DEVELOPMENT OF A GENERIC DIGITAL CONTROLLER FOR POWER ELECTRONIC APPLICATIONS

CHARL R. JOOSTE

Cape Peninsula
University of Technology

DEVELOPMENT OF A GENERIC DIGITAL CONTROLLER FOR POWER ELECTRONIC APPLICATIONS

by

CHARL ROELOF JOOSTE

Thesis submitted in fulfilment of the requirements for the degree

Master of Technology: Electrical Engineering

in the Faculty of Engineering

at the Cape Peninsula University of Technology

Supervisor: Dr R.H. Wilkinson

Cape Town
April 2011

Declaration

I, Charl Roelof Jooste, declare that the contents of this thesis represent my own unaided work, and that the thesis has not previously been submitted for academic examinations towards any qualification. Furthermore, it represents my own opinions and not necessarily those of the Cape Peninsula University of Technology.

Signature: .....
C.R. Jooste

Date: 2011/04/15 .....

Abstract

Development of a Generic Digital Controller for Power Electronic Applications

C.R. Jooste

Thesis: MTech (Electrical Engineering)

April 2011

This thesis presents an investigation into the generic tools, hardware and firmware, involved in power electronic converter control and feedback. The aim was to determine the optimal controller architecture through research of existing controllers. As soon as the architecture was established, design of the controller commenced. Explanations for the various components selected were provided. The design considerations when designing a printed circuit board (PCB) with mixed signals was also presented. The theory behind the control of a multicell converter as well as the practical implementation of the control scheme in firmware was presented.

Acknowledgements

I would like to express my sincere gratitude to the following people and organisations:

My family and friends for their continued support and motivation.

My supervisor, Dr Richardt Wilkinson, for his guidance and support.

The Centre for Instrumentation Research (CIR) staff, store members and students.

Special thanks to my colleagues Jason Quibell and Rory Pentz.

The financial assistance of the National Research Foundation, Cape Peninsula University of Technology and F'SATI towards this research is acknowledged. Opinions expressed in this thesis and the conclusions arrived at, are those of the author, and are not necessarily to be attributed to the respective mentioned establishments.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

ac	Alternating Current
ADC	Analog-to-Digital Converter
AIC	Analog Interface Controller
ASIC	Application-Specific IC
CAD	Computer Aided Design
CAS	Column Address Strobe
CCS	Code Composer Studio
CIR	Centre for Instrumentation Research
CIRPEC	CIR Power Electronics Controller
CMOS	Complementary Metal Oxide Semiconductor
CSV	Comma Separated Values
CPES	Centre for Power Electronics Systems
CPU	Central Processing Unit
DAC	Digital-to-Analog Converter
dc	Direct Current
DMA	Direct Memory Access
dMAX	Dual Data Movement Accelerator
DP	Double-Precision
DPWM	Digital Pulse-Width Modulator
DSP	Digital Signal Processor
EEPROM	Electrically Erasable Programmable ROM
EMI	Electromagnetic Interference
EMIF	External Memory Interface

ENOB	Effective Number of Bits
ER	Effective Resolution
EVM	Evaluation Module
FFT	Fast Fourier Transform
FIFO	First In First Out
FPGA	Field Programmable Gate Array
FSR	Full-Scale Range
GPIO	General Purpose Input Output
GTO	Gate-Turn-Off Thyristor
I^2C/I2C	Inter-Integrated Circuit
IC	Integrated Circuit
ICM	Imbricated Cells Multilevel
IDE	Integrated Development Environment
IGBT	Insulated Gate Bipolar Transistor
IP	Intellectual Property
JTAG	Joint Test Action Group
LDO	Low Dropout
LVCMOS	Low-Voltage CMOS
LVDS	Low-Voltage Differential Signalling
LVTTL	Low-Voltage Transistor-Transistor Logic
McASP	Multichannel Audio Serial Port
MFLOPS	Million Floating Operations Per Second
MIPS	Million Instructions Per Second
MMACS	Million Multiply-Accumulates Per Second
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
PCB	Printed Circuit Board
PEC	Power Electronics Controller
PEG	Power Electronics Group
PLD	Programmable Logic Device
PLL	Phase-Lock Loop
PQFP	Plastic Quad Flat Pack

PWM	Pulse-Width Modulation
RAM	Random Access Memory
RAS	Row Address Strobe
ROM	Read Only Memory
RSDS	Reduced Swing Differential Signalling
S/H	Sample-and-Hold
SAR	Successive Approximation Register
SDRAM	Synchronous Dynamic RAM
SIMO	Slave In Master Out
SINAD	Signal-to-(Noise + Distortion)
SNR	Signal-to-Noise Ratio
SOIC	Small Outline IC
SOMI	Slave Out Master In
SP	Single-Precision
SPI	Serial Peripheral Interface
SQNR	Signal-to-Quantization-Noise Ratio
SRAM	Static Random Access Memory
SSTL	Stub Series Terminated Logic
THD	Total Harmonic Distortion
TI	Texas Instruments
TQFP	Thin Quad Flat Pack
TSOP	Thin Small Outline Package
TSSOP	Thin-Shrink Small Outline Package
TTL	Transistor-Transistor Logic
USB	Universal Serial Bus

Constants

$\pi =$	3.141 592 654

Symbols

C_f	Filter capacitor

d	Duty cycle
d_c	Duty-cycle command
e_q	Quantization error
f_{clk}	FPGA clock frequency
f_r	Reference frequency
f_s	Switching frequency
F_s	Sampling frequency
i_L	Current through the inductor
L_f	Filter inductor
s	Switching function
s_t	Total switching function
t_{conv}	Conversion time
T_s	Sampling period
V_{C_i}	i^{th} cell capacitor voltage
V_{dc}	dc voltage
V_o	Voltage out
V_{ref}	Reference voltage
V_s	Filtered output voltage
Z_{load}	Load impedance

Chapter 1

Introduction

1.1 Introduction to Digital Control in Power Electronics

Power electronics can be said to be the processing of electrical power using electronic devices, where emphasis is placed on the control of energy flow (Krein, 1998). It also deals with the efficient processing and control of electric power in applications ranging from sub-watt dc-dc power management devices in battery-operated portable equipment to tens, hundreds and thousands of watts for computer and office power supplies, to kilowatts and megawatts for variable-speed drives and 1,000 megawatt inverters and rectifiers used in the utility power system.

An important element of power electronics and more specifically digitally controlled power electronics is the *switching converter*. A switching converter is a power electronics circuit consisting of semiconductor 'switches' which convert an input voltage to a voltage of another level. Basic functions that are implemented by switching converters are included in Table 1.1: (Erickson & Maksimovic, 2004; Mohan *et al.*, 2003).

Figure 1.1 is an illustration of a basic switching converter block. It consists of input power and output power ports as well as a control input port. The raw input power is processed by the control input specifications to produce the required conditioned output power.

Figure 1.2 illustrates the inclusion of a controller block into the basic power processing block of a switching power converter. Control is necessary to pro-

Table 1.1: Basic functions of switching converters.

	Basic functions of switching converters
dc-dc conversion	Direct current (dc) input voltage is converted to dc output voltage possibly of smaller or larger magnitude and potentially opposite polarity or reference isolation between input and output grounds
ac-dc rectifier	Alternating current (ac) input voltage is rectified to give a dc output voltage
dc-ac inversion	Transformation of a dc input voltage into an ac output voltage of controllable magnitude and frequency
ac-ac cyclocon-version	Converts an input ac voltage into an ac output voltage of controllable magnitude and frequency

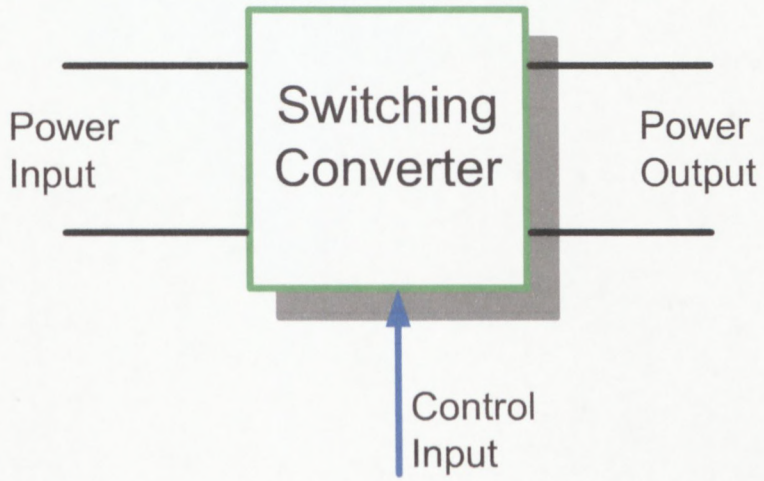


Figure 1.1: A basic block diagram of a switching power converter.

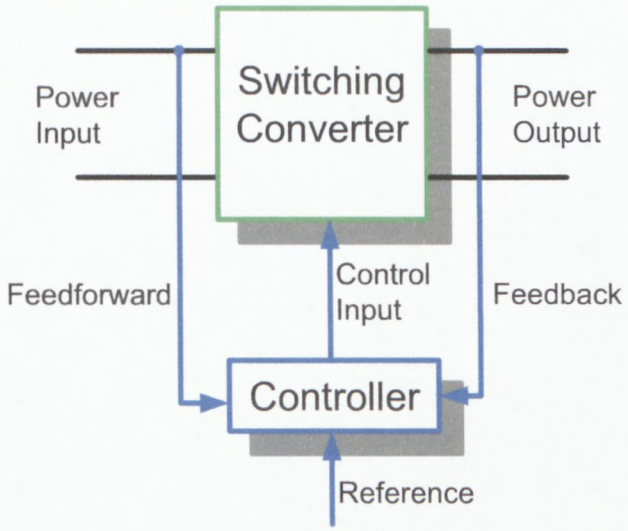


Figure 1.2: Inclusion of the controller block.

vide a well conditioned and well regulated output voltage where there may be variations in the input voltage or load current.

Switching converters are being implemented in more branches of power electronics as these high-frequency switching converters can transfer energy in high power density and efficiency. They operate in the discrete-time domain since the energy and power delivery from the source to the load are no longer in the continuous-time mode. Digital control methods therefore need to be applied. (Luo *et al.*, 2005)

Digital control of switching converters is becoming very common in high-power low-frequency applications as well as in low-to-medium power high-frequency applications (Prodic *et al.*, 2003).

High efficiency is desirable in power electronic systems. The implications of converter design with low efficiency, especially in high-power applications, is evident from the subsequent equations. The power-transfer efficiency (η) of a converter can be derived from the following equation: (Erickson & Maksimovic, 2004; Luo *et al.*, 2005)

$$\eta = \frac{P_{out}}{P_{in}} \tag{1.1.1}$$

where P_{out} is the *output power* and P_{in} the *input power*.

The power loss in a converter can be calculated by equation (1.1.2) and is plotted in Figure 1.3:

$$P_{loss} = P_{in} - P_{out} = P_{out} \cdot \left(\frac{1}{\eta} - 1\right) \tag{1.1.2}$$

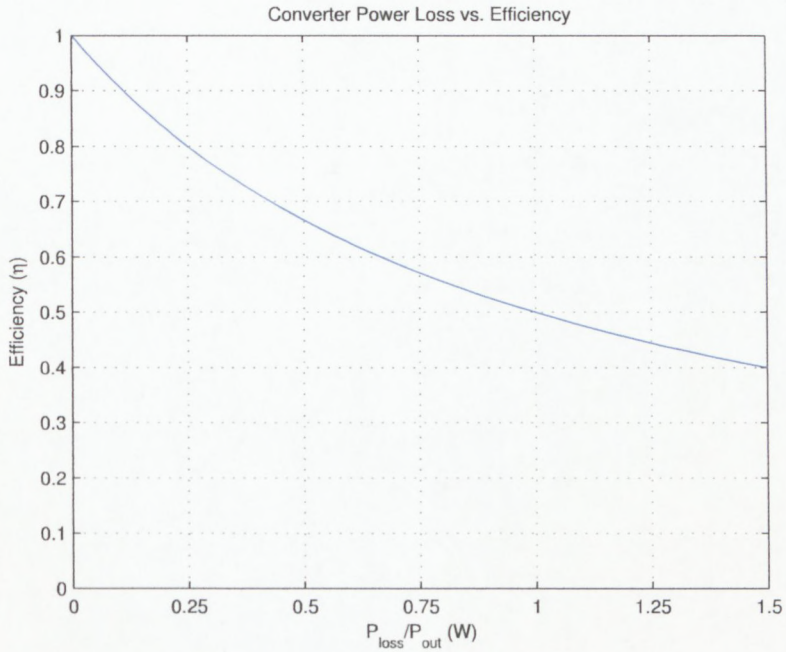


Figure 1.3: Converter power loss vs. efficiency. (Erickson & Maksimovic, 2004)

From equations (1.1.1) and (1.1.2) and analysing Figure 1.3 it can be determined that a converter with an efficiency of 50% will have a P_{loss} equal to that of its P_{out}. The power that is lost is inevitably converted into heat which may require large and expensive cooling methods depending on the output power and loss. Operating a converter at high temperatures will lead to instability and reliability issues.

However, if the converter is 90% efficient then the loss power is only 11% of the output power as calculated in equation (1.1.3); while considering an η of one represents 100% efficiency.

$$P_{loss} = P_{out} \cdot \left(\frac{1}{\eta} - 1\right) = P_{out} \cdot \left(\frac{1}{0.9} - 1\right)$$
$$\frac{P_{loss}}{P_{out}} = 0.1\dot{1}$$
$$\therefore \frac{P_{loss}}{P_{out}} = 11.\dot{1}\% \tag{1.1.3}$$

Low power loss enables the converter components to be more densely packed allowing for a smaller and lighter converter with minimal temperature rise.

In order to achieve this efficiency, switched-mode semiconductor devices are operated as 'switches'. These power semiconductor devices are listed in Table 1.2 (Luo *et al.*, 2005; Mohan *et al.*, 2003).

Table 1.2: Types of switched-mode semiconductor devices.

	Device	Power Capability	Switching Speed
GTOs	Gate-Turn-Off Thyristors	High	Slow
BJTs	Bipolar Junction Transistors	Medium	Medium
IGBTs	Insulated Gate Bipolar Transistors	Medium	Medium
MCTs	MOS-Controlled Thyristors	Medium	Medium
MOSFETs	MOS Field Effect Transistors	Low	Fast

By actively controlling the on/off states of these 'switches' it is possible to regulate the input or output. When the semiconductor operates in the 'off' state, the current through it is zero and so is the power dissipation. When operating in the 'on' (saturated) state, its voltage drop is small and so is its power dissipation. The power dissipated by these devices is therefore very low and well suited for high-efficiency power converters.

Examples of power converters where these switching devices are used include buck, boost, buck-boost, half-bridge, full-bridge and multilevel topologies including the multicell converter.

The majority of this introductory section research was sourced from the references of Erickson & Maksimovic (2004); Maksimovic *et al.* (2004)

1.2 Project Background

In the last few years much research in the area of multicell converters has been conducted at the Cape Peninsula University of Technology's Centre for Instrumentation Research (CIR) laboratory. It was established that there was a need for a digital controller to provide the necessary control signals.

The focus of this thesis is the design and development of a generic digital controller for power electronic applications. The controller will provide engineers and students with a hardware platform to implement power electronic applications without having to design their own application specific board, as this can be a time consuming exercise. A few firmware modules have also been developed that will greatly reduce the time taken to produce working application specific code.

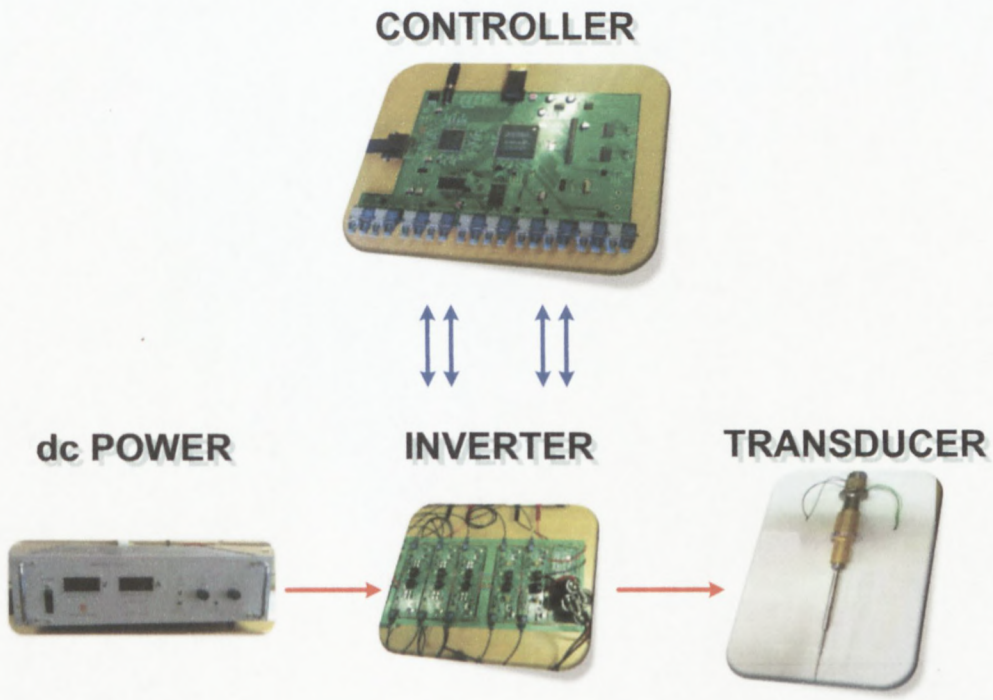


Figure 1.4: System overview diagram.

Figure 1.2 illustrates the system overview for the primary application. A dc voltage source is converted to ac by means of a multicell inverter. The reference to an inverter specifically implies that the converter will be used for

dc to ac inversion. The terms *converter* and *inverter* may therefore be used interchangeably in this text depending on the context of its appearance.

A controller is required to generate the digital pulse-width modulation (PWM) control signals. From there the stepped sinusoidal output signal from the inverter is filtered to produce a conditioned sinusoidal output voltage. The conditioned sinusoidal output signal is used to drive an ultrasonic transducer load. Applications for the mentioned system include, amongst others, ultrasonic welding and drilling.

The focus of this research is purely on the controller. It encompasses the controller design and implementation and also the firmware development involved in the generation of the PWM signals that are necessary for control of the multicell inverter.

1.2.1 Multicell Topology

Although the purpose of this thesis is not the development of a multicell converter, understanding is required of the operation of this topology since the primary application is the control thereof. An overview of the topology is therefore covered in this section. The control signals required and their implementation will be covered in Chapter 4.

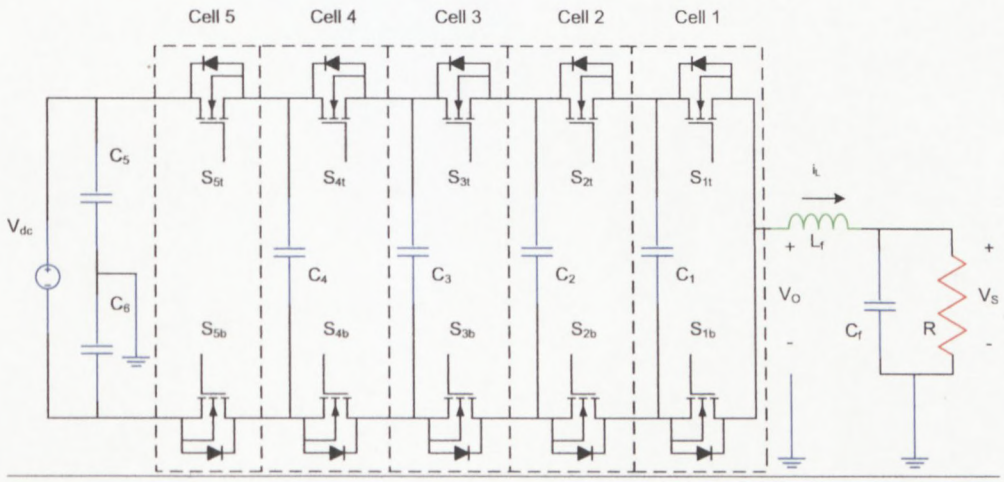


Figure 1.5: The multicell inverter topology.

The multicell converter topology is also referred to as the "flying-capacitor multilevel topology", "nested cells multilevel topology" or "imbricated cells

multilevel topology" (ICM) (Meynard & Foch, 1995; Wilkinson, 1997; Walker, 1999; Wilkinson, 2004; Hansmann, 2004). The multicell topology can be used for many applications including dc-dc, dc-ac and also ac-ac converters for unidirectional and bidirectional power flow (Walker, 1999). To re-emphasise, the multicell converter will often be referred to as an inverter in this thesis implying it will be used for dc-ac inversion.

In this topology semiconductor switches are connected in series and the top and bottom switches are connected by floating capacitors. The floating capacitors act as dc voltage sources which need to be charged to a particular voltage in order for the switching devices not to be destroyed (Wilkinson, 2004). This topology can be seen as multiple generic cells made up of a half-bridge configuration and a capacitor across the input bus. Figure 1.5 illustrates a *five*-cell multicell inverter with each cell being demarcated.

A single cell consists of two switching devices forming what is called a complementary pair. When the top switch is conducting, the bottom switch must be blocking. If both of the switching devices are conducting at the same time it will cause a short circuit which could have destructive implications. To ensure that only one switching device in a complementary pair is conducting at any one instant, a period of dead time is inserted. Dead time allows the conducting switch to enter the 'off' state before the other switch begins conducting. Although the rise and fall times of a switching device may differ, it is prudent to allow the switch that is transitioning from the 'on' state to the 'off' state to switch off completely before allowing the other device to start conducting. This is due to the fact that the conducting device takes a period of time, T_f, before it stops conducting. Typically the rise time, T_r, is shorter than the fall time, T_f, and if switched simultaneously the one switch may still be transitioning to the 'off' state while the other switch starts conducting.

The ideal voltage across a cell capacitor under balanced conditions can be expressed as follows: (Hamma *et al.*, 1995; Meynard *et al.*, 1997)

$$V_{C_i} = \frac{i \cdot V_{dc}}{p} \quad \text{for } i = 1, \ldots, p \tag{1.2.1}$$

Where V_{C_i} represents the i^{th} cell capacitor voltage, V_{dc} represents the input voltage and i the index for the number of cells up to the p^{th} cell.

Commutation cells have a voltage-drop across the switching devices of $\frac{V_{dc}}{p}$ where p represents the number of cells. The cells are overlapped to form an

inverter leg capable of switching the input voltage V_{dc}. The number of cells *i.e.* p can be increased depending on the blocking voltage required. The output voltage ripple has an amplitude of $\frac{V_{dc}}{p}$. The apparent switching frequency is p times the switching frequency, f_s, which results in a reduction in the output filter size by a factor of p^2. (Meynard & Foch, 1995; Molepo, 2003; Wilkinson, 2004)

The multicell converter relies on the phase-shift and duty cycle of the control signals in order to operate correctly. The phase-shift between successive control signals needs to be $\frac{2\pi}{p}$. This also results in the cancellation of harmonics centred at multiples of f_s and up to and including $(p-1) \cdot f_s$ (Wilkinson, 2004).

The phase-shift between control signals of successive cells can be expressed as follows: (Wilkinson, 2004)

$$\phi = \frac{2\pi}{p} \tag{1.2.2}$$

and the phase-shift of the control signal for a single cell can be expressed as follows: (Wilkinson, 2004)

$$\phi_i = (i-1) \cdot \phi \text{ for } i = 1, \ldots, p \tag{1.2.3}$$

Where i is an index for the cells up to the p^{th} cell.

1.3 Project Objectives

The objectives of the project **include**:

- An investigation into the generic tools, hardware and firmware, involved in power electronic control and feedback is required to determine the optimal controller architecture necessary for a generic power electronics controller.

  A thorough investigation of controllers designed at other institutions will be performed in order to determine the applications the hardware and firmware may have been designed for and from that establish a generic controller architecture.

- From this investigation, the trade-offs that exist as a result of the design of a generic controller rather than a controller designed for a specific

power electronics application will be established. These trade-offs may include time to development and cost factors.

- The design and development of a generic high-performance controller for various power electronic applications.

  In order to establish whether the hardware functions as desired, it will be used to control a multicell inverter. Development of the multicell inverter falls outside of the scope of this project, therefore, the controller will be tested on a multicell inverter developed in-house.

- The development of various firmware modules to be implemented on the controller, this will include a multicell inverter module as the primary application. Other modules may include a half-bridge, full-bridge and three-phase motor control module.

  In order to determine whether the firmware functions as desired it will be implemented on the hardware controller mentioned in the previous point. Since the multicell inverter module is an advanced form of the half- and full-bridge modules, as presented in Chapter 4, only the multicell inverter module will be tested with a multicell inverter. If the multicell module functions as desired it is safe to assume the half- and full-bridge modules will function as desired.

The project objectives **do not include**:

- The development of a multicell inverter.
- Feedback from the multicell inverter, although this was incorporated into the design of the hardware and a basic firmware implementation was carried out.

1.4 Thesis Outline

- **Chapter 2** focuses on researching the controller architecture and determining the key components that are necessary for control.
- **Chapter 3** focuses on the controller design. The controller specifications, and the printed circuit board (PCB) design and considerations are presented.

- **Chapter 4** focuses on the firmware development. The various modules as well as the interleaved switching method are presented.
- **Chapter 5** focuses on the simulation and results obtained from the hardware and firmware.
- The thesis is concluded in **Chapter 6**.

Chapter 2

Digital Control Systems

2.1 Introduction

Digital controllers and digital control methods in power electronics have become widespread. They are especially useful in applications requiring complex control and monitoring such as motor drives and three-phase power converters. Digital controllers may include processors such as the microprocessor, programmable logic devices (PLD), field programmable gate arrays (FPGA) and digital signal processors (DSP). Technical advantages coupled with increasing processing power and decreasing cost have made digital control popular in power electronics at relatively high power levels (Maksimovic *et al.*, 2004).

2.2 Digital Control

In order to establish the components necessary for a digital controller an investigation into the control of switching converters is required. The objective of control is to operate the controlled system safely and optimally (Kaye & Smith, 1989). This is done by regulating the output of the converter by sensing at the output and compensating the signal so that it is within an error margin of the desired output. Control is necessary to achieve a desirable output voltage despite disturbances which could be introduced in the load, input voltage or converter circuit element values (Erickson & Maksimovic, 2004).

The basic control process block of a switching converter, illustrated in Figure 2.1, is presented again for quick reference. A controller is needed to

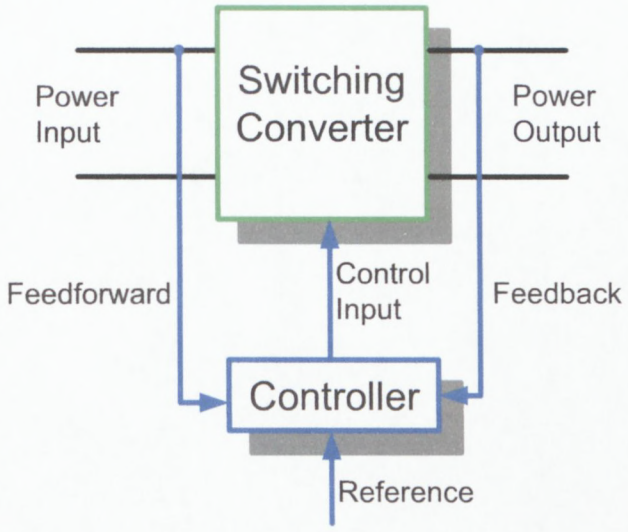


Figure 2.1: Switching converter and controller block.

provide the control input signals to the switching converter as well as process any feedback or feed-forward and make adjustments to the control signals.

An example of a digital controller block is illustrated in Figure 2.2. It consists of an analog-to-digital converter (ADC), a discrete-time compensator and a digital PWM gate signal generator.

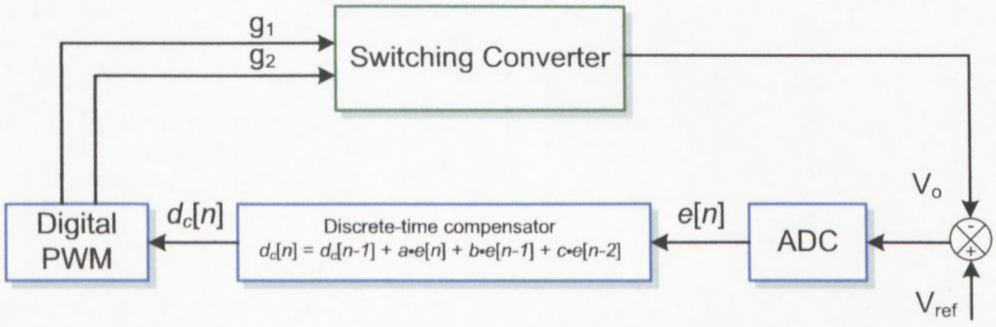


Figure 2.2: Digital PWM controller. (Maksimovic *et al.*, 2004)

An ADC samples the difference between the sensed output voltage, V_o, and the reference voltage, V_{ref}, providing the digital error signal, e. The duty-cycle

command, d_c, is then determined by a discrete-time compensator by applying the received digital value from the ADC and computing that with the control algorithm on the DSP. A digital pulse-width modulator (DPWM) is used to generate the output gating signals at the desired switching frequency as well as duty cycle as determined by d_c. The control input signals can either be converted back into analog signals using a digital-to-analog converter (DAC) or the digital signals can be used for the gating signals, g_1 and g_2, of the switching devices. In this system the DPWM serves as the DAC. A high-frequency DPWM can be constructed using a fast clocked counter and a digital comparator. (Maksimovic *et al.*, 2004)

In order to achieve fast dynamic response the ADC must sample at least at the same rate as the switching frequency. The ADC resolution also needs to be high enough in order to achieve good voltage regulation requirements. (Maksimovic *et al.*, 2004)

The discrete-time compensation function of

$$d_c[n] = d_c[n-1] + a \cdot e[n] + b \cdot e[n-1] + c \cdot e[n-2] \tag{2.2.1}$$

that runs on the DSP is that of a digital filter. $e[n]$, $e[n-1]$, $e[n-2]$ represent consecutive samples of e. The coefficients a, b, and c are what determine the compensator's frequency response. This computation will need to be computed in a fraction of the switching frequency such that d_c can be updated without much delay. For this filter it can be seen that there are three multiplications and 3 additions. DSPs are capable of performing such calculations, however the greater the processing needs the greater the cost and complexity of the design. (Maksimovic *et al.*, 2004)

2.3 Key Components of a Digital Controller

From the general controller block illustrated in Figure 2.2 it was determined that there are three key components that define the general architecture of a digital controller. These include the ADC, the DSP and lastly the DAC. In this section a brief overview of these components is given.

Figure 2.3: Block diagram of the analog-to-digital conversion process. (Luo *et al.*, 2005)

2.3.1 The Analog-to-Digital Converter

An ADC is necessary to convert real-world signals from the analog (continuous-time) domain to the digital (discrete-time) domain ready for processing by a digital controller. Important factors when selecting an ADC include conversion time t_{conv} and quantization error e_q. Figure 2.3.1 shows the conversion process from analog to digital.

Firstly, the analog signal is sampled. Sampling occurs at regular time intervals also known as the sampling period T_s. The sample-and-hold (S/H) process samples the analog signal at each interval and holds that value until the next sample interval arrives. A phenomenon known as aliasing can occur if the analog signal is not sampled fast enough. This is a result of important information being lost or not sampled due to the slow sampling rate. The sampling frequency, F_s, should be selected to be at least twice the frequency of the highest frequency component of the analog signal to be able to exactly recover the analog signal from the sample values. This is referred to as the *Shannon sampling theorem* or the *Nyquist rate* (Proakis & Manolakis, 1996:29) (Tan, 2008:19).

Quantization and digitization of the signal takes place next. Once the signal has been acquired by the sampler and held, the converter assigns a quantization level that approximates the signal as closely as possible in the form of a binary code. Where N bits are used, 2^N possibilities can be represented by the converter. The more bits, the more closely the digital signal will correspond to the analog signal. The gap between the levels is known as the *quantization step.*

The quantization step or resolution of an ADC is defined as:

$$\Delta = \frac{R}{2^N} \tag{2.3.1}$$

where R represents the *full-scale range* (FSR) of the converter and N is

the number of converter bits. (Bester, 1999; Van Heerden, 2003; Pease, 2008; Luo *et al.*, 2005)

A quantization error (e_q) can occur due to the limited quantization levels the converter has available to select from, meaning the signal is either truncated or rounded. The quantization error is the difference between the quantized value and the actual sampled value. (Luo *et al.*, 2005)

$$e_q = \text{quantized value - actual value} \tag{2.3.2}$$

The error will always be in the range

$$-\frac{\Delta}{2} < e_q < \frac{\Delta}{2} \tag{2.3.3}$$

(Bester, 1999; Van Heerden, 2003). From equation (2.3.3) it is evident that with every added bit, the quantization error will be halved.

When selecting an ADC, ac domain specifications such as effective resolution (ER), signal-to-(noise + distortion) or SINAD and effective number of bits (ENOB) will provide a measure of how repeatable the ADC conversion will be. Dc domain specifications such as offset error, gain error, differential nonlinearity and integral nonlinearity will provide a measure of how accurately the input signal will be matched to the output code. Noise can however result in varying results. (Pease, 2008)

SQNR (signal-to-quantization-noise ratio) is a ratio of the signal power to noise power of a signal. The quality of the output of the ADC can be determined by this value. The SQNR of a converter in decibel is ideally defined as:

$$6.02 \cdot n \; + \; 1.76 \; dB \tag{2.3.4}$$

where n is equal to the number of bits in the converter. This noise is as a result of the quantization noise within the converter. (Pease, 2008)

The SQNR must be large to prevent unwanted quantization noise. The various types of ADCs and their characteristics are presented in Table 2.1.

2.3.2 The Digital Signal Processor

A DSP is necessary to process and execute control algorithms and make adjustments to the control algorithm as determined by the feedback or feed-forward.

Table 2.1: Types of ADCs (Luo *et al.*, 2005).

ADC type	Characteristics	Examples	Resolution	Sample rate
Serial	Convert 1 LSB at a time, starting from MSB 1 Bit per clock	Single slope, dual slope, successive approximation (SAR), delta-sigma ($\Delta\Sigma$)	Best	Slowest
Flash	Conversion period of one clock cycle	Flash	Medium	Medium
Subranging	Combination of serial and parallel techniques	Two-step, multistep, pipeline	Lowest	Fastest

DSPs are well suited as high-performance controllers. They have been designed for real-time digital processing and can handle complex control algorithms. Their architecture has specifically been optimised for digital signal processing tasks allowing them to perform multiple instructions per clock-cycle. However, they do not generally contain ADCs that are necessary for closed-loop control of power converters (du Toit *et al.*, 1995).

The way DSPs perform arithmetic can be classified as either floating- or fixed-point. Floating-point offers a wider dynamic range and increased accuracy. It is much simpler to implement control algorithms in floating-point arithmetic than it is to perform in fixed-point; this is due to the fact that fixed-point requires careful monitoring of overflow and underflow errors by the scaling of variables. Floating-point processors tend to be more expensive due to their increased complexity although development costs can probably be saved thanks to their ease of programming. (Karipidis, 2001)

The decision whether to use a floating-point or fixed-point DSP was formerly based on cost and ease of use. This however is not as significant anymore. What defines the decision is rather based on the data set required. If the data set requires greater mathematical flexibility and accuracy a floating-point format is required. Floating-point DSPs offer real arithmetic, higher precision and a wider dynamic range than fixed-point. (Frantz & Simar, 2004)

The fixed- and floating-point terms refer to the way in which the devices

represent their numeric data. A fixed-point DSP will perform integer arithmetic, while floating-point DSPs support either integer or real arithmetic. (Frantz & Simar, 2004)

2.3.3 The Digital-to-Analog Converter

The structure of a digital-to-analog converter (DAC) is simple in comparison with an ADC. It can be divided into two components as seen in Figure 2.3.3. The digital signal gets converted into an analog signal of magnitude corresponding to the digital code. The decoder is used to convert the digital word into a number of an amplitude-modulated pulse. The S/H maintains the voltage for the duration of the sampling period. (Luo *et al.*, 2005)

Figure 2.4: Block diagram of a digital-to-analog converter. (Luo *et al.*, 2005)

The number of bits determines the accuracy of the DAC. DACs are considered simpler than ADCs since they need only decode the digital input signal, there is little degradation to the output signal accuracy. The only degradation that may occur is if the processor's data word is greater than that of the DAC in which case the signal would have to be compensated in order to be the correct width. (Van Heerden, 2003)

DACs can be classified into serial and parallel devices. A serial device typically requires N or $2^N - 1$ clock cycles to output for an N bit converter. A parallel device typically requires one clock cycle. (Luo *et al.*, 2005)

2.4 Research in the Area of Digital Controllers

Tables 2.2 and 2.3 list a comparison of controllers developed at various institutions for the control of power electronic converters. These controllers were used to determine the specifications for the CIR power electronics controller (CIRPEC), which was designed by the author.

Table 2.2: Comparison of current digital controllers. Table adapted from Van Heerden (2003).

Comparison of current digital controllers		
Name	PEC33	UC
Manufacturer	University of Stellenbosch (PEG)	Virginia Polytechnic (CPES)
DSP	TMS320VC33-150	ADSP-21160M
Precision Fixed/Floating point	32-bit Floating	32-bit Floating
MFLOPS	150	600
MIPS	75	-
Data bus width	32-bit	64-bit
Serial ports	1×SPI	2×SPI, 6×link port
ADC channels	-	-
PWM channels	-	-
PLDs		
FPGAs	2×EP1K50	1×XCV400-4BG560C
EPLDs	2×EPM7256B	-
Memory **DSP (internal)**		
RAM	32 kB SRAM	4 Mb
ROM/FLASH/EEPROM	-	-
DSP (External)		
RAM (SRAM/SDRAM)	-	-
FLASH/EEPROM	512 k (8-bit)	EEPROM (Unknown)
Communication		
RS-232/SPI/I^2C/USB	RS-232/USB	PCI via mezzanine card
Emulator (DSP/FPGA)	JTAG	JTAG
Fibre optics	Ext. 18×Tx, 18×Rx. Local 2×Tx, 2×Rx	4 (2×Tx, 2×Rx)
Analog inputs	32 (10-bit)	ext.
Analog outputs	8 (12-bit)	On-board unknown
PWM outputs	2×2×4 + 2 (18)	ext.
Power supply	5 V, 2 A (max)	5 V
Expansion header	Yes	Yes

Table 2.3: Comparison of current digital controllers continued.

Comparison of current digital controllers continued		
Name	MU-DSP240-LPI	ISEADSP
Manufacturer	Monash (PEG)	RWTH Aachen (ISEA)
DSP	TMS320F240	ADSP-21060/1/2 (max 2)
Precision	16-bit	32-bit
Fixed/Floating point	Fixed	Floating
MFLOPS	-	80 (120 peak)
MIPS	20	40
Data bus width	16-bit	48-bit
Serial ports	1×SPI, 1×RS422/RS485/ RS232	2×RS232, RS485/RS422, VMEbus, 4×link port, 1×TDM serial port
ADC channels	16 (10-bit)	-
PWM channels	12	-
PLDs		
FPGAs	-	6×Orca 2CA
EPLDs	-	-
Memory **DSP (internal)**		
RAM	544 (16-bit)	2 Mb - 4 Mb
ROM/FLASH/EEPROM	16 kB (16-bit)	-
DSP (External)		
RAM (SRAM/SDRAM)	128 k (16-bit)	32 k - 256 k (48-bit)
FLASH/EEPROM	64 k (16-bit)	512 k - 4 MB (Flash) 16 kB (EEPROM)
Communication		
RS-232/SPI/I^2C/USB	RS232	2×RS232, RS485/RS422, VMEbus
Emulator (DSP/FPGA)	JTAG	JTAG
Fibre-optics	-	-
Analog inputs	10 (10-bit)	64 (12/14-bit) I/O
Analog outputs	-	64 (16-bit) I/O
PWM outputs	2×4	32 single
Power supply	±15 V, +5 V	-
Expansion header	No	Yes

The comparison includes the PEC33 controller developed by Van Heerden (2003) at Stellenbosch University. It succeeded the PEC31 controller presented in Bester *et al.* (1998); Bester (1999). The PEC33 has a more powerful DSP and faster ADC converters in comparison to its predecessor. The PEC33 implements a modular design through expansion headers with all of its 18 PWM outputs being sent via a fibre optic expansion board.

Modular designs have advantages including the ability to have separate boards for separate functions that can be removed or added from or to a system whenever necessary. It also serves the purpose of ease of future expansion and upgrading; when a section of the hardware becomes obsolete or outdated only the expansion boards are affected thereby not having to redevelop the entire PCB's contents.

However, there are issues inherent to a modular design. By severing the connection between boards, via cable or a header, signal integrity is compromised and unwanted noise is introduced into the system. Track lengths between devices on a single PCB tend to be shorter and there is the added advantage of a potential solid ground and power plane to further reduce noise. The author of this thesis has attempted to place as much as is necessary on the controller PCB while at the same time providing modular functionality through the use of expansion headers.

The PEC33 controller makes use of two transmit and receive fibre optic devices on board as communication devices, most likely for the distributed control of power electronics as presented in du Toit *et al.* (1998).

The next controller is the Universal Controller (UC) presented in Francis & Boroyevich (2001), Francis (2004) and Francis *et al.* (2005) developed at the Centre for Power Electronics Systems (CPES) at the Virginia Polytechnic Institute and State University. This particular controller succeeded the previous UC developed by Celanovic (2000); Celanovic *et al.* (2000) which was built around an Analog Devices 21062 SHARC DSP EZ-LAB development system. Improvements included a more powerful DSP in the form of an ADSP-21160. The ADSP-21160 has the ability to be connected in parallel with six more of the same DSPs, allowing for parallel processing. However, the controller only consists of one DSP and one FPGA. The ADSP-21160 also comes with a large amount of on-board static random access memory (SRAM) (4 Mb). As a result of the UCs architecture being specifically designed as an application

manager (main controller) in a distributed control environment it does not have any on-board ADCs and features two transmit and receive fibre optics.

The next controller in the table is the MU-DSP240-LPI developed by the Power Electronics Group at Monash University of Australia (Holmes, 2000). It is a free-standing single-board design based around the digital motor/motion control DSP from Texas Instruments, the TMS320F240. The only device external to the DSP is the addition of an electrically erasable programmable read only memory (EEPROM). The MU-DSP240-LPI board was designed to control three-phase inverters. However, it does not make provision for I/O expansion. This board is an example of a design relying purely on a dedicated motion control DSP.

The final controller in the table is the ISEADSP board developed by Karipidis (2001). The ISEADSP forms the basic control hardware for various research projects at the Institute for Power Electronics and Electrical Drives (ISEA) at the Aachen University of Technology. Karipidis proposed a versatile digital architecture/structure for the rapid prototyping of control units and real-time simulation of loads for power electronic and electrical drive systems. A single board was proposed as the basis for a powerful and flexible, modular hardware structure. It features a DSP and FPGA design. All I/O interfaces are implemented through the FPGAs of which there are six. The ISEADSP boards can be stacked up to five boards deep allowing for a total of ten processors, 320 analog and 320 digital inputs or outputs. A VMEbus rack can include 20 ISEADSP boards. In this structure a single ISEADSP board can feature up to two DSPs. The DSPs selected were from the Analog Devices range of SHARC ADSP-2106× devices. These DSPs are capable of 120 million floating operations per second (MFLOPS) peak and 80 MFLOPS sustained operation.

Karipidis established that it was advantageous to complement a high-performance DSP with a device such as an FPGA as this would offer greater flexibility for I/O related tasks due to the reconfigurable nature of the FPGA devices. Karipidis also established that the DSP would at least need to be capable of a sustained performance of 40-80 MFLOPS in order to implement control and simulation algorithms. The ISEADSP does not make provision for any fibre optic connectors on the controller board itself.

A modular application of a digital controller was investigated at the North

Table 2.4: Features of the EVMs used by Liu (2005).

EVM (central controller)	TMS320C6701
Manufacturer	Texas Instruments
Fixed/Floating point DSP	Floating
Precision	32-bit
Clock	120/150/167 MHz
MFLOPS	1000
MIPS	-
SRAM	1 Mb
External data bus	32-bit
Interface	JTAG
EVM (daughterboard)	AED-106
Manufacturer	Signalware
FPGA	Xilinx XCV300
Analog inputs	4×16-channel (12-bit)
Sample rate	6 MSPS

Carolina State University for distributed power electronic systems by Liu (2005); Liu *et al.* (2005). A new architecture for modular distributed control was proposed by Liu (2005). It is referred to as a hybrid multitap-inverse star structure in the literature. The central controller transmits signals to the local controllers at each phase-leg using a fibre optic multitap bus interface. Using an asynchronous serial communication interface each local controller receives the same information from the central controller and interprets the information accordingly. Return signals are sent from each individual local controller straight back to the central controller via a fibre optic interface.

A development board, the TMS320C6701 evaluation module (EVM), designed by Texas Instruments was used as the central controller. It consists of a TMS320C6701 floating-point DSP. This board was then complemented by an FPGA daughterboard, the AED-106 developed by SIGNALWARE. The FPGA present on this board is the XCV300 from Xilinx. Some of the features of the two boards are listed in Table 2.4.

Another instance where an EVM was used as the digital controller is by the Colorado Power Electronics Center at the University of Colorado at Boulder. Prodic *et al.* (2001) investigated a digital controller for a high-frequency switching power supply. The ADMC401 evaluation module developed by Analog Devices (2000) was used as the controller for a test bed to validate a de-

sign targeted at a low cost standalone application-specific integrated circuit (ASIC). Some of the features are presented in Table 2.5. The DSP core of the ADMC401 is the ADSP-2171. It is a 26 million instructions per second (MIPS) 16-bit fixed-point DSP intended for motor control applications. The ADMC401 features an eight channel 12-bit ADC. An external DPWM chip was used as the built-in PWM units on the ADMC401 could only offer 2-bit resolution at the desired switching frequency of 1 MHz (Prodic *et al.*, 2001).

Table 2.5: Features of the EVM used by Prodic *et al.* (2001).

EVM	**ADMC401**
Manufacturer	Analog Devices
DSP	ADSP-2171
MIPS	26
Fixed/Floating	Fixed
Precision	16-bit
Analog inputs	8-channel (12-bit)

In a Journal paper published by Mohan *&* Ang (1994), a TMS320C30 floating-point DSP EVM by Texas Instruments was used to control a Ćuk converter. This DSP is capable of 33.3 MFLOPS and 16.7 MIPS. The DSP performed the necessary signal processing algorithms and generated the PWM signals. The DSP was interfaced to a TLC32044 analog interface controller (AIC) which contains 14-bit resolution ADC and DAC converters. Mohan *&* Ang (1994) concluded that a DSP could be used as a controller for a high-performance switching converter.

2.5 Advantages and Disadvantages of Digital Control Systems

Digital control systems have many advantages over analog control systems, some of which are presented in Table 2.6. (Martin *&* Ang, 1995; Duan *&* Jin, 1999; Wilkinson, 1997; Peterchev *&* Sanders, 2001; Karipidis, 2001; Peterchev *&* Sanders, 2003; Prodic *et al.*, 2003; Miao *et al.*, 2004; Syed *et al.*, 2004; Luo *et al.*, 2005; Milanovic *et al.*, 2005, 2007)

Table 2.6: Advantages of digital control systems.

Advantages
• There are many design tools that can help shorten the design procedure.
• Analog control system circuitries are affected by manufacturer tolerances and external factors such as temperature; whereas software based digital control systems remain largely unaffected by these problems.
• Digital systems are smaller and consume less power when compared to their analog counterparts.
• Highly reproducible/repeatable.
• Highly reprogrammable.
• Highly flexible/versatile; modification of control algorithms can be done without the need to alter hardware.
• Less susceptible to aging and environmental variations.
• Digital control systems are less susceptible to noise.
• Results in a reduction in the number of external components required.
• Results in improved dynamic response.

Some of the disadvantages are presented in Table 2.7: (du Toit *et al.*, 1995, 1997; Martin & Ang, 1995; Bester *et al.*, 1998; Duan & Jin, 1999; Karipidis, 2001)

Table 2.7: Disadvantages of digital control systems.

Disadvantages
• Real-time operation is not inherent to digital control systems and requires careful engineering to balance software and hardware implementations.
• Delays occur when converting from the analog to the digital domain.
• Signal resolution is determined by a finite word length of a digital controller.

2.6 Conclusion

In this chapter the basic control block for a switching converter was presented. This allowed for the basic components required for digital control to be determined, namely an ADC, DAC and DSP. A brief investigation was done on these components to establish what characteristics need to be present when selecting these components for the design. A thorough investigation was done on existing digital controllers to determine which components and what characteristics the components selected had and also for which end applications they had been designed for. This allowed for a generic structure for a digital controller to be conceived. From the investigation a list of advantages and disadvantages are presented and it could be seen that the advantages introduced by digital control outweigh the disadvantages.

Chapter 3

Controller Design and Considerations

3.1 Introduction

This chapter focuses on design and development of the CIRPEC hardware. Controller specifications as determined by the previous chapter are listed and mention is given to all the components of importance. The controller architecture overview is provided to give the reader a visual understanding of how all the components are linked. The power supply requirements including voltage and current are presented. This chapter concludes with the PCB design and layout and the challenges that need to be considered at high-switching frequencies.

3.2 Controller Specifications

In order to make the controller as generic, reprogrammable and flexible as possible, the specifications presented in Table 3.1 were decided upon. The component selections that satisfy these specifications are explained in more detail in this chapter.

Table 3.1: Controller specifications.

Controller specifications	
Programmable signal processor	Floating-point processor to compute control algorithms.
Programmable logic device	To perform co-processing and to add design flexibility.
Analog-to-digital conversion Sample rate (resolution)	> 100 kSPS (10-bit).
Analog input channels	> 8.
Digital-to-analog conversion Update rate (resolution)	> 100 kHz (12-bit).
Analog output channels	2.
Pulse-width modulation outputs	10, five complementary pairs.
inputs	10 for error signalling.
Connectors	20×Fibre optic (10×Rx/Tx) 1×expansion header 1×Universal Serial Bus 2×JTAG.
Memory storage	Non-volatile, volatile memory for program and data storage.
User interfaces	Push button.
Audio	Route audio pins from signal processor to PLD.

3.3 Controller Overview

A simplified block diagram of the CIRPEC is illustrated in Figure 3.3. It features a hybrid design consisting of a DSP and FPGA. One of the reasons for this design was due to the fact that this controller would be used as a tool for students and engineers. This allows for either the FPGA or DSP to be largely programmed as separate pieces of hardware.

A large high-speed data and address bus has been implemented in order to service the major components. These include the DSP, FPGA, Flash and synchronous dynamic random access memory (SDRAM). The data bus operates at 100 MHz. The DSP has direct access to the FPGA, SDRAM and EEPROM. In order for the DSP to access the upper address regions of the Flash memory, it needs to do so through the FPGA since there are not enough address pins on the DSP. This is useful since it allows the FPGA to have read

and write access to the Flash memory.

The DSP generally performs computationally intensive algorithms and therefore needs direct access to external memory such as SDRAM. The FPGA instead is used as a co-processor to the DSP and performs glue-logic functionality for many components. For instance the DSP would need to communicate via the FPGA to have access to the ADCs. In this case registers may need to be configured in the FPGA to allow communication to occur.

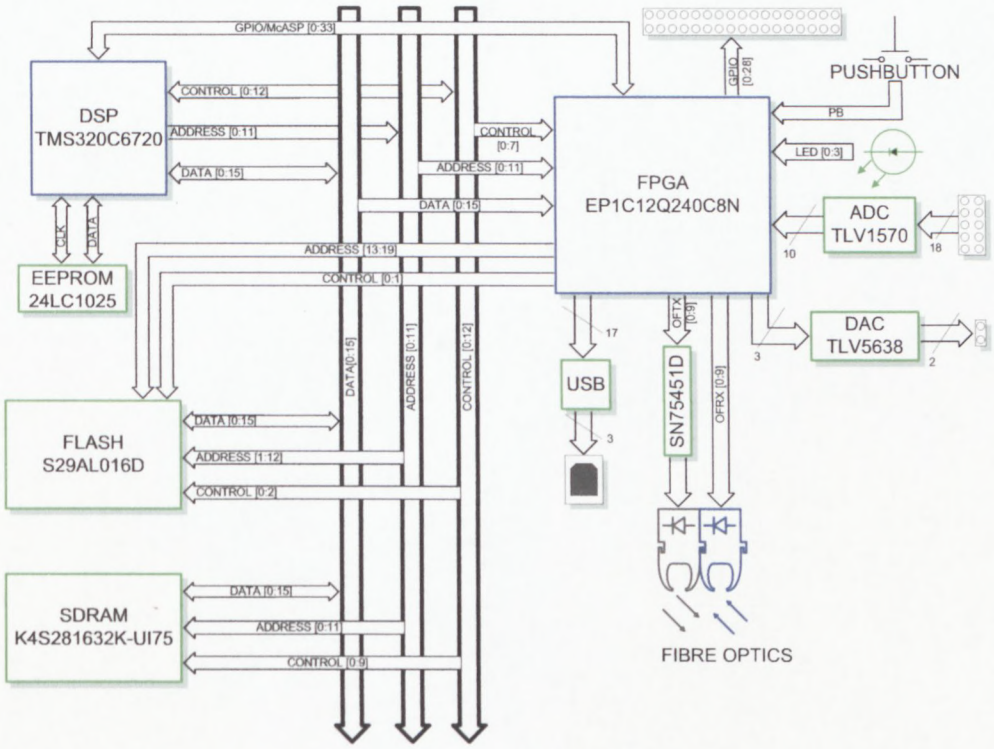


Figure 3.1: Block diagram of the CIRPEC.

3.4 Controller Components

This section provides more detail into the components selected for the CIRPEC in order to satisfy the specifications. Features of some of the devices are presented and explanation is given as to how and where these devices fit into the design of the controller.

3.4.1 DSP

A DSP is ideal for high-performance controllers. They have been designed for raw computational throughput and are ideal for processing real-time control algorithms. One of the things to consider when selecting a DSP is whether to select floating- or fixed-point. This was mentioned in Chapter 2. A floating-point DSP was thus selected for its greater dynamic range, mathematical flexibility, high accuracy and relative ease of programming. The DSP which was selected is the TMS320C6720 from Texas Instruments. A block diagram of the DSP is provided in Appendix C.1. This device will be referred to as the C6720 in the text. The TMS320C67× floating-point DSP's architecture operates by dividing a 32-bit data path into two parts in the form of scientific notation. The one part is a 24-bit mantissa which can either be used for integer values or as the base of a real number and the second part is an 8-bit exponent. The 24-bits allow for a precision range in the order of 16 M and coupled with the 8-bit exponent allows for a greater dynamic range than is possible with a fixed-point format. The C6720 DSP natively supports 32-bit fixed-point, 32-bit single-precision (SP) floating-point, and 64-bit double-precision (DP) floating-point arithmetic (Texas Instruments, 2008*b*). 64-Bit DP utilises a 53-bit mantissa and an 11-bit exponent. DP achieves much greater precision and dynamic range at the expense of speed as this requires multiple cycles per operation. (Frantz & Simar, 2004)

Table 3.2 presents some of the DSP's features. The C6720 is a high-performance 32-bit floating-point DSP featuring the enhanced C67×+ ('plus') central processing unit (CPU). The DSP operates at 200 MHz at which the CPU is capable of a maximum of 1200 MFLOPS (Texas Instruments, 2006*b*). The CPU is capable of executing up to eight instructions, six of which are floating-point, in parallel each cycle. It has 64 kB of internal random access memory (RAM) and 32 kB of instruction cache. Both are necessary when performing calculation intensive algorithms.

The C6720 has a 16-bit External Memory Interface (EMIF) allowing it to connect to external memories such as Single Data Rate SDRAM (SDR SDRAM), and asynchronous RAM or read only memory (ROM). The FPGA has been configured as an asynchronous memory so that the two devices can communicate. The data movement accelerator controller (dMAX) on the DSP can be used to transfer data between internal data memory and any address-

Table 3.2: DSP features.

TMS320C6720	
CPU Core	C67×+
Precision	32-bit Floating-point
Clock	200 MHz
MFLOPS	1200
MMACS	400
Architecture	VLIW
Memory RAM	64 kB
On-Chip L1/SRAM	32 kB
EMIF	16-bit
GPIO	64
External memory support	Async RAM/ROM, SDR SDRAM
I^2C	2
SPI	2
McASP	2
dMax	Yes
Timers	1×RTI
Core voltage	1.2 V
IO voltage	3.3 V
Package	144-TQFP

able memory space including external memory. These transfers include one-dimensional (1D), two-dimensional (2D) and three-dimensional (3D) transfers.

The DSP has a volatile program memory. If power is removed from the board then the DSP needs to load the program from a non-volatile memory device before it can operate normally. The controller has been designed such that it can boot from either a serial EEPROM or from Flash memory. Another method to boot the DSP is using the USB emulator which connects via the joint test action group (JTAG) pins.

3.4.1.1 Booting

TMS320C672× devices contain an on-chip ROM, identified as 'C9230C100', which contains a core set of Texas Instruments (TI) software for rapid software development. The contents of the ROM are presented in Table 3.3.

The C6720 supports one hardware boot mode option. When the device is

Table 3.3: Components of the C9230C100 ROM. Texas Instruments (2006*a*)

Component	Description	Address
Bootloader (BOOT ROM)	On-chip bootloader	0×0000 0000
DSP/BIOS	DSP/BIOS real-time operating system	0×0003 0000
FastRTS	Optimised math library containing common math functions e.g. cosine, sine, etc.	0×0002 C000
DSPLIB	Optimised set of common DSP functions e.g. FIRs, FFTs, vector MAX, etc.	0×0002 0000

reset, a program counter is set to start reading at the beginning of the ROM, address space 0×0000 0000, where execution of the bootloader begins. The boot modes which are supported by the bootloader are included in Table 3.4. To summarise, the C6720 can either be booted from a parallel FLASH device or from an EEPROM device using either the serial peripheral interface (SPI) or inter-integrated circuit (I^2C) protocol, in either slave or master mode. The CIRPEC provides both an EEPROM, in I^2C mode, and a parallel FLASH device as standalone bootable options. Thus far only the EEPROM has been used to boot the controller.

Table 3.4: Boot options supported by the on-chip bootloader for the C6720.

Boot mode	Description
Parallel Flash	Support for 8-/16-bit flash
SPI0 Master	4-wire SPI, standard protocol. 16-bits of address/8 bits of data
SPI0 Slave	DSP sends/receives 16-bit data
I2C1 Master	Standard I2C bus protocol. 16-bits of address/8 bits of data
I2C1 Slave	Addressing done through the I2C protocol. The DSP sends/receives 8-bit data

The bootloader uses the registers CFGPIN0 and CFGPIN1 to capture the state of various pins at reset. For the C6720 these pins include the SPI0_SOMI (slave out master in), SPI0_SIMO (slave in master out) and SPI0_CLK pins. To select which device to boot from on the CIRPEC three external boot mode

jumpers are provided relating to the three pins. The jumper positions, as displayed in Table 3.5, determine in what mode the device will boot on the CIRPEC. More information on using the bootloader can be found in Baldwin (2009).

Table 3.5: Boot mode selection for jumper pins.

Bootmode Selection			
Boot Mode	**JP1**	**JP2**	**JP3**
Parallel Flash	2-3	1-2	2-3
SPI Master	2-3	2-3	1-2
SPI Slave	2-3	1-2	1-2
I2C1 Master	1-2	2-3	1-2
I2C1 Slave	1-2	1-2	1-2

3.4.1.2 The Emulator

Another method to boot the DSP is to use a JTAG emulator. The XDS510USB PLUS emulator used was developed by Spectrum Digital. The emulator allows communication between a computer and the DSP and is connected in-between these pieces of hardware. An emulator is necessary to program the DSP and is also used for source code debugging. It allows stepping through the code on a line by line basis and the ability to monitor the values of variables and registers through the Code Composer Studio Integrated Development Environment (CCS IDE) software from Texas Instruments.

The JTAG interface has been standardised as IEEE 1149.1 (Spectrum Digital Inc., 2008). The side that gets connected to the computer is a universal serial bus (USB) connector. The other side of the emulator has a 14-pin (7×2) header which needs to be made provision for on the DSP board. A description of the JTAG emulator pins is given in Table 3.6. The JTAG interface for the DSP on the CIRPEC is presented in Figure 3.4.1.2 and has been designed to conform with the guidelines set out in Texas Instruments (2003).

3.4.1.3 Audio Signals

The C6720 has two McASPs (Multichannel Audio Serial Ports), of which one port which can provide up to 16 stereo channels using the IIS (I2S) format.

Table 3.6: DSP 14-pin emulator connector pin description. (Spectrum Digital Inc., 2007)

DSP 14-pin emulator connector pin description				
Pin #	Signal	Description	Emulator State	Target State
1	TMS	JTAG test mode select.	Output	Input
2	TRST-	JTAG test reset.	Output	Input
3	TDI	JTAG test data input.	Output	Input
4, 8, 10, 12	GND			
5	PD	Presence detect. Indicates the emulator cable is connected and the target is powered up.	Input	Output
6	KEYED			
7	TDO	JTAG test data output.	Input	Output
9	TCK_RET	JTAG test clock return. Test clock input to emulator.	Input	Output
11	TCK	JTAG test clock. 12 MHz clock from emulation pod.	Output	Input
13	EMU0	Emulation pin 0.	I/O	I/O
14	EMU1	Emulation pin 1.	I/O	I/O

The McASP can seamlessly interface with CODECs, DACs, ADCs and other devices. The McASP supports many variants of the IIS format including time division multiplex (TDM) formats with up to 32 time slots.

Considering the CIRPEC was designed with future audio applications in mind at the CIR, these multiplexed signals have been routed to the FPGA where they can then be routed off-board if necessary.

3.4.2 FPGA

An FPGA adds to the flexibility of the design as many external components can be interfaced to its large number of I/O pins. The FPGA selected is an EP1C12Q240C8 Cyclone device developed by Altera. It will be referred to as an EP1C12 in the text. Table 3.7 lists some of the features of the EP1C12 device.

The architecture of the FPGA is configured using complementary metal oxide semiconductor (CMOS) SRAM which requires configuration data to

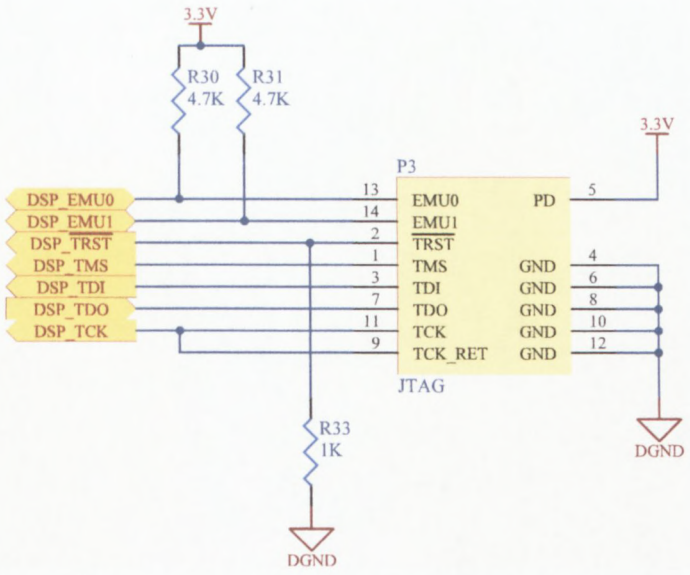


Figure 3.2: JTAG header interface to DSP.

Table 3.7: EP1C12 device features.

Cyclone EP1C12 device features	
Logic Elements	12,060
M4K RAM blocks (128×36 bits)	52
Total RAM bits	239,616
PLLs	2
Maximum user I/O pins	173
Package	240-PQFP
Core voltage	1.5 V
I/O buffer voltage	3.3 V

achieve logic, circuitry and interconnects. This is necessary each time the FPGA powers up.

FPGAs currently make provision for some DSP applications such as finite impulse response (FIR) filters, pseudo-random number generators, multi-channel filtering and auto-correlation and cross-correlation functions by implementing shift-register blocks in embedded memory. This helps to save logic cells that would otherwise be used to implement flip-flop logic (Altera Corporation, 2003).

The latest FPGAs allow for an embedded soft-core processor block, in Altera's case the NIOS® II processor can be logically programmed into the

device. The C/C++ language can then be used to program the processor as a hardware processor would. In a DSP/FPGA hybrid design however, it is recommended that the FPGA be used as a co-processor to the DSP (Hansmann, 2004). Altera also provides the software tools in the form of intellectual property (IP) functions to get the design up and running quickly.

As previously mentioned the function of the FPGA on the CIRPEC is to function as a co-processor to the DSP. It was therefore necessary to route the 16-bit data and 12-bit address lines to the FPGA. The FPGA would then have application specific modules running concurrent to the DSP.

The other function of the FPGA is to provide glue-logic and interface with other external devices. These included providing additional address lines for the Flash memory if the DSP was needing access to the full address range. The FPGA has been configured such that it has full access to the Flash memory. Other pins routed from the DSP include the multiplexed McASP0/1 signals, which include the SPI1, and SPI0 signals and other control signals for external asynchronous memories. A large majority of the DSP general purpose input output (GPIO) pins have thus been routed to the FPGA where they can be used to interface to devices such as ADCs and DACs.

Other devices which have been interfaced to the FPGA include two ADCs, one DAC, a header to allow for expansion, four LEDs for troubleshooting or general signaling, ten fibre optic transmitters and ten receivers, a USB device to interface with a computer, and a pushbutton for a reset or other user interaction.

3.4.2.1 Configuration Device

Each time the FPGA loses power the configuration of the FPGA is lost. At system power-up, a configuration device is required to load the FPGA with SRAM configuration data. This process is known as configuration. The device used is an Altera EPCS4SI8N and acts as a controller in an active serial configuration. Table 3.8 presents the memory size of the device. The configuration typically requires less than 120 ms using serial data at a rate of 20 MHz (Altera Corporation, 2003).

After configuration, the FPGA enters an initialisation mode where the registers are reset and the I/O pins are enabled. The FPGA will now operate as a logic device. The configuration and initialisation processes together are

Table 3.8: Serial configuration device features.

Serial configuration device	
Device	EPCS4SI8
Memory size (bits)	4,194,304

referred to as *command mode* and normal device operation is referred to as *user mode*.

Since the Cyclone devices use SRAM configuration elements they can be reconfigured in-circuit by loading the new configuration data into the device. When real-time configuration is occurring, the device is placed into command mode by a device pin. After the new configuration data has been loaded and initialised the device will return to user mode.

3.4.2.2 Emulator

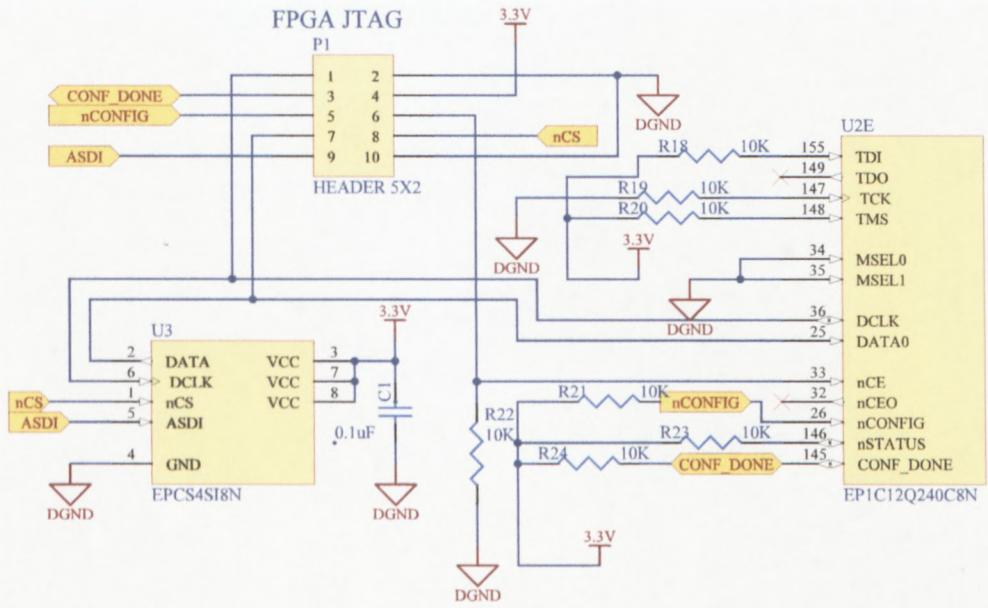


Figure 3.3: Connections for the EPCS4 configuration device.

In order to program the serial configuration device, EPCS4, a JTAG in-

terface (IEEE 1149.1) is required. Figure 3.4.2.2 illustrates how the JTAG header and configuration device have been connected to the FPGA. The programmer used was an Altera USB Blaster download cable. During in-system programming of the EPCS4 using the USB Blaster download cable, the cable pulls *nCONFIG* low which resets the FPGA and overrides the 10 kΩ pull-down resistor on the FPGA's *nCE* pin. The cable then uses the *DATA, nCS, ASDI,* and *DCLK* pins to program the EPCS4. As soon as the programming is complete, the cable releases the EPCS4's four interface pins and the FPGA's *nCE* pin, and pulses *nCONFIG* to begin the configuration.

3.4.3 ADC

An ADC is necessary to convert signals from the analog domain to the digital domain so that it can be processed by a digital processor. When considering the selection of an ADC the factors mentioned in Section 2.3.1 need to be taken into account. To revise, these include factors such as conversion time and quantization error.

The ADC selected was the TLV1570 developed by Texas Instruments. It was selected since it satisfied the specifications presented at the beginning of the chapter. Some of the features are presented in Table 3.9.

Table 3.9: ADC features.

TLV1570 features	
Architecture	SAR
Max. sample rate	1.25 MSPS
Channels	8
Resolution	10 bits
Reference	Int. 2.3 V, 3.8 V Ext.
Analog input range	+5 V
SINAD	60 dB
SNR	61 dB
SPI interface	Yes
CCS plug-in	Yes
Package	20-SOIC

The TLV1570 is a successive approximation register (SAR) architecture. It has a maximum sample rate of 1.25 MSPS with a maximum input clock

frequency of 20 MHz. The ADC interfaces serially with other devices such as a DSP using the SPI interface. There is no additional glue-logic required to interface the ADC to the C6720 and can thus be routed directly through the FPGA. The C6720 has two SPI ports which operate at 10 MHz, so if the ADCs are interfaced to the DSP then the maximum sample rate that can be achieved will be $^{10\ \text{MHz}}/_{16} = 625$ kSPS where each conversion requires 16 clock cycles. Were all eight input channels to be used this would reduce the sample rate to $^{625\ \text{kSPS}}/_{8} = 78.125$ kSPS.

Two ADCs have been included in the design so if all 16 channels are not required then the inputs can be split among the two devices. The clock input for one of the ADCs has been connected to the output of one of the FPGA's phase-lock loops (PLL), thus if a higher frequency clock signal is required for a different application it can be achieved.

Since the TLV1570 has a fixed supply voltage of +5 V it has a single internal reference of 3.8 V. The external reference pin has been routed off-board for application flexibility so that a different reference voltage can be selected if required. Table 3.10 presents the configuration register (CR). A configuration word with the corresponding bits are selected and sent to the SDIN pin to configure the ADC as desired. A new configuration word must be sent after each conversion.

3.4.4 DAC

A DAC is necessary to convert digital values back into the analog domain. When selecting the DAC it was important to consider factors such as update rate and settling time. A fast update rate and settling time was desired. The DAC selected for the CIRPEC was the TLV5638 developed by Texas Instruments, it was selected since it satisfied the specifications. Features of the DAC are presented in Table 3.11.

The TLV5638 is a dual 12-bit single supply DAC. It is based on a resistor string architecture. It also features speed and power-down control logic, a resistor string and rail-to-rail output buffer. There is also a programmable internal reference which can be set to internal or external reference. Since the DAC has been configured for +5 V the internal voltage selectable is 2.048 V. The external reference pin has been configured to +2.5 V by using a resistor divide network connected externally to the device.

Table 3.10: ADC configuration register. (Texas Instruments, 1998)

Configuration register (CR)	
Bit #	Description
15	Software power down: 0: Normal 1: Power down enabled
14	Reads out values of the internal register: 1: Read (15:1 are read out)
13:12	Self-test voltage is applied to the ADC input during next clock cylcle: 00: Allow AIN to come in normally 01: Apply AGND to AIN 10: Apply VREF/2 to AIN 11: N/A
11	Choose speed application: 0: High speed (higher power consumption) 1: Low speed (lower power consumption)
10	Enable channel auto-scan: 0: Disabled 1: Enabled
9:7	Channel selection/sequence selection
6	Internal or external reference voltage: 0: External 1: Internal
5	Select internal reference voltage value: 0: 2.3 V (3 V operation) 1: 3.8 V (5 V operation)
4	Auto-power function down 0: Disable 1: Enable
3	Performance optimizer - linearity 0: AV_{DD} = 5.5 V to 3.6 V 1: AV_{DD} = 3.5 V to 2.7 V
2:0	Reserved bits (Write 0)

The output voltage is determined by the following formula: Texas Instruments (2004).

$$2 \times Ref \cdot \frac{CODE}{0 \times 1000} \text{ (V)} \tag{3.4.1}$$

Where *Ref* refers to the reference voltage and *CODE* refers to the digital

Table 3.11: TLV5638 features.

TLV5638 features	
Architecture	Resistor string
Resolution	12-bit
Channels	2
Settling time	1 μs (Fast mode) 3.5 μs (Slow mode)
Update rate (max)	1.25 MHz
Reference	Int. 1.024 V, 2.048 V Ext. 2.5 V
SPI interface	Yes
SNR	74 dB
SFDR	72 dB
CCS plug-in	Yes
Package	8-SOIC

value sent from the DSP in the range 0×000 to 0×FFF.

The DAC has a SPI interface which allows it to be interfaced directly to the C6720 without any additional glue-logic. It would however have to be routed through the FPGA in order for the DSP to have access to it. The maximum serial clock rate is 20 MHz. Since it is a serial device it requires 16 clock cycles before the device will be configured and the settling time would have to be considered.

Table 3.12 presents the 16-bit configuration word that is required to configure the DAC as desired. Data bits 15:12 represent the program bits and bits 11:0 represent the data bits.

3.4.5 SDRAM

The inclusion of SDRAM was to provide the DSP with additional memory in the form of fast access off-chip memory. This additional memory could be used as a temporary storage place for program and data memory. SDRAM is volatile so as soon as power is removed from the board the data currently in the memory will be lost.

The way the CIRPEC architecture has been designed results in only the DSP having access to the SDRAM. The external memory interface (EMIF) of the DSP supports 133 MHz SDRAM either 16- or 32-bit interface with 1, 2,

Table 3.12: DAC configuration word.

TLV5638 data word format	
Bit #	Description
15, 12	Register select (R1, R0): 00: Write data to DAC B and Buffer 01: Write data to Buffer 10: Write data to DAC A and update DAC B with Buffer data 11: Write data to control register
14	Speed control: 0: Slow mode 1: Fast mode
13	Power control bit: 0: Normal operation 1: Power down
11:0	Data bits (Register dependent): DAC A/B or Buffer: New DAC value 15, 12 = 11 (Control): 1:0 = REF1, REF0
REF×	Reference voltage select: 00/11: External 01: 1.024 V (+3 V) 10: 2.048 V (+5 V)

or 4 banks and supports SDRAM devices up to 128 Mb.

The SDRAM selected was the K4S281632K-UI75 manufactured by Samsung Electronics. Table 3.13 presents some of the features. It is a 128 Mb synchronous high data rate dynamic RAM. The SDRAM is organised as 2 Mb × 4 banks × 16-bits. The interface to the SDRAM is therefore 16-bits. The maximum frequency the SDRAM can be driven at is 133 MHz. The frequency of the EMIF however is 100 MHz and that is therefore the frequency that is used to drive the SDRAM on the CIRPEC. SDRAM will lose the values stored if periodic refreshing does not take place. The SDRAM features an automatic refresh rate every 64 ms (4k cycles) and there is also the option for a self refresh.

3.4.6 Flash Memory

Flash memory is a non-volatile memory. It provides a larger and faster means to store data when compared to an EEPROM. It has a parallel interface with

Table 3.13: SDRAM features. (Samsung Electronics, 2008)

SDRAM features	
Manufacturer	Samsung
Device	K4S281632K
Density	128 Mb (16 MB)
Organisation	8 M×16-bit
Banks	4
Max. frequency	133 MHz
CAS latency	2 & 3
Refresh	Auto & self
Interface	LVTTL
Package	54-TSOP-II

which to access the memory rather than a serial one in the case of the EEPROM. This memory also provides an alternative device for the DSP to boot from. The Flash memory selected is the S29AL016D manufactured by Spansion. Some of its features are presented in Table 3.14.

Table 3.14: Flash memory features. (Spansion Inc., 2007)

Flash memory features	
Manufacturer	Spansion
Device	S29AL016D
Density	16 Mb (2 MB)
Organisation	1 M×16-bit
Access speed	70 ns
Power saving	Automatic sleep, standby mode
Package	48-TSOP

It is a 16-Mb Flash organised as a 1 M×16-bit interface. On the CIRPEC it is connected to the main 16-bit data bus and the address bus. Additional address lines and control lines are necessary from the FPGA since the DSP does not have enough memory address pins to interface directly to the upper address lines of the Flash memory.

This is performed on the DSP by configuring some of the GPIO pins that are routed to the FPGA as GPIO. In order for a pin to be configured as a GPIO it needs to be pulled low and held low during reset until configured as

an output. Although this may not be necessary, as a precaution in the hardware design some pins have been selected for this upper address line purpose and provision has been made for a pull-down resistor. The pins selected are illustrated in Figure 3.4.6.

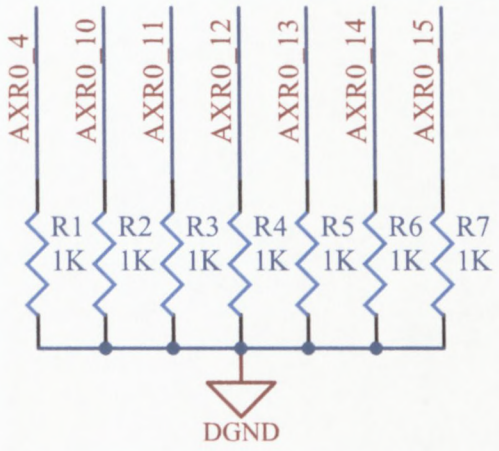


Figure 3.4: GPIO pins selected for Flash memory address pins.

3.4.7 EEPROM

Since the DSP has a volatile memory there was a need for a device to store the boot image that will be loaded by the DSPs bootloader. The Flash memory can serve this purpose but a device more suited to this purpose is a serial EEPROM. When the DSP is booting there is no real need for a parallel interface and fast access speeds.

The serial EEPROM selected is the 24LC1025 by Microchip. Some of the device's features are presented in Table 3.15.

The I^2C interface requires only two lines, namely *SDA* (data) and *SCL* (clock) to transfer data between the devices. The protocol has two speeds, *standard mode* has a frequency of 100 kHz and *fast mode* 400 kHz. The EEPROM has been configured for fast mode on the CIRPEC by providing a 1 kΩ pull-up resistor to the clock and data lines.

The EEPROM also provides a write protect input. When this input is tied to V_{SS} (Gnd) write operations are enabled. When tied to V_{CC} (+3.3 V) write operations are inhibited, however read operations are still possible. A jumper

Table 3.15: Serial EEPROM features. (Microchip Technology, 2007)

Serial EEPROM features	
Manufacturer	Microchip
Device	24LC1025
Density	1024 kb (1 Mb)
Configuration	128 kb×8-bit
Interface	I^2C
Max freq.	400 kHz
Erase/Write cycles	> 1 M
Package	8-SOIJ

has been provided on the CIRPEC to provide the user with the option to write protect data on the chip to prevent accidental erasure.

Transfer of data to the EEPROM, including the boot image, needs to be performed through the DSP. A new boot image is uploaded to the EEPROM while being connected to the USB Emulator and CCS software. The DSP is the only device which has access to the EEPROM. Once the Emulator is removed and the DSP is allowed to boot standalone it needs to retrieve the boot image from the EEPROM.

The EEPROM is thus configured as an I^2C slave device. It monitors the *SDA* bus for a control byte that will be sent from the master device. The control byte consists of a 4-bit control code. In the case of the 24LC1025 the control code is '1010'. The control code is then followed by a *block select* bit (B0), two *chip select* bits (A1, A0) and finally a *read/write* bit. The control byte is illustrated in Figure 3.4.7. The bit before the control byte is referred to as the start condition. Depending on whether *SDA* is high or low during a *SCL* high determines whether there is a start or stop condition. Once the control byte has been correctly received the EEPROM transmits an *acknowledge* signal. In the case of the bootloader, the DSP requests a read operation followed by the start address. The EEPROM then transmits the data.

3.4.8 USB

In order for the CIRPEC to be able to communicate with a computer at relatively high speeds, a USB peripheral controller device was added to the

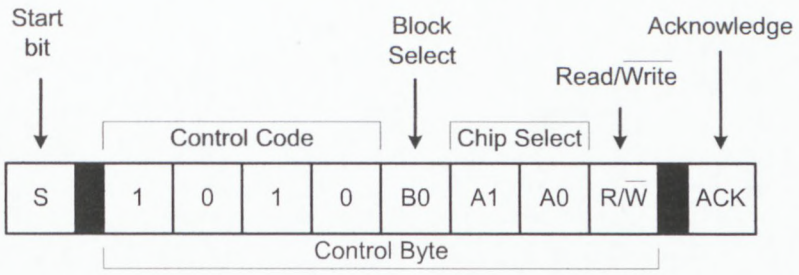


Figure 3.5: EEPROM configuration byte.

controller. The device selected was the PDIUSBD12 manufactured by ST-NXP Wireless. Some of the device's features are presented in Table 3.16.

Table 3.16: USB device features. (ST-NXP Wireless, 2006)

USB peripheral controller features	
Manufacturer	ST-NXP Wireless
Device	PDIUSBD12
USB Specification Rev.	2.0
Transfer interface	12 Mb/s Full-speed 2 MB/s High-speed parallel
Data transfer rates	1 MB/s bulk mode 1 Mb/s isochronous
Package	28-TSSOP28

The device is USB revision 2.0 compatible which supports a transfer rate of 12 Mb/s. It has a high-speed parallel interface which is used to interface with a microcontroller. The USB device on the CIRPEC is directly interfaced with the FPGA using the parallel interface. The device also features a *GoodLink* output which allows for an external LED to be connected to the device. This will provide an indication of the data traffic through the device. The USB device also features a local direct memory access (DMA) controller to efficiently transfer blocks of data between the host and local memory. Transfers occur autonomously meaning there is no need for local CPU intervention. The device also features 320 bytes of multi-configuration first in first out (FIFO) memory.

3.4.9 PLL Clock Driver

It is not advisable to drive multiple devices using the same clock, therefore a phase-lock loop (PLL) clock driver has been implemented on the CIRPEC. The clock driver used was a CDCVF2505 manufactured by Texas Instruments. Some of the features are summarised in Table 3.17.

Table 3.17: Clock driver features. Texas Instruments (2005*a*)

Clock driver features	
Manufacturer	Texas Instruments
Device	CDCVF2505
Operating frequency	24 - 200 MHz
Clock outputs	5
Low jitter	$<$ 150 ps
Package	8-TSSOP

The CDCVF2505 is a high-performance clock driver and has low skew and jitter. An advantage of this clock driver is that it does not require any external RC network. There is however a small margin of time that the PLL requires to stabilise and achieve a lock on the input signal. The driver has one clock input which it then distributes to the output clocks. The output clocks are aligned in both frequency and phase to the input clock.

In the case of the CIRPEC the input clock to the driver comes from the DSPs EMIF clock. This clock has a frequency of 100 MHz which is within the range of frequency that the driver operates in. Two clock outputs are then distributed to the SDRAM and the Flash memory respectively. As there is a lot of high-frequency switching taking place in the device precautions were taken to isolate the power supplies so as to prevent any noise from affecting other power supplies.

Even though the device has internal series damping resistors, additional 33-Ω external damping resistors were added.

3.4.10 Off-Board Connector

Figure 3.6 illustrates the connections of the off-board connector. The connector makes provision for 29 GPIO pins which are directly linked to the FPGA. The I/O pins of the FPGA can be configured as low-voltage transistor-transistor

logic (LVTTL), low-voltage CMOS (LVCMOS), 2.5 V, 1.8 V, 1.5 V, 2.5 V low-voltage differential signalling (LVDS), stub series terminated logic (SSTL-2) class I/II, differential SSTL-2, SSTL-3 class I/II and reduced swing differential signalling (RSDS) (Altera Corporation, 2003). It is also possible for the FPGA to interface with multiple-voltage systems, making it possible to interface with a 5.0 V, 3.3 V or 1.5 V device. This makes it possible for the FPGA to interface to a multitude of technologies.

The flexibility of the FPGA allows the CIRPEC to have access to off-board devices while at the same time it could provide a daughterboard access to a device on the CIRPEC. Most signals are made available from the DSP and will allow for many DSP related applications. Examples of daughterboards include measurement, additional fibre optic devices and audio.

Power is also provided through the off-board connector. +5 V, +3.3 V and Gnd is provided and will allow the daughterboard to be powered.

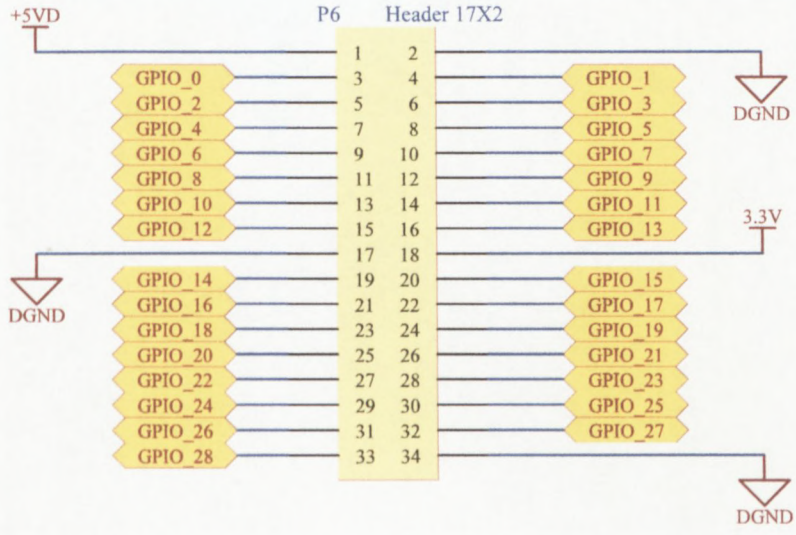


Figure 3.6: Off-board connector.

3.4.11 Fibre Optics

Fibre optics were necessary as a means to communicate with the power converter. They provide a link which is electrically (voltage) isolated between the

two boards. Another advantage is it allows the controller to be placed a fair distance from the converter.

On the CIRPEC there are ten transmit and ten receive devices. The devices which were used are the HFBR-1521 for transmit and HFBR-2521 for receive manufactured by Avago Technologies. Some of the features are presented in Table 3.18. In order to drive the transmit devices a dual peripheral driver integrated circuit (IC) was used. The device used was the SN75451B manufactured by Texas Instruments and can be used to drive a TTL load up to 300 mA.

Table 3.18: Fibre optic features. (Avago Technologies, 2006)

Fibre optic features		
	Transmit	Receive
Manufacturer	Avago	
Device	HFBR-1521	HFBR-2521
Operating wavelength	660 nm	
Distance	27 m (min.)	48 m (typ.)
Signal rate	5 Megabaud	
Maximum supply current	60 mA	10 mA

3.4.12 LEDs

Four LEDs have been added off the FPGA as general purpose LEDs. They could be used to indicate a fault, troubleshoot or provide a general indication.

3.4.13 Pushbutton

A pushbutton has been included and could be used for various applications. A possible application is providing a reset for the DSP.

3.5 CIRPEC Features

Table 3.19 presents a summary of the CIRPEC features. It has a more powerful DSP in comparison to the researched controllers, capable of 1200 MFLOPS.

Table 3.19: CIRPEC features.

CIRPEC features	
Name	CIRPEC
Manufacturer	CPUT (CIR)
DSP	TMS320C6720
Precision	32-bit
Fixed/Floating point	Floating
MFLOPS	1200
MIPS	-
Data bus width	16-bit
Serial ports	2×I2C, 2×SPI
ADC channels	-
PWM channels	-
PLDs	
FPGAs	1×EP1C12Q240C8N
EPLDs	-
Memory	
DSP (internal)	
RAM	64 kB, 32 kB instruction cache
ROM/FLASH/EEPROM	384 kB
DSP (External)	
RAM (SRAM/SDRAM)	128 Mb SDRAM (16-bit)
FLASH/EEPROM	1 Mb EEPROM, 16 Mb Flash (16-bit)
Communication	
RS-232/SPI/I^2C/USB	USB/SPI/I^2C
Emulator (DSP/FPGA)	JTAG (DSP/FPGA)
Fibre optics	20 (10×Rx, 10×Tx)
Analog inputs	16 (2×8 Channel) (10-bit)
Analog outputs	2 (12-bit)
PWM outputs	10
Power supply	5 V, 3 A (max)
Expansion header	Yes

This reduces the need for additional processing power through the use of multiple DSPs.

The CIRPEC features a DSP/FPGA co-processor design. This architecture was utilised in most of the controller designs. Karipidis (2001) explained how it is advantageous to complement a high-performance DSP with an FPGA as this configuration allows for greater design flexibility.

If necessary the CIRPEC could be implemented as a standalone controller or as a central controller in a distributed control structure on a single PCB, making use of its hybrid DSP and FPGA architecture along with its fibre optic transmit and receive devices.

The CIRPEC makes provision for ten transmit and receive fibre optic devices on the PCB while allowing for expansion capabilities through the expansion header.

3.6 Power Supply, Reset Circuitry and Considerations

This section describes the power supply requirements for the components on the CIRPEC. The voltages and current requirements of the components are presented. Reset circuitry of the controller is also presented.

3.6.1 Power Supply

Power distribution can be a complex design obstacle especially when there are many components that may require different operating voltages. The design of the CIRPEC had to make provision for four voltage levels including an analog and digital +5 V supply. Table 3.20 presents the different voltages and current requirements of the components on the controller.

In order to provide these voltages, voltage regulators and supervisors were incorporated into the design. Table 3.21 is a list of the voltage regulators used. All the regulators selected were low dropout (LDO) regulators. Even though they may not be as efficient as switching regulators they emit much less noise onto the supply and have fast transient response to load changes. They are also a cheaper solution to their switching regulator counterparts.

Table 3.20: Voltage and current requirements of CIRPEC components in mA.

Device voltage and current requirements				
Device	**+5 V**	**+3.3 V**	**+1.5 V**	**+1.2 V**
DSP Core				~440
DSP I/O		~58		
FPGA Core			~200	
FPGA I/O		~100		
FPGA config device		5		
SDRAM		200		
Flash		35		
USB		15		
EEPROM		5		
5×Fibre optic driver	650			
2×ADC	17			
DAC	7			
Clock PLL driver		40		
2×OSC		45		

Table 3.21: Voltage regulators on the CIRPEC.

Regulator voltage and current characteristics		
Regulator	**Voltage**	**Current**
LM1085	+5 V	3 A
UA78M05	+5 V	500 mA
TPS70448	+3.3 V, +1.5 V	1 A, 2 A
TPS70445	+3.3 V, +1.2 V	1 A, 2 A

Figure 3.6.1 provides a graphical representation of the power distribution in the CIRPEC. There is a single point where power is connected to the board. In order to protect the components on the board as well as to provide a regulated 5 V to the controller a LM1085 manufactured by National Semiconductor was selected (National Semiconductor, 2005). This particular device was selected because it has a large maximum input to output voltage differential of 25 V and is capable of supplying 3 A with a maximum dropout of 1.5 V. It was necessary that this regulator provide a large amount of current as this is the regulator that supplies digital +5 V to the controller including other lower voltage regulators.

The UA78M05 manufactured by Texas Instruments was selected to provide

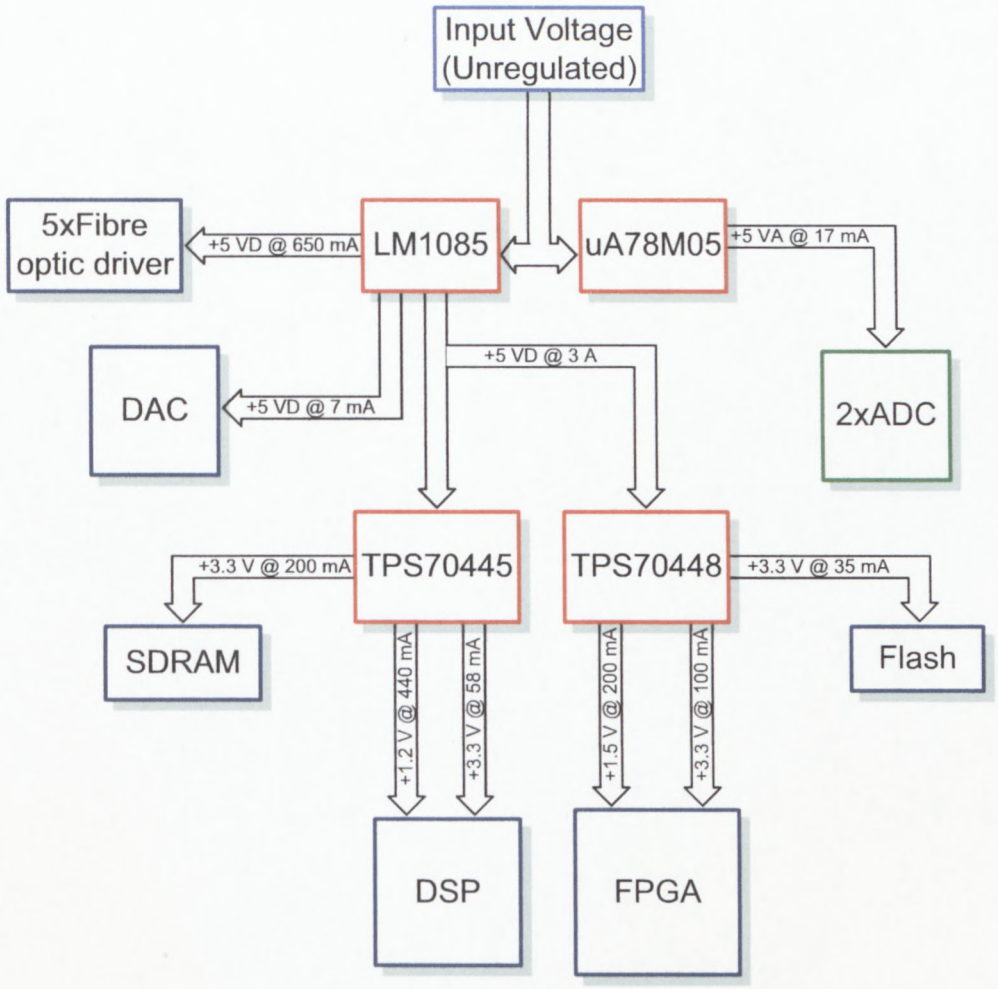


Figure 3.7: Graphical representation of the CIRPEC power distribution.

---

an isolated +5 V to the analog components (Texas Instruments, 2005*c*). It is capable of providing 500 mA of current. Since this regulator is also connected at the point where power is connected to the board it had to be selected with a high input voltage rating. This particular device is capable of operating up to a maximum of 35 V. As with the LM1085, this device also features current-limiting and thermal-shutdown circuitry.

The TPS7044× set of regulators are dual output LDOs manufactured by Texas Instruments (Texas Instruments, 2007). They also provide integrated supervisory circuitry (SVS) which includes a reset, power on reset (POR) and power good (PG) functions. These dual regulators can supply 1 A for regulator

1, which is connected to the I/O supply rails of the DSP and FPGA (+3.3 V). Regulator 2 can supply 2 A. This regulator is connected to the core supply rails of the DSP (+1.2 V) and FPGA (+1.5 V). It is necessary that the core supply regulator is capable of supplying a large current as the core typically uses more current than the I/O peripherals and when the FPGA and DSP are switched on there is a larger than normal current required for a small period of time. In the case of the FPGA at startup it may draw a maximum of 900 mA typically for a few milliseconds.

An advantage of using the dual regulators is for the power supply sequencing of the DSP which states that neither supply rail should be powered up for longer than one second while the other supply rail powered down as defined in the datasheet. (Texas Instruments, 2008*b*)

Power supply indicators in the form of LEDs have been implemented on the CIRPEC for +5 VD, +5 VA and 3.3 V. This feature provides an indication of whether all the power levels are up and running or not.

3.6.2 Reset Circuitry

Figure 3.6.2 illustrates the reset circuitry used in the CIRPEC.

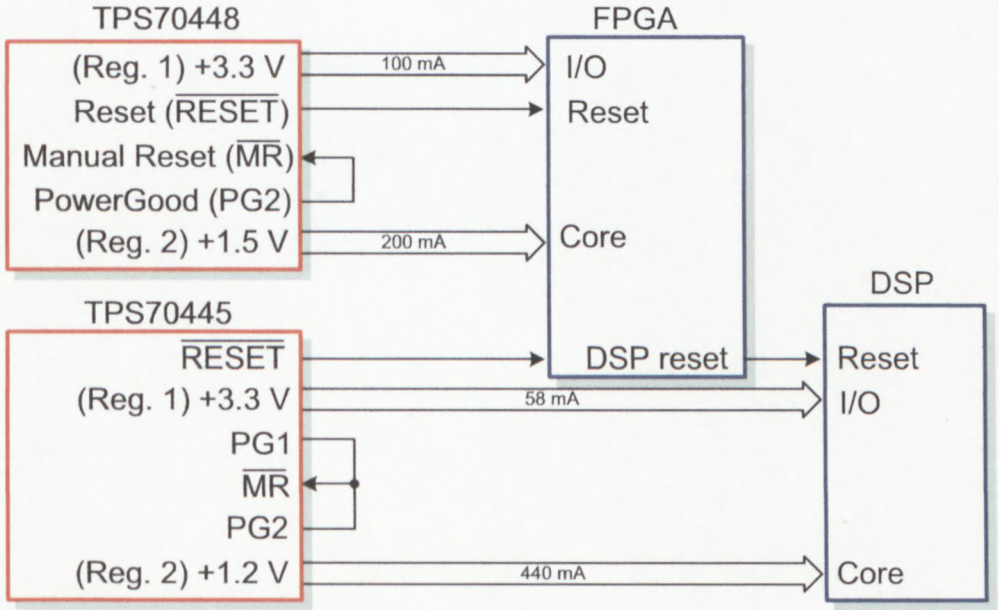


Figure 3.8: Reset circuitry of the CIRPEC.

For the FPGA's supply regulator, TPS70448, the $\overline{RESET}$ output from the regulator is driven low when regulator 1 is below the undervoltage condition and the $\overline{MR}$, manual reset, is not active. The reset pin is used to drive the *nCONFIG* pin of the FPGA which means that the FPGA is held in reset until it is released by the supervisor circuitry. The *PG2*, power-good, pin is used to monitor the voltage of the second regulator, 1.5 V, or rather the FPGA's core supply. When an overload condition occurs the power-good pin will be pulled low. This pin has been connected to the manual reset pin so that reset of this power supply occurs if this happens.

The supply regulator for the DSP, TPS70445, has been configured in much the same way. However, when the reset signal of the regulator is kept low it will pull a normal I/O pin on the FPGA low. The reason for this is so that the reset of the DSP can be controlled with some combinational logic inside the FPGA. A separate signal is sent from the FPGA to reset the DSP. This implies that this logic needs to be enabled in the FPGA before the DSP can be used. The FPGA therefore has control over when the DSP is allowed to run and can be held in reset if it is not needed. Another difference with the power supply configuration is that both the regulator 1 and regulator 2 voltages are monitored and will generate a reset when an overload condition occurs.

3.7 PCB Design and Considerations

Much research went into the physical design of the controller. There are many factors to consider when designing a controller with both analog and high-frequency switching digital components on the same PCB. Factors include current return paths, electromagnetic interference (EMI) suppression, board zoning, component placement, number of layers, power distribution.

Systems which have high-speed clocks, transmit and receive signals that operate in the high megahertz range will generate a lot of noise and radiation which can degrade the system's performance and create EMI. Good board design techniques can help to reduce the amount of noise and EMI generated.

3.7.1 Printed Circuit Board Layout

A powerful computer aided design (CAD) software tool in the form of Altium Designer 6 was used in the layout of the controller. The CIRPEC schematics and PCBs can be found in Appendices A and B respectively.

The layout was challenging in the fact that there were numerous devices that had large pin counts and there were analog and digital components in the design. As a result a four layer PCB was decided upon with two signal, ground and power layers. All the active components bar one were placed on the top layer and a majority of the passive components were placed on the underside of the board. This ensured that a majority of the signal traces existed on the top layer, adjacent to the ground plane.

The DSP and FPGA formed core components of the design since they had the most connections between each other. It was therefore decided to place these components relatively centre to the board. The FPGA had connections to many other components and these were placed in close proximity. Off-board connectors such as the header, ADCs, DAC and USB were placed on the edge of the board for easy access to off-board connections. Due to space constraints the fibre optic transmitters were placed at the edge along the length of the board.

The DSP had to have access to the FPGA, SDRAM and Flash using a large data and address bus. These components were therefore placed in close proximity to each other.

Important to consider when working with analog and digital components on the same board is the zoning of the board. For this reason the digital and analog components were separated and the analog components placed away from the digital components so as to reduce the interference that would be experienced by them. Routing of digital signals in the analog section was avoided.

The input power connector to the board was placed on the side of the board closest to the digital components. This would allow for the shortest return path for the digital currents so that they would not interfere with the analog section of the board.

3.7.2 Power Distribution

The CIRPEC has one ground plane and one power plane. The ground plane is adjacent to the top signal trace layer and the power plane is adjacent to the bottom signal trace layer.

A solid ground plane was opted for as the best solution for a mixed signal PCB design as explained in Ott (2001). Analog components were placed away from the digital components to prevent them being interfered with by noise. A solid ground polygon pour was placed over the top layer to further reduce EMI and radiation. A large number of vias were also used between ground polygons and the ground plane to reduce the impedance between the grounds.

The power plane was configured as the +3.3 V plane since a majority of the components operate from this supply voltage. From this solid +3.3 V plane smaller isolated islands were created where an isolated or different voltage supply was needed. The bottom signal layer also served as the main supply of digital +5, +1.2, +1.5 V and also the analog +5 V by placing polygon pours. A large section of the bottom layer was also covered by a ground polygon pour to reduce noise.

The effects of crosstalk needed to be taken into account and that current return paths differ depending on their frequency. Crosstalk refers to the unwanted interference from nearby circuitry. This can be in the form of electromagnetic radiation which generates unwanted signals on the power and ground planes. For low-speed signal currents, typically less than 10 MHz, the current returns on the path of least resistance which is normally the shortest path back to the source. Whereas for high-speed signal currents, typically greater than 10 MHz, the current returns on the path of least inductance which is normally beneath the signal trace. This is illustrated in Figure 3.7.2 (Texas Instruments, 2005*b*).

These return currents will run on the plane adjacent to the signal path regardless of whether it is a ground plane or power plane (Ott, 2001). In the case of the CIRPEC this will be the ground and +3.3 V power plane as the signal layers are on the outsides.

Caution was therefore applied to reduce crosstalk by preventing the placement of slots in the ground and power planes. Where there were power islands care was taken to prevent running a trace over the slot on the layer adjacent to the power layer, as this would result in a longer current return loop which

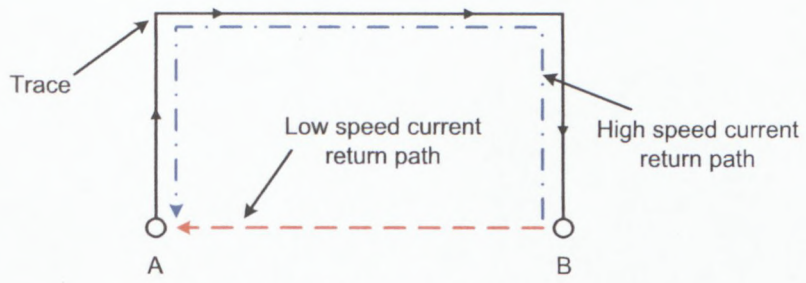


Figure 3.9: Current return paths.

---

would act as an antenna and emit radiation (Ott, 2001).

To reduce the phenomenon known as ground bounce as many V_{CC}/Gnd pairs as possible of the active devices have been decoupled, especially around the FPGA and DSP. Ground bounce occurs as a result of high-transient currents generated by many fast switching device outputs transitioning from a logic high to a logic low as they discharge load capacitances (Altera Corporation, 2001). Figure 3.7.2 illustrates the placement of bulk and decoupling capacitors underneath the board around the DSP and FPGA.

3.7.3 Filtering Techniques

All power supplies and PLL pins were filtered using a ferrite bead and capacitors. A ferrite bead has similar characteristics to an inductor but has negligible parasitic capacitance. It can be said to behave as an inductor over a wide range of frequencies (Texas Instruments, 2005*b*). In the case of the DSP PLL an external '*T*' filter was suggested in the datasheet. This filter implements a single capacitor and two ferrite beads. A '*Pi*' filter was used to isolate the PLL of the FPGA and clock driver supply, this filter is implemented using a single ferrite bead and two capacitors on either side. Filtering was necessary to isolate supplies to prevent noise from entering or escaping the particular supply.

Large value bulk capacitors were used to filter low-frequency noise while small value capacitors were used for high-frequency noise filtering. These decoupling capacitors were placed as close to the V_{CC} pins as possible.

In order to further reduce crosstalk all high-frequency clocking signals have been filtered using a 33 Ω series terminating resistor to reduce the rise and fall times of the signal.

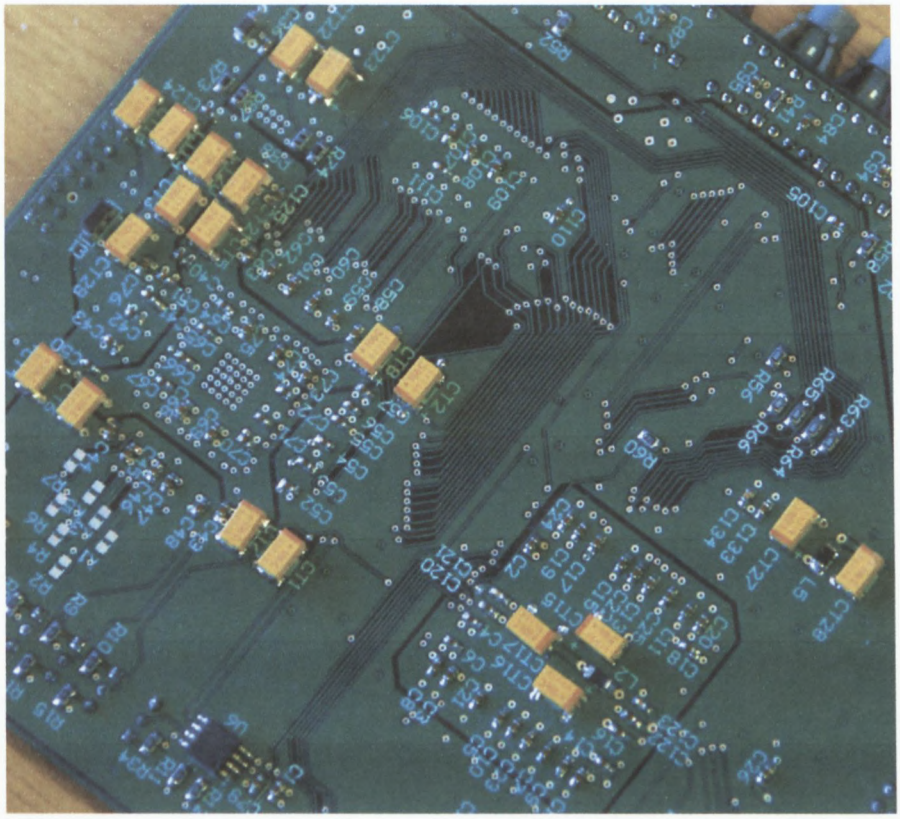


Figure 3.10: Decoupling V_{CC}/Gnd pairs around the DSP (left) and FPGA (right).

3.7.4 Thermal Considerations

Some of the devices which produce a lot of heat, such as the power supplies and the DSP need some form of heat dissipation. The DSP as well as the power supplies for the DSP and FPGA are TI devices. They provide what is referred to as a PowerPad (thermal pad) underneath the device (Texas Instruments, 2008*a*). The thermal pad provides a low thermal resistance path between the die and the exterior of the package. It is soldered directly on the PCB and connected to the ground plane of the PCB using vias. The ground plane performs the function of a heat sink. The same principle was applied to the remaining power supply components.

3.8 Conclusion

The specifications for the CIRPEC were decided upon based on research performed on other controllers as presented in Chapter 2. It was established that the DSP as a primary processor and an FPGA as the co-processor was the best configuration for processing control algorithms while still adding to flexibility. The number of analog inputs and digital outputs were determined as well as the individual specifications of the ADCs and DAC. Important to their selection was their sample and update rates as well as the time it took for conversions to complete.

The number of PWM outputs were determined by the application. Ten fibre optic transmitters were placed on-board to allow for the control of a *five*-cell multicell converter. Expansion capability was provided through the use of an expansion header. This will allow for additional ADC or DAC daughterboards to be connected. Communication ports included USB and JTAG ports to interface with a computer.

A block diagram of the controller was included to give an idea of how the various components are linked. The major components were discussed and explained how they fit into the design.

The voltage and current supply requirements were presented as a breakdown of each component. The distribution of the current and voltage was also illustrated. The reset circuitry was presented to illustrate the procedure at startup.

Finally the PCB design and techniques used were presented. These design techniques had to be implemented considering the complexity of the design.

Chapter 4

Firmware Development

4.1 Introduction

This chapter introduces the pulse-width modulation (PWM) control scheme that is necessary for the control of a multicell inverter as well as the firmware implementation on the CIRPEC. Implementations of other firmware modules including half- and full-bridge are presented. The means of configuring the DSP and FPGA for communication as well as configuring the DSP to boot standalone is presented.

4.2 Pulse-width Modulation

This section sets out to explain the theory behind the interleaved switching method necessary to generate the gating signals for the multicell inverter. It will also present the equivalent implementation on the controller.

4.2.1 Interleaved Switching

The interleaved switching method was selected for generating the PWM gating signals of the respective cells as it has harmonic advantages when used with the multicell topology (Wilkinson, 2004). The switching frequency is effectively multiplied by the number of cells in the multicell converter such that for a p-cell converter the switching frequency becomes $p \cdot f_s$ (Wilkinson, 1997). Also, when the successive gating signals are phase-shifted by $\frac{2\pi}{p}$ radians it promotes

the natural balance of the cell capacitor voltages of the multicell converter (Wilkinson, 2004).

Interleaved switching implements sinusoidal modulation. In order to explain the method of interleaved switching it can be simplified for a "single"-cell and then adapted for a p-cell case.

4.2.1.1 *Single*-cell

Interleaved switching for a "single"-cell or half-bridge converter is a good starting point as this is the simplest case. In order to produce a sinusoidal output voltage, a sinusoidal control or *reference* signal (f_r) is required at the desired frequency and gets compared to a triangular or *carrier* signal (f_c) as illustrated in Figure 4.1 (Mohan *et al.*, 2003). This figure only represents a single period of f_r but this sequence continues through time.

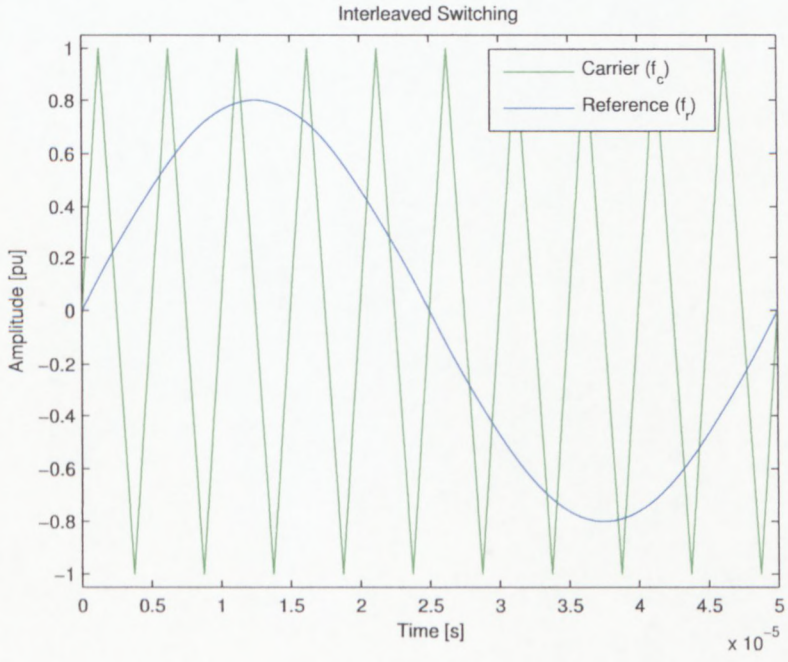


Figure 4.1: Interleaved switching.

The frequency of the f_c signal establishes the switching frequency (f_s). This is the frequency at which the inverter switches within the half-bridge

are switched. The signal's amplitude and frequency remain constant through time. The carrier signal can be expressed as follows: (Wilkinson, 2004)

$$f_c(t) = \frac{2}{\pi} \arcsin\left[\sin(\omega_s t)\right] \tag{4.2.1}$$

The f_r signal is used to modulate the duty cycle of the switches and has a frequency of f_0 which is the desired fundamental frequency of the inverter voltage output (Mohan *et al.*, 2003). The reference signal can be expressed as follows: (Wilkinson, 2004)

$$f_r(t) = m_a \sin(\omega_r t) \tag{4.2.2}$$

The modulation index m_a (also known as the amplitude-modulation ratio) is defined as: (Mohan *et al.*, 2003; Luo *et al.*, 2005)

$$m_a = \frac{\widehat{V}_r}{\widehat{V}_c} \tag{4.2.3}$$

where $\widehat{V}_r$ and $\widehat{V}_c$ denotes the peak amplitudes of the respective signals. Generally linear modulation is considered such that m_a is smaller than unity, typically 0.8 (Luo *et al.*, 2005).

The normalised carrier frequency index m_f (also known as the frequency-modulation ratio) is defined as: (Mohan *et al.*, 2003; Luo *et al.*, 2005)

$$m_f = \frac{f_c}{f_r} \tag{4.2.4}$$

where f_c and f_r are the frequencies of the carrier and reference waveform respectively. Generally m_f is taken as a large value (e.g. $m_f > 9$) in order to achieve low total harmonic distortion (THD) (Luo *et al.*, 2005).

Figure 4.2 illustrates the resulting PWM. The switching function is generated as follows: (Wilkinson, 2004)

$$s_1(t) = \begin{cases} 1 & \text{if } f_r(t) > f_c(t) \\ -1 & \text{if } f_r(t) < f_c(t) \end{cases} \tag{4.2.5}$$

Analysis of the switching function on a *single*-cell, presented in Figure 4.3, is defined as: (Wilkinson, 2004)

$$s_1(t) = \begin{cases} 1 & \text{if } S_{1t} \text{ is closed} \\ -1 & \text{if } S_{1b} \text{ is closed} \end{cases} \tag{4.2.6}$$

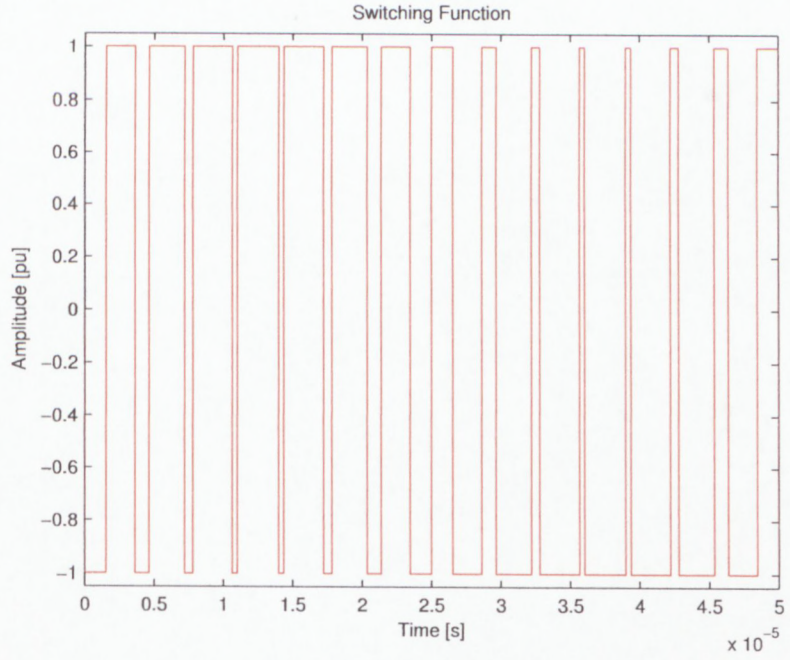


Figure 4.2: Resulting switching function.

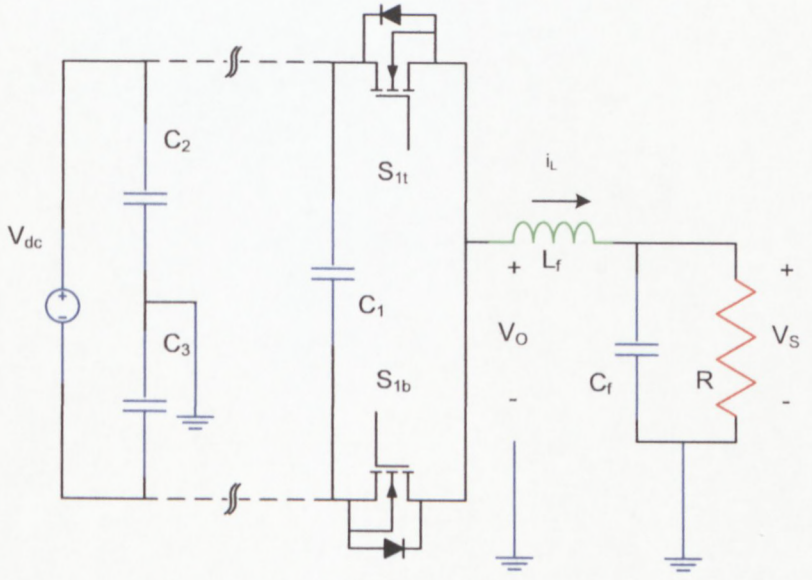


Figure 4.3: *Single*-cell multicell converter.

4.2.1.2 *p*-cell

The *p*-cell multicell is based on the analysis of the *single*-cell model. The difference in the case of a *p*-cell multicell converter is that there are *p* switching functions. The switching function for *p*-cells can therefore be defined as: (Wilkinson, 2004)

$$s_i(t) = \begin{cases} 1 & \text{if } f_r(t) > f_{c_i}(t) \\ -1 & \text{if } f_r(t) < f_{c_i}(t) \end{cases} \quad \text{for } i = 1, \ldots, p \tag{4.2.7}$$

The carrier signals of the *p*-cells are phase-shifted by $\frac{2\pi}{p}$ radians. This implies that a particular carrier of a cell lags the carrier prior to it by $\frac{2\pi}{p}$ radians i.e. the p^{th} cell will lag the $(p-1)^{th}$ cell by $\frac{2\pi}{p}$ radians.

Analysis of the switching function on a *p*-cell, presented in Figure 4.2.1.2, is defined as: (Wilkinson, 2004)

$$s_i(t) = \begin{cases} 1 & \text{if } S_{it} \text{ is closed} \\ -1 & \text{if } S_{ib} \text{ is closed} \end{cases} \quad \text{for } i = 1, \ldots, p \tag{4.2.8}$$

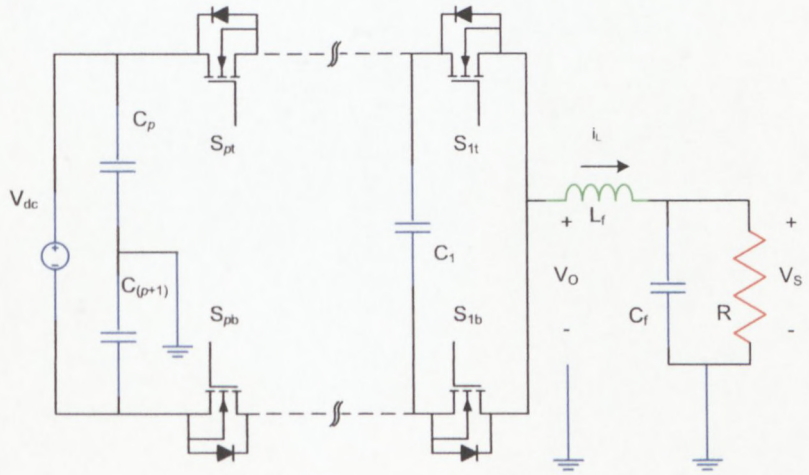


Figure 4.4: *p*-cell multicell converter.

Finally, a "total" switching function can be defined as the scaled sum of all the switching functions: (Wilkinson, 2004)

$$s_t(t) = \frac{1}{2}\left\{s_1(t) + s_2(t) + \ldots + s_{(p-1)}(t) + s_p(t)\right\} \tag{4.2.9}$$

4.2.1.3 *Five*-cell

Considering the *five*-cell is the primary end application, the interleaved switching method waveforms are presented. Since there are five cells, each carrier signal needs to be phase shifted by $\frac{2\pi}{p}$ radians, where $p = 5$, thus $\frac{360°}{5}$ degrees which results in a phase shift of 72°.

Figure 4.2.1.3 presents the interleaved switching waveform. The f_r signal, which is configured at 20 kHz, is compared against the five phase shifted f_c signals. The f_c signals have been configured at 50 kHz to better illustrate the waveforms. Ideally the frequency of the f_c signals should be much higher than the f_r signal.

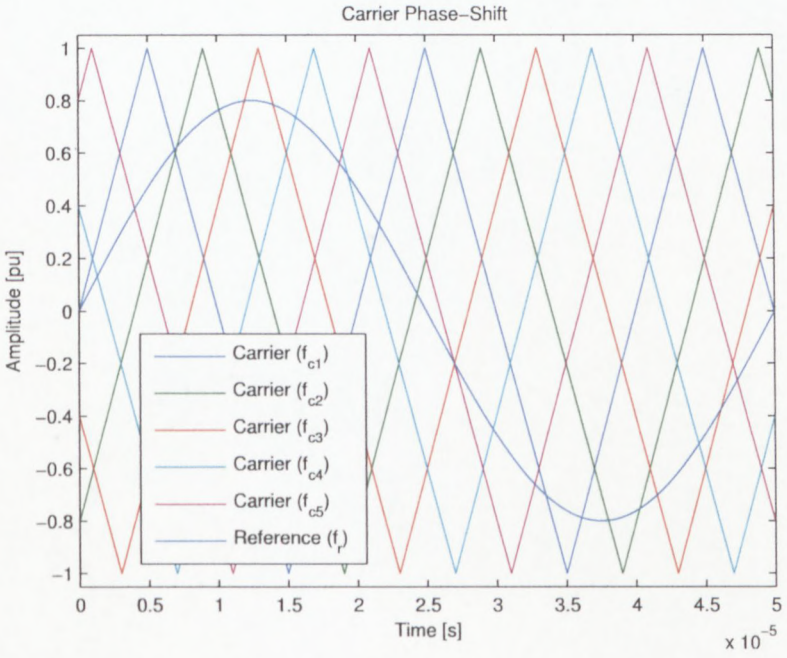


Figure 4.5: Interleaved switching waveforms for a *five*-cell converter.

The interleaved switching and resulting switching functions of the five waveforms are illustrated on subplots in Figure 4.2.1.3. The frequency of the f_c signals have been configured at 200 kHz. There is only one f_r signal but has been placed in each subplot for illustrative purposes.

The *total* switching function which is the sum of all the switching functions is illustrated in Figure 4.2.1.3.

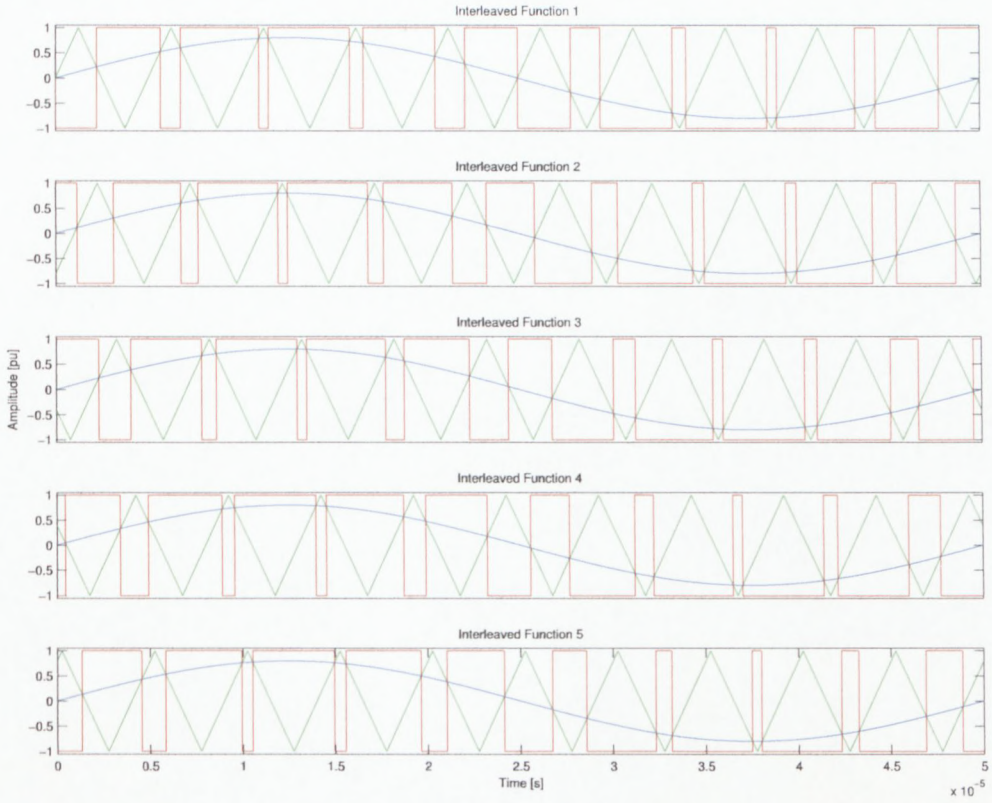


Figure 4.6: A subplot of the interleaved switching method and resulting switching function waveforms for a *five*-cell multicell converter.

4.3 Firmware Modules

From Section 4.2.1 it was determined that a sinusoidal reference, f_r, and triangular carrier, f_c, signals were necessary to generate the PWM that is required for the control of a multicell converter.

The frequency of f_r is determined by the end application. As mentioned in the introductory chapter of the document was that the multicell converter would be used to drive an ultrasonic transducer. When driving the ultrasonic transducer it is necessary to drive it at a frequency that is close to its main resonance mode frequency. This was previously determined to be 22,1 kHz.

Due to the real-time processing capability of the DSP and the fact that feedback would need to be implemented it was decided to generate the f_r signal on the DSP. This implementation would allow the controller to more accurately track the resonant frequency of the ultrasonic transducer as it is

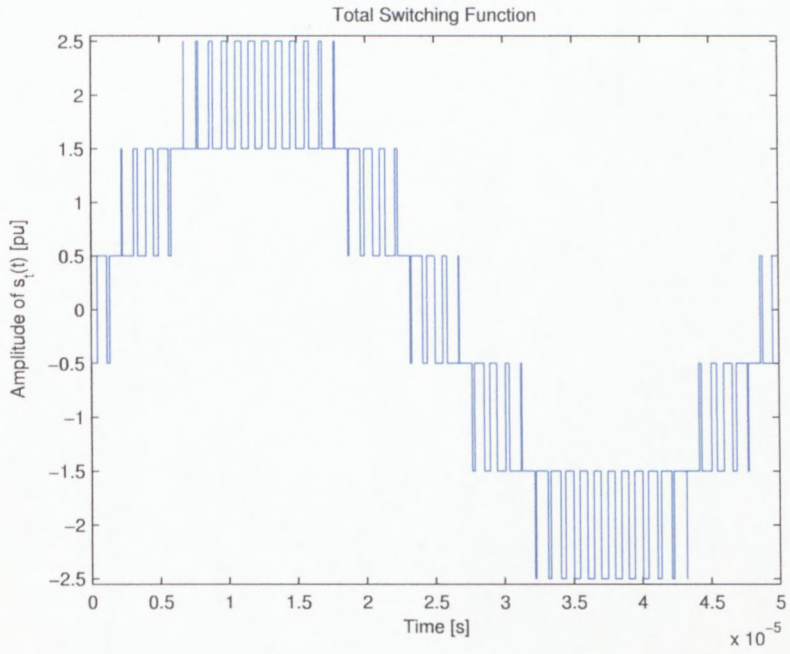


Figure 4.7: *Total* switching function of a *five*-cell multicell converter.

---

much easier to adjust the sinusoidal frequency on the DSP.

The frequency of the f_c signal was chosen to be a number of times larger than the f_r signal (300 kHz) so as to prevent aliasing. Other reasons includes harmonic advantages where the switching frequency of the multicell converter, f_s, is effectively multiplied by the number of cells.

The triangular f_c signals were decided to be implemented on the FPGA as a counter could relatively simply be implemented in logic to generate the triangular signal.

Before the implementation of the PWM code could be started there had to be communication between the DSP and the FPGA. This is explained in the following section.

4.3.1 Configuring the DSP for Transfers

The FPGA has been configured as an asynchronous memory with respect to the DSP. This implies that it must be addressed according to the *memory map* of the DSP. Appendix C.1 includes the memory map of the DSP. The FPGA can therefore be addressed between the memory regions 0×9000 0000

and 0×9FFF FFFF. Various registers had to be configured correctly in order for the DSP to operate properly and secondly to be able to interface with the FPGA.

One of the first registers that needed to be configured on the DSP was that of the PLL. This register is used to configure the operating clock frequency of the device as well as configure the clock frequency of the external memory interface (EMIF). The EMIF clock determines the read and write frequency between the DSP and the FPGA, SDRAM and Flash memory. As mentioned in the hardware section 3.4.9 this is the same clock that gets routed to the clock driver IC which then redistributes the clock to the FPGA and the SDRAM. A 25 MHz oscillator is routed to the CLKIN pin of the DSP. The internal PLL then multiplies that frequency to achieve a frequency of 200 MHz which is then used for the system clock frequency after going through a divide-by-one divider. The initial 200 MHz clock is then divided by two after going through a divide-by-two divider in order to achieve a frequency of 100 MHz for the EMIF clock.

The asynchronous memory register (*A1CR*) also needed to be configured in order to be able to communicate with the Flash memory and the FPGA. Important register values included configuring the EMIF data bus for 16-bit operation. The *strobe select mode* bit was configured for select strobe so that the EM_$\overline{WE}$_DQM pins function as byte enables and pin $\overline{\text{EM\_CS[2]}}$ acts as a strobe. The *extended wait mode* was disabled since the C6720 does not have a EM_WAIT pin. The remaining bits configured were that of the timing which included the write and read setup, strobe and hold times. The *TA* bit determines the minimum turnaround time between read and write cycles. This setting helps to avoid contention on the bus.

The SDRAM configuration register (*SDCR*) also had to be configured. The *self refresh mode* bit (*SR*) has been disabled to prevent the EMIF from issuing the self refresh command to the SDRAM. The *narrow mode* bit *NM* has been configured for a 16-bit bus. CAS latency (*CL*) has been configured for 2 *EM_CLK cycles*. The *internal bank size* (*IBANK*) field has been configured for a 4-bank SDRAM device while the *page size* (*PAGESIZE*) has been configured for 512 elements per row to conform to the SDRAM connected. The SDRAM timing register (*SDTIMR*) was used to configure many of the timing parameters.

The dual data movement accelerator (dMAX) controller is configured to transfer data between the internal data memory controller and the device peripherals of the C6720 including external memory. It has the capability to perform one-, two- and three-dimensional transfers including two concurrent requests provided they are in different sources/destinations. A block diagram of the C6720 is provided in Appendix C.1. In order to define the transfer parameters an event entry needs to be configured. An event entry table exists for events of both high- and low-priority events, each table has 32 entries starting from word address zero in the parameter RAM (PaRAM). The event used for transfer has been configured for a general purpose 16-bit 1-D transfer. It has been configured to generate an interrupt on completion of the transfer and also to reload the active element counter, source (*SRC*) and destination (*DST*) addresses when the transfer is complete. There are also two transfer entry tables, each capable of holding eight transfer entries. One is associated with the high-priority events and the other low-priority events. The transfer entry table is used to configure and update the *SRC* and *DST* address as well as the transfer counter value. Also configured is the reload addresses for *SRC* and *DST*.

4.3.2 DSP Implementation

Figure 4.3.2 illustrates the code flow of the DSP. After initialising the chip support library (CSL) and configuring the dMAX mentioned in Section 4.3.1 the program enters the function to generate the sine waves. In order for the PWM design to be able to adjust the frequency of the f_r signal it was decided to generate a few sine wave tables of incremental lengths, although with identical amplitudes, on the DSP. When the FPGA runs through the table the influence of a longer table would result in a lower frequency.

It was decided to implement the frequency adjustment on the DSP rather than generating a large table and using the FPGA to determine the frequency as the DSP method provides better accuracy. The FPGA implementation would result in too much of a frequency step for each adjustment. The DSP method can also be implemented to focus more on a particular frequency to provide a smaller frequency step.

The *sinsp* function was used to generate the sine waves. This function returns the sine of a floating-point argument. The sine wave output has been

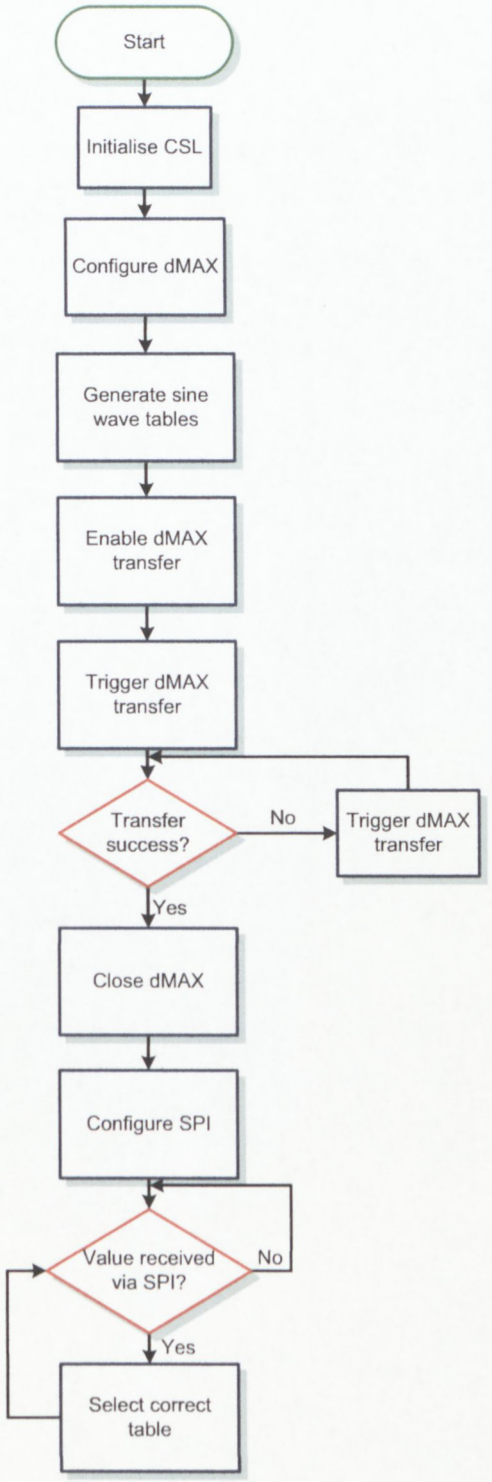


Figure 4.8: DSP flow chart.

configured as a *signed* 16-bit word implying the output value swings between positive and negative values.

After the sine tables have been generated and stored in internal memory of the DSP they can be transferred to the FPGA using the dMAX. A RAM block has been configured in the FPGA to accept the values written by the DSP. All the registers for the dMAX were configured at the beginning of the code including the source, which is the start location of the sine tables, and the destination, which is the address of the FPGA configured as 0×9000 0000 as depicted in the memory map for the DSP. The FPGA could have been configured for any region within the addressable space of the DSP for asynchronous memories which is likely to be the case if the Flash memory is used.

The dMAX transfers the data as a 1-D block to the FPGA and is much faster than transmitting one table value at a time.

As soon as the dMAX transfer is complete a read of the destination is performed to verify the values in the FPGA's RAM block. This is performed on a value-by-value basis and if there is a discrepancy with regard to the value in the internal memory of the DSP and the FPGA then the dMAX transfer is repeated. If the transfer is a success then the dMAX transfer handle is closed.

Apart from the values of the sine tables, an index value (*indexValue*) and table length value (*tableLength*) are transferred to the FPGA's RAM block. Using this information the FPGA can determine the location of each table in its memory as well as the length of table so that it can wrap the waveform.

To perform basic feedback, a microcontroller was connected externally to the CIRPEC to emulate a controller at the ultrasonic transducer end. The microcontroller was used to determine whether to adjust the frequency of the f_r signal or not. This was implemented by means of the microcontroller transmitting a value, using the SPI protocol, via fibre optics to the CIRPEC. The value corresponded to the *indexValue* used in the DSP code. This value was then used to select a more suitable reference frequency.

Since the fibre optic receivers are connected to the FPGA the signals had to be routed internally to the relevant DSP pins. The SPI module would have already been configured on the DSP and be running in an infinite while loop checking if there was any activity on the SPI module by polling the SPI module flag bit *SPIFLG*. Once a value was received and it was within the bounds of

the table index values then the *indexValue* and *tableLength* values would be updated in the FPGA RAM. This would automatically update the table that the FPGA points to and will now run the newly selected table and subsequent frequency.

The program then continues to poll the *SPIFLG* for activity.

4.3.3 FPGA Implementation

All FPGA code was programmed in VHDL using the Quartus® II software from Altera.

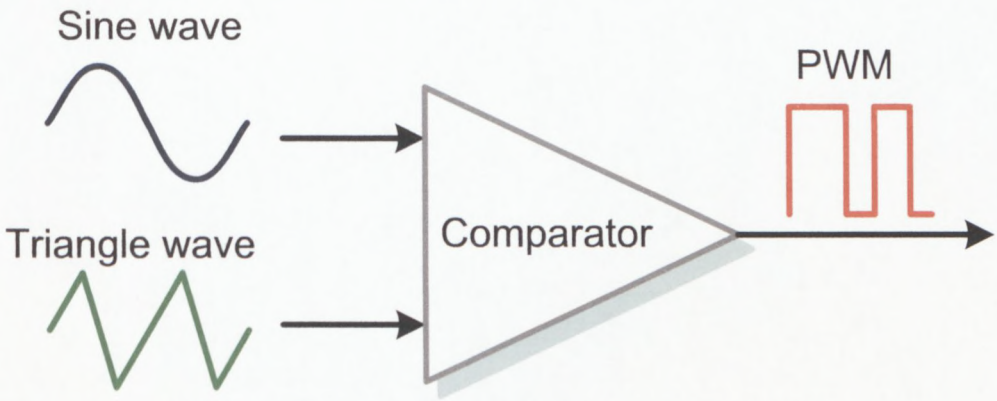


Figure 4.9: Basic blocks necessary to generate PWM.

Figure 4.3.3 illustrates the basic process and blocks necessary to generate the interleaved PWM method in hardware. Apart from generating the sine wave, all the logic represented in the figure is generated on the FPGA. The reason for this was due to the fact that the remaining logic could relatively simply be implemented using logic elements.

The triangular signals, f_c, were generated by configuring an up/down counter. The counter was then also given a positive and negative *state* to allow the signal to swing negative. Each triangle signal could therefore be configured to have an initial count value as well as a direction, either up or down, and an initial state, either negative or positive. This is illustrated in Figure 4.3.3. A sawtooth could also have been used but a triangle waveform provides better harmonic performance (du Toit & Enslin, 1996; du Toit *et al.*,

1997). The sawtooth can be used where higher switching frequencies are required.

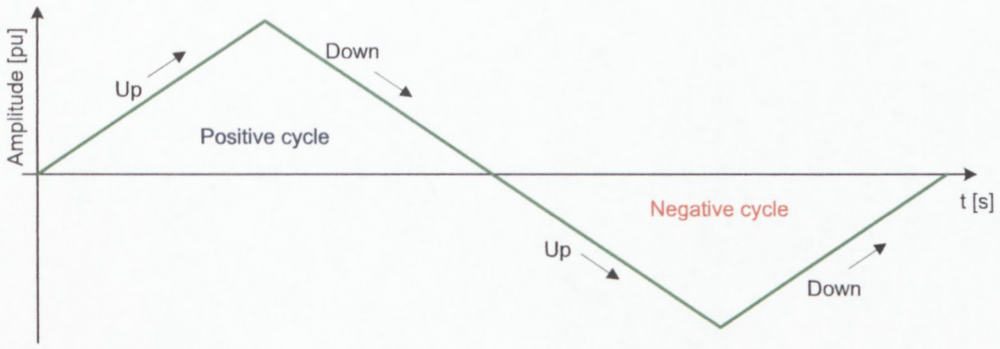


Figure 4.10: Triangle positive and negative cycle.

Calculations to determine the maximum count value for f_c are determined as follows: The switching period is defined as:

$$\begin{aligned} f_s &= 300 \text{ kHz} \\ \therefore T_s &= 3.\dot{3} \ \mu\text{s} \end{aligned} \tag{4.3.1}$$

One FPGA clock period is defined as:

$$\begin{aligned} f_{clk} &= 100 \text{ MHz} \\ \therefore T_{clk} &= 10 \text{ ns} \end{aligned} \tag{4.3.2}$$

The maximum count value and number of bits required for the counter are determined by:

$$\begin{aligned} Count &= {}^{T_s}/_{T_{clk}} \\ &= {}^{3.\dot{3} \ \mu\text{s}}/_{10 \text{ ns}} \\ &= 333.\dot{3} \approx 332 \text{ units} \end{aligned}$$

which equates to a 9 bit value.

Considering the triangle has four quadrants:

$$\begin{aligned} Count &= {}^{332}/_{4} \\ &= 83 \end{aligned}$$

which equates to a 7 bit value.

An external 40 MHz clock is provided to the internal PLL of the FPGA which then multiplies it to generate a system clock frequency (f_{clk}) of 100 MHz.

An advantage of using the negative and positive cycle method is that the number of bits required to produce the f_c signal are reduced. Although, to maintain the negative information of the value a signed bit needs to be added.

A comparator block is then implemented to determine when the f_r signal is greater than the f_c signal. The values of the sine wave are passed to the comparator block from the RAM block.

A dual-port RAM (DPRAM) block as illustrated in Figure 4.3.3 has been implemented on the FPGA as a means to store the sine table values generated by the DSP. The DSP uses the data (*data_a*) and address (*address_a*) bus signals in conjunction with the write enable (*wren_a*) and byte enable (*byteena_a*) control signals in order to write to the RAM block. The DSP also had to have the ability to read from the RAM block most notably in order to verify the values transferred. Since the data signals that are used for read and write use the same physical bus the data bus pins (*dsp_ed*) needed to be tri-stated. The EMIF clock sent from the DSP is used to latch the data into the RAM.

The FPGA has access to the RAM block using separate data, address and write enable signals. The FPGA pulls the *wren_b* signal low to read from RAM. The output from the RAM is a 16-bit value received via signal *q_b* which gets passed to the comparator. The FPGA uses an internal clock to determine the read rate. The rate at which the read clock runs and the number of values that constitute a sine wave determine the frequency of the sine wave.

The variables *indexValue* and *tableLength* are also stored in RAM. The *indexValue* is used to point to the correct address in the RAM. The *tableLength* variable is then used to clear a counter which is used to incrementally read from the RAM until the length of the table is reached.

Both the current sine wave and triangle wave values are compared using a logic *compare* block. Both values are signed values. The output is a logic high when the sine wave value is larger than the triangular value and a logic low when the inverse is true.

The output of the compare block is the resulting PWM. The PWM is then

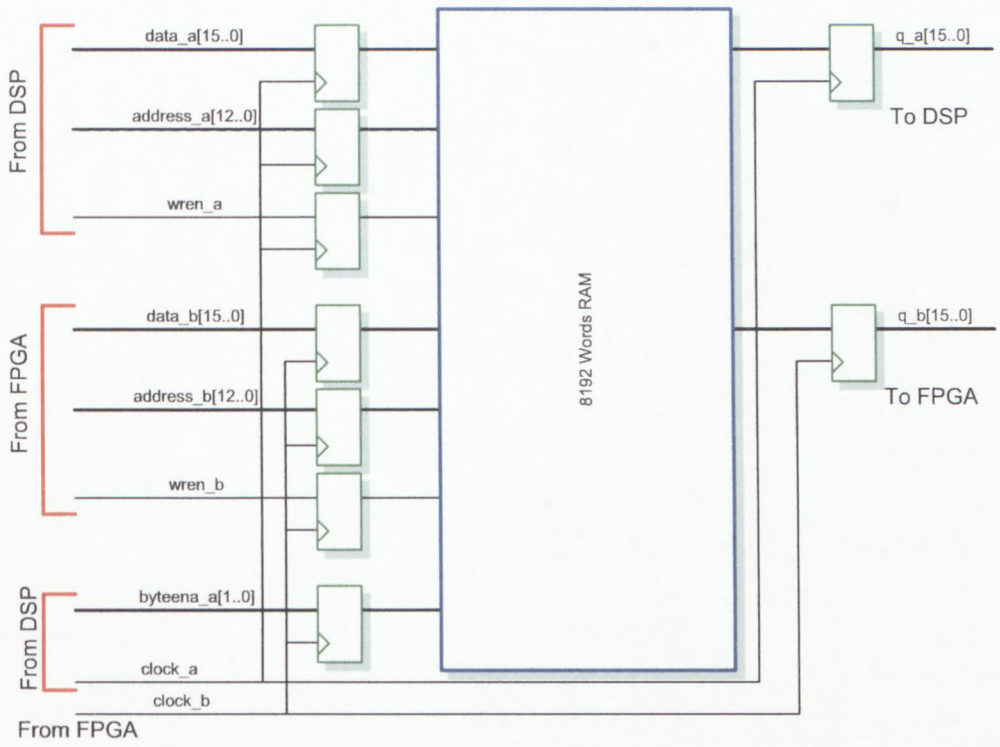


Figure 4.11: Dual-port RAM block implemented on FPGA.

routed out the FPGA through to the fibre optics.

4.3.4 Modulator

4.3.4.1 *single*-cell Implementation

Figure 4.3.4.1 illustrates the PWM blocks required for a *single*-cell of a multicell converter. Evidently there are two PWM output signals which are generated from a single PWM output. This is due to the fact that each cell has two MOSFETs and they need to be in complementary states at all stages.

In order to generate the complementary logic, the PWM signal that is output from the *compare* block is split and one signal routed through an inverter block. This will result in the output having the inverted logic level with reference to the input. Both the PWM output and the inverse PWM output are routed to a *dead time* block.

Dead time is necessary to compensate for rise and fall times of the MOSFETS in each cell. Usually the fall times, T_f, are larger than the rise times, T_r,

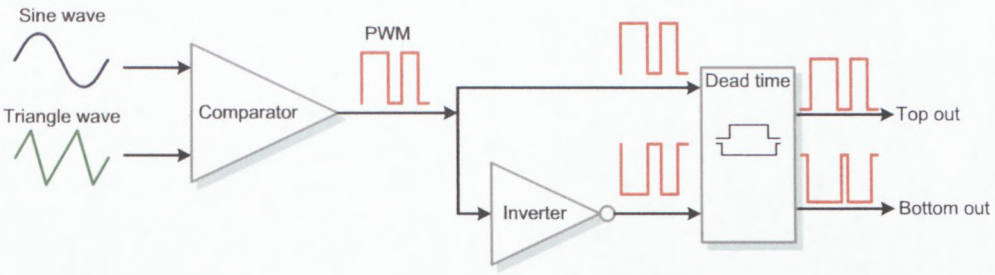


Figure 4.12: PWM blocks required for a *single*-cell.

which implies that the signal that is transitioning from a logic high to a logic low needs to transition first, this is illustrated in Figure 4.3.4.1. Although, implementing this in logic is not as trivial as it may sound. The dead time block has been divided into two processes, one handles the top and the other the bottom PWM signal. The implementation is identical.

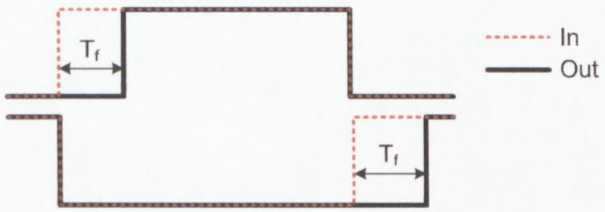


Figure 4.13: Dead time implementation.

When a transition of the input PWM signal to the dead time block is detected from low to high, a variable is loaded with a value. The value of the variable is the period of dead time required. Each MOSFET is different and the datasheet should be consulted. This value can range from low nanoseconds to microseconds depending on the application. Each clock pulse decrements the variable until its value is zero, at this time the output will be allowed to transition to high as depicted in the figure. It will remain high until there is a transition from high to low.

When there is a transition from high to low, the signal is allowed to immediately transition to the low state. The inverse signal will remain low until its dead time period is finished. Important to note is that at no stage are both the signals high as this would result in both MOSFETS conducting which could have destructive consequences.

The PWM now incorporates dead time and is output from the dead time block. The PWM signals are then routed out via the fibre optic transmitters.

4.3.4.2 *p*-cell Implementation

In the case of a p-cell implementation the only difference is that there are p triangle signals needed. These also need to be phase-shifted by $\frac{2\pi}{p}$ degrees. This is implemented by configuring the initial value, cycle; whether positive or negative, and state; whether up or down, for each triangle wave. A MATLAB® script (Appendix D.1) was developed to provide the information required. Calculating this information by hand can be error prone and time consuming as there are many variables to consider. For a p-cell multicell module there are $p\times$PWM outputs each with their complementary pairs.

4.3.5 Half-bridge

The half-bridge implementation is identical to the *single*-cell multicell implementation. There is a single f_r and f_c signal which are compared to each other. The resulting PWM needs to have a complementary signal.

4.3.6 Full-bridge

The full-bridge module was implemented using the *two*-cell module. This module has a single f_r signal which is then compared against two f_c signals that are phase shifted by 180° with respect to each other. This produces a three-level output voltage.

4.4 Booting the DSP

In order to get the CIRPEC to boot standalone, i.e. to be able to boot without being connected via an emulator, the bootloader needed to be used. In Section 3.4.1.1 the hardware of the bootloader was explained. To recap, the bootloader resides in the internal ROM of the DSP starting at address 0×0000 0000. When power is connected to the device or a reset takes place, the program counter is set to the start of the ROM address and execution of the bootloader commences. The CIRPEC has been configured to boot from

an external I^2C device in master mode. This means the bootloader will read the data from the I^2C device when it is required.

The bootloader checks on the I^2C bus address 0×50 for a *magic word* (0×41504954). If the magic word is not found boot mode fails. If it is correct the bootloader expects the data to be in the application image script (AIS) format. AIS is an application image transfer format developed by Texas Instruments. The bootloader will continue to read from the I^2C bus until the AIS *JUMP_CLOSE* command is encountered.

Texas Instruments provides two software utility tools to assist in creating the AIS data stream, namely *genBootCfg* and *genAIS*. The utility *genBootCfg* is a Perl TK GUI which assists in configuring the PLL, EMIF SDRAM and ASYNC RAM as well as GPIO pin configuration for extended Parallel Flash booting. The genBootCfg tool produces *.c* and *.cfg* outputs. The *.cfg* file including the application object *.out*, generated by the CCS compiler, are used to generate the AIS boot file. The genAIS tool is used to generate the AIS boot format file in ASCII format.

After generating the AIS file the image can be programmed onto the I^2C EEPROM device on the CIRPEC. This was done using the *progI2C.c* program developed by TI. The whole operation is performed using the USB Emulator while compiling and running the program in CCS.

The program initially scans the AIS image into memory before it starts programming on the I^2C line using the address sequence of '1010000' or 0×50. Once the AIS image has been programmed onto the EEPROM the program attempts to read the information back to verify its validity. A message is displayed if there is an error alternatively the program completes.

The AIS image now exists in the EEPROM ready for booting off when the power and USB Emulator are removed and the power reconnected to the board. The correct jumper configuration also needs to be set to boot from the I^2C device.

4.5 Conclusion

In this chapter the theory behind the interleaved PWM scheme used to generate the gating signals for a multicell converter was introduced. Cases for a *single*-cell, *p*-cell and the end application of a *five*-cell converter were ex-

plained. Implementation of the firmware on the DSP and FPGA was then discussed, starting off with the configuration of the devices to allow communication. The firmware modules presented included a description of a *single*-cell modulator and the differences compared to the implementation of a p-cell modulator. Implementation of the half- and full-bridge modules were provided and explained how they could be implemented as variants of the multicell modules. The steps followed to get the DSP to boot standalone were explained. This included writing an image file to the EEPROM device. This file contains all the information necessary to configure the DSP registers and start execution of the program.

Chapter 5

Results

5.1 Introduction

The results of the various firmware modules developed are presented in this chapter, along with the corresponding simulations generated in MATLAB® to verify their accuracy. An experimental setup which included the CIRPEC being used to control a multicell inverter was established. The experimental test was used to determine whether the CIRPEC was capable of controlling a multicell inverter and presenting the results thereof. The experimental results are presented in this chapter. Since most of the results in this chapter are presented using MATLAB®, the notation used was to plot simulated results in red and experimental results in blue.

5.2 Experimental Setup

A description of the experimental setup is provided in this section. This will provide the reader with an idea of where in the experimental chain the various results presented were obtained.

A graphical illustration of the test setup is presented in Figure 5.1. A computer was required to run all the relevant software necessary for programming the DSP and FPGA. This included Code Composer Studio (CCS) and Quartus® software packages for the DSP and FPGA respectively. MATLAB® was used to simulate the PWM implementation. When the measured results of the PWM were obtained, they were then verified using the simulated results.

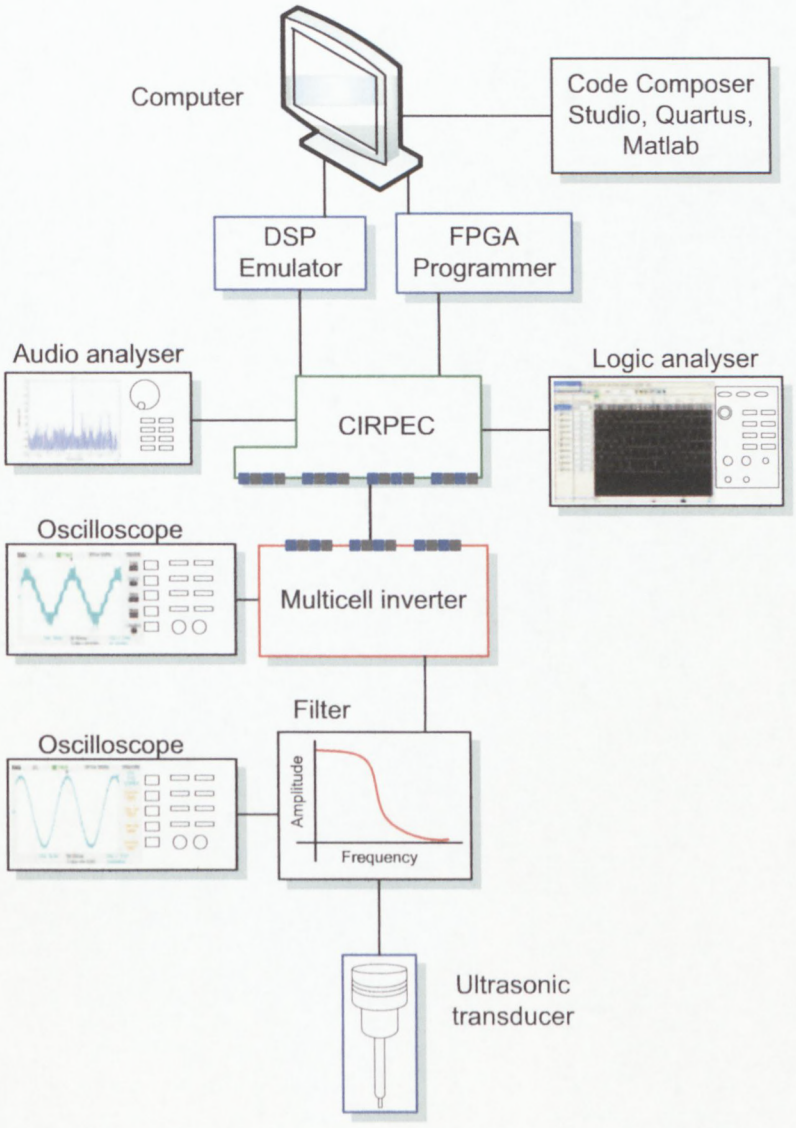


Figure 5.1: Experimental setup.

In order to program and debug the DSP and FPGA a piece of hardware was required, in the case of the DSP it is an emulator and for the FPGA a USB Blaster cable. The DSP emulator is manufactured by Spectrum Digital while the FPGA USB Blaster by Altera. In respect of the DSP, the emulator is a very useful debugging tool as it allows the programmer to step through code and view register and memory values. The CCS development environment therefore provided the first means of knowing whether the code was operating as it was intended.

A logic analyser was used to capture the digital PWM waveform as output by the CIRPEC controller. A logic analyser is ideal for measuring and debugging digital signals. It is capable of capturing a long period of results at a very high sample rate and provides a clear indication of the logic level of the signals; represented as either a *one* for a logic level HIGH or a *zero* for a logic level LOW. Another advantage of the logic analyser was the ability to capture multiple signals using the many available probes, this allowed for a whole signal bus to be monitored which made debugging easier.

An audio analyser was used to determine whether the fundamental frequency of the PWM signal was at the frequency it was designed for, in the case of the firmware modules the desired frequency was 22.1 kHz. In order to do so a single PWM signal needed to be measured by the audio analyser. The audio analyser plots a fast fourier transform (FFT), which displays the frequency content of the signal. This allows for the harmonics of the signal to be analysed. The amplitude of the fundamental should be much larger than any of the other harmonics in the frequency spectrum.

The PWM signals generated by the firmware module on the CIRPEC are sent via fibre optic connections to the multicell inverter. An oscilloscope was used to measure the unfiltered voltage providing a stepped sinusoidal signal. Another measurement was taken after the filter providing a smooth sinusoidal signal. This is the driving signal that is used for the ultrasonic transducer.

The measurement equipment used in the experimental setup is listed in Table 5.1.

Table 5.1: Measurement equipment.

Measurement equipment	
Oscilloscope	Tektronix TDS2024B
Logic analyser	Agilent Technologies 1683A
Audio analyser	Rohde & Schwarz UP350

5.3 Experimental Results

5.3.1 *single*-cell and Half-Bridge

The MATLAB® simulations for the *single*-cell scenario were presented in Section 4.2.1.1 of Chapter 4, however, the resulting PWM signal is repeated in Figure 5.2 for convenience. The PWM for the half-bridge is generated the same way as that for the *single*-cell. A single reference (f_r) signal is compared against a single, much higher frequency carrier (f_c) signal resulting in a duty cycle resembling a sinusoidal signal as can be seen in the figure. The figure only presents the resulting PWM for a *single* period of the f_r signal.

The resulting PWM from the firmware module was captured using the logic analyser in the form of a comma separated values (CSV) file and exported into MATLAB®. A MATLAB® script (Appendix D.2) was written to read in all the values from the CSV file and extract the data necessary for a single iteration of the reference signal. This data was then plotted and is presented in Figure 5.3.1.

A comparison can now be made by analysing the simulated and experimental results obtained. Through analysis it can be seen that the results obtained from the firmware module were desirable. The amplitude of the two graphs differ since the simulated signal is derived from a previously defined equation which is repeated for convenience in equation (5.3.1).

$$s_1(t) = \begin{cases} \quad 1 & \text{if } f_r(t) > f_c(t) \\ -\ 1 & \text{if } f_r(t) < f_c(t) \end{cases} \tag{5.3.1}$$

Where $s_1(t)$ represents the switching function for a single *cell*, f_r the reference frequency and f_c the carrier frequency. It states the upper limit of the

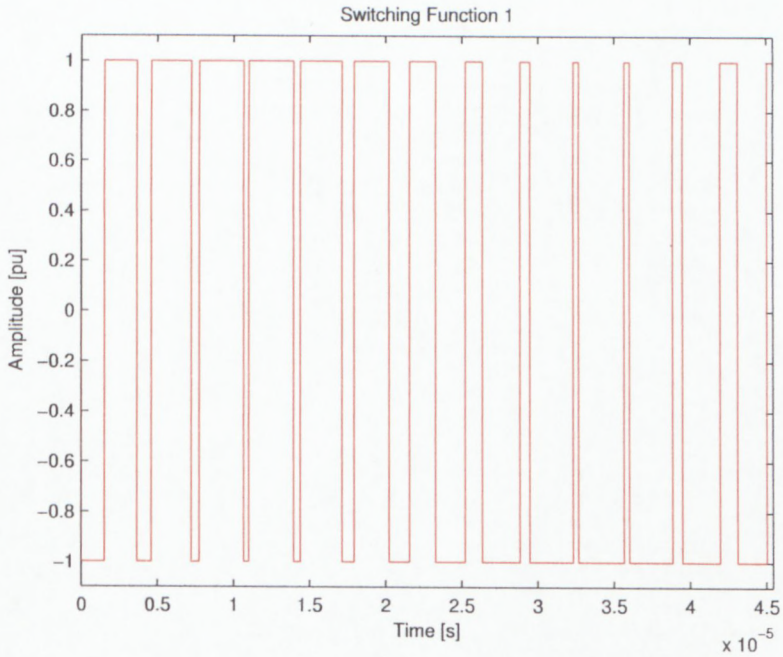


Figure 5.2: Resulting PWM of a single gating signal.

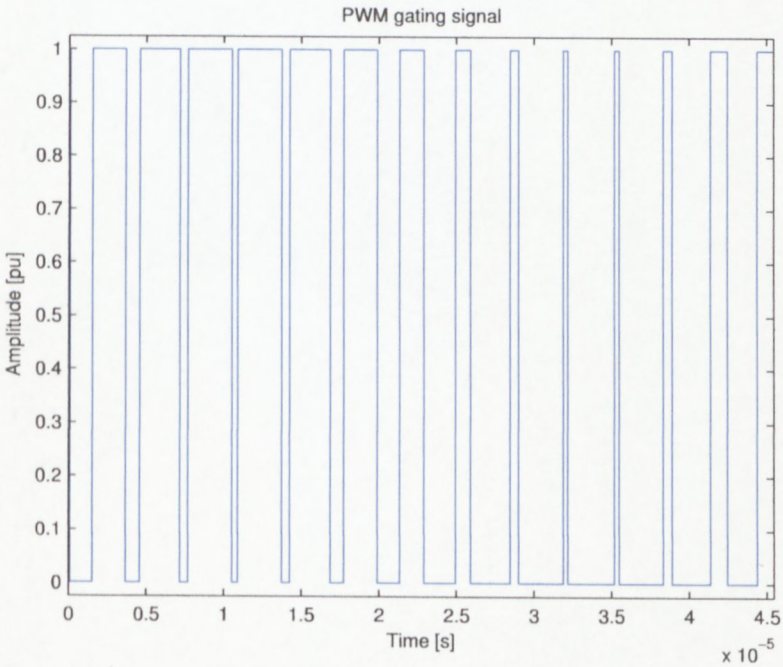


Figure 5.3: PWM gating signal generated by the *single*-cell firmware module.

function has a value of '1' when the reference signal is greater than the carrier signal and the lower limit of the function is '-1' when reversed.

The same comparison of the f_r and f_c signals occur within the FPGA but since it is a digital logic signal the upper limit has a value of '1' representing a *high* and a lower limit of '0' representing a *low*.

The firmware module does not only output a single PWM signal since a *single*-cell inverter consists of two switching devices. These switching devices must always be in opposite states; when the top switch is conducting, the bottom switch is blocking current and vice versa. Figure 5.3.1 presents the PWM signal as presented in Figure 5.3.1 along with its complementary pair. The complementary signal is generated by inverting the PWM gating signal. Dead time is also included in the PWM signals.

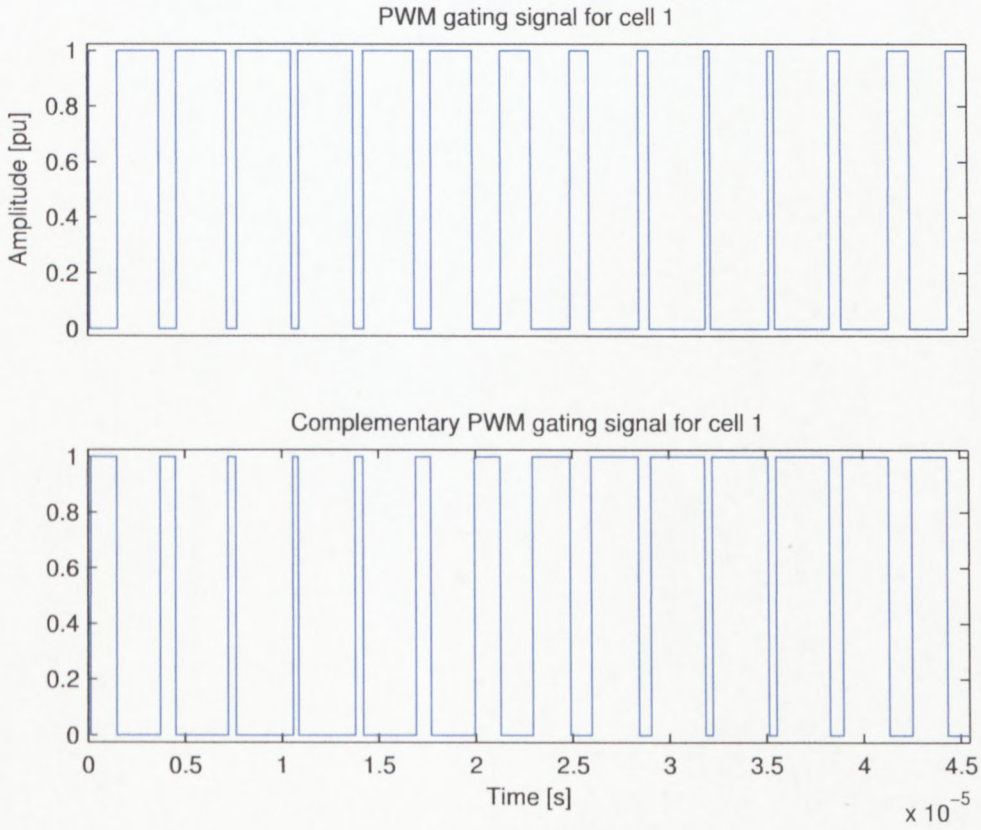


Figure 5.4: PWM gating and complementary gating signal generated by the *single*-cell firmware module.

In order to generate the PWM, the f_c signal was chosen to be 300 kHz while the f_r was selected to be in the region of 22 kHz. Figure 5.3.1 illustrates the resulting fast fourier transform of the PWM output signal. This measurement was captured using the audio analyser by measuring a single PWM gating signal output. The data obtained from the audio analyser was also output as a CSV file. It was therefore necessary to plot the values in a MATLAB® figure to present the data more appropriately.

Analysing the figure it is clear that the fundamental frequency was around the previously specified 22 kHz. The FFT was taken of the first frequency in the index of sine tables on the FPGA, i.e. where *indexValue* was equal to *one*. Beyond that frequency feedback would seek to select a more appropriate frequency by cycling through the remaining sine tables by incrementing the *indexValue*. The noise floor of the other frequency components is low with respect to the fundamental which is what is desired.

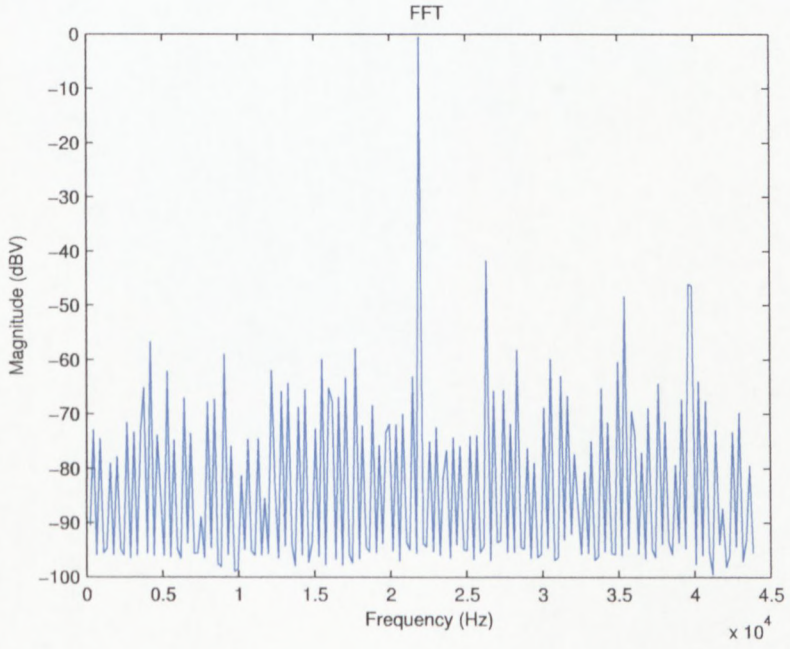


Figure 5.5: Measured FFT of a single PWM gating signal.

The previous chapter mentioned the importance of dead time in the PWM gating signals and the destructive impact it could have if omitted. To recap,

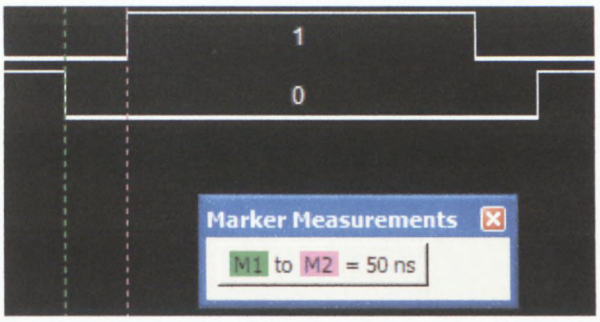


Figure 5.6: Measured dead time.

if both the MOSFETs of a single cell were conducting at the same it would result in a short-circuit which will damage the inverter.

Figure 5.3.1 illustrates the measured dead time that exists for every time there is a transitioning logic level. This figure was captured by the logic analyser and a screenshot was taken thereof. In the case of the figure the dead time was configured for five clock cycles or 50 ns, which is what was measured.

The dead time is programmable by changing a value in the code. The reconfigurability of the FPGA is highlighted in this instance since each switching device type, e.g. MOSFET or IGBT etc., has different characteristics when it comes to rise and fall time requirements; based on the semiconductor itself. By being able to alter a value in the code means no hardware components need to be changed to facilitate a different semiconductor device.

5.3.2 *two-cell* and Full-bridge

The full-bridge module is implemented using the same principle as that of the *two*-cell multicell module. In order to generate the PWM gating signals, a single f_r signal is compared against two f_c signals each phase-shifted by $\frac{360°}{2}$ or 180 degrees. Two f_c signals are required since there are two cells. In the case of a *two*-cell multicell inverter there are four switching devices, two per cell, meaning that four PWM signals are required. The PWM gating signals generated in MATLAB®, without the complementary pairs, are presented in Figure 5.3.2.

The PWM gating signals, also minus their complementary signals, generated by the firmware module for a single f_r iteration are presented in Figure 5.3.2 to provide a comparison with the simulated results. Analysis of the simu-

lated and experimental results illustrate a desired output for the PWM gating signals for the *two*-cell firmware module. The 180 degree phase-shift between the gating signals is evident.

Figure 5.3.2 presents the complete output from the *two*-cell multicell inverter firmware module. Four PWM gating signals are generated since there are four MOSFETs. The PWM gating signals presented in Figure 5.3.2 are listed as PWM gating signal for cell one and cell two respectively. These signals will be used to control the top MOSFETs of their respective cells and the complementary signals the bottom MOSFETs.

The total switching function of the *two*-cell and half-bridge module will result in a waveform as illustrated in the MATLAB® simulated Figure 5.3.2. To recap, the total switching function is the sum of all the switching functions. It results in a three level output. This is the waveform that will be seen at the output of the multicell inverter before filtering takes place.

5.3.3 *three*-cell

Another firmware module developed was that of a *three*-cell multicell inverter module. As with the two prior firmware modules the same approach was followed to generate the PWM gating signals, namely using the interleaved switching method.

Three cells imply that three f_c signals are necessary. The f_c signals are phase-shifted by $\frac{360^\circ}{3}$ or 120 degrees. The carrier signals are then compared to a single f_r signal. When the f_r signal is greater than the f_c signal the PWM signal is represented as a *high*. The PWM signal is represented as a *low* when the opposite is true.

In order to verify the firmware results obtained, MATLAB® was once again used to simulate the PWM signals. The switching function waveforms for the *three*-cell multicell inverter as simulated in MATLAB® are illustrated in Figure 5.3.3.

The PWM gating signals generated by the firmware module were captured using the logic analyser. The data was then imported into MATLAB® where it was plotted. Figure 5.3.3 presents the PWM gating signals without their complementary signals in order to perform a comparison with the simulated results. A comparison of the two figures yields a favourable result.

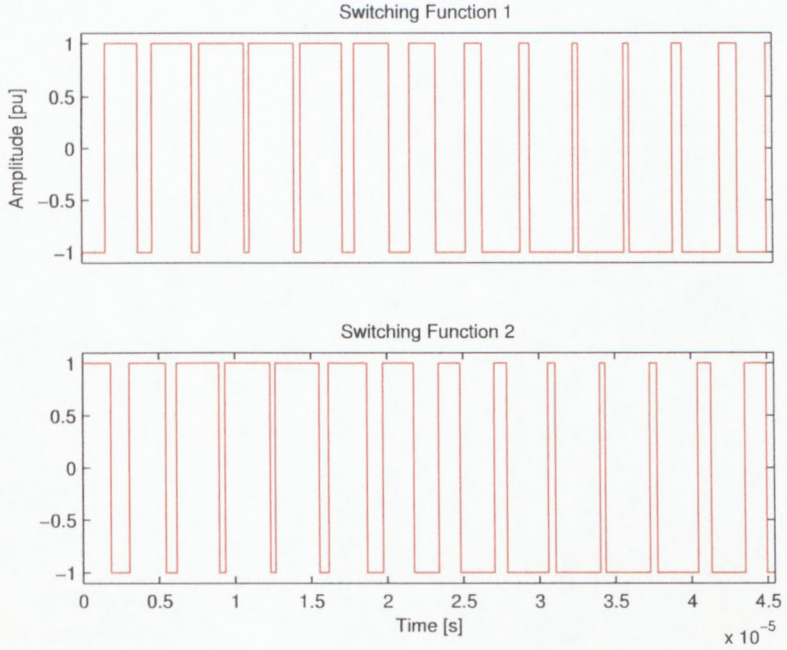


Figure 5.7: Switching function of a *two*-cell simulated in MATLAB®.

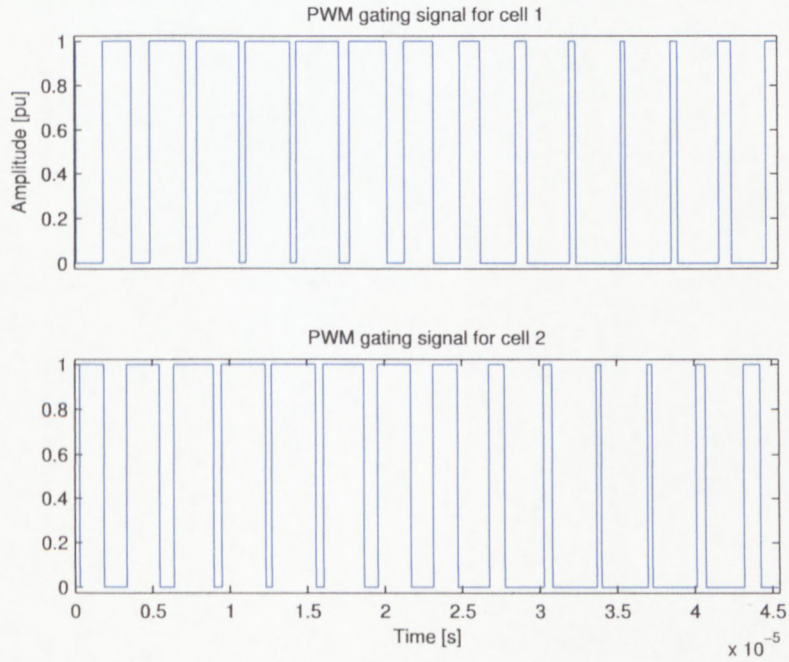


Figure 5.8: *Two*-cell multicell inverter firmware PWM as output by the logic analyser.

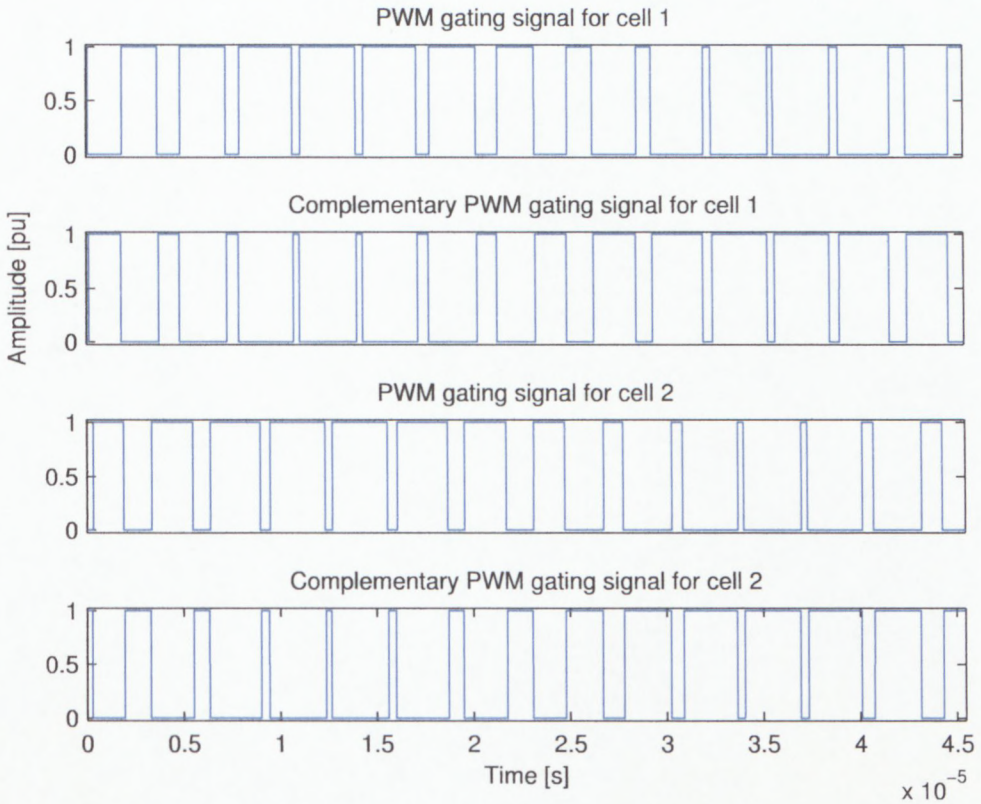


Figure 5.9: PWM gating and complementary gating signals generated by the *two*-cell firmware module.

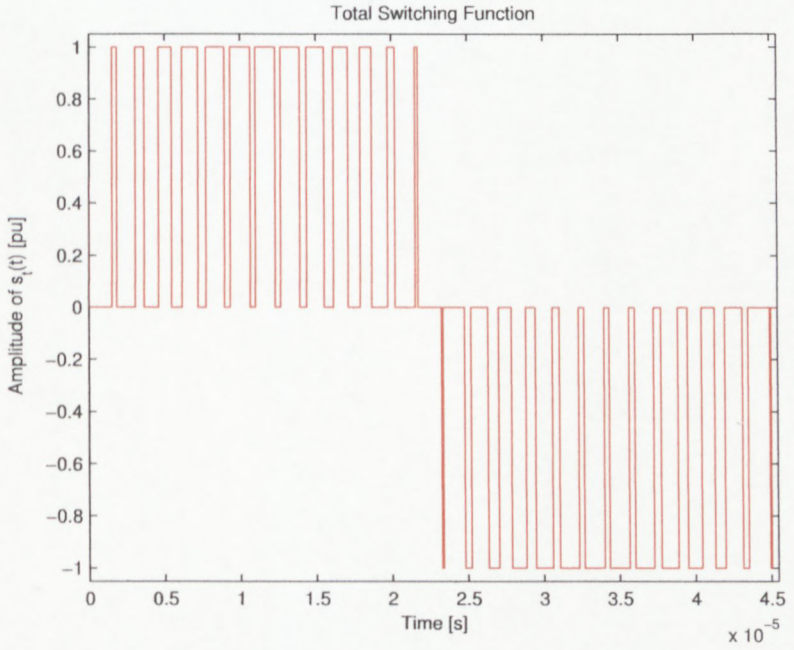


Figure 5.10: Total switching function of the *two*-cell.

The phase-shift of the PWM signals for successive cells is important in order to acquire the desired output from the multicell inverter. The phase-shift as previously mentioned is determined by the triangular carrier signals. The triangular signals are generated in the FPGA using a counter that had a negative and positive cycle. In order to implement a phase-shift among the carrier signals, the period of a single triangle signal needed to be determined. A phase-shift could then be implemented by starting at different count values of the triangle carrier signal, also depending on whether the counter must count up or down. A MATLAB® script was written to help determine these values quickly and accurately. Presented in Appendix D.1.

The complete output from the multicell inverter firmware module is presented in Figure 5.3.3. Since this firmware module is controlling a *three*-cell inverter, there are six PWM gating signals required, two for each cell. The PWM gating signals illustrated in Figure 5.3.3 will be used to control the top MOSFETs of the successive cells of the multicell inverter and the complementary signals the bottom MOSFETs.

5.3.4 *five*-cell

As stated in the introductory chapter, the primary application of the CIRPEC was to provide the gating signals necessary to control a *five*-cell multicell inverter. This was due to the fact that a *five*-cell multicell inverter existed which could be used to test the operation of the CIRPEC hardware and firmware on. Another fact was due to the *five*-cell firmware module being the most testing implementation for both the hardware and firmware.

The *five*-cell multicell firmware module implementation requires more hardware interfaces, more memory resources and more power than any of the other developed modules. In fact, all ten available fibre optic transmitters are utilised for this application.

This does not mean however that the controller is limited to a *five*-cell firmware module implementation, since additional fibre optic transmitters can be implemented using a daughterboard that will plug into the available GPIO header.

As with the previous multicell firmware modules, the process of generating the PWM is the same. Five f_c signals, phase-shifted by $\frac{360^\circ}{5}$ or 72° are compared against a single f_r signal. The f_r signal was selected to be in the region of

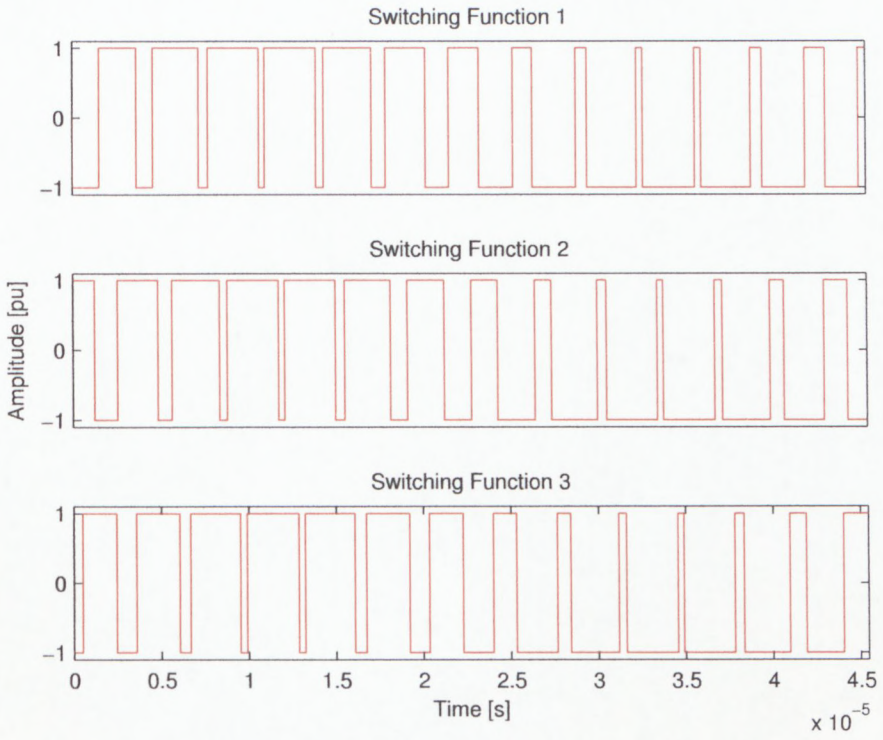


Figure 5.11: Switching function of a *three*-cell multicell inverter.

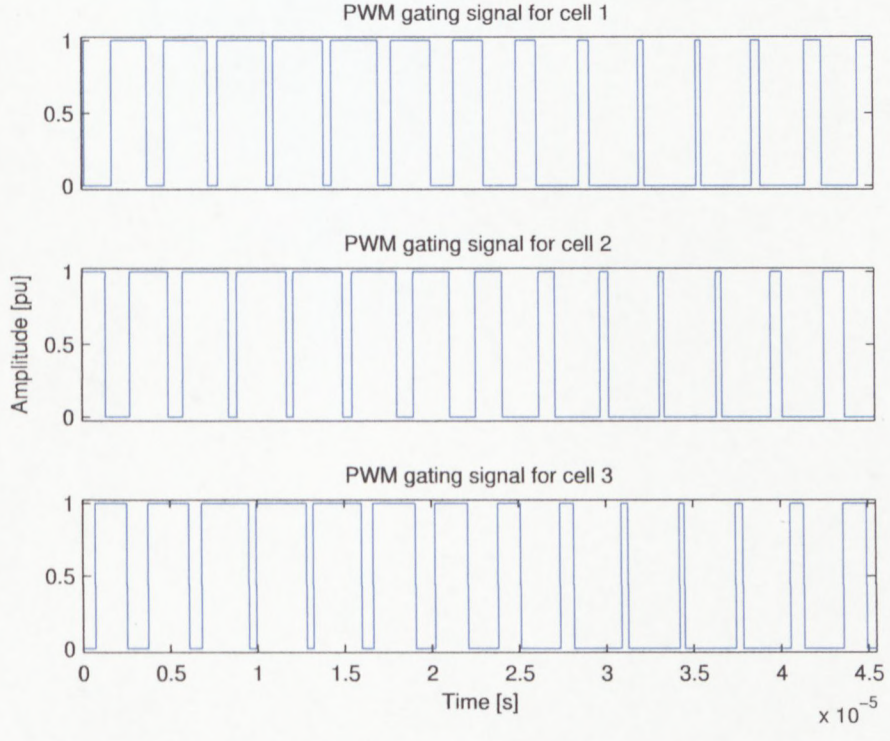


Figure 5.12: *Three*-cell multicell inverter firmware PWM gating signals.

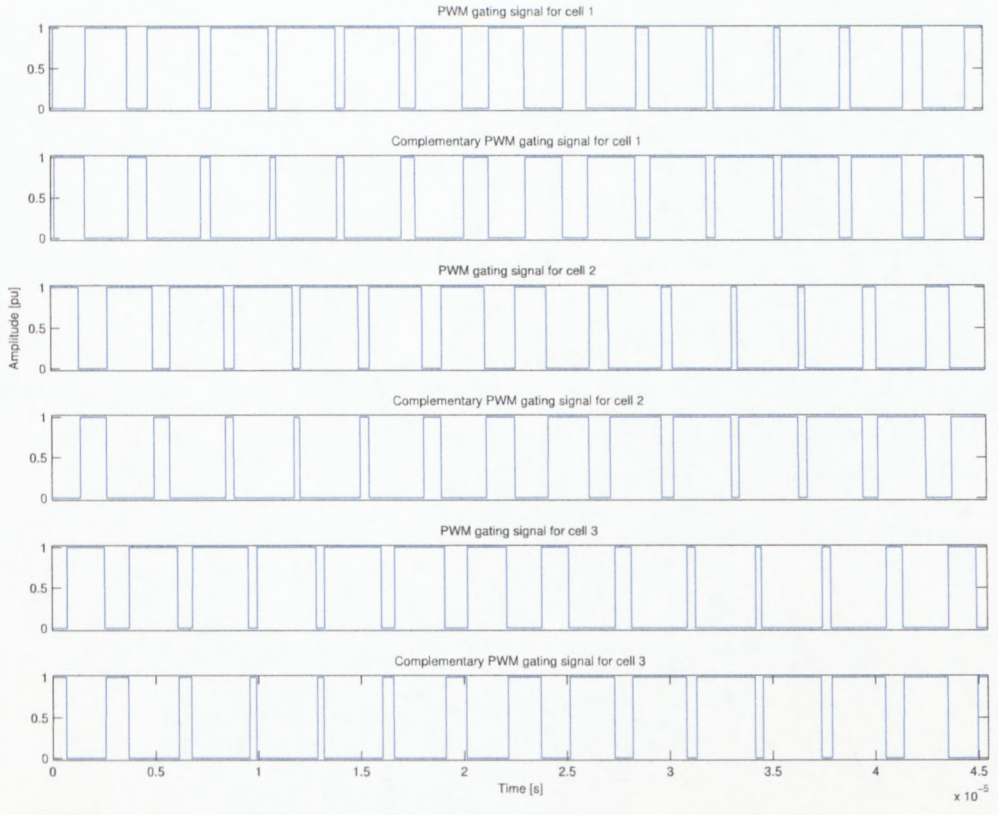


Figure 5.13: PWM gating and complementary gating signals generated by the *three*-cell firmware module.

---

22 kHz, since this was the resonant frequency of the ultrasonic transducer load. The f_c signal was chosen as 300 kHz since a much higher carrier frequency has certain harmonic advantages and also to achieve a good representation of the reference signal.

The switching functions, excluding their complementary pairs, for a *five*-cell multicell inverter as simulated in MATLAB® are presented in Figure 5.3.4.

The resulting PWM gating signals, also excluding their complementary signals, generated by the *five*-cell firmware module are presented in Figure 5.3.4 for comparison. A quick analysis of the captured waveforms with the simulated waveforms presents a favourable result.

Included in the complete output from the firmware module are the five PWM gating signals as presented in Figure 5.3.4, one for each cell, and their complementary gating signals as shown in Figure 5.3.4. The PWM gating

signals will be used to control the top MOSFETs of each successive cell and the complementary signals the bottom MOSFETs.

To fully establish whether the firmware module was operating as desired, the PWM gating signals generated to control the multicell inverter would need to be applied.

Before this was done though, a MATLAB® simulation of the total switching function was performed in order to determine what the unfiltered output from the multicell inverter was going to look like in order to make a comparison with the measured values. The result of the simulation for the *five*-cell module was presented in Chapter 4, however it is presented again for convenience in Figure 5.3.4. From the figure it can be seen that the *five*-cell multicell inverter has a six level output.

The PWM gating signals were then applied to the multicell inverter and the unfiltered output voltage was measured. Figure 5.3.4 presents the measured voltage. Analysis of the figure reveals the six level output as predicted by the simulation. The frequency of the output signal was measured to be 22.1 kHz. This is however not an accurate frequency to go on since it is quite a noisy signal and the frequency measurement would not have been steady.

Before that output voltage can be used to drive the transducer load it needs to be filtered. A sinusoidal signal is obtained by passing the unfiltered signal through a low-pass filter to remove all the high-frequency components. The measured filtered output voltage, after applying the filter, is illustrated in Figure 5.3.4. The frequency of the sine wave was captured at 22.05 kHz.

The *five*-cell multicell inverter used to acquire the above measurements was developed by Rory Pentz, a fellow masters student at the CIR. Modifications are still being made to the inverter and filter such that future waveforms obtained will be of better quality.

5.3.5 Feedback

In Section 4.3.2 of Chapter 4, it was mentioned that basic feedback functionality was implemented in the DSP. Feedback is used to make a change to the input of the system based on some response fed back from the output of the system with the aim to bring the output of the system back to an acceptable error margin.

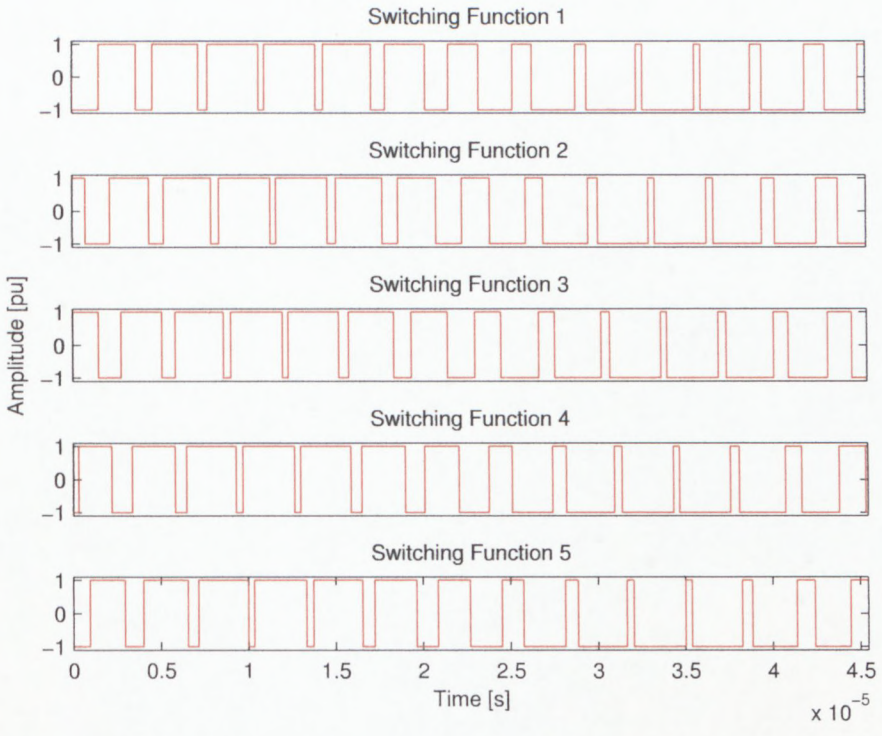


Figure 5.14: Simulated switching functions for the *five*-cell multicell converter.

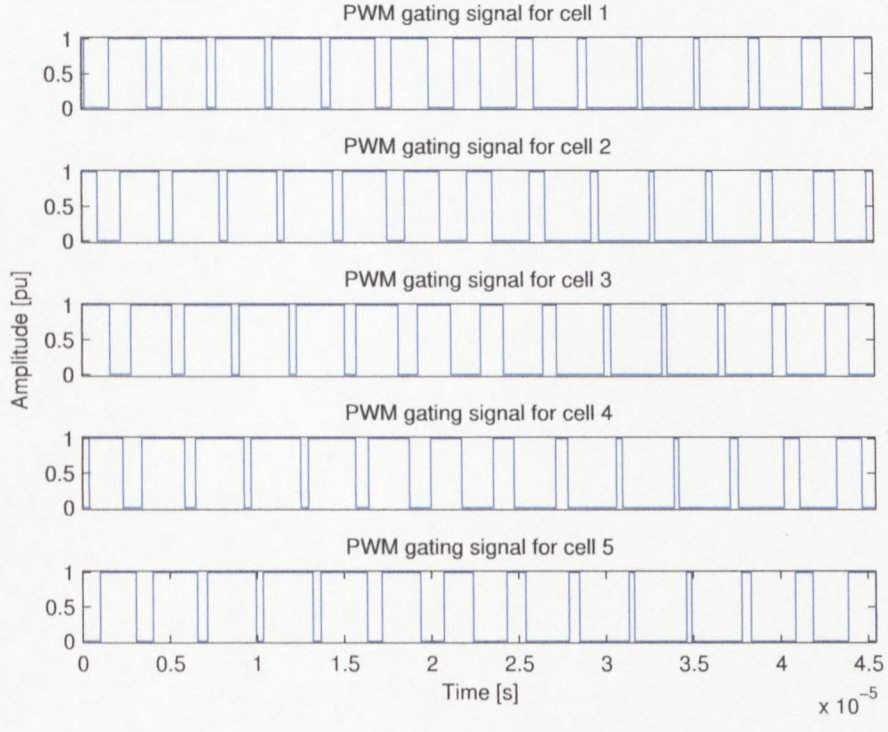


Figure 5.15: *Five*-cell multicell inverter firmware PWM gating signals.

Figure 5.16: *Five*-cell firmware PWM gating and complementary gating signals.

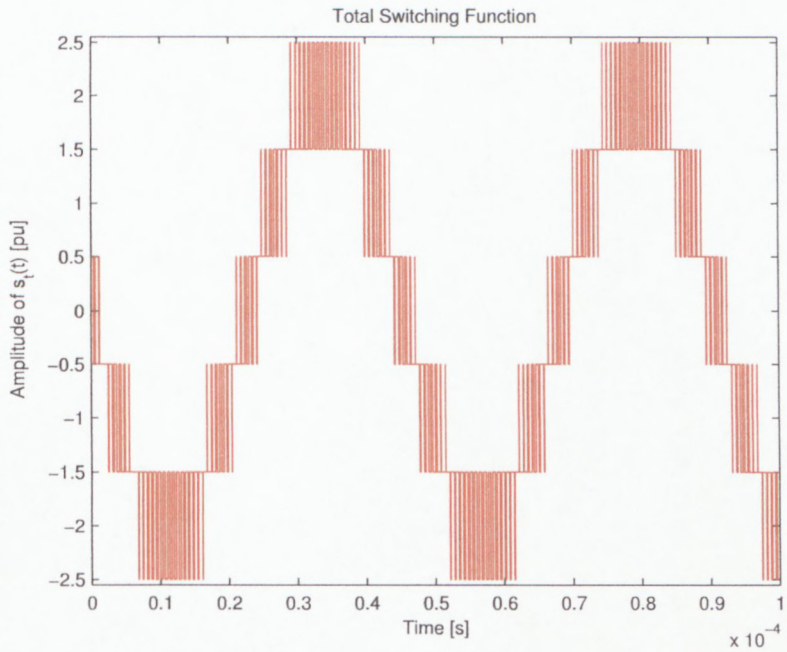


Figure 5.17: Total switching function for a *five*-cell multicell inverter.

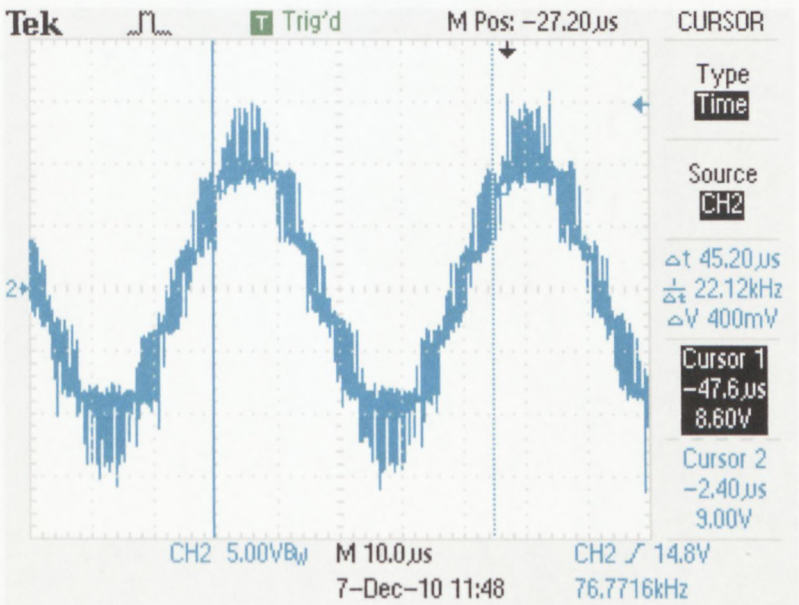


Figure 5.18: Unfiltered output voltage measured from the *five*-cell multicell inverter.

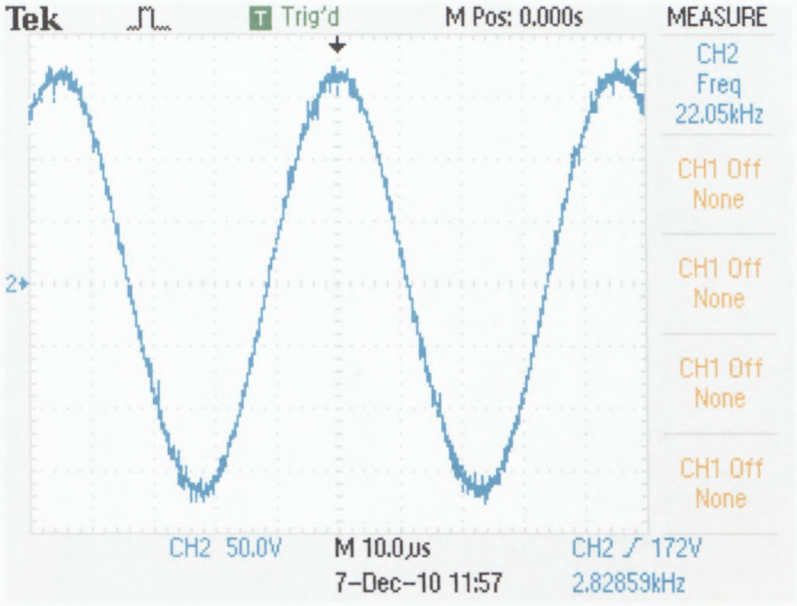


Figure 5.19: Filtered output voltage measured from the *five*-cell multicell inverter.

A simple form of feedback was chosen and that was to have the DSP change the sinusoidal reference signal's (f_r) frequency based on a value sent to it by a microcontroller measuring at the output of the transducer load. Future implementations include the CIRPEC performing all closed loop tasks and algorithms, since feedback was out of the scope of this project.

In order for the DSP to be able to select different sinusoidal f_r frequencies involved the DSP generating multiple sinusoidal waveforms at startup and storing these *tables* in RAM. Once all the tables had been generated and stored locally to the DSP, a block transfer using the DSPs dMAX module or data movement accelerator module was initiated to send the tables to the FPGA. The tables were stored in a RAM block generated in the FPGA.

As the location of the start and stop of every table could be determined since the size of each table was known, a table index value variable (*indexValue*) was created and used as a table pointer. The DSP would directly update the *indexValue* variable on the FPGA based on the value received from the microcontroller. The *indexValue* variable is used by the firmware module in the FPGA to run a specific sinusoidal table and subsequently generate the PWM signals.

To demonstrate that changing the table index value (*indexValue*) on the

DSP results in a different sinusoidal frequency being output, a series of three consecutive index values were captured on an oscilloscope. The following sinusoidal waveforms were obtained by passing a single PWM gating signal out via a GPIO pin and low-pass filtering it. The sequence of results are presented in Figures 5.3.5, 5.3.5 and 5.3.5.

The frequencies of the three sine signals are 22.0021, 22.0789 and 22.1580 kHz respectively. This results in increments of 77.7 Hz and 78.2 Hz between signals respectively. The frequency increments are dependent on the length of the sine table and the rate at which it is read out from the RAM block.

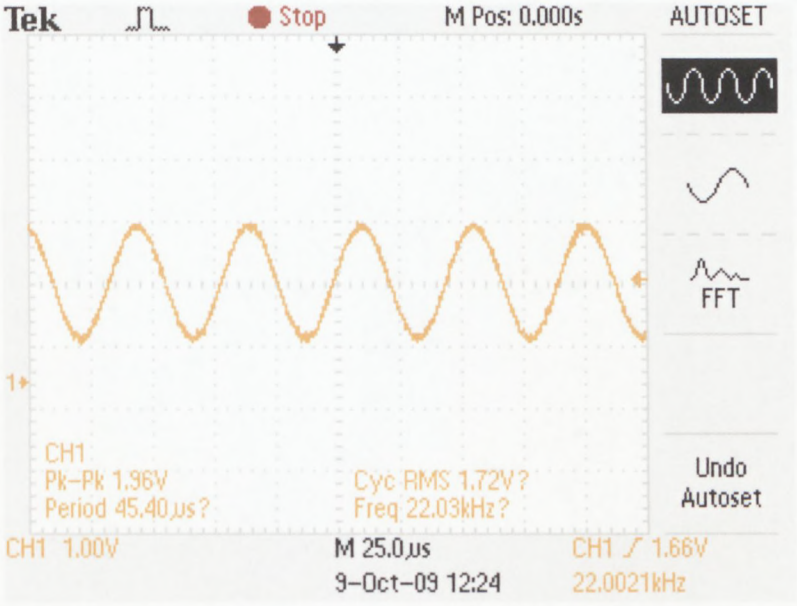


Figure 5.20: First sine table.

5.4 Conclusion

The results obtained from the various firmware modules captured by the logic analyser were presented, along with the MATLAB® simulated waveforms to verify the results. Through analysis of the simulated and practical PWM gating signals obtained it can be seen that the results are desirable. The frequency of the f_r signal was measured using an audio analyser and confirmed the fundamental frequency to be at 22 kHz.

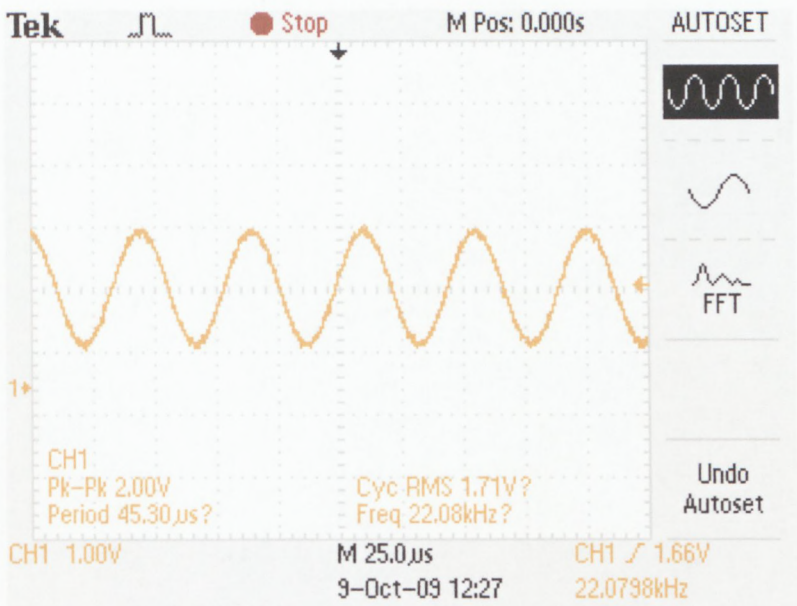


Figure 5.21: Second sine table.

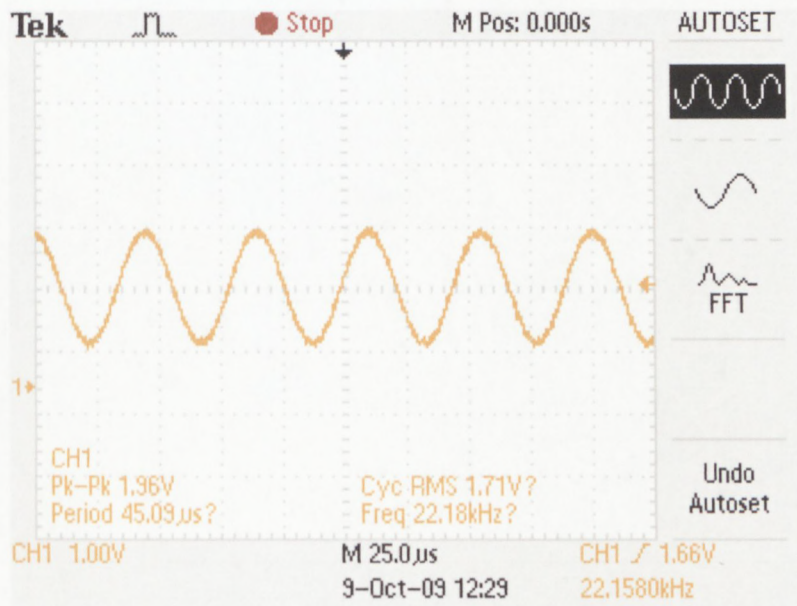


Figure 5.22: Third sine table.

The experimental setup was explained for testing the *five*-cell firmware module on a *five*-cell multicell inverter. The output voltage waveform captured by the oscilloscope was accurate according to the MATLAB® simulation. Also the filtered output voltage was presented and illustrated a desirable output from the inverter. The results verify that the controller hardware and firmware module designed were successfully implemented in the control of a *five*-cell multicell inverter.

A basic feedback implementation was presented, by adjusting the *index-Value* variable, the FPGA was successfully able to select a different frequency sinusoidal table from its memory and integrate it into PWM firmware module. The results of three such frequencies were presented.

Chapter 6

Conclusion

6.1 Introduction

A generic digital controller for power electronic applications was investigated and designed in this thesis. A flexible digital controller was developed by the author and is referred to as the CIRPEC (Centre for Instrumentation Research Power Electronics Controller) in the text. This chapter provides the findings of the research, contributions in terms of hardware and firmware, and conclusions.

6.2 Hardware

An investigation was carried out to determine the key components of a power electronics controller. The controller also needed to be able to incorporate feedback functionality. Many existing controllers were researched to determine their architecture and develop an understanding of the components used based on their end applications.

Research established that three components were important for control. These were an ADC, DAC and a processor. An ADC is necessary to sample an analog signal and provide a digital output to a processor in order to apply a control algorithm. A DAC is necessary to convert the post-processed signal back into an analog signal to be fed back into the control loop. A DSP and an FPGA were selected as a hybrid combination for the processor. One of the reasons for the hybrid selection was as a result of the controller being developed as a tool for students and engineers. This allows for either the FPGA or DSP to be largely programmed as separate pieces of hardware allowing for greater

application flexibility. The DSP allowed for raw computational processing of intensive algorithms. The FPGA was selected as a co-processor to the DSP to add flexibility to the design.

The FPGA can be used for performing processing while at the same time being used to interface to external devices. An FPGA helps to reduce the need for external components which has positive implications including simplifying board design complexity, time and cost reductions. The more circuitry that can be implemented in a single chip will result in better signal integrity as well as a more compact board design. This in turn will save costs on printed circuit board manufacturing as well as the price of additional external components. Both the DSP and FPGA are reconfigurable allowing for application flexibility.

The addition of fibre optic transmitters and receivers allow for the control and feedback of power electronic converters while providing electrical isolation. Electric isolation is necessary in the event of a voltage problem on the converter that the controller is isolated from damage. Another advantage of using fibre optics as a data transfer medium is that the controller can be placed a fair distance away from the converter. The light signal transmitted along a fibre optic cable is not attenuated as severely as a voltage signal along a wired connection, this allows for greater distances to be achieved. The data is also transmitted at a high data rate and can be in the order of five Megabaud. The ten fibre optic transmitters provided on the CIRPEC are sufficient to control many types of power electronic converters. To add to the flexibility of the design, an expansion header was provided that is linked directly to the FPGA providing 29 GPIO pins. These pins can be used for measurement daughterboards or additional fibre optics and other general purpose applications. Access to the audio functionality of the DSP can also be achieved via the header.

6.3 Firmware

Considering the controller was designed for power electronics applications and that there was an end application in mind, some firmware modules had to be programmed to demonstrate its capability for control. A programmed module included a scalable multicell module which, with minimal adjustments, could be adjusted for control of a p-cell multicell converter. As an experimental

setup, a *five*-cell multicell inverter firmware module was used to successfully control a *five*-cell multicell inverter. Other firmware modules programmed included a half- and full-bridge module. A MATLAB® script was written to assist in verifying the various firmware modules developed. The programmed modules will also serve as an example for future users of the hardware on how to configure communication between the DSP, SDRAM and the FPGA. This includes setting up the relevant EMIF and dMAX registers for data transfers and configuring the FPGA to accept incoming data.

6.4 Thesis Contribution

An engineering hardware tool in the form of a flexible generic digital controller has been developed along with some generic firmware modules. The aim was to assist both students and engineers in implementing their designs without having to develop a controller for a specific application as this may require much time and skill resources as controller hardware design may not be their niche area or expertise. The firmware modules will also provide the necessary reference code to commence design, shortening production time.

6.5 Future Work

All the hardware components necessary for the design of the firmware modules on the CIRPEC were verified experimentally. These included the DSP, FPGA, SDRAM, EEPROM, fibre optics, LEDs and GPIOs. However, some of the functionality incorporated into the hardware design was not tested. These components include the ADCs, DACs, USB and Flash memory. These components should be tested to verify the design.

Future board revisions could include modifying the design to include a higher-end FPGA capable of embedding a soft processor, such as the Altera NIOS® II, as this will provide engineers with greater choice when it comes to designs. Such embedded processors are capable of implementing DSP functions. The designer can therefore choose whether to use the hardware DSP or the soft processor for designs or both.

Future firmware designs could include implementing registers on the FPGA that could change the module used on the fly, currently if the module needs to

be changed the FPGA must be reprogrammed. The DSP would then be used to modify the register values.

List of References

Altera Corporation. 2001. High-speed board designs. Application note 75.
Available at: http://www.altera.com/

Altera Corporation. 2003. Cyclone device handbook, volume 1.
Available at: http://www.altera.com/

Analog Devices. 2000. ADMC401: single-chip, DSP-based high performance motor controller.
Available at: http://www.analog.com/static/imported-files/data_sheets/ADMC401.pdf

Avago Technologies. 2006. HFBR-0500Z series versatile link the versatile fiber optic connection.
Available at: http://www.avagotech.com/

Baldwin, K. 2009. Texas Instruments application report: using the TMS320C672x bootloader. SPRAA69D.
Available at: http://www.ti.com/

Bester, D.D. 1999. Control of series compensator for power quality conditioner. Master's thesis, University of Stellenbosch, Stellenbosch, South Africa.

Bester, D.D., du Toit, J.A. & Enslin, J.H.R. 1998. High performance DSP/FPGA controller for implementation of computationally intensive algorithms. In *Proceedings of the IEEE International Symposium on Industrial Electronics*, volume 1 of *ISIE'98*, pages 240 – 244. Pretoria, South Africa.
Available at: http://dx.doi.org/10.1109/ISIE.1998.707784

Celanovic, I. 2000. A distributed digital control architecture for power electronics systems. Master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA, USA.

Celanovic, I., Milosavljevic, I., Boroyevich, D., Cooley, R. *&* Guo, J. 2000. A new distributed digital controller for the next generation of power electronics building blocks. In *Proceedings of the 15th IEEE Applied Power Electronics Conference and Exposition*, volume 2 of *APEC'00*, pages 889 – 894. New Orleans, LA, USA.
Available at: `http://dx.doi.org/10.1109/APEC.2000.822610`

du Toit, J., Bester, D. *&* Enslin, J. 1997. DSP based controller for back to back power electronic converters with FPGA integration. In *Proceedings of the 12th IEEE Applied Power Electronics Conference and Exposition*, volume 2 of *APEC'97*, pages 699 – 705. Atlanta, GA, USA.
Available at: `http://dx.doi.org/10.1109/APEC.1997.575661`

du Toit, J. *&* Enslin, J. 1996. DSP based controller for high dynamic bandwidth power electronic applications. In *Proceedings of the 4th IEEE AFRICON Conference in Africa*, volume 2 of *AFRICON'96*, pages 665 – 670. Stellenbosch, South Africa.
Available at: `http://dx.doi.org/10.1109/AFRCON.1996.562968`

du Toit, J., le Roux, A. *&* Enslin, J. 1998. An integrated controller module for distributed control of power electronics. In *Proceedings of the 13th IEEE Applied Power Electronics Conference and Exposition*, volume 2 of *APEC'98*, pages 874 – 880. Anaheim, CA, USA.
Available at: `http://dx.doi.org/10.1109/APEC.1998.654001`

du Toit, J.A., Enslin, J.H.R. *&* Spee, R. 1995. Experimental evaluation of digital control options for high power electronics. In *Proceedings of the 21st IEEE International Conference on Industrial Electronics, Control, and Instrumentation*, volume 1 of *IECON'95*, pages 674 – 679. Orlando, FL, USA.
Available at: `http://dx.doi.org/10.1109/IECON.1995.483489`

Duan, Y. *&* Jin, H. 1999. Digital controller design for switchmode power converters. In *Proceedings of the 14th IEEE Applied Power Electronics Conference and Exposition*, volume 2 of *APEC'99*, pages 967 – 973. Dallas, TX, USA.
Available at: `http://dx.doi.org/10.1109/APEC.1999.750486`

Erickson, R.W. *&* Maksimovic, D. 2004. *Fundamentals of Power Electronics*. 2nd edition. Springer.
Available at: `http://common.books24x7.com/book/id_16231/book.asp`

Francis, G. 2004. A synchronous distributed digital control architecture for high power converters. Masters thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA, USA.

Francis, G., Burgos, R., Celanovic, I., Wang, F. *&* Boroyevich, D. 2005. A universal controller for distributed control of power electronics systems in electric ships. In *Proceedings of the American Control Conference*, volume 3 of *ACC'05*, pages 1999 – 2004. Portland, OR, USA.
Available at: `http://dx.doi.org/10.1109/ACC.2005.1470263`

Francis, J. *&* Boroyevich, D. 2001. Design of a universal controller for distributed control and power electronics applications. In *CPES Power Electronics Seminar and NSF/Industry Annual Review.*
Available at: `http://web-cat.cs.vt.edu/PEBB/publications.php`

Frantz, G. *&* Simar, R. 2004. Texas Instruments whitepaper: comparing fixed- and floating-point DSPs. SPRY061.
Available at: `http://focus.ti.com/lit/wp/spry061/spry061.pdf`

Hamma, F., Meynard, T.A., Tourkhani, F. *&* Viarouge, P. 1995. Characteristics and design of multilevel choppers. In *Proceedings of the 26th IEEE Power Electronics Specialists Conference*, volume 2 of *PESC'95*, pages 1208–1214. Atlanta, GA, USA.
Available at: `http://dx.doi.org/10.1109/PESC.1995.474968`

Hansmann, C.H. 2004. Active capacitor voltage stabilisation in a medium-voltage flying-capacitor multilevel active filter. Master's thesis, University of Stellenbosch, Stellenbosch, South Africa.

Holmes, G. 2000. Technical brief: MU-DSP240-LPI inverter controller card. Technical Report, PEG, Monash University, Victoria, Australia.

Karipidis, C.U. 2001. A versatile DSP/FPGA structure optimised for rapid prototyping and digital real-time simulation of power electronic and electrical drive systems. Ph.D. thesis, RWTH Aachen University, Aachen, Germany.

Kaye, M.E. *&* Smith, T. 1989. Design of a digital signal processor based digital controller. In *Proceedings of the 21st Southeastern Symposium on System Theory*, SSST'89, pages 353 – 358. Tallahassee, FL, USA.
Available at: `http://dx.doi.org/10.1109/SSST.1989.72491`

Krein, P.T. 1998. *Elements of Power Electronics.* Oxford University Press.
Available at: `http://common.books24x7.com/book/id_9294/book.asp`

Liu, W. 2005. Distributed modular controller architecture for high power converter applications. Master's thesis, North Carolina State University, Raleigh, NC, USA.

Liu, W., Jayakar, R., Song, W. *&* Huang, A.Q. 2005. A modular digital controller architecture for multi-node high power converter applications. In *31st Annual Conference of the IEEE Industrial Electronics Society*, IECON'05, pages 715–720. Raleigh, NC, USA.
Available at: `http://dx.doi.org/10.1109/IECON.2005.1568992`

Luo, F.L., Ye, H. *&* Rashid, M.H. 2005. *Digital Power Electronics and Applications.* Academic Press.
Available at: `http://common.books24x7.com/book/id_17875/book.asp`

Maksimovic, D., Zane, R. *&* Erickson, R. 2004. Impact of digital control in power electronics. In *Proceedings of the 16th International Symposium on Power Semiconductor Devices and ICs*, ISPSD'04, pages 13–22. Kitakyushu, Japan.
Available at: `http://dx.doi.org/10.1109/ISPSD.2004.1332844`

Martin, T.W. *&* Ang, S.S. 1995. Digital control for switching converters. In *Proceedings of the International Symposium on Industrial Electronics*, volume 2 of *ISIE'95*, pages 480 – 484. Athens, Greece.
Available at: `http://dx.doi.org/10.1109/ISIE.1995.497232`

Meynard, T.A., Fadel, M. *&* Aouda, N. 1997. Modelling of multilevel converters. In *IEEE Transactions on Industrial Electronics*, volume 44, pages 356–364.

Meynard, T.A. *&* Foch, H. 1995. Multilevel converters and derived topologies for high power conversion. In *Proceedings of the 21st IEEE International Conference on Industrial Electronics, Control, and Instrumentation*, volume 1 of *IECON'95*, pages 21–26. Orlando, FL, USA.
Available at: `http://dx.doi.org/10.1109/IECON.1995.483327`

Miao, B., Zane, R. *&* Maksimovic, D. 2004. A modified cross-correlation method for system identification of power converters with digital control. In *Proceedings of the 35th Annual Power Electronics Specialists Conference*, volume 5 of *PESC'04*, pages 3728 – 3733. Aachen, Germany.
Available at: `http://dx.doi.org/10.1109/PESC.2004.1355134`

Microchip Technology. 2007. 24AA1025/24LC1025/24FC1025 1024K I2C CMOS serial EEPROM.
Available at: `http://www.microchip.com/`

Milanovic, M., Truntic, M. *&* Slibar, P. 2005. FPGA implementation of digital controller for dc-dc buck converter. In *Proceedings of the 5th International Workshop on System-on-Chip for Real-Time Applications*, IWSOC'05, pages 439 – 443. Banff, Alta., Canada.
Available at: `http://dx.doi.org/10.1109/IWSOC.2005.67`

Milanovic, M., Truntic, M., Slibar, P. *&* Dolinar, D. 2007. Reconfigurable digital controller for a buck converter based on FPGA. In *Microelectronics Reliability*, volume 47, pages 150 – 154. Elsevier.
Available at: `http://dx.doi.org/10.1016/j.microrel.2006.09.019`

Mohan, N. *&* Ang, S. 1994. A digital-signal-processor based controller for a switching converter. In *International Journal of Electronics*, volume 77, pages 643 – 656.
Available at: `http://dx.doi.org/10.1080/00207219408926092`

Mohan, N., Undeland, T. *&* Robbins, W. 2003. *Power Electronics: Converters, Applications, and Design.* 3rd edition. John Wiley & Sons, Inc.

Molepo, S.A. 2003. A multilevel inverter for dc reticulation. Master's thesis, University of Stellenbosch, Stellenbosch, South Africa.

National Semiconductor. 2005. LM1085 3A low dropout positive regulators. DS100947.
Available at: `http://www.national.com/`

Ott, H.W. 2001. Partitioning and layout of a mixed-signal pcb. In *Printed Circuit Design*, pages 8 – 11.
Available at: `http://www.hottconsultants.com/papers.html`

Pease, R.A. 2008. *Analog Circuits: World Class Designs.* Newnes.
Available at: `http://common.books24x7.com/book/id_32302/book.asp`

Peterchev, A.V. *&* Sanders, S.R. 2001. Quantization resolution and limit cycling in digitally controlled pwm converters. In *Proceedings of the 32nd IEEE Power Electronics Specialists Conference*, volume 2 of *PESC'01*, pages 465 – 471. Vancouver, BC, Canada.
Available at: `http://dx.doi.org/10.1109/PESC.2001.954158`

Peterchev, A.V. *&* Sanders, S.R. 2003. Quantization resolution and limit cycling in digitally controlled pwm converters. In *IEEE Transactions on Power Electronics*, volume 18, pages 301 – 308.
Available at: `http://dx.doi.org/10.1109/TPEL.2002.807092`

Proakis, J.G. *&* Manolakis, D.G. 1996. *Digital signal processing principles, algorithms, and applications.* Prentice-Hall, Inc.

Prodic, A., Maksimovic, D. *&* Erickson, R.W. 2001. Design and implementation of a digital pwm controller for a high-frequency switching dc-dc power converter. In *Proceedings of the 27th Annual IEEE Industrial Electronics Society Conference*, volume 2 of *IECON'01*, pages 893 – 898. Denver, CO, USA.
Available at: `http://dx.doi.org/10.1109/IECON.2001.975878`

Prodic, A., Maksimovic, D. *&* Erickson, R.W. 2003. Dead-zone digital controller for improved dynamic response of power factor preregulators. In *Proceedings of the 18th Applied Power Electronics Conference and Exposition*, volume 1 of *APEC'03*, pages 382 – 388.
Available at: `http://dx.doi.org/10.1109/APEC.2003.1179242`

Samsung Electronics. 2008. K4S281632K 128Mb K-die SDRAM specification.
Available at: `http://www.samsung.com/`

Spansion Inc. 2007. S29AL016D 16 megabit (2 M x 8-bit/1 M x 16-bit) CMOS 3.0 volt-only boot sector flash memory.
Available at: `http://www.spansion.com/`

Spectrum Digital Inc. 2007. XDS510USB PLUS JTAG emulator installation guide.
Available at: `http://www.spectrumdigital.com/`

Spectrum Digital Inc. 2008. Spectrum Digital emulator selection guide.
Available at: `http://www.spectrumdigital.com/`

ST-NXP Wireless. 2006. PDIUSBD12 universal serial bus peripheral controller with parallel bus.
Available at: `http://www.stnwireless.com`

Syed, A., Ahmed, E. *&* Maksimovic, D. 2004. Digital pwm controller with feedforward compensation. In *Proceedings of the 19th IEEE Applied Power Electronics Conference and Exposition*, volume 1 of *APEC'04*, pages 60 – 66. Anaheim, CA, USA.
Available at: `http://dx.doi.org/10.1109/APEC.2004.1295788`

Tan, L. 2008. *Digital signal processing fundamentals and applications.* Academic Press.

Texas Instruments. 1998. TLV1570 2.7 V to 5.5 V 8-channel 10-bit 1.25-MSPS serial analog-to-digital converter. SLAS169A.
Available at: `http://www.ti.com/`

Texas Instruments. 2003. TMS320C6000 DSP designing for JTAG emulation reference guide. SPRU641.
Available at: `http://www.ti.com/`

Texas Instruments. 2004. 2.7-V to 5.5-V low-power dual 12-bit digital-to-analog converter with internal reference and power down. SLAS225C.
Available at: `http://www.ti.com/`

Texas Instruments. 2005*a*. CDCVF2505 3.3-V clock phase-lock loop clock driver. SCAS640E.
Available at: `http://www.ti.com/`

Texas Instruments. 2005*b*. High-speed DSP systems design reference guide. SPRU889.
Available at: `http://www.ti.com/`

Texas Instruments. 2005*c*. uA78M00 series positive-voltage regulators. SLVS059P.
Available at: `http://www.ti.com/`

Texas Instruments. 2006*a*. C923C100 TMS320C672x floating-point digital signal processor ROM. SPRS277C.
Available at: `http://www.ti.com/`

Texas Instruments. 2006*b*. Texas Instruments product bulletin: TMS320C672x floating-point DSPs. SPRT349D.
Available at: `http://www.ti.com/`

Texas Instruments. 2007. TPS70445, TPS70448, TPS70451, TPS70458, TPS70402 dual-output, low dropout voltage regulators with integrated SVS for split voltage systems. SLVS307D.
Available at: `http://www.ti.com/`

Texas Instruments. 2008*a*. Application report: Powerpad thermally enhanced package. SLMA002C.
Available at: `http://www.ti.com/`

Texas Instruments. 2008*b*. TMS320C6727, TMS320C6726B, TMS320C6722B, TMS320C6720 floating-point digital signal processors. SPRS370E. Available at: `http://www.ti.com/`

Van Heerden, G. 2003. Design and implementation of a DSP based controller for power electronic applications. Master's thesis, University of Stellenbosch, Stellenbosch, South Africa.

Walker, G.R. 1999. Modulation and control of multilevel converters. Ph.D. thesis, University of Queensland, Queensland, Australia.

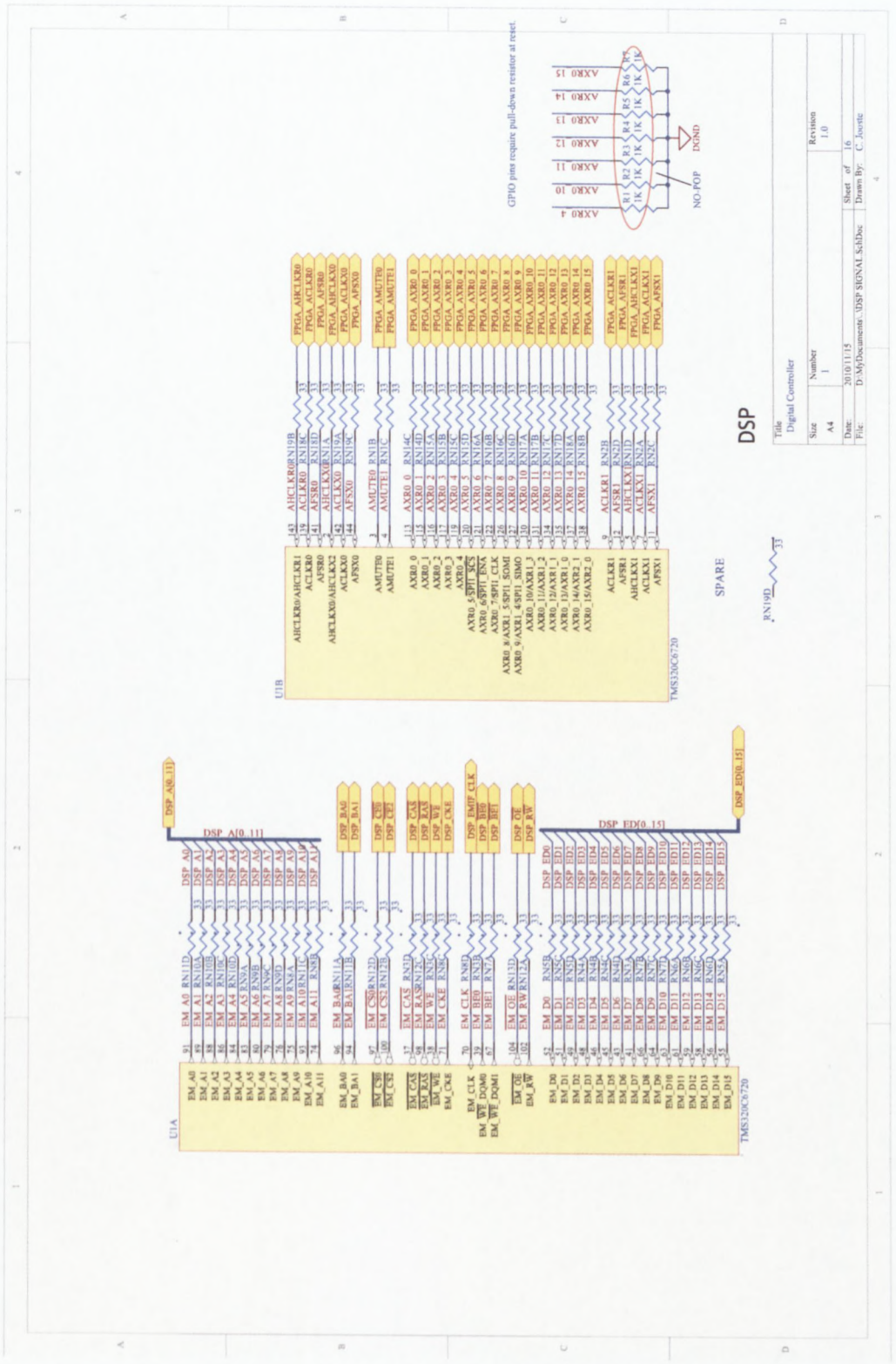
Wilkinson, R.H. 1997. Topology, control and development of high power multilevel converters. Master's thesis, University of Stellenbosch, Stellenbosch, South Africa.

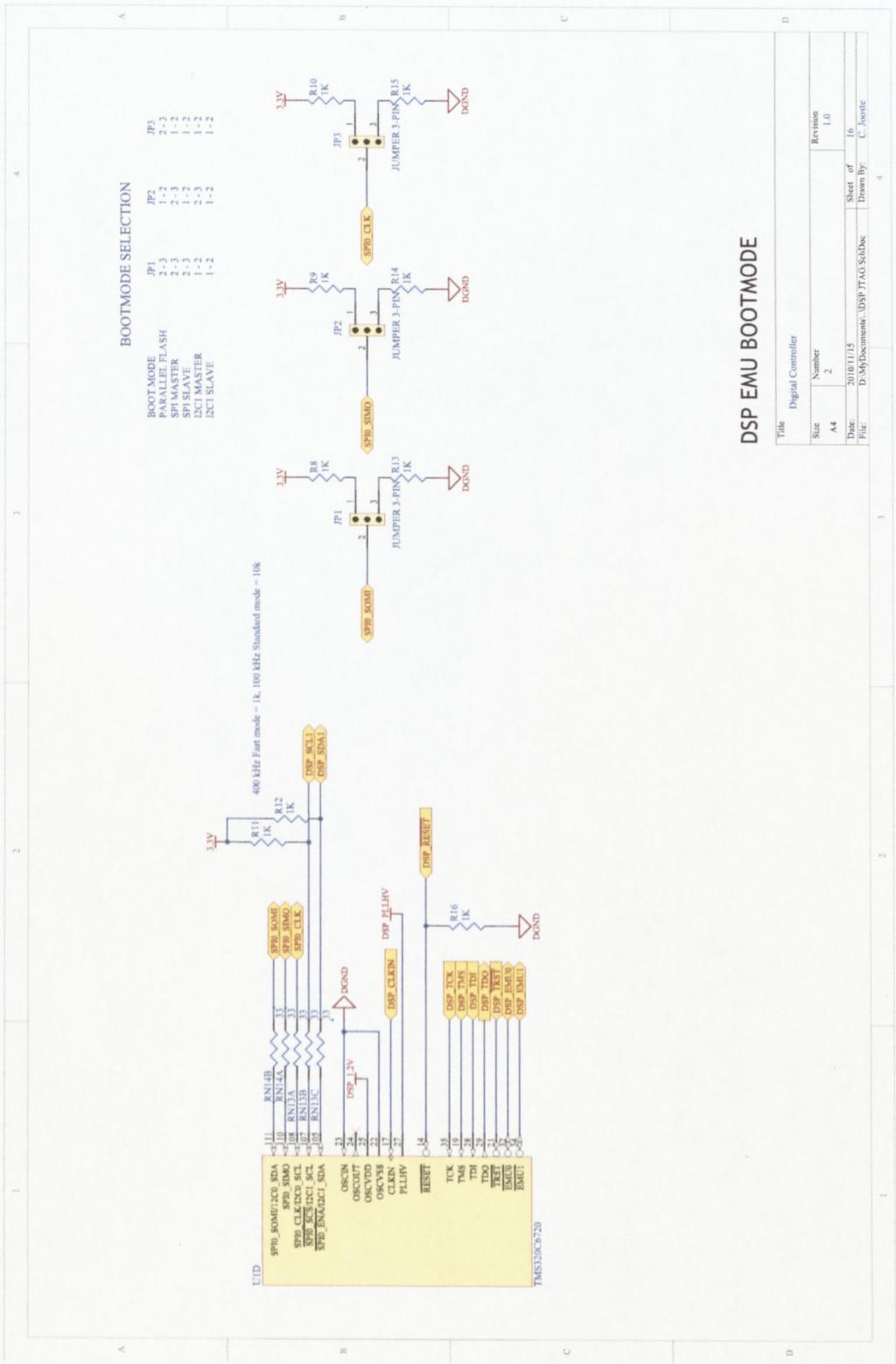
Wilkinson, R.H. 2004. Natural balancing of multicell converters. Ph.D. thesis, University of Stellenbosch, Stellenbosch, South Africa.

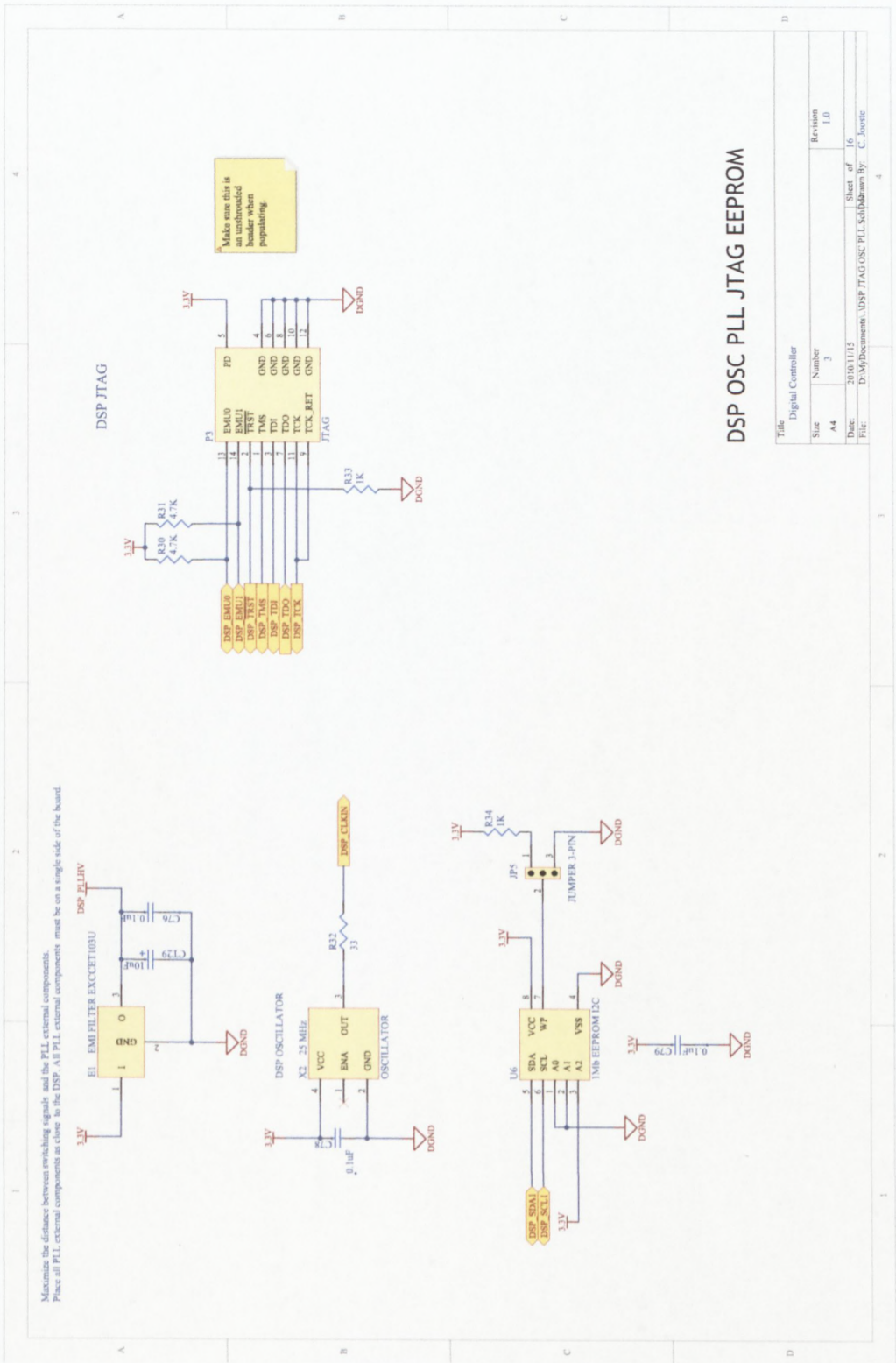
Appendix A

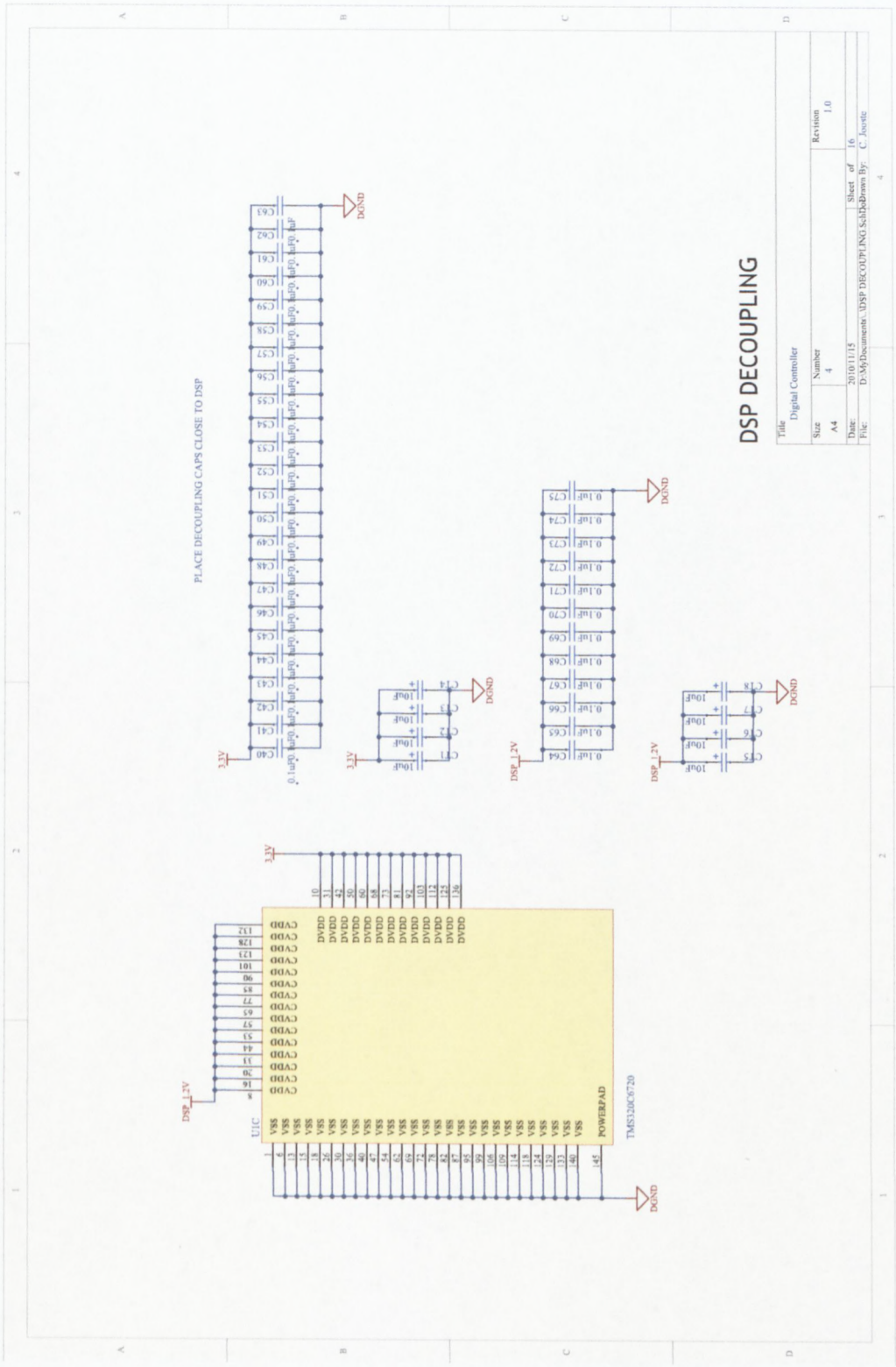
Schematics

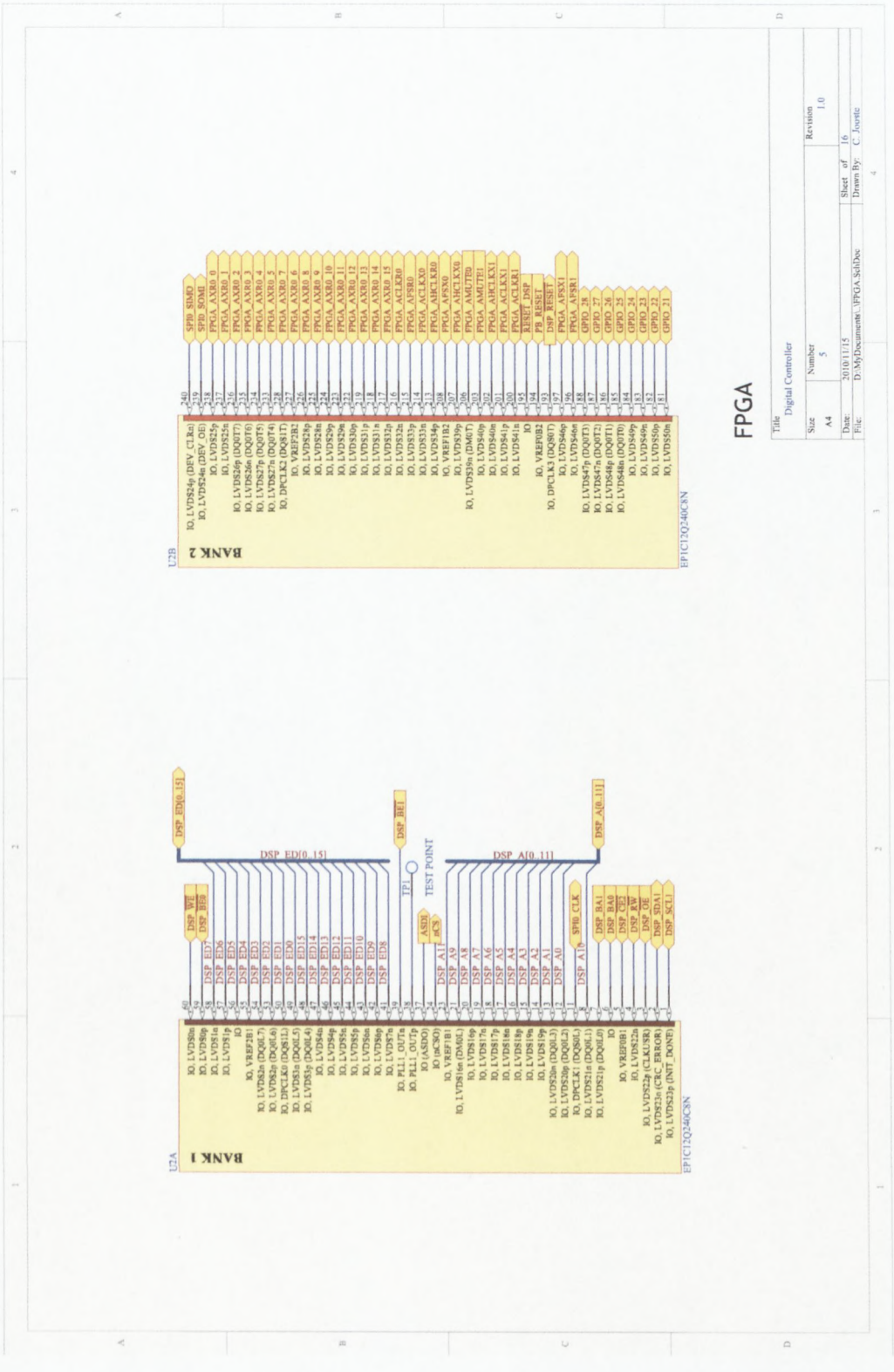
A.1 CIRPEC Schematics


DSP
Title
Digital Controller
Size
A4
Number
1
Revision
1.0
Date:
2010/11/15
Sheet of 16
File:
D:\MyDocuments\..\DSP SIGNAL Sch.Doc
Drawn By:
C. Jacobs
GPIO pins require pull-down resistor at reset.
NO-POP
DGND
SPARE
U1A
U1B
TMS320C6720
DSP_A[0..11]
DSP_ED[0..15]
AHCLKR0/AHCLKR1
ACLKR0
AFSR0
AHCLKX0/AHCLKX1
ACLKX0
AFSX0
AMUTE0
AMUTE1
ACLKR1
AFSR1
AHCLKX1
ACLKX1
AFSX1

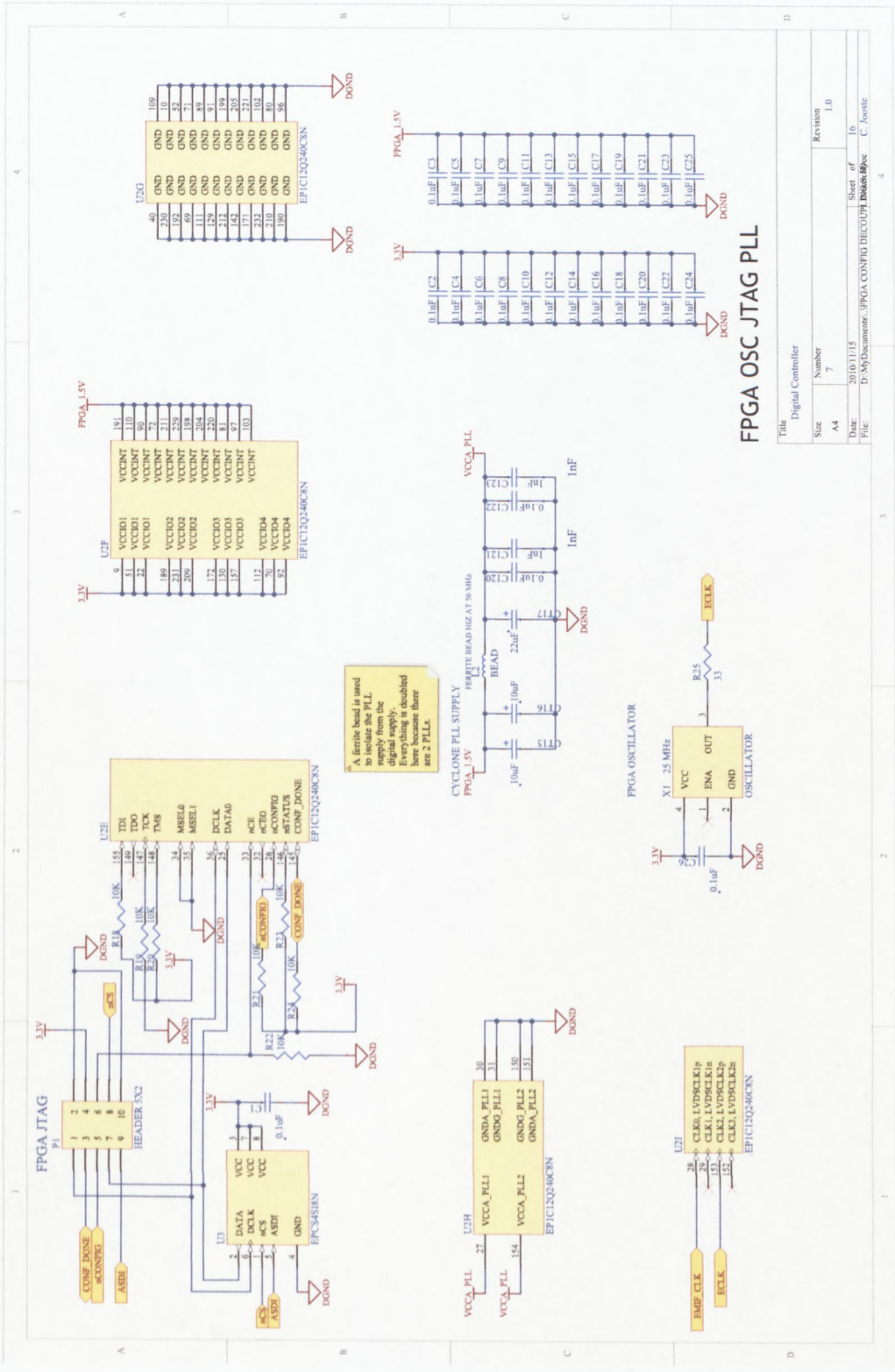

BOOTMODE SELECTION
BOOT MODE JP1 JP2 JP3
PARALLEL_FLASH 2 - 3 1 - 2 2 - 3
SPI MASTER 2 - 3 2 - 3 1 - 2
SPI SLAVE 2 - 3 1 - 2 1 - 2
I2C1 MASTER 1 - 2 2 - 3 1 - 2
I2C1 SLAVE 1 - 2 1 - 2 1 - 2
400 kHz Fast mode = 1k, 100 kHz Standard mode = 10k
DSP EMU BOOTMODE
Digital Controller
Size A4
Number 2
Revision 1.0
Date: 2010/11/15
Sheet of 16
File: D:\MyDocuments\...\DSP JTAG Sch1.Doc
Drawn By: C. Jacobs

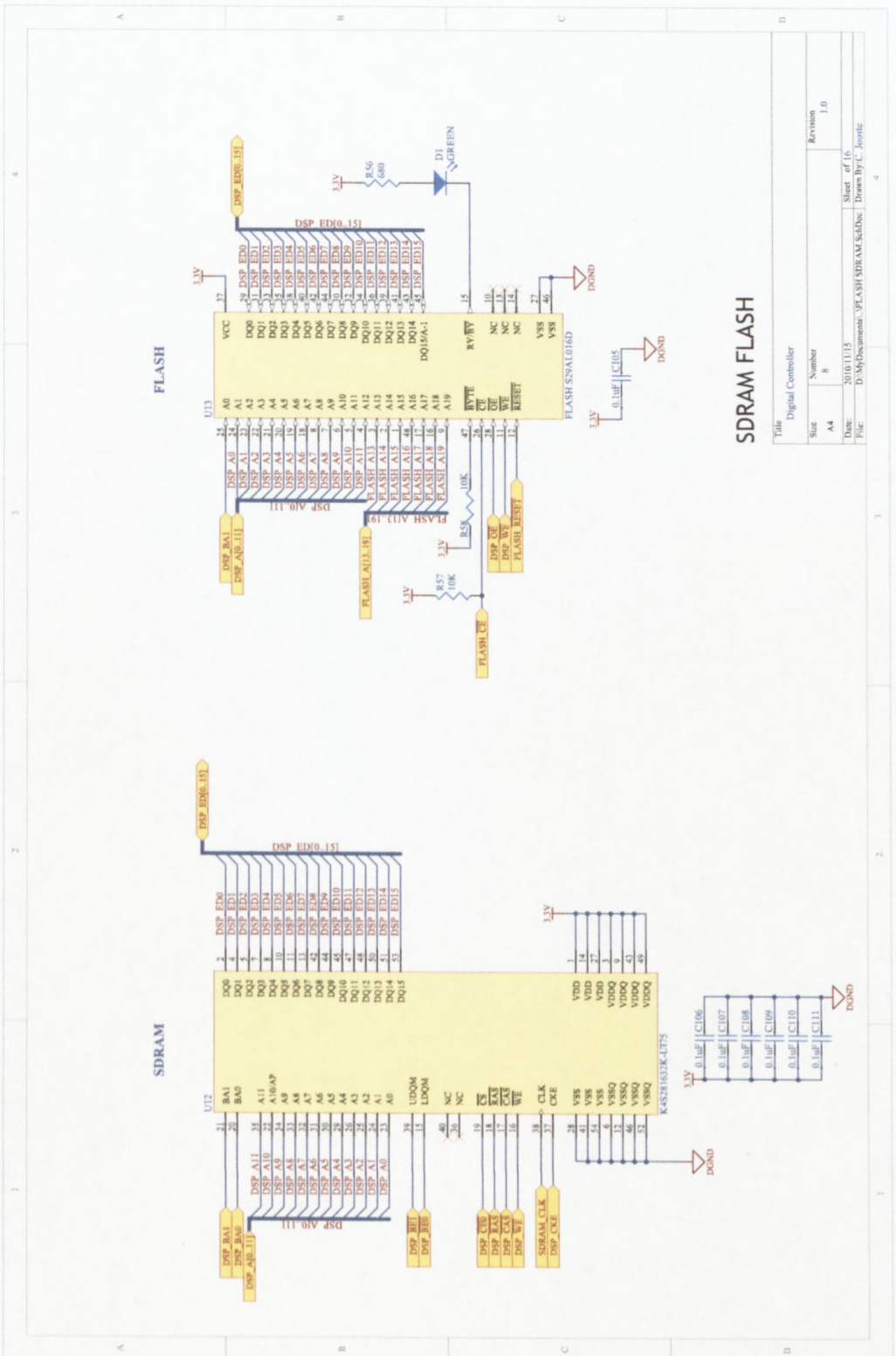

DSP OSC PLL JTAG EEPROM
DSP JTAG
Make sure this is an unshrouded header when populating.
P3
JTAG
R31 4.7K
R30 4.7K
R33 1K
DSP_EMU0
DSP_EMU1
DSP_TRST
DSP_TMS
DSP_TDI
DSP_TDO
DSP_TCK
DSP OSCILLATOR
X2 25 MHz
OSCILLATOR
R32 33
DSP_CLKIN
C75 0.1uF
E1 EMI FILTER EXCCET103U
DSP_PLLHV
C76 0.1uF
C129 10uF
JP5
JUMPER 3-PIN
R34 1K
U6
1Mb EEPROM I2C
C79 0.1uF
DSP_SDA1
DSP_SCL1
3.3V
DGND
Maximize the distance between switching signals and the PLL external components.
Place all PLL external components as close to the DSP. All PLL external components must be on a single side of the board.
Title
Digital Controller
Size A4
Number 3
Revision 1.0
Date: 2010/11/15
Sheet of 16
File: D:\MyDocuments\...\DSP JTAG OSC PLL SchDoc
Drawn By: C. Jocelle

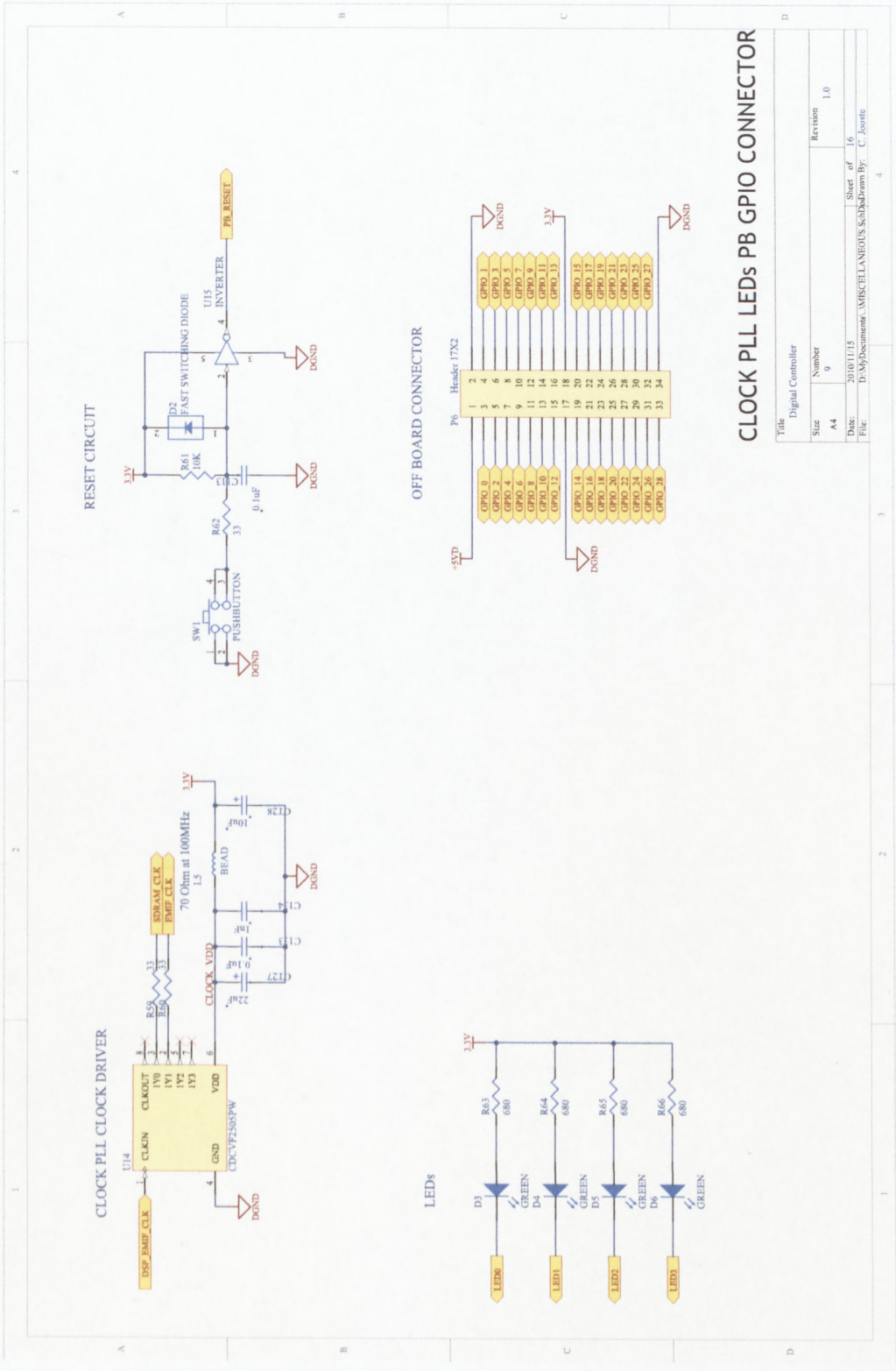

DSP DECOUPLING
Title
Digital Controller
Size
A4
Number
4
Revision
1.0
Date:
2010/11/15
Sheet of
16
File:
D:\MyDocuments\...\DSP DECOUPLING.SchDoc
Drawn By:
C. Jacobs
PLACE DECOUPLING CAPS CLOSE TO DSP
DGND
3.3V
DSP_1.2V
U1C
TMS320C6720
POWERPAD

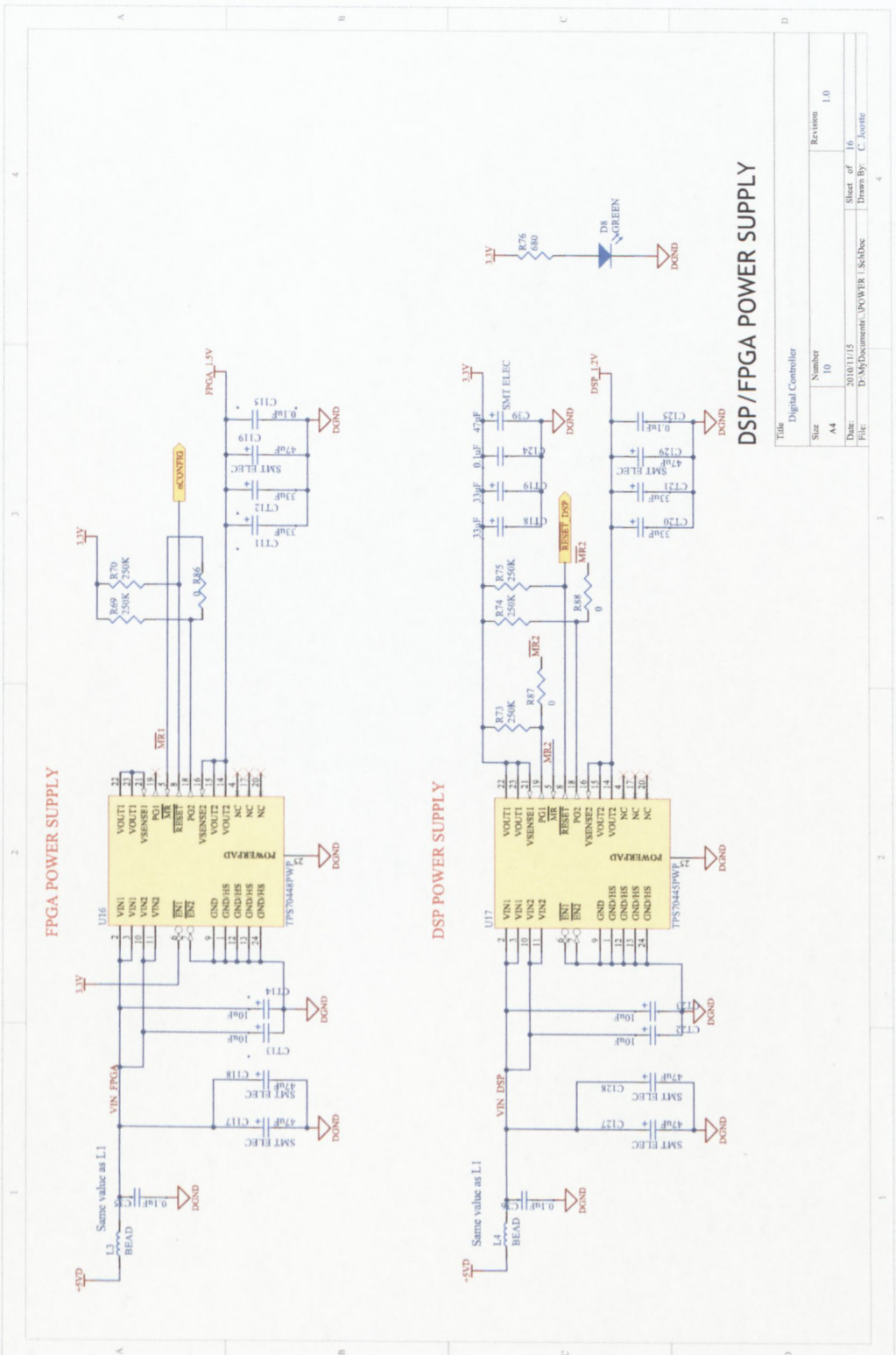

FPGA
Title
Digital Controller
Size
A4
Number
5
Revision
1.0
Date:
2010/11/15
Sheet of
16
File:
D:\MyDocuments\...\FPGA.SchDoc
Drawn By:
C. Jouste
BANK 2
U2B
EP1C12Q240C8N
BANK 1
U2A
EP1C12Q240C8N
TEST POINT
TP1

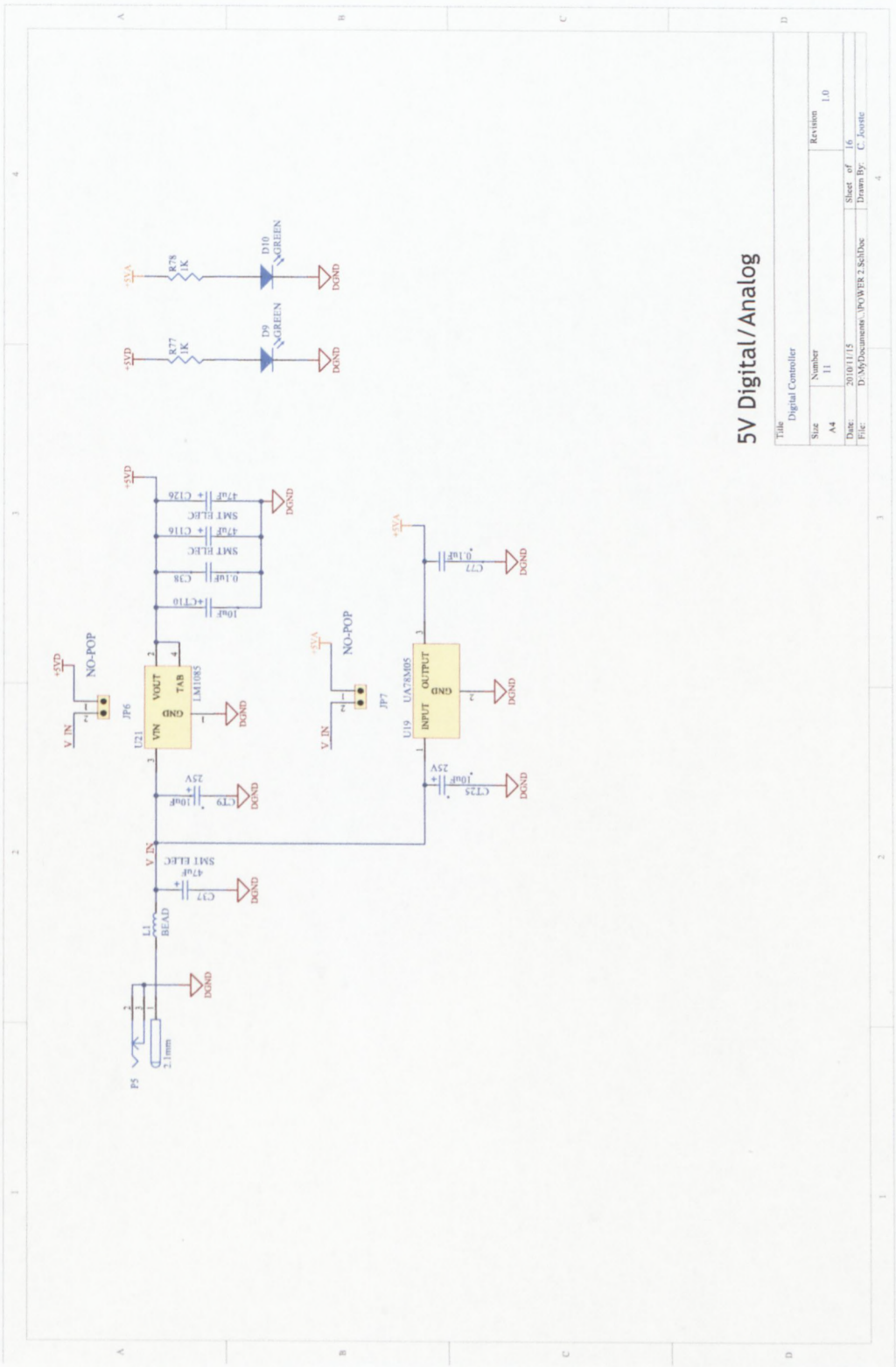

FPGA
Title
Digital Controller
Size
A4
Number
6
Revision
1.0
Date:
2010/11/15
Sheet of
16
File:
D:\MyDocuments\...\FPGA2.SchDoc
Drawn By:
C. Jootle
BANK 4
U2D
EP1C12Q240C8N
BANK 3
U2C
EP1C12Q240C8N
TEST POINT
USB_DATA[0..7]
FLASH_A[13..19]
USB_DATA2
USB_DATA3
USB_DATA4
USB_DATA5
USB_DATA6
USB_DATA7
USB_CS
USB_SUSPEND
USB_INT
FLASH_A16
FLASH_A15
FLASH_A14
FLASH_A13
FLASH_A19
FLASH_RESET
FLASH_A18
FLASH_A17
FLASH_OE
GPIO_20
GPIO_19
GPIO_18
GPIO_17
GPIO_16
GPIO_15
GPIO_14
GPIO_13
GPIO_12
GPIO_11
GPIO_10
GPIO_9
GPIO_8
GPIO_7
GPIO_6
GPIO_5
GPIO_4
GPIO_3
GPIO_2
GPIO_1
GPIO_0
ADC0_FS
TP2
ADC0_SCLK
ADC0_SDIN
ADC1_CS
ADC0_SDOUT
ADC1_FS
ADC1_SCLK
ADC1_SDIN
ADC1_CS
ADC1_SDOUT
DAC_DIN
DAC_SCLK
DAC_CS
USB_A0
USB_EOT
USB_DMACK
USB_DMAREQ
USB_WR
USB_RD
USB_DATA0
USB_DATA1

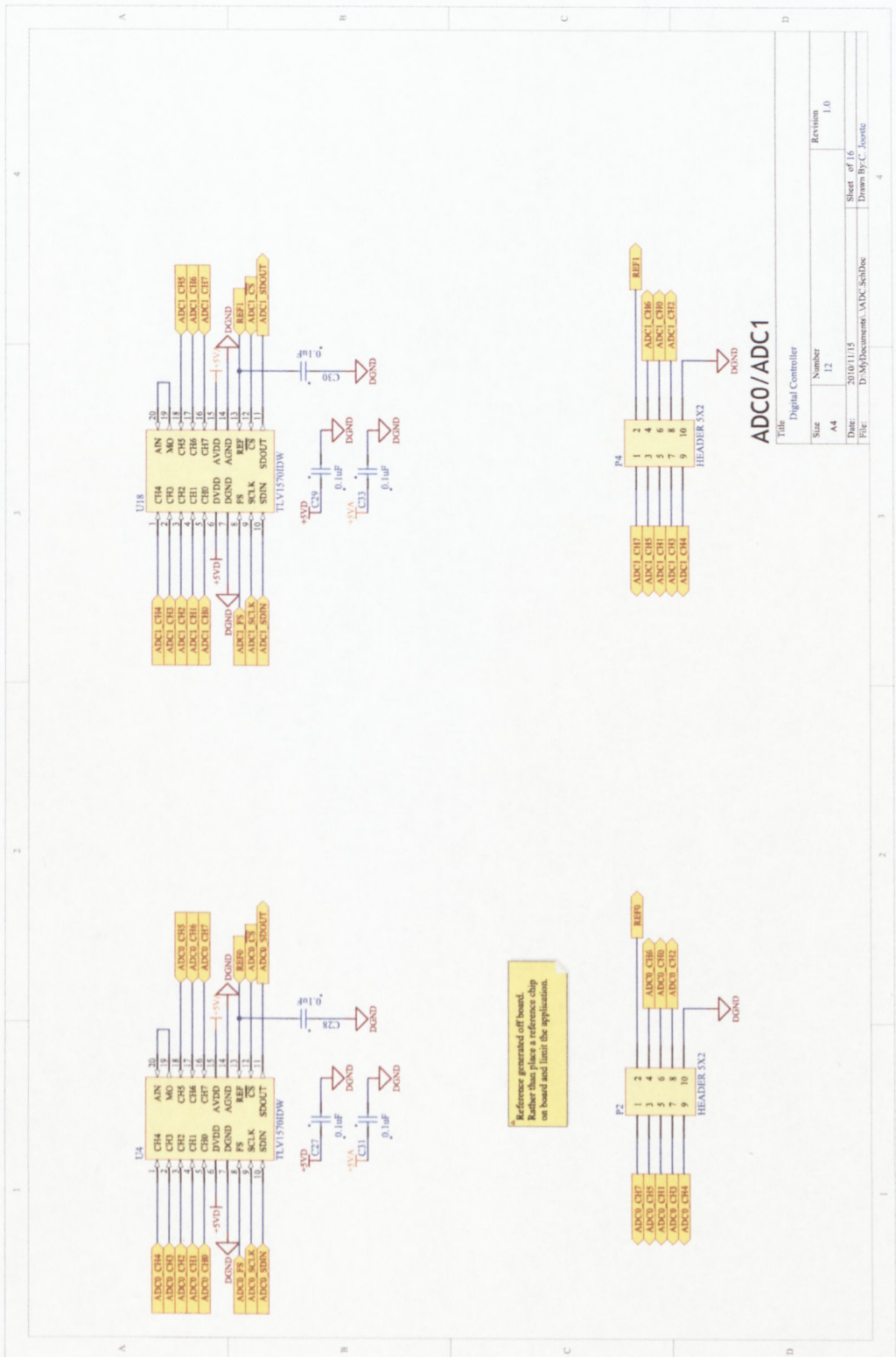

FPGA OSC JTAG PLL
Digital Controller
FPGA JTAG
HEADER 5X2
EPCS4SI8N
EP1C12Q240C8N
FPGA OSCILLATOR
OSCILLATOR
25 MHz
CYCLONE PLL SUPPLY
FERRITE BEAD HIZ AT 50 MHz
A ferrite bead is used to isolate the PLL supply from the digital supply. Everything is doubled here because there are 2 PLLs.
FPGA_1.5V
3.3V
DGND
EMIF CLK
ECLK
CONF_DONE
nCONFIG
nCS
ASDI

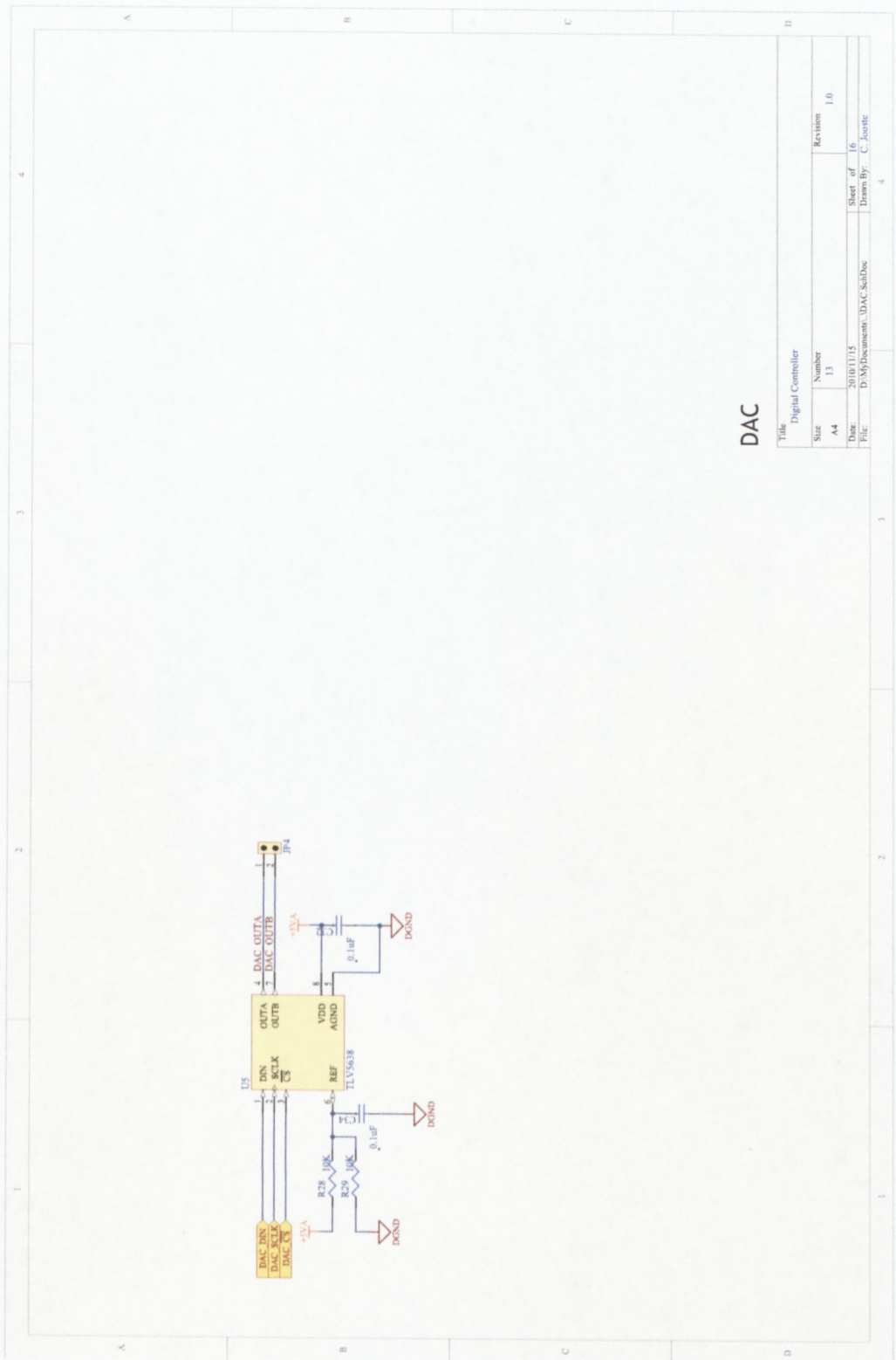

SDRAM FLASH
Title
Digital Controller
Size
A4
Number
8
Revision
1.0
Date:
2010/1/15
Sheet of 16
File:
D:\MyDocuments\...\FLASH SDRAM.SchDoc
Drawn By:
C. Jostle
FLASH
SDRAM
U13
FLASH S29AL016D
U12
K4S281632K-UI75
DSP_ED[0..15]
DSP_EA[0..11]
FLASH_A[0..19]
3.3V
DGND
R56
680
D1
GREEN
R58
10K
R57
10K
C105
0.1uF
C106
C107
C108
C109
C110
C111
FLASH_CE
FLASH_RESET

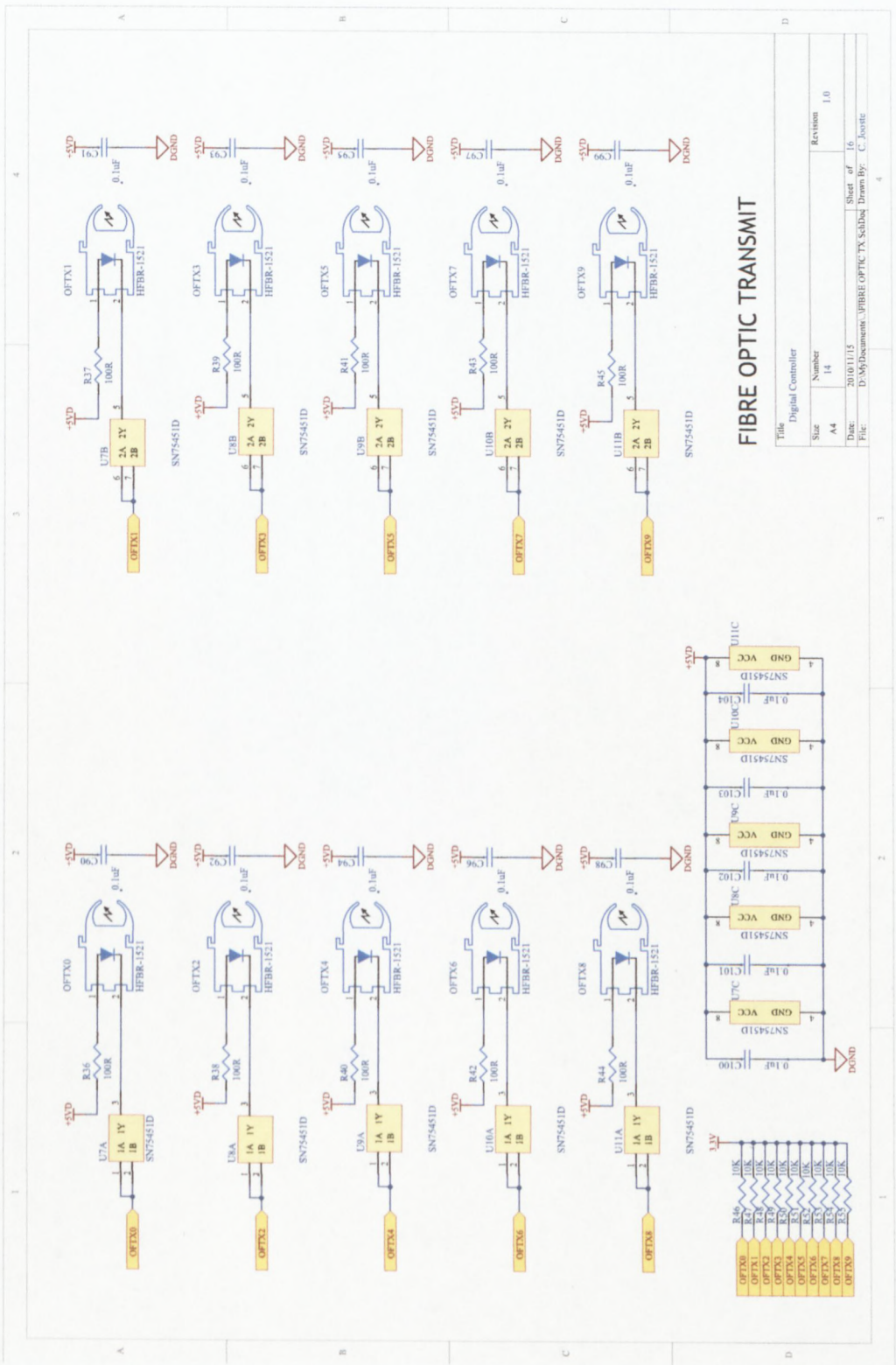

CLOCK PLL LEDs PB GPIO CONNECTOR
RESET CIRCUIT
OFF BOARD CONNECTOR
CLOCK PLL CLOCK DRIVER
LEDs
PB_RESET
U15
INVERTER
D2
FAST SWITCHING DIODE
R61
10K
R62
33
C113
0.1uF
SW1
PUSHBUTTON
P6
Header 17X2
U14
CDCVF2505PW
CLKIN
CLKOUT
VDD
GND
DSP_FMIF_CLK
SDRAM_CLK
FMIF_CLK
70 Ohm at 100MHz
L5
BEAD
CLOCK_VDD
R58
R59
C127
C124
C125
C128
10uF
22uF
0.1uF
1nF
R63
R64
R65
R66
680
D3
D4
D5
D6
GREEN
LED0
LED1
LED2
LED3
3.3V
DGND
-5VD
Title
Digital Controller
Size
A4
Number
9
Revision
1.0
Date:
2010/11/15
Sheet of
16
File:
D:\MyDocuments\..\MISCELLANEOUS.SchDoc
Drawn By:
C. Jooste

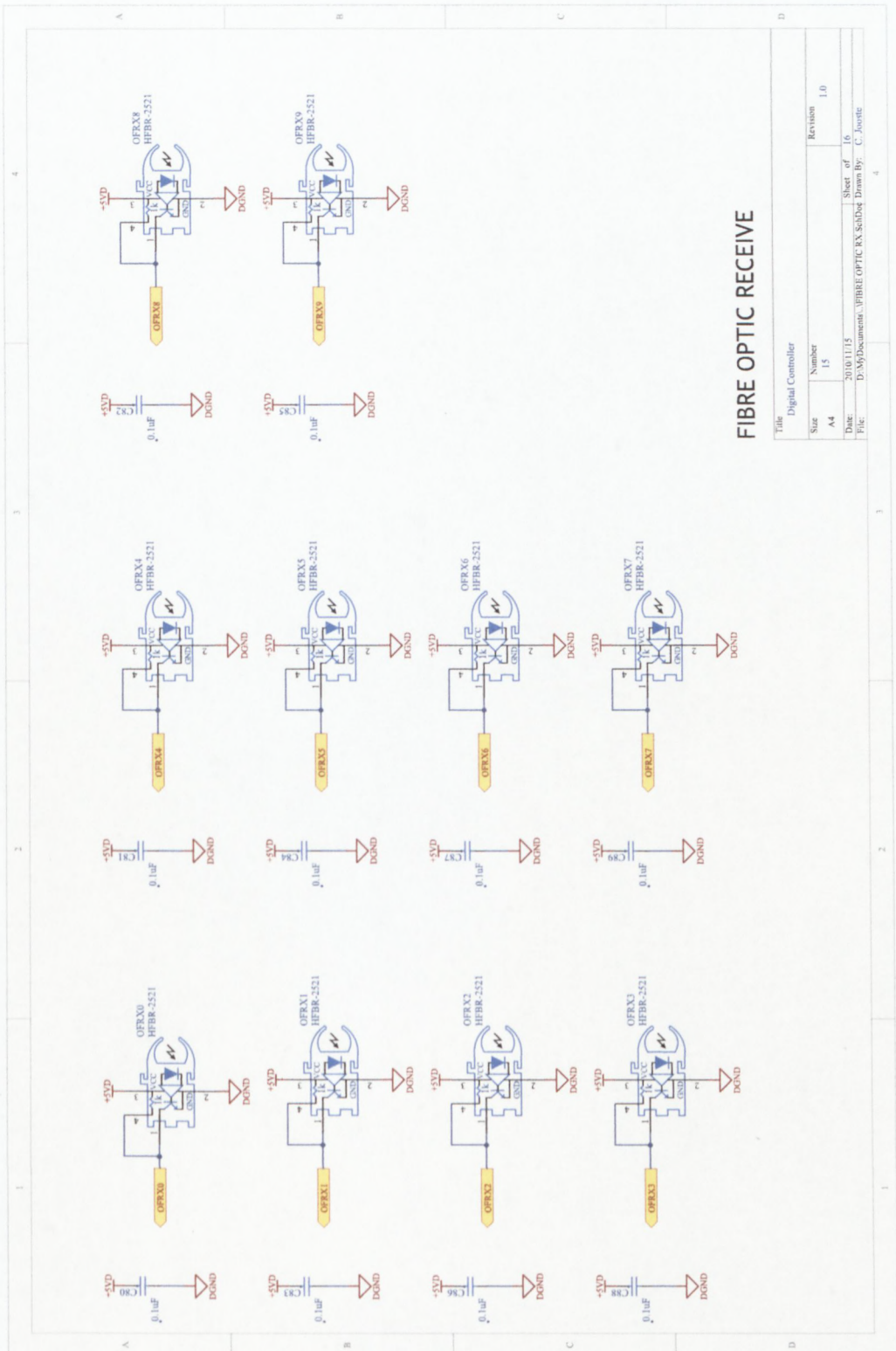

DSP/FPGA POWER SUPPLY
FPGA POWER SUPPLY
DSP POWER SUPPLY
Title
Digital Controller
Size
A4
Number
10
Revision
1.0
Date:
2010/11/15
Sheet of
16
File:
D:\MyDocuments\...\POWER1.SchDoc
Drawn By:
C. Joystic
U16
U17
TPS70448PWP
TPS70445PWP
POWERPAD
VIN_FPGA
VIN_DSP
FPGA_1.5V
DSP_1.5V
3.3V
+5VD
DGND
nCONFIG
RESET_DSP
MR1
MR2
Same value as L1
L3
L4
BEAD
D8
GREEN
R76
680
SMT ELEC

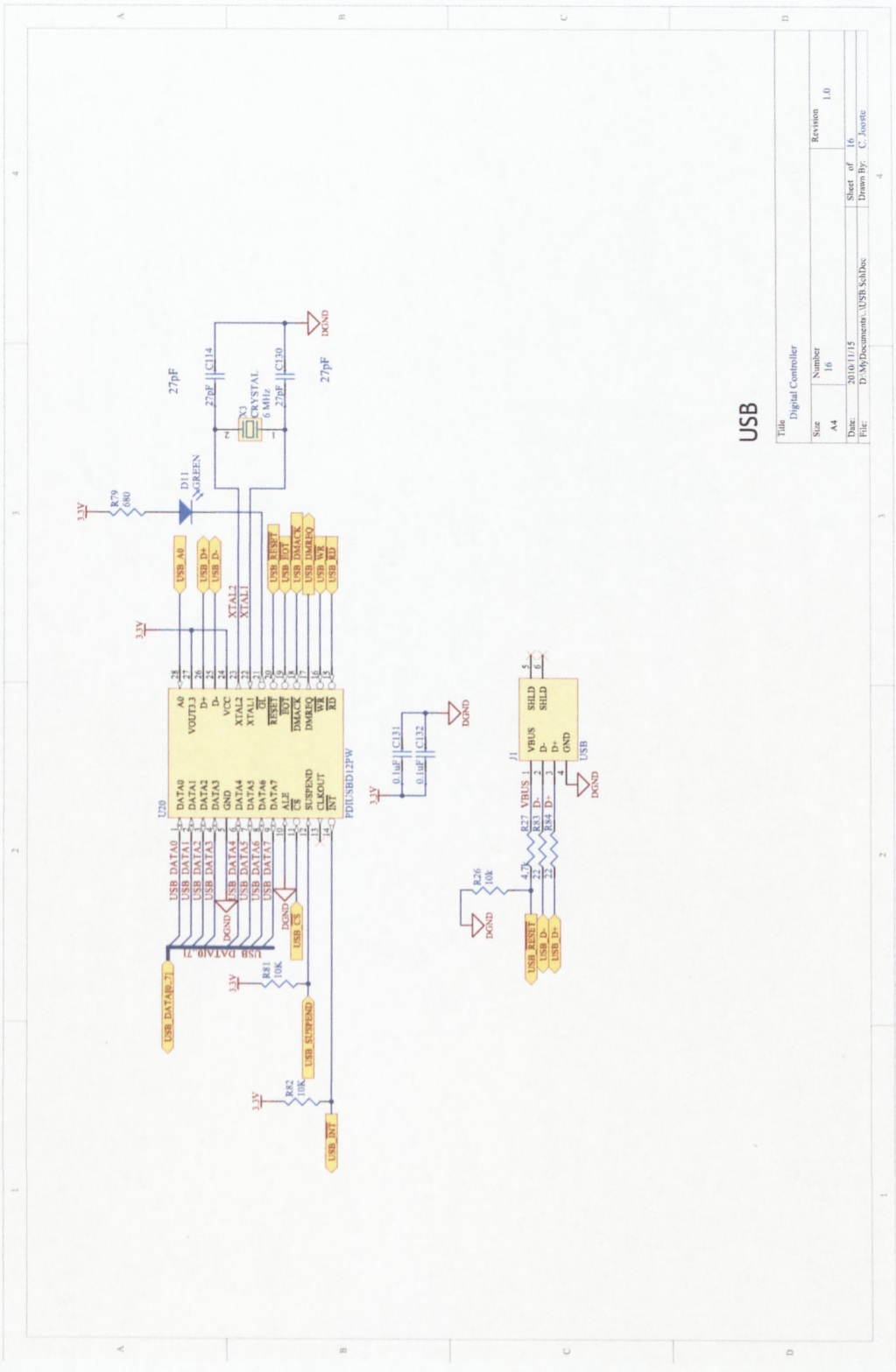

5V Digital/Analog
Title
Digital Controller
Size
A4
Number
11
Revision
1.0
Date:
2010/11/15
Sheet of
16
File:
D:\MyDocuments...\POWER 2.SchDoc
Drawn By:
C. Jooste
+5VD
R78
1K
D10
GREEN
DGND
+5VA
R77
1K
D9
GREEN
DGND
+5VD
C126
47uF
SMT ELEC
C116
47uF
SMT ELEC
C38
0.1uF
C110
10uF
DGND
+5VD
NO-POP
JP6
V_IN
U21
LM1085
VIN
VOUT
TAB
GND
DGND
C79
10uF
25V
DGND
V_IN
C117
47uF
SMT ELEC
DGND
L1
BEAD
DGND
P5
2.1mm
+5VA
C77
0.1uF
DGND
+5VA
NO-POP
JP7
V_IN
U19
UA78M05
INPUT
OUTPUT
GND
DGND
C125
10uF
25V
DGND


ADC0/ADC1
Digital Controller
TLV1570IDW
HEADER 5X2
Reference generated off board.
Rather than place a reference chip
on board and limit the application.


DAC
Title
Digital Controller
Size
A4
Number
13
Revision
1.0
Date: 2010/11/15
Sheet of 16
File: D:\MyDocuments\DAC.SchDoc
Drawn By: C. Jostle
JP4
DAC_OUTA
DAC_OUTB
DAC_DIN
DAC_SCLK
DAC_CS
U5
TLV5638
OUTA
OUTB
VDD
AGND
DIN
SCLK
CS
REF
+5VA
DGND
0.1uF
R28 10K
R29 10K


FIBRE OPTIC TRANSMIT
Title
Digital Controller
Size
A4
Number
14
Revision
1.0
Date:
2010/11/15
Sheet of
16
File:
D:\MyDocuments\...\FIBRE\OPTIC TX.SchDoc
Drawn By:
C. Foote
SN75451D
HFBR-1521
0.1uF
100R
10K
+5VD
DGND
1.1V


FIBRE OPTIC RECEIVE
Title
Digital Controller
Size
A4
Number
15
Revision
1.0
Date:
2010/11/15
Sheet of
16
File:
D:\MyDocuments\..\FIBRE OPTIC RX.SchDoc
Drawn By:
C. Joyrie
OFRX0
OFRX1
OFRX2
OFRX3
OFRX4
OFRX5
OFRX6
OFRX7
OFRX8
OFRX9
HFBR-2521
+5VD
DGND
0.1uF


USB
Title
Digital Controller
Size
A4
Number
16
Revision
1.0
Date:
2010/11/15
Sheet of 16
File:
D:\MyDocuments\USB.SchDoc
Drawn By:
C. Joerke
U20
PDIUSBD12PW
J1
USB
X3
CRYSTAL
6 MHz
27pF
C114
C130
C131
C132
0.1uF
R79
680
D11
GREEN
R81
10K
R82
10K
R26
10k
R27
R33
R34
4.7
22
22
3.3V
DGND
USB_A0
USB_D+
USB_D-
USB_RESET
USB_EOT
USB_DMACK
USB_DMREQ
USB_WR
USB_RD
USB_CS
USB_SUSPEND
USB_INT
USB_DATA[0..7]
XTAL1
XTAL2
VBUS
SHLD
GND

Appendix B

Printed Circuit Boards

B.1 CIRPEC Printed Circuit Boards

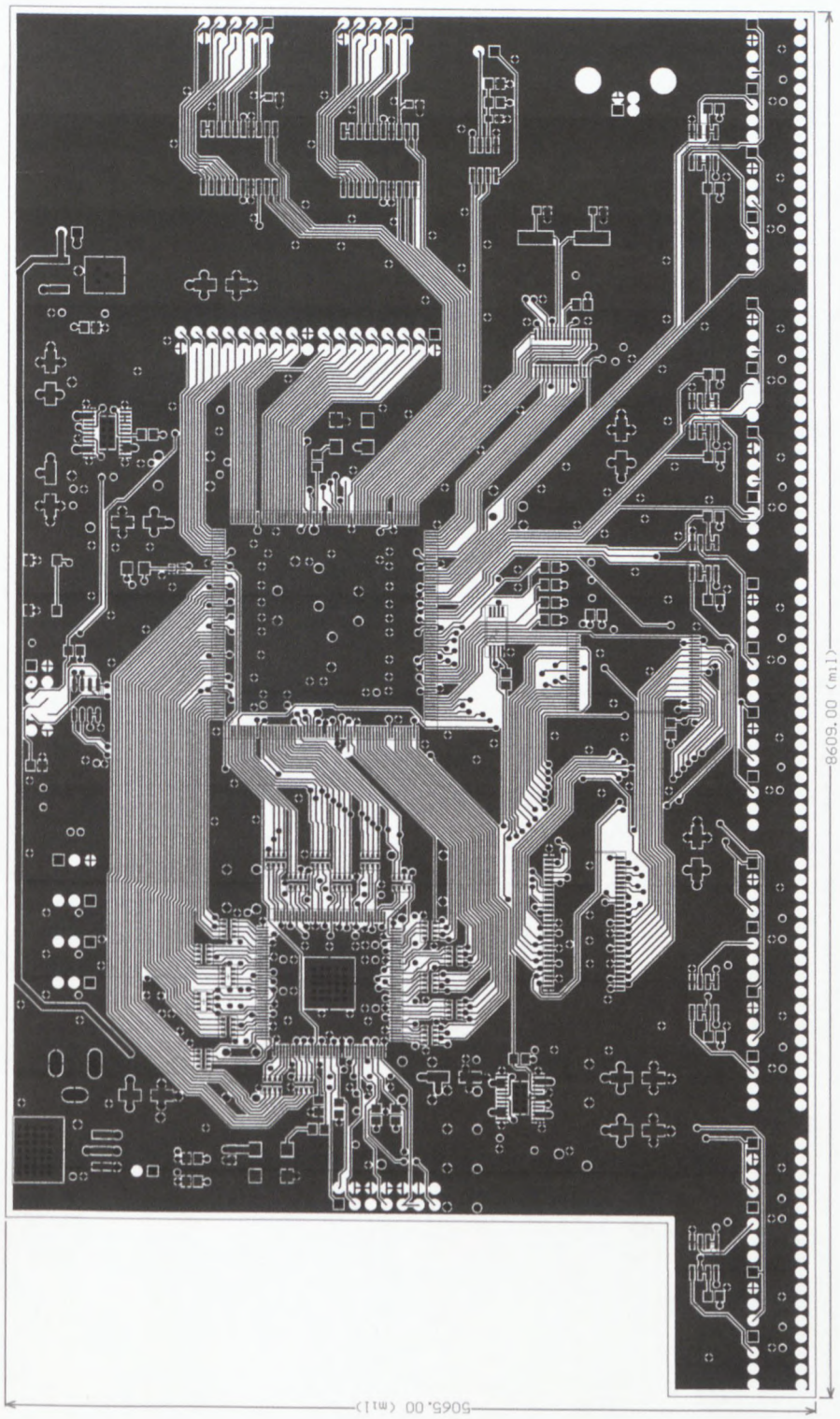


Figure B.1: CIRPEC top layer.

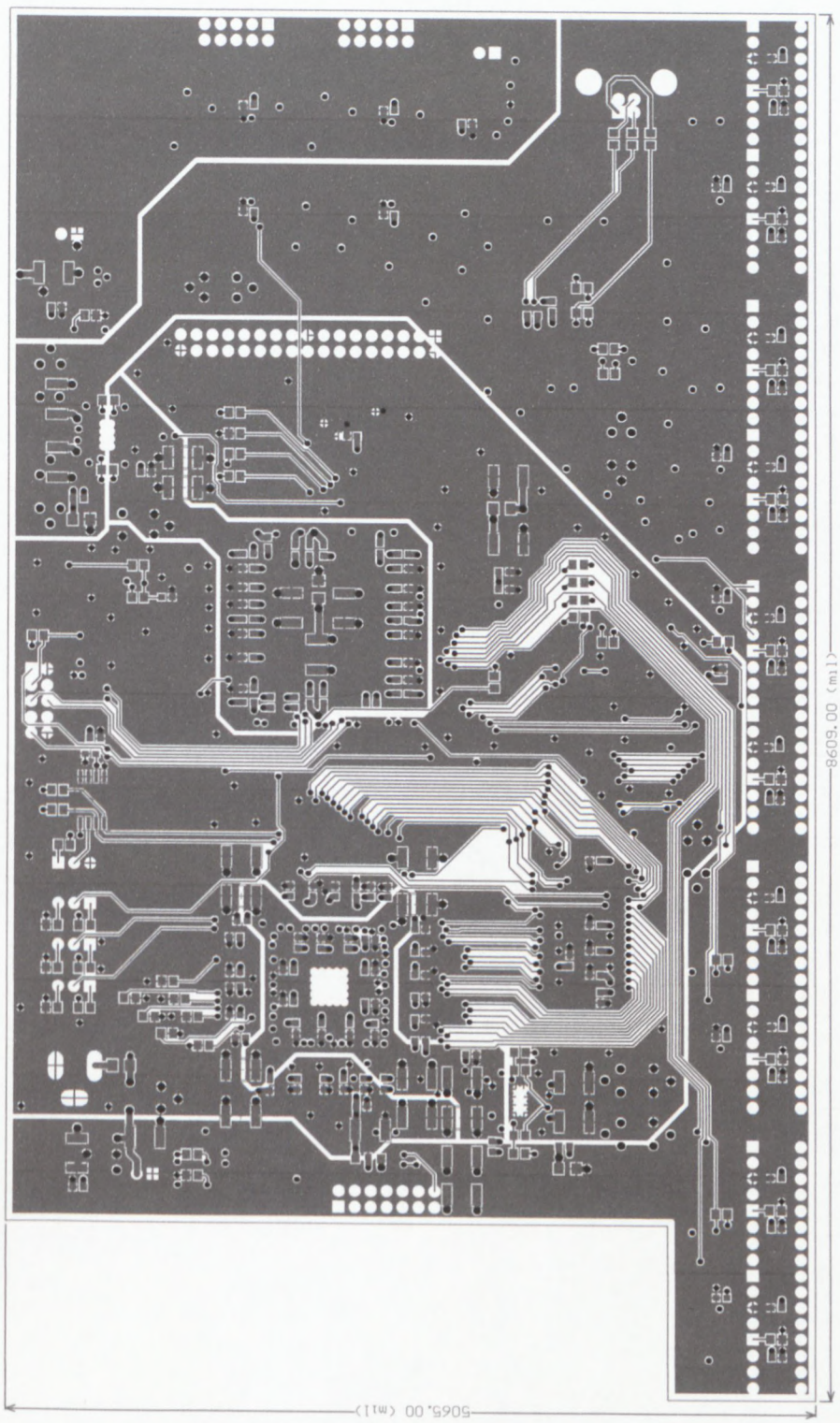


Figure B.2: CIRPEC bottom layer.

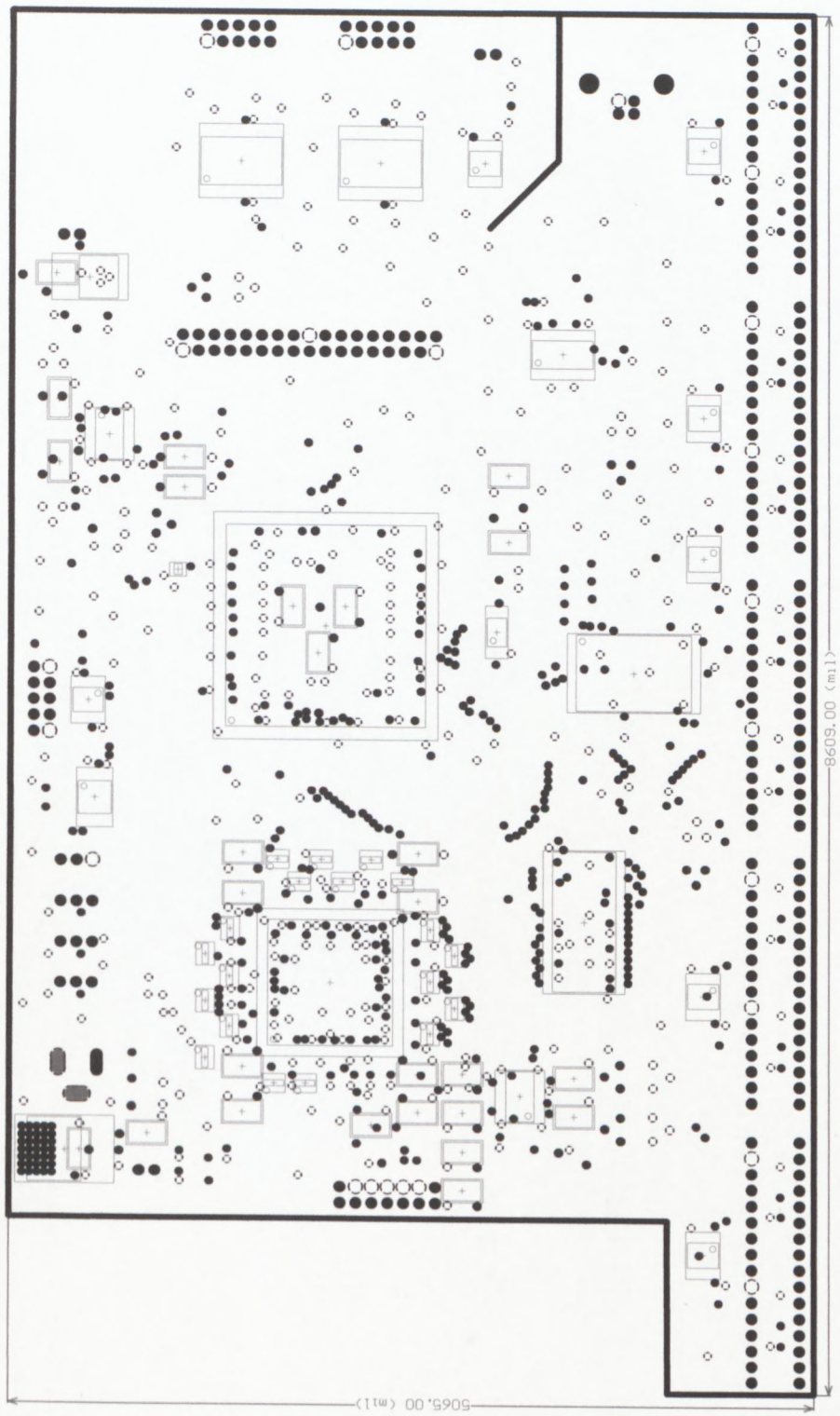


Figure B.3: CIRPEC ground plane.

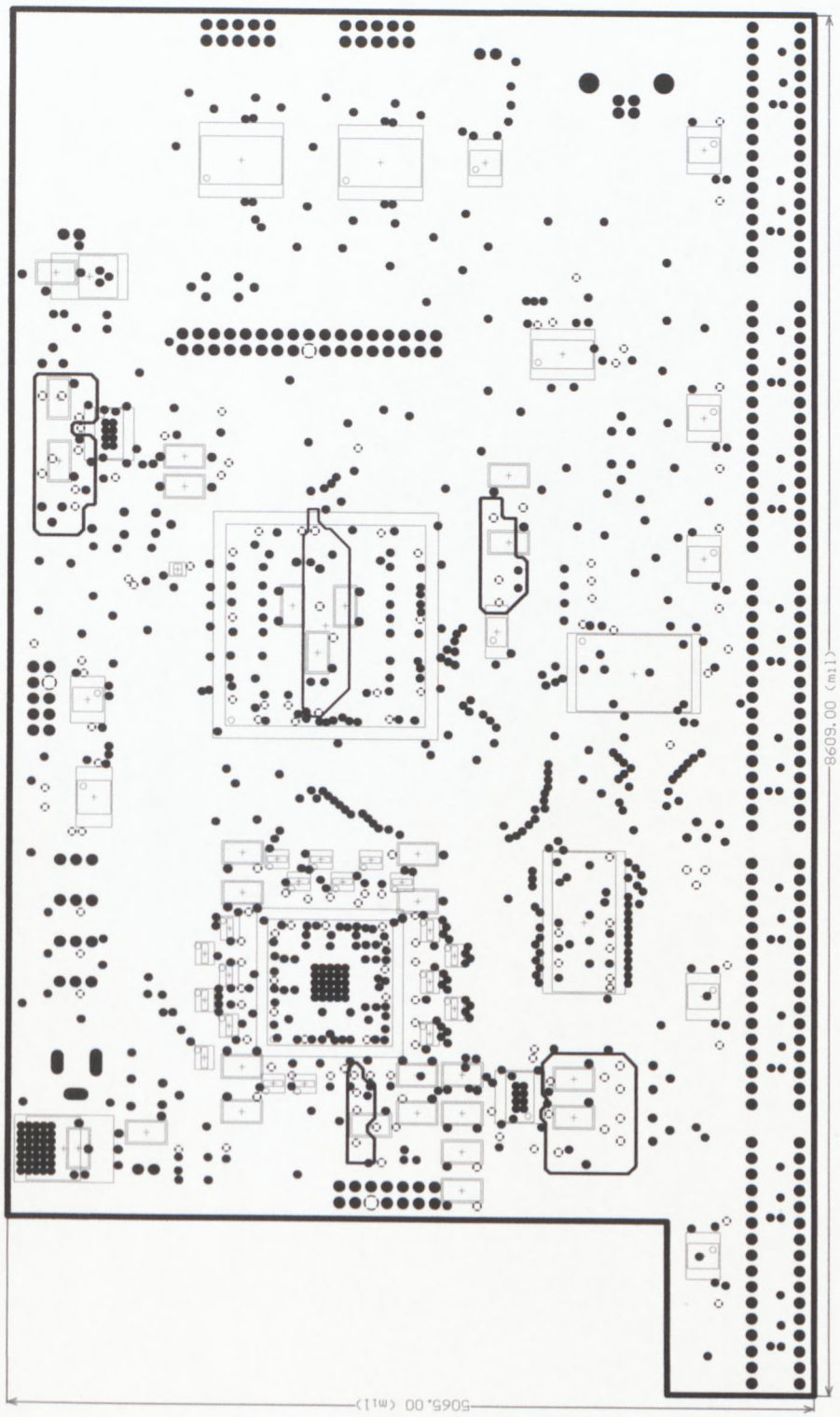


Figure B.4: CIRPEC power layer.

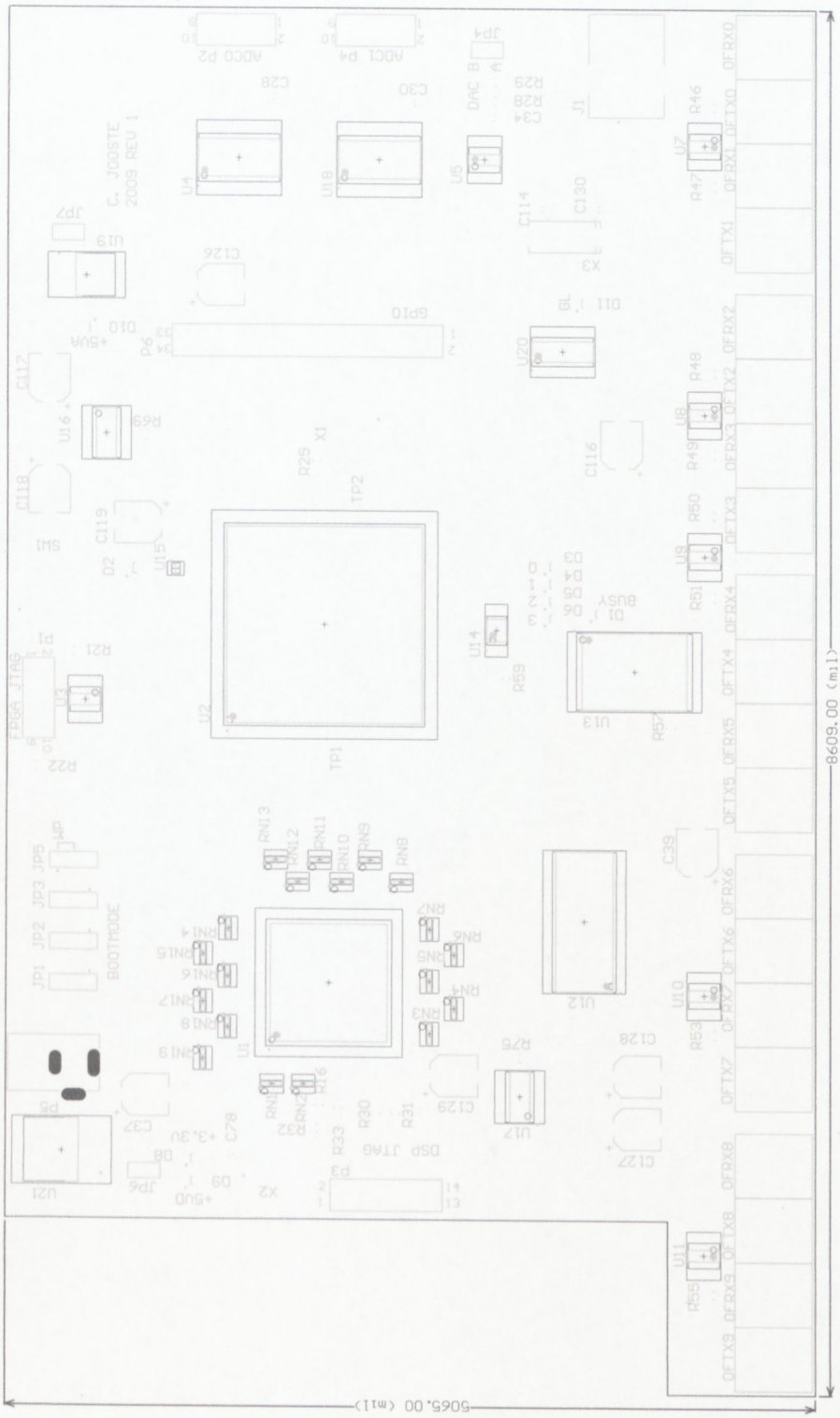


Figure B.5: CIRPEC top silkscreen overlay.

Figure B.6: CIRPEC bottom silkscreen overlay.

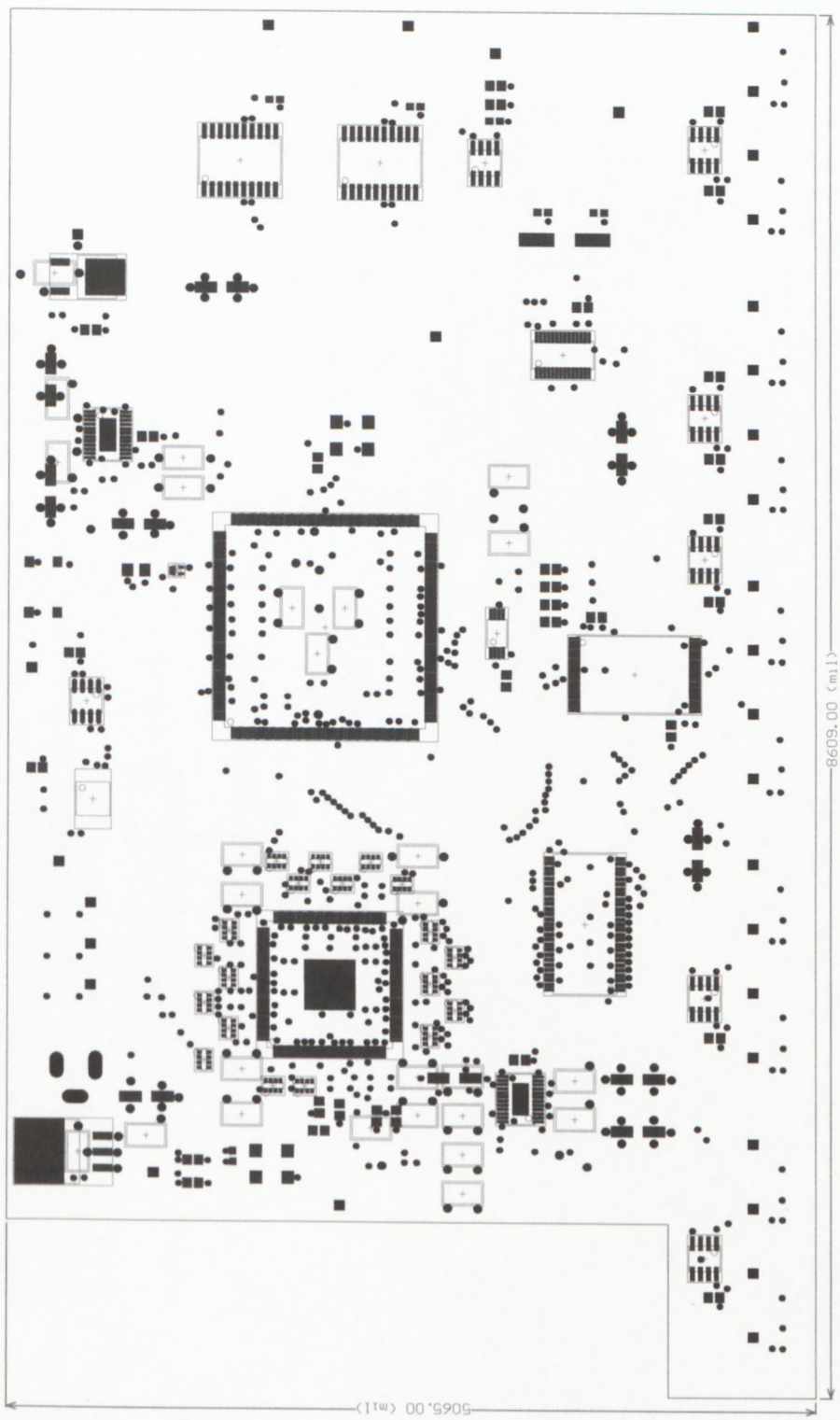


Figure B.7: CIRPEC top solder mask.

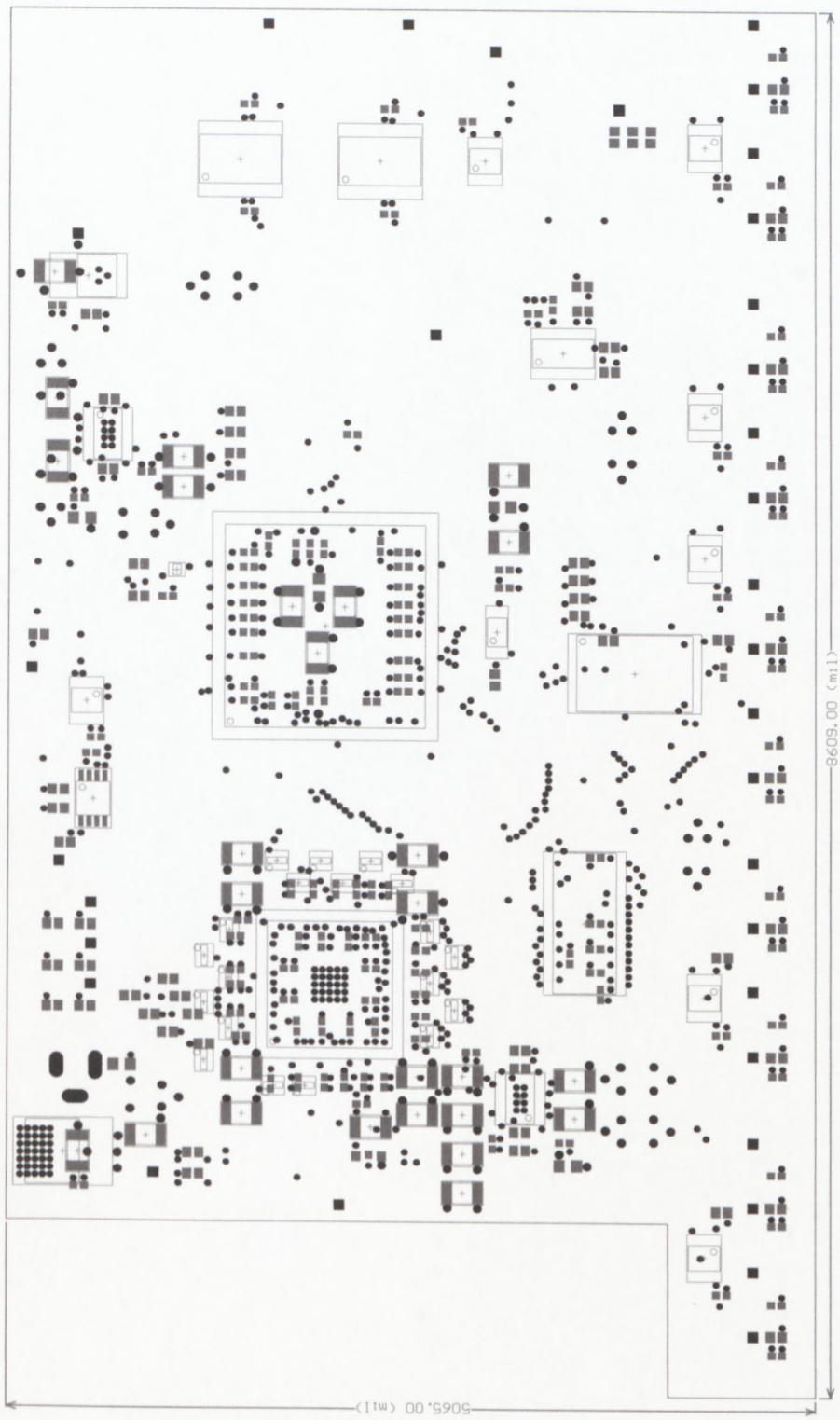


Figure B.8: CIRPEC bottom solder mask.

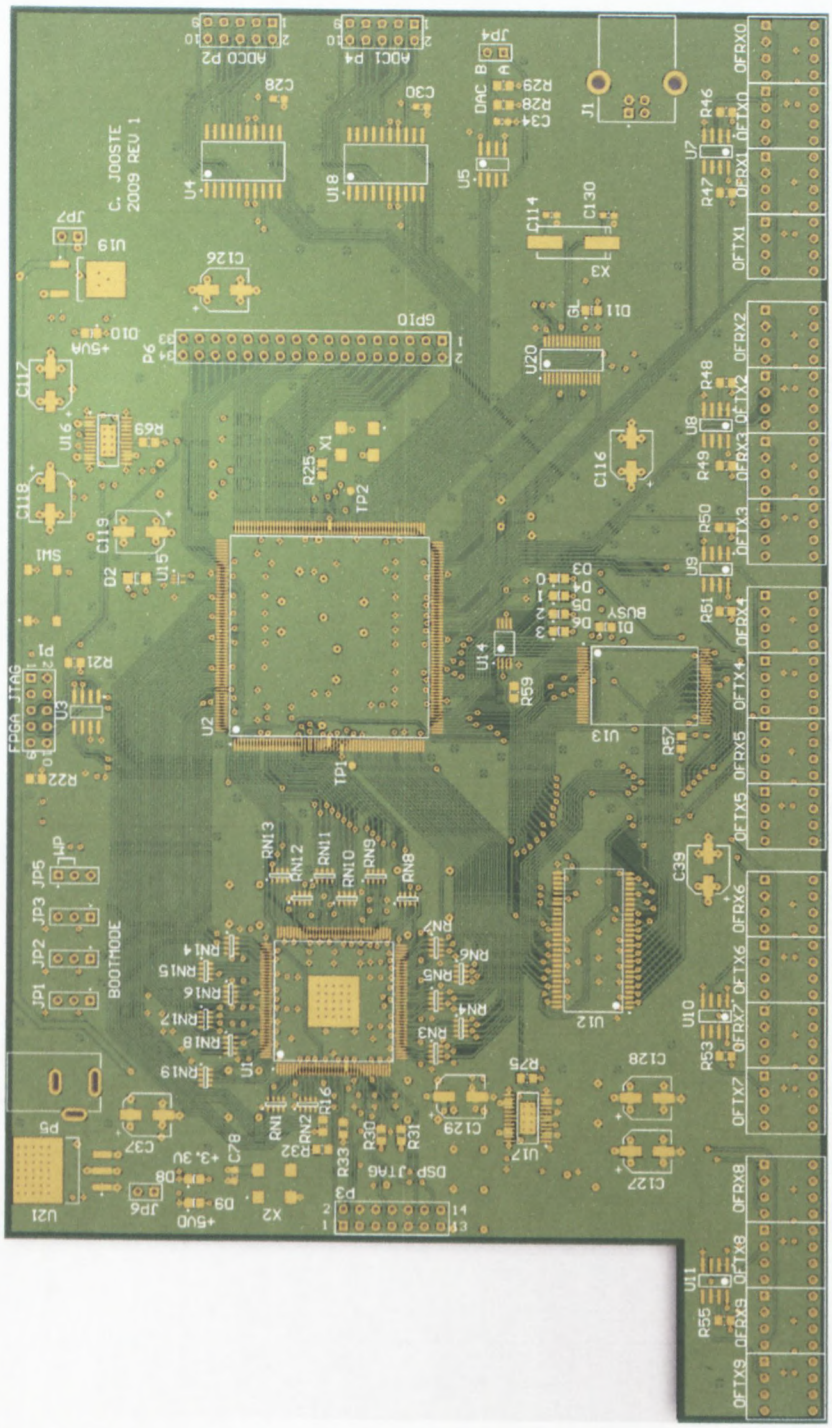


Figure B.9: 3D representation of the top side of the CIRPEC as generated by the Altium Designer software.

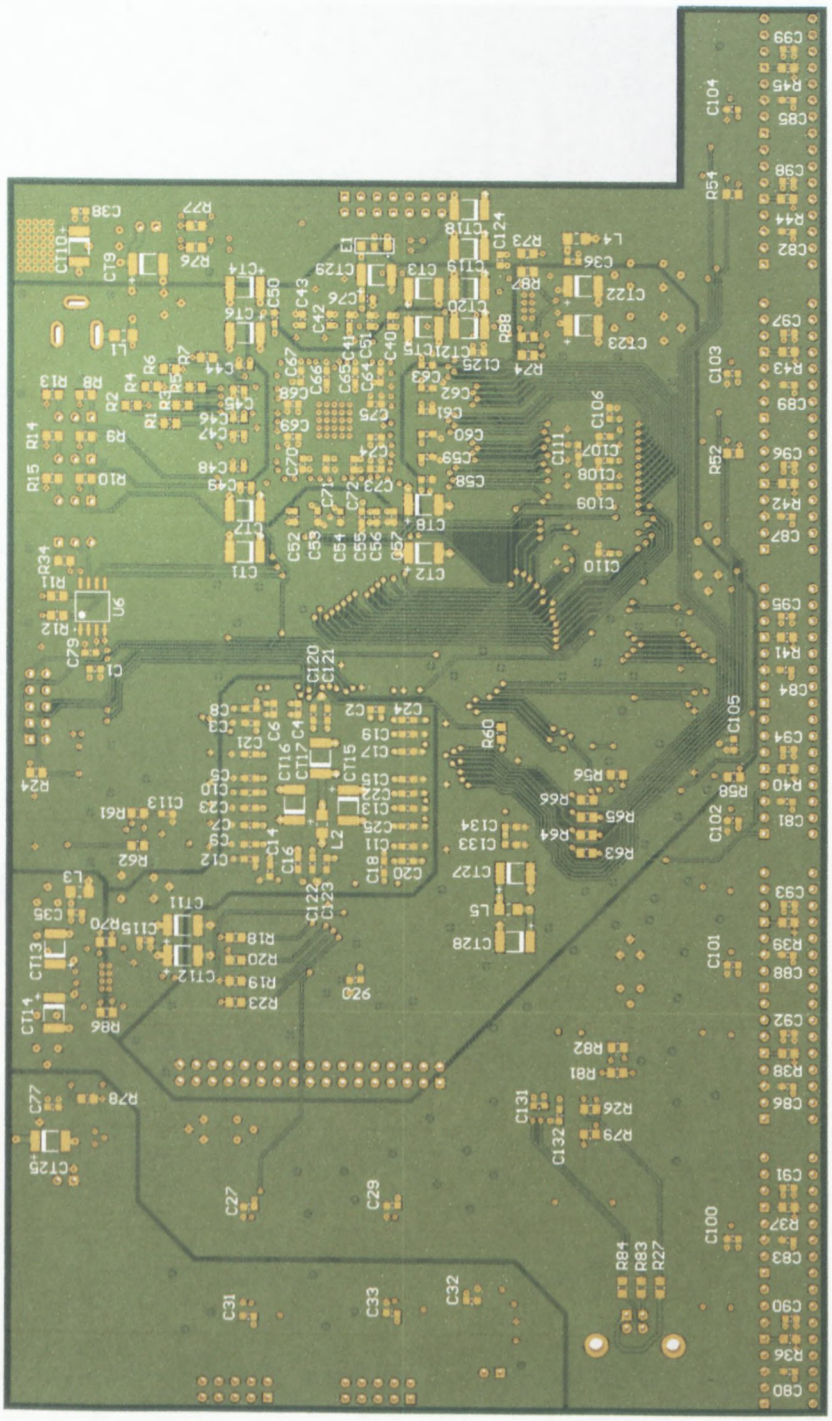


Figure B.10: 3D representation of the bottom side of the CIRPEC as generated by the Altium Designer software.

B.2 CIRPEC Photos

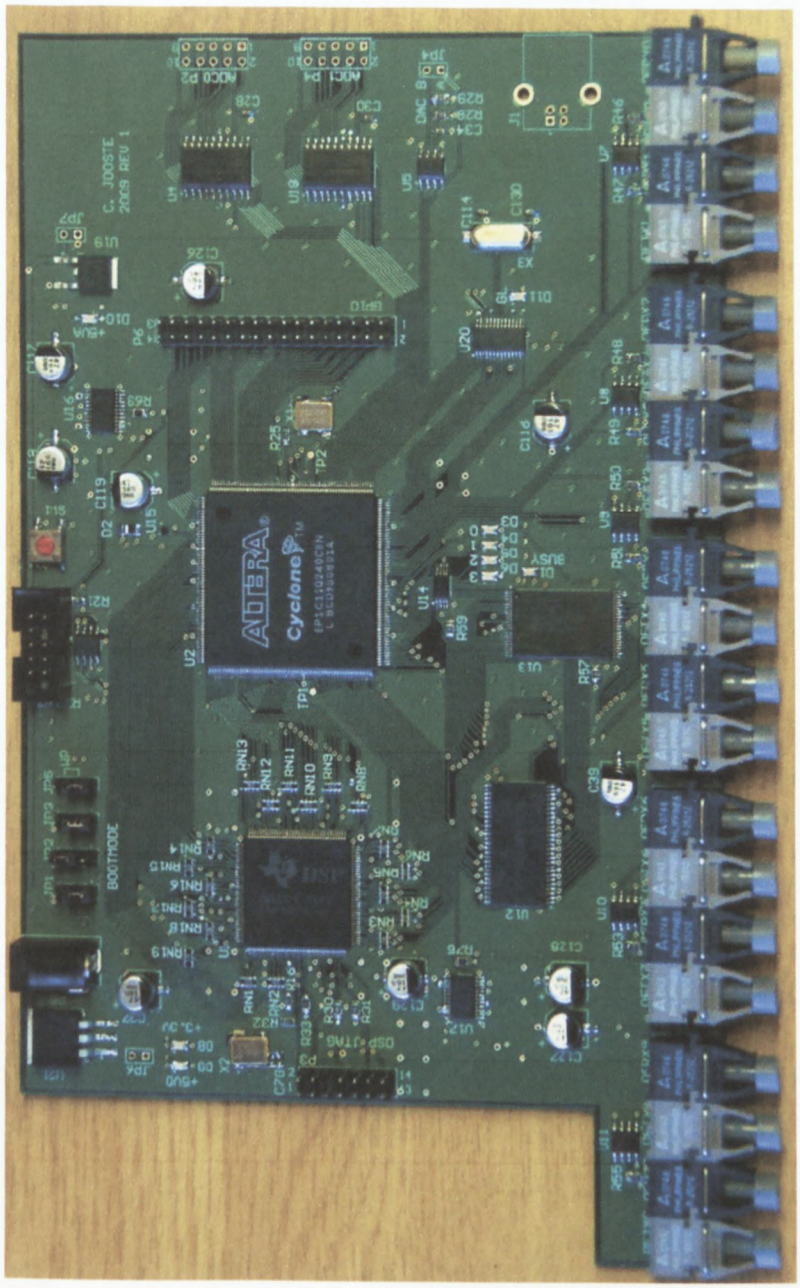


Figure B.11: Top side of the CIRPEC.

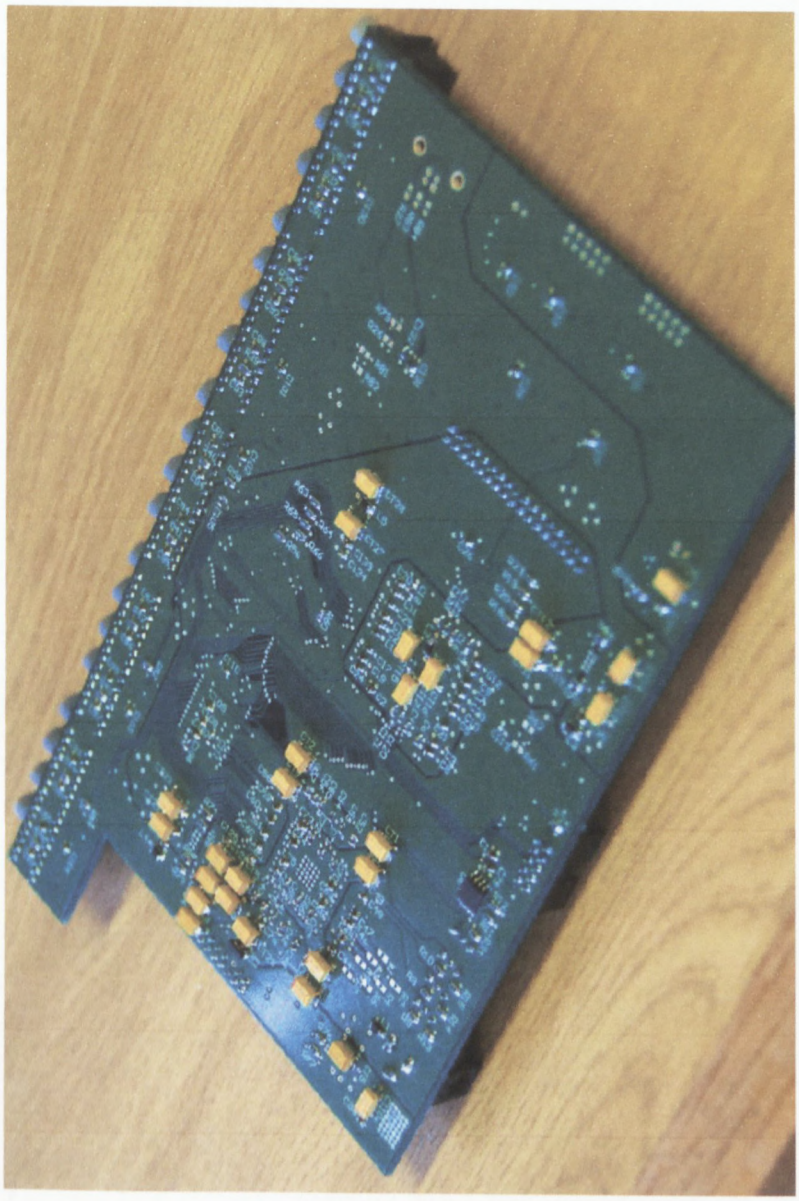


Figure B.12: Bottom side of CIRPEC.

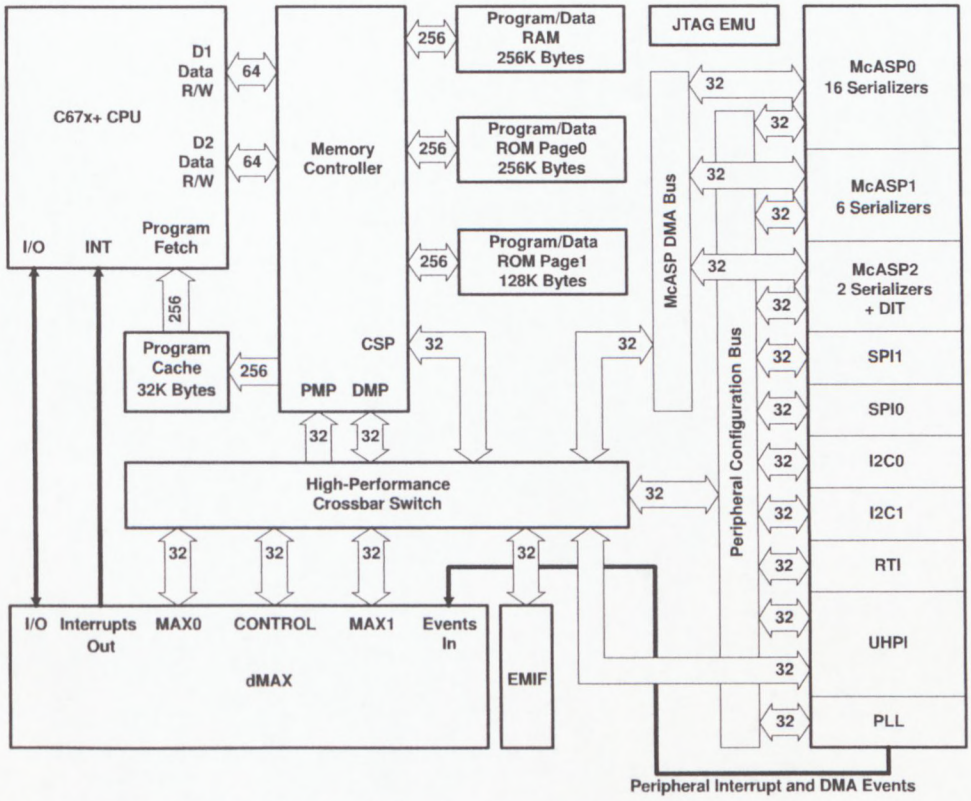
Appendix C

Datasheet Information

C.1 DSP

1.3 Functional Block Diagram

Figure 1-1 shows the functional block diagram of the C672x device.



A. UHPI is available only on the C6727B. McASP2 is not available on the C6722B and C6720.

Figure 1-1. C672x DSP Block Diagram

2.7 Memory Map Summary

A high-level memory map of the C672x DSP appears in Table 2-8. The base address of each region is listed. ***Any address past the end address must not be read or written.*** The table also lists whether the regions are word-addressable or byte- and word-addressable.

Table 2-8. C672x Memory Map

DESCRIPTION	BASE ADDRESS	END ADDRESS	BYTE- OR WORD-ADDRESSABLE
Internal ROM Page 0 (256K Bytes)	0x0000 0000	0x0003 FFFF	Byte and Word
Internal ROM Page 1 (128K Bytes)	0x0004 0000	0x0005 FFFF	Byte and Word
Internal RAM Page 0 (256K Bytes)	0x1000 0000	0x1003 FFFF	Byte and Word
Memory and Cache Control Registers	0x2000 0000	0x2000 001F	Word Only
Emulation Control Registers (Do Not Access)	0x3000 0000	0x3FFF FFFF	Word Only
Device Configuration Registers	0x4000 0000	0x4000 0083	Word Only
PLL Control Registers	0x4100 0000	0x4100 015F	Word Only
Real-time Interrupt (RTI) Control Registers	0x4200 0000	0x4200 00A3	Word Only
Universal Host-Port Interface (UHPI) Registers	0x4300 0000	0x4300 0043	Word Only
McASP0 Control Registers	0x4400 0000	0x4400 02BF	Word Only
McASP1 Control Registers	0x4500 0000	0x4500 02BF	Word Only
McASP2 Control Registers	0x4600 0000	0x4600 02BF	Word Only
SPI0 Control Registers	0x4700 0000	0x4700 007F	Word Only
SPI1 Control Registers	0x4800 0000	0x4800 007F	Word Only
I2C0 Control Registers	0x4900 0000	0x4900 007F	Word Only
I2C1 Control Registers	0x4A00 0000	0x4A00 007F	Word Only
McASP0 DMA Port (any address in this range)	0x5400 0000	0x54FF FFFF	Word Only
McASP1 DMA Port (any address in this range)	0x5500 0000	0x55FF FFFF	Word Only
McASP2 DMA Port (any address in this range)	0x5600 0000	0x56FF FFFF	Word Only
dMAX Control Registers	0x6000 0000	0x6000 008F	Word Only
MAX0 (HiMAX) Event Entry Table	0x6100 8000	0x6100 807F	Byte and Word
Reserved	0x6100 8080	0x6100 809F	
MAX0 (HiMAX) Transfer Entry Table	0x6100 80A0	0x6100 81FF	Byte and Word
MAX1 (LoMAX) Event Entry Table	0x6200 8000	0x6200 807F	Byte and Word
Reserved	0x6200 8080	0x6200 809F	
MAX1 (LoMAX) Transfer Entry Table	0x6200 80A0	0x6200 81FF	Byte and Word
External SDRAM space on EMIF	0x8000 0000	0x8FFF FFFF	Byte and Word
External Asynchronous / Flash space on EMIF	0x9000 0000	0x9FFF FFFF	Byte and Word
EMIF Control Registers	0xF000 0000	0xF000 00BF	Word Only(1)

(1) The upper byte of the EMIF's SDRAM Configuration Register (SDCR[31:24]) is byte-addressable to support placing the EMIF into the Self-Refresh State without triggering the SDRAM Initialization Sequence.

Appendix D

MATLAB

Listing D.1: PWM simulation script

```
% author: Charl Jooste
function [out] = determinePWM(cells,fr,fc,fclk,...
        tri_max_ampl,sine_max_ampl,scale)
% eg. output = determinePWM(5, 20e3, 300e3,
% 100e6, 164, 127, 2)
% cells = no. of cells of inverter
% fr = reference frequency
% fc = carrier frequency
% fclk = frequency of the counter clk
% tri_max_ampl = maximum amplitude of carrier wave
% sine_max_ampl = maximum amplitude of reference wave.
%       Note: sine wave must be 0.8 times the carrier
% waveform (modulation index)
% scale = scaling of amplitude of carrier
% wave in design (comparator block)
close all;

if (nargin ~= 7)
    error('Incorrect number of input arguments');
end

period = 1/fr; % [s]
fsample = 1/fc; % [s]
fclk_s = 1/fclk; % [s]
sine_waveforms = 1; % change to plot more sine waves
total_period = period * sine_waveforms; % total
% period of plotted waveform
```

```
time = 0:fclk_s:total_period;
n=length(time);
for t=1:1:n;

f_r(t)=sine_max_ampl*sin(2*pi*fr*time(t));
for i=1:cells      f_c(i,t)=tri_max_ampl*(2/pi)*...
asin(sin(2*pi*fc*time(t)-((i-1)*2*pi/cells)));
    if f_c(i,t)<f_r(t) s(i,t)=1;
    elseif f_r(t)<f_c(i,t) s(i,t)=-1;
    end
end % end for i
end % end for t

% Figure 1: Plotting a single waveform 'period'
% carrier & reference
 figure;
 plot(time,f_c(1,:),'g');
 hold on;
 plot(time,f_r,'b');
 hold on;
 axis([0 period -(tri_max_ampl+0.05*tri_max_ampl)...
  (tri_max_ampl+0.05*tri_max_ampl)]); % change period
% to total_period if plotting < period.
 title('Interleaved Switching');
 ylabel('Amplitude [pu]');
 xlabel('Time [s]');
 legend('Carrier (f_{c})','Reference (f_r)',...
 'Location','Best');

% Figure 2: Plotting a single waveform 'period'
% of switching function
 figure;
 plot(time,s(1,:)*tri_max_ampl,'r'); %plot
 % switching function
 hold on;
 axis([0 period -(tri_max_ampl+0.05*tri_max_ampl)...
  (tri_max_ampl+0.05*tri_max_ampl)]); % change
 % period to total_period if plotting < period.
 title('Switching Function');
 ylabel('Amplitude [pu]');
 xlabel('Time [s]');

 % Figure 3:plotting all cells
 figure;
```

```
plot(time,f_c(:,:));
hold on;
plot(time,f_r,'b');
hold on;
%  plot(time,s(:,:)*tri_max_ampl);
% plot switching function
%  hold on;
axis([0 period -(tri_max_ampl+0.05*tri_max_ampl)...
 (tri_max_ampl+0.05*tri_max_ampl)]);
%set(gca,'XGrid','on','YGrid','on');
title('Carrier Phase-Shift')
ylabel('Amplitude [pu]');
xlabel('Time [s]');
legend('Carrier (f_{c1})','Carrier (f_{c2})',...
'Carrier (f_{c3})','Carrier (f_{c4})',...
'Carrier (f_{c5})','Reference (f_{r})',...
'Carrier (f_c5)','Location','Best');

% Figure 4: Subplot of interleaved
figure;
for i=1:cells
    subplot(cells,1,i);
    plot(time,f_c(i,:),'g');
    hold on;
    plot(time,f_r,'b')
    hold on;
    plot(time,s(i,:),'r')
    set(gca,'xtick',[],'ytickMode','auto');
     axis([0 period -(tri_max_ampl+0.05*tri_max_ampl)...
      (tri_max_ampl+0.05*tri_max_ampl)]);
    title(['Interleaved Function ',num2str(i)])
    if (round(cells/2) == i)
        ylabel('Amplitude [pu]');
    end
end % for i
% Reset the bottom subplot to have xticks
set(gca,'xtickMode', 'auto')
xlabel('Time [s]');

% Figure 5: Subplot of individual triangles
figure;
for i=1:cells
    subplot(cells,1,i);
    plot(time,f_c(i,:),'g');
```

```
    set(gca,'xtick',[],'ytickMode','auto');
117   axis([0 period -(tri_max_ampl+0.05*tri_max_ampl)...
      (tri_max_ampl+0.05*tri_max_ampl)]);
119   title(['Carrier ',num2str(i)])
      if (round(cells/2) == i)
121       ylabel('Amplitude [pu]');
      end
123 end % for i
  % Reset the bottom subplot to have xticks
125 set(gca,'xtickMode', 'auto')
  xlabel('Time [s]');
127
  % Figure 6: Subplot of individual switching function
129 figure;
  for i=1:cells
131   subplot(cells,1,i);
      plot(time,s(i,:),'r');
133   set(gca,'xtick',[],'ytickMode','auto');
      axis([0 total_period -1.1 1.1]);
135   title(['Switching Function ',num2str(i)])
      if (round(cells/2) == i)
137       ylabel('Amplitude [pu]');
      end
139 end
  % Reset the bottom subplot to have xticks
141 set(gca,'xtickMode', 'auto')
  xlabel('Time [s]');
143
  % Figure 7: Total switching function = 1/2*
145 % (sit +..+ spt) where p are #cells
  figure;
147 total(1,:) = (1/2)*(sum(s(:,:)));
  plot(time,total(1,:),'b');
149 %set(gca,'xtick',[],'ytickMode','auto');
  axis([0 total_period -(max(total)+0.05)...
151  (max(total)+0.05)]);
  title('Total Switching Function');
153 xlabel('Time [s]');
  ylabel('Amplitude of s_{t}(t) [pu]');
155



157
  % return value
159 out = [f_c(1:cells,1)]/scale; % output
```

```
% starting values for design.
end
```

Listing D.2: Logic analyser script

```
% author: Charl Jooste
function [data, pwm] = logicAnalyser(cells,
complementary, index)
% [data, pwm] = logicAnalyser(1,0,9180);
% [data, pwm] = logicAnalyser(2,0,13420);
% [data, pwm] = logicAnalyser(3,0,4250);
% [data, pwm] = logicAnalyser(5,0,11500);
% [data, pwm] = logicAnalyser(5,1,11500);
% for complementary signals
% cells = no. of cells to be presented
% complementary = must the complementary signals be
%displayed? yes=1/no=0
% index = start index of sample
% include a select period/segment of results.
close all;

if (nargin ~= 3)
    error('Incorrect number of input arguments');
end %if nargin

% options for exportfig
if (complementary == 1 && (cells == 3 || cells == 5))
% make height of figure larger for the 3-cell and four
% cell comp signals.
    opts = struct('LockAxes',0,'FontMode','fixed',
    'FontSize',8,'height',20,'color','rgb');
else
    opts = struct('LockAxes',0,'FontMode','fixed',
    'FontSize',8,'height',10,'color','rgb');
end;

switch (cells)
    % 1 cell
    case 1
    data = csvread('D:\!Repositories\Thesis Repository
    \Thesis\figs\Results\LogicAnalyser\csv\1-cell\
    bus100.csv',1,0); % 1 to skip first row
    % 2 cell
    case 2
```

```
40      data = csvread('D:\!Repositories\Thesis_Repository
    _____\Thesis\figs\Results\LogicAnalyser\csv\2-cell\
42  _____bus100.csv',1,0); % 1 to skip first row
        % 3 cell
44      case 3
        data = csvread('D:\!Repositories\Thesis_Repository
46  _____\Thesis\figs\Results\LogicAnalyser\csv\3-cell\
    _____bus100.csv',1,0); % 1 to skip first row
48      % 5 cell
        case 5
50      data = csvread('D:\!Repositories\Thesis_Repository
    _____\Thesis\figs\Results\LogicAnalyser\csv\5-cell\
52  _____bus100.csv',1,0);
    end; % switch
54
    fs = 400e6; % sample rate of logic analyser
56  Ts = 1/fs; % 2.5 ns for logic analyser
    samples = 0:20000; % 20000 samples = 50e-9
58  n = length(samples);
    period = 1/fs;
60
    t=1:1:n;
62
     for i=1:cells*2 % *2 for the complementary pair
64   %ie, 1 cell has 2 columns of data.
         pwm(t,i) = data(index:(index-1+n),i);
66       %pwm(t,i) = data(1:n,i); % was this
     end;
68
     timeaxis = samples.*Ts;
70   column = 1; % start at this column

72   if (complementary == 1) % if complementary is selected
     % as yes then display
74       columnskip = 1;      % every column;
         cells = cells * 2;
76   else
         columnskip = 2;
78   end;

80   count = 1;
     for i=1:cells
82   subplot(cells,1,i);
     plot(timeaxis, pwm(:,column));
```

```
84  set(gca,'xtick',[],'ytickMode','auto');
    axis([0 45.45e-6 -0.025 1.025]);
86  if (cells > 1)
        if (complementary == 1)
88          if ((mod(i,2)) == 0)
                title(['Complementary PWM gating signal for
90  cell ',num2str(count-1)]);
            else
92              title(['PWM gating signal for cell ',
                num2str(count)]);
94              count = count+1;
            end;
96      else
            title(['PWM gating signal for cell ',
98          num2str(i)]);
        end;
100 else
        title('PWM gating signal');
102 end;
    if (round(cells/2) == i)
104         ylabel('Amplitude [pu]');
    end
106
        if (i ~= cells)
108         column = column + columnskip;
%           figure;
110     end;
    end;
112 % Reset the bottom subplot to have xticks
    set(gca,'xtickMode', 'auto');
114 xlabel('Time [s]');


116
    if (complementary == 1)
118     cells = cells/2;
    end;
120
    switch (cells)
122 % 1 cell
        case 1
124         if (complementary == 1)
                exportfig(gcf,'logicAnalyser_1cell_comp.eps'
126             ,opts);
            else
```

```
128             exportfig(gcf,'logicAnalyser_1cell.eps'
                ,opts);
130         end;
    % 2 cell
132     case 2
            if (complementary == 1)
134             exportfig(gcf,'logicAnalyser_2cell_comp.eps'
                ,opts);
136         else
                exportfig(gcf,'logicAnalyser_2cell.eps'
138             ,opts);
            end;
140 % 3 cell
        case 3
142         if (complementary == 1)
                exportfig(gcf,'logicAnalyser_3cell_comp.eps'
144             ,opts);
            else
146             exportfig(gcf,'logicAnalyser_3cell.eps'
                ,opts);
148         end;
    % 5 cell
150     case 5
            if (complementary == 1)
152             exportfig(gcf,'logicAnalyser_5cell_comp.eps'
                ,opts);
154         else
                exportfig(gcf,'logicAnalyser_5cell.eps'
156             ,opts);
            end;
158 end; %switch
    end %end function
```