B8
B9
B10
B11
GND_RCP
R85
R87
R88
R89
470R
100k
C96
C97
C98
C99
220p
C100
100n
VDD1
GND1
VIA
VIB
VIC
VID
NC
GND11
VDD2
GND2
VOA
VOB
VOC
VOD
VE2
GND22
GND_MID
+5V_MID
C103
ADC_3V3
J_B8
J_B9
J_B10
J_B11
GND
+5V_RCP
B12
B13
B14
B15
GND_RCP
R93
R95
R96
R97
C104
C105
C106
C107
C108
C111
J_B12
J_B13
J_B14
GPIO_28
TODO:
SEE FPGA
PORT NAMES
RCP_HS_Interface_11
not saved!
Sheet: 10/15

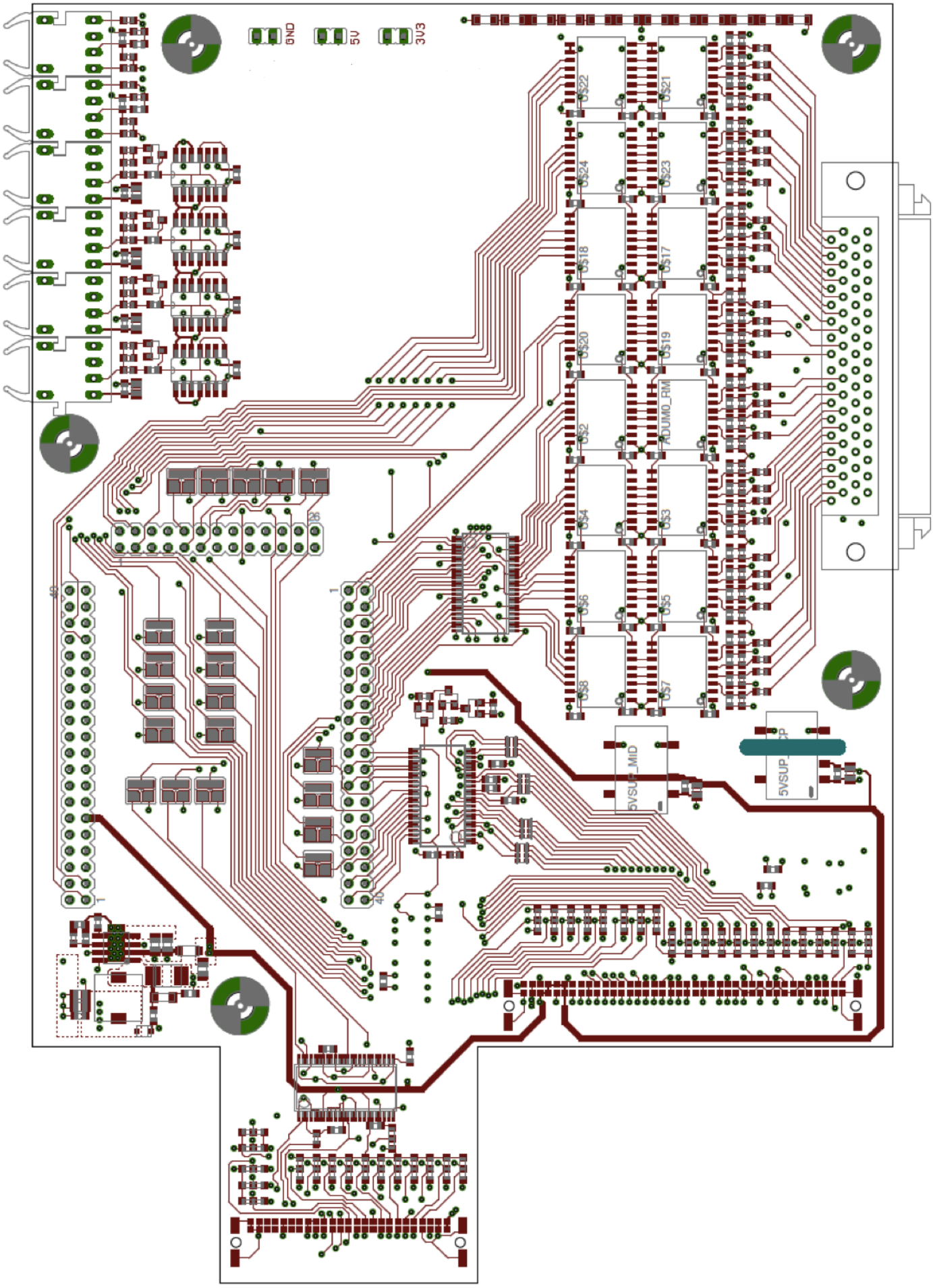

1
2
3
4
5
6
A
B
C
D
+5V_RCP
INT
CS0
CS1
C0
GND_RCP
R101
470R
R111
470R
R112
470R
R113
470R
C112
100n
C115
100n
GND22
GND11
VE2
NC
VOD
VID
VOC
VIC
VOB
VIB
VOA
VIA
GND2
GND1
VDD2
VDD1
C93
100n
C91
100n
GND_MID
+5V_MID
ADC_3V3
GPIO_211
GPIO_210
GPIO_212
J_C0
GND
C1
C2
C3
C4
R114
470R
R115
470R
R116
470R
R117
470R
C116
100n
C119
100n
C117
100n
C2101
J_C1
J_C2
J_C3
J_C4
RCP_HS_Interface_11
not saved!
Sheet: 11/15

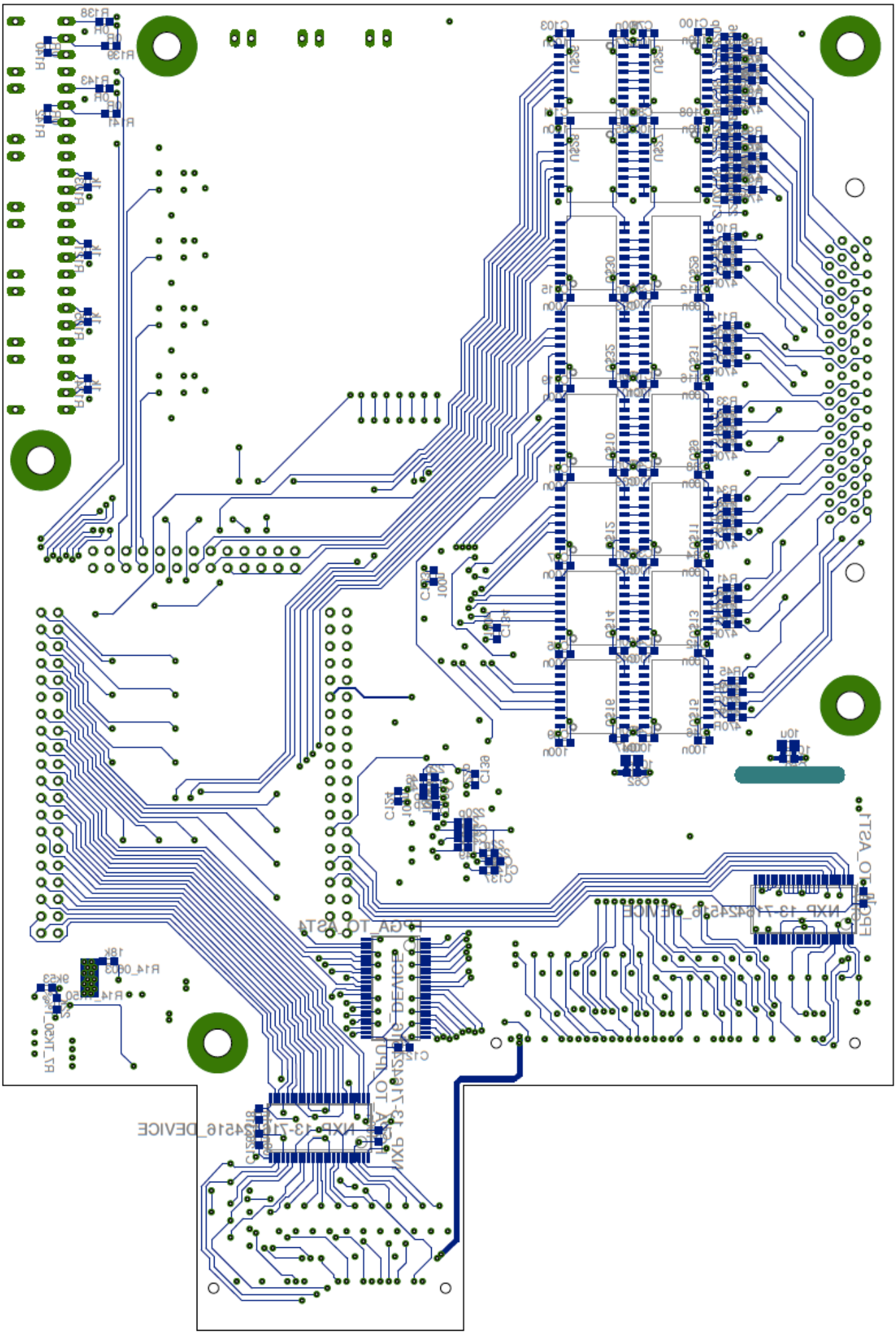

FGPA_TO_IPU1
FPGA_TO_IPU2
FPGA_TO_AST4
FPGA_TO_AST1
FPGA_TO_AST2
ADC_3V3
+5V_BFS
100n
GND
%OF%
VCCA VCCB
1DIR /1OE
2DIR /2OE
NXP_13-716424516_DEVICE
RCP_HS_Interface_11
not saved!
Sheet: 12/15


+5V_BFS
10K
P$2
GATE_A1
GATE_A2
/BFSET_BC
GATE_BC1
GATE_A3
GATE_BC2
GATE_A4
GATE_BC3
GATE_B1
GATE_C1
GATE_B2
GATE_B3
GATE_C2
GATE_B4
GATE_C3
GATE_C4
GND
+5V_BFS
10K
P$2
/BFSET_BR
GND
ADC_3V3
10K
P$2
%RST_BFS_TO_AST%
%SPI1_CS%
SPI1_CLK
%SAFE_BREAK%
%INSRISCTL%
FANINT_CTL
FANEXT_CTL
SPIX_CS_AST_PARAM
AFCL_CLK
RLYBCCTL
RLYCACTL
RLYABCTL
INSRISRLYNGCTL
INSRISRLYPSCTL
DIW_DGS_CTL
DIW_TSTCUR_CTL
GND
ADC_3V3
10K
P$2
%SPI3_CS%
SPI3_MOSI
ADC23_CLK
SPI2_MOSI
%SPI2_CS%
GND
ADC_3V3
R129
10K
R130
10K
R102
10K
%CONF%
Q6
BSS84
%OE%
GPIO_122
Q9
BSS84
Q10
BSS138
R128
10K
GND
GND
RCP_HS_Interface_11
not saved!
Sheet: 13/15


1
2
3
4
5
6
A
B
C
D
4R7
R120
R139
0R
R140
0R
GND
C135
470n
R123
4R7
+5V_BFS
C155
100n
D7
HFBR2528
VCC
VO
GND
R81
680
GPIO_029
R82
1k
0R
R138
GND
HFBR-2528Z: R138+R139
HFBR-252X6Z: R120+R140
4R7
R131
R141
0R
R142
0R
GND
C136
470n
R83
4R7
+5V_BFS
C159
100n
D5
HFBR2528
VCC
VO
GND
R84
680
GPIO_026
R124
1k
0R
R143
GND
HFBR-2528Z: R138+R139
HFBR-252X6Z: R120+R140
+5V_BFS
C154
100n
IC5P
GND
GPIO_20
IC5A
IC5B
IC5C
IC5D
74HC00D
BAT165
C157
D1
43p
300
R108
300
R107
15
R104
R103
1k
+5V_BFS
C18_00053
10u
100n
GND
D8
HFBR1528
A
C
GND
+5V_BFS
C158
100n
IC1P
GND
GPIO_21
IC1A
IC1B
IC1C
IC1D
74HC00D
BAT165
C156
D6
43p
300
R126
300
R125
15
R122
R121
1k
+5V_BFS
C18_00057
10u
100n
GND
D3
HFBR1528
A
C
GND
RCP_HS_Interface_11
not saved!
Sheet: 14/15


1
2
3
4
5
6
A
B
C
D
MOUNT-PAD-ROUND4.3
H6
MOUNT-PAD-ROUND4.3
H7
MOUNT-PAD-ROUND4.3
H8
MOUNT-PAD-ROUND4.3
H9
MOUNT-PAD-ROUND4.3
H10
GND
GND_RCP
GPIO_22
+5V_BFS
IC2A
IC2B
IC2C
IC2D
IC2P
74HC00D
C162
100n
BAT165
C163
43p
R133
300
R132
300
R119
15
R105
1k
C18_00061
10u
100n
D2
HFBR1528
GPIO_23
IC3A
IC3B
IC3C
IC3D
IC3P
C165
C164
R137
R136
R135
R134
C18_00064
D9
RCP_HS_Interface_11
not saved!
Sheet: 15/15


GND
5V
3V3
U22
U21
U24
U23
U18
U17
U20
U19
ADUM0_RM
U2
U4
U3
U6
U5
U8
U7
5VSUP_MID
5VSUP
J6
J9
1
40



Bibliography

[1] L. Owens, "Vannevar Bush and the differential analyzer: The text and context of an early computer," Technol. Culture, vol. 27, no. 1, pp. 63–95, Jan. 1986.
[2] W. Dirk and M. Kratz, "A real time development platform for next generation of power systems control functions," IEEE Trans. Ind. Electron., vol. 57, no. 4, pp. 1159–1167, Apr. 2010.
[3] J. Bélanger, V. Lapointe, C. Dufour, and L. Schoen, "eMEGAsim: An open high-performance distributed real-time power grid simulator. Architecture and specification," presented at the Int. Conf. Power System, Bangalore, India, Dec., 2007.
[4] S. Abourida and J. Belanger, "Real-time platform for the control proto- typing and simulation of power electronics and motor drives," presented at the 3rd Int. Conf. Modeling, Simulation and Applied Optimization, Sharajah, U.A.E., Jan., 2009.
[5] C. Bordas, C. Dufour, and O. Rudloff, "A 3-level neutral-clamped inverter model with natural switching mode support for the real-time simulation of variable speed drives," in Proc. Planet RT, Jul. 2009.
[6] C. Dufour, G. Dumur, J. N. Paquin, and J. Bélanger, "A PC-based hardware-in-the-loop simulator for the integration testing of modern train and ship propulsion systems," in Proc. IEEE Power Electron. Spec. Conf., Island of Rhodes, Greece, Jun. 2008, pp. 444–449.
[7] Y. Liu, Z. Xi, Z. Liang, W. Song, S. Bhattacharya, A. Huang, M. Steuer, W. Litzenberger, L. Anderson, R. Adapa, and A. Sundaram, "Controller hardware-in-the-loop validation for a 10 MVA ETO-based STATCOM for wind farm application," in Proc. IEEE ECCE, San Jose, CA, Sep. 2009, pp. 1398–1403.
[8] A.-L. Allégre, A. Bouscayrol, J.-N. Verhille, P. Delarue, E. Chattot, and S. El-Fassi, "Reduced-scale-power hardware-in-the-loop simulation of an innovative subway," IEEE Trans. Ind. Electron., vol. 57, no. 4, pp. 1175– 1185, Apr. 2010.
[9] S. Lentijo, S. D'Arco, and A. Monti, "Comparing the dynamic performances of power hardware in the loop interfaces," IEEE Trans. Ind. Electron., vol. 57, no. 4, pp. 1195–1207, Apr. 2010.
[10] Y. S. Rao and M. C. Chandorkar, "Real-time electrical load emulator using optimal feedback control technique," IEEE Trans. Ind. Electron., vol. 57, no. 4, pp. 1217–1225, Apr. 2010.
[11] S. Grubic, B. Amlang, W. Schumacher, and A. Wenzel, "A high performance electronic hardware-in-the-loop drive-load-simulation using a linear inverter (LinVerter)," IEEE Trans. Ind. Electron., vol. 57, no. 4, pp. 1208–1216, Apr. 2010.
[12] M. Steurer, C. S. Edrington, M. Sloderbeck, W. Ren, and J. Langston, "A megawatt-scale power hardware-in-the-loop simulation setup for mo- tor drives," IEEE Trans. Ind. Electron., vol. 57, no. 4, pp. 1254–1260, Apr. 2010.
[13] C. Dufour, J. Belanger, S. Abourida, and V. Lapointe, "FPGA-based real-time simulation of finite-element analysis permanent magnet syn- chronous machine drives," in Proc. IEEE Power Electron. Spec. Conf., Orlando, FL, Jun. 2007, pp. 909–915.
[14] G. G. Parma and V. Dinavahi, "Real-time digital hardware simulation of power electronics and drives," IEEE Trans. Power Del., vol. 22, no. 2, pp. 1235–1246, Apr. 2007.
[15] A. Myaing and V. Dinavahi, "FPGA-based real-time emulation of power electronics systems with detailed representation of device characteristics," IEEE Trans. Ind. Electron., vol. 58, no. 1, pp. 358–368, Jan. 2011.
[16] R. Ruelland, G. Gateau, T. A. Meynard, and J. C. Hapiot, "Design of FPGA-based emulator for series multicell converters using co-simulation tools," IEEE Trans. Power Electron., vol. 18, no. 1, pp. 455–463, Jan. 2003.
[17]D.Majstorovic´,Z.Pele,A.Kovac˘evic´,andN.C˘elanovic´,"Computer based emulation of power electronics hardware," in Proc. IEEE ECBS- EERC, Novi Sad, Serbia, Sep. 2009, pp. 56–64.
[18] J. G. Kassakian, "Simulating power electronics systems—A new approach," Proc. IEEE, vol. 67, no. 11, pp. 1428–1439, Oct. 1979.
[19] P. Pejovic and D. Maksimovic, "A method for fast time-domain simulation of networks with switches," IEEE Trans. Power Electron., vol. 9, no. 4, pp. 449–456, Jul. 1994.
[20] J. Allmeling and W. Hammer, "PLECS—Piece-wise linear electrical circuit simulation for Simulink," in Proc. IEEE PEDS, Hong Kong, Jul. 1999, pp. 355–360.
[21] M. O. Faruque, V. Dinavahi, and X. Wilsun, "Algorithms for the accounting of multiple switching events in digital simulation of power-electronic- systems," IEEE Trans. Power Del., vol. 20, no. 2, pp. 1157–1167, Apr. 2005.
[22] M. O. Faruque and V. Dinavahi, "Hardware-in-the-loop simulation of power electronic systems using adaptive discretization," IEEE Trans. Ind. Electron., vol. 57, no. 4, pp. 1146–1158, Apr. 2010.
[23] C. Graf, J. Maas, T. Schulte, and J. Weise-Emden, "Real-time HIL- simulation of power electronics," in Proc. 34th Annu. IECON Conf., Nov. 2008, pp. 2829–2834.

[24] M. Harakawa, H. Yamasaki, T. Nagano, S. Abourida, C. Dufour, and J. Bélanger, "Real-time simulation of a complete PMSM drive at 10 μs," presented at the International Power Electronics Conf. (IPEC), Niigata, Japan, 2005.
[25] K. D. Underwood and K. S. Hemmert, "Closing the gap: CPU and FPGA trends in sustainable floating-point BLAS performance," in Proc. 12th Annu. IEEE Symp. FCCM, 2004, pp. 219–228.
[26] G. Lienhart, D. Gembris, and R. Männer, "Perspectives for the use of field programmable gate arrays for finite element computations," presented at the COMSOL Conf., Frankfurt, Germany, 2005.
[27] K. Strunz and E. Carlson, "Nested fast and simultaneous solution for time- domain simulation of integrative power-electric and electronic systems," IEEE Trans. Power Del., vol. 22, no. 1, pp. 277–287, Jan. 2007.
[28] M. Armstrong, J. R. Marti, L. R. Linares, and P. Kundur, "Multilevel MATE for efficient simultaneous solution of control systems and nonlin- earities in the OVNI simulator," IEEE Trans. Power Syst., vol. 21, no. 3, pp. 1250–1259, Aug. 2006.
[29] W. J. Dally and B. Towles, "Route packets not wires: On-chip intercon- nection networks," in Proc. Design Automation Conf., 2001, pp. 684–689. [30] W. Lai and C.-T. Lea, "A programmable state machine architecture for packet processing," IEEE Micro, vol. 23, no. 4, pp. 32–42, Jul./Aug. 2003. [31] B. Soewito, L. Vespa, A. Mahajan, N. Weng, and H. Wang, "Self- addressable memory-based FSM: A scalable intrusion detection engine,"
IEEE Netw., vol. 23, no. 1, pp. 14–21, Jan./Feb. 2009.
[32] M. Boden, A. Gleich, S. Rulke, and U. Nageldinger, "A low-cost realiza- tion of an adaptable protocol processing unit," in Proc. 19th IEEE IPDPS,2005, vol. 4, p. 161b.
[33] M. Su, L. Xia, Y. Sun, H. Qin, and H. Xie, "Carrier modulation of four-leg matrix converter based on FPGA," in Proc. ICEMS, 2008, pp. 1247–1250. [34] L. O. Chua and P. Lin, Computer Aided Analysis of Electronics Cir- cuits: Algorithms and Computational Techniques. Englewood Cliffs, NJ: Prentice-Hall, 1975.
[35] C. He, G. Qin, M. Lu, and W. Zhao, "Group-alignment based accurate
floating-point summation on FPGAs," in Proc. ERSA, 2006, pp. 136–142. [36] M. R. Bodnar, J. R. Humphrey, P. F. Curt, J. P. Durbano, and D. W. Prather, "Floating-point accumulation circuit for matrix applica- tions," in Proc. 14th IEEE Symp. Field-Program. Custom Comput. Mach., 2006, pp. 303–304.
[37] M. de Lorimier and A. De Hon, "Floating-point sparse matrix vector multiply for FPGAs," in Proc. Int. Symp. FPGA, 2005, pp. 75–85.
[38] S. Sun and J. Zambreno, "A floating-point accumulator for FPGA-based high performance computing applications," in Proc. Int. Conf. Field-Program. Technol., 2009, pp. 493–499.
[39] A. Paidimarri, A. Cevrero, P. Brisk, and P. Ienne, "FPGA implementation of a single-precision floating-point multiply-accumulator with single- cycle accumulation," in Proc. IEEE Symp. FCCM, Napa, CA, 2009, pp. 267–270.
[40] L. Zhuo, G. R. Morris, and V. K. Prasanna, "High-performance reduction circuits using deeply pipelined operators on FPGAs," IEEE Trans. Paral- lel Distrib. Syst., vol. 18, no. 10, pp. 1377–1392, Oct. 2007.
[41] F. de Dinechin, B. Pasca, O. Cret, and R. Tudoran, "An FPGA-specific approach to floating-point accumulation and sum-of-products," in Proc. Int. Conf. ICECE Technol., FPT, Taipei, Taiwan, 2008, pp. 33–40.
[42] T. Ogita, S. M. Rump, and S. Oishi, "Accurate sum and dot product," SIAM J. Sci. Comput., vol. 26, no. 6, pp. 1955–1988, 2005.
[43] ANSI/IEEE, IEEE Standard for Binary Floating Point Arithmetic, Std. 754-1985,1985.
[44] H. C. Stanley, "An analysis of the induction motor," AIEE Trans., vol. 57, pp. 751–755, 1938.
[45] P. C. Krause, O. Wasynczuk, and S. D. Sudhoff, Analysis of Electrical Machinery and Drive Systems, 2nd ed. IEEE Press Power Engineering Series, A. Hoboken, NJ: Wiley, 2002, ser. IEEE Press Power Engineering Series, A.
[46] D. Majstorovic, I. Celanovic, N. D. Teslic, N. Celanovic, and V. A. Katic, "Ultralow-latency hardware-in-the-loop platform for rapid validation of power electronics designs," IEEE Trans. Ind. Electron., vol. 58, no. 10, pp. 4708–4716, Oct. 2011.