The Design of a High-Performance, Floating-Point Embedded System for Speech Recognition and Audio Research Purposes

William Duckitt

Thesis presented in partial fulfilment of the requirements for the degree
Master of Science in Electronic Engineering
at the University of Stellenbosch

Supervisor: Dr TR Niesler
Co-supervisor: Mr W Smit

December 2007

Declaration

I, the undersigned, hereby declare that the work contained in this thesis is my own original work and that I have not previously in its entirety or in part submitted it at any university for a degree.

Signature:.. Date:.....................................

Abstract

This thesis describes the design of a high performance, floating-point, standalone embedded system that is appropriate for speech and audio processing purposes.

The system successfully employs the Analog Devices TigerSHARC TS201 600MHz floating point digital signal processor as a CPU, and includes 512MB RAM, a Compact Flash storage card interface as non-volatile memory, a multi-channel audio input and output system with two programmable microphone preamplifiers offering up to 65dB gain, a USB interface, a LCD display and a push-button user interface.

An Altera Cyclone II FPGA is used to interface the CPU with the various peripheral components. The FIFO buffers within the FPGA allow bulk DMA transfers of audio data for minimal processor delays. Similar approaches are taken for communication with the USB interface, the Compact Flash storage card and the LCD display.

A logic analyzer interface allows system debugging via the FPGA. This interface can also in future be used to interface to additional components. The power distribution required a total of 11 different supplies to be provided with a total consumption of 16.8W. A 6 layer PCB incorporating 4 signal layers, a power plane and ground plane was designed for the final prototype.

All system components were verified to be operating correctly by means of appropriate testing software, and the computational performance was measured by repeated calculation of a multi-dimensional Gaussian log-probability and found to be comparable with an Intel 1.8GHz Core2Duo processor.

The design can therefore be considered a success, and the prototype is ready for development of suitable speech or audio processing software.

Opsomming

Die tesis beskryf die ontwerp van ‘n hoë werk verrigting, wisselpunt stelsel wat geskik is vir spraak-en klank toepassings.

Die stelsel maak gebruik van ‘n Analog Devices TigerSHARC TS201 600MHz wisselpunt digitale seinverwerker as sentrale verwerker met 512MB RAM. Dit gebruik ‘n Compact Flash geheuekaart as lees-alleen-geheue en het ‘n multi-kanaal klank inset/uitset poorte met twee programmeerbare mikrofoon voor-versterkers wat tot 65dB aanwins lewer. Dit het ook ‘n USB koppelvlak, ‘n LCD skerm en ‘n gebruikerskoppelvlak.

‘n Altera Cyclone II FPGA word gebruik om met die CPU en verskeie ander komponente te kommunikeer. Die FIFO buffers binne die FPGA laat DMA blokoordrag van klank data toe met minimale verwerkervertragings. Die kommunikasie word op ‘n soortgelyke manier word gedoen met die USB koppelvlak, die Compact Flash kaart en die LCD skerm.

Die logiese analiseerderkoppelvlak laat stelselontfouting toe via die FPGA. Die koppelvlak kan ook in die toekoms gebruik word om met bykomende komponente te kommunikeer. Die kragstelsel het ‘n totaal van 11 verskillende toevoere benodig, met ‘n totale verbruik van 16.8W. Die finale prototipe gebruik ‘n 6 vlak gedrukte etsbaanboard met 4 sein lae, een krag laag en ‘n grond laag.

Alle stelselkomponente is geverifieer om korrek te funksioneer met gepaste toets sagteware, en die berekeningswerksverrigting is gemeet deur herhaalde berekening van ‘n multi-dimensionele Gauss logwaarskynlikheid. Dit is gevind om vergelykbaar te wees met ‘n Intel 1.8GHz Core2Duo verwerker.

Die ontwerp is daarom ‘n sukses en die prototipe is reg vir ontwikkeling van gepaste spraak en klank verwerkingssagteware.

Acknowledgements

To my supervisor, Dr Thomas Niesler, for your tremendous support and guidance, and for providing all the necessary equipment to successfully implement this design.

To Willem Smit, for all your advice in the field of embedded design.

To my parents and my brother, for their enduring support throughout my studies.

Contents

List of Figures

List of Tables

Abbreviations

ACK	Acknowledge
ADC	Analog-to-Digital Converters
ALU	Arrhythmic Logic Unit
AS	Active Serial
ASM	Algorithmic State Machine
BCLK	Bit Clock
BGA	Ball Grid Array
CAFIFO	Command Address FIFO
CAS	Column Address Strobe
CE	Chip Enable
CF	Compact Flash
CLU	Communications Logic Unit
CMOS	Complementary Metal Oxide Semiconductor
CPU	Central Processing Unit
DAC	Digital-to-Analog Converters
DC	Direct Current
DIFIFO	Data Input FIFO
DIMM	Dual Inline Memory Module
DMA	Direct Memory Access
DRAM	Dynamic Random Access Memory
DSP	Digital Signal Processor
EMIF	External Memory Interface
FIFO	First In First Out Buffer
FPGA	Field Programmable Gate Array
GB	Giga-Byte
GFLOPS	Giga-Floating Point Operations Per Second
GPO	General Purpose Output
I2C	Inter-Integrated Component Communication
IFIFO	Input First In First Out Buffer
IO	Input Output
IRQ	Interrupt Request
JTAG	Joint Test Access Group
KB	Kilobyte
LCD	Liquid Crystal Display
LDF	Linker Descriptive File
LED	Light Emitting Diode
LRCLK	Left-Right Channel Clock
LSB	Least Significant Bit
MB	Mega-Byte
MHz	Mega Hertz
MIPS	Million Instructions Per Second
MMU0	Memory Management Unit

MS0	Memory Space 0
MS1	Memory Space 1
MSB	Most Significant Bit
MSSD0	Memory Space SDRAM 0
MSSD1	Memory Space SDRAM 1
MSSD2	Memory Space SDRAM 2
MSSD3	Memory Space SDRAM 3
nF	Nano-Farad
ns	Nano-Seconds
OE	Output Enable
OFIFO	Output First In First Out Buffer
OVR	Over Voltage Register
PCB	Printed Circuit Board
PGA	Programmable Gain Amplifier
PLD	Programmable Logic Device
PLL	Phase Locked Loop
PRC	Pre-charge
PS	Passive Serial
PWM	Pulse Width Modulation
RAM	Ram Access Memory
RAS	Row Address Strobe
SCLK	System Clock
SDATA	Serial Audio Data
SDI	Serial Data In
SDO	Serial Data Out
SDRAM	Synchronous Dynamic Random Access Memory
SIMD	Single Instruction Multiple Data
SPDIF	Sony/ Phillips Digital Interface
SPI	Serial Peripheral Interface
SPORTS	Serial Ports
SRAM	Static Random Access Memory
SRC	Sample Rate Converter
UART	Universal Asynchronous Receiver and Transmitter
USB	Universal Serial Bus
V	Volts
VLIW	Very large instruction word
W	Watt
WE	Write Enable

1 Introduction

Modern speech recognition systems employ a number of fundamental techniques and mathematical models, such as diagonal covariance Gaussian mixture models and hidden Markov models. The use of these techniques is highly processor intensive and is most suited to systems that can perform floating point computations.

Implementations of speech recognition systems range from state-of-the-art research systems, to word processing applications that have embedded speech recognition (such as Microsoft Office), to applications that can recognize isolated words on a cellular phone. Research systems typically process the speech off-line, whilst word processing and cellular phones attempt to process the speech in real time. For real time systems, performance accuracy is dependent on the processing power of the system. Cellular phones typically employ processors that can only perform fixed-point calculations, requiring the floating point computations to be emulated or approximated. This leads to system performance and/or accuracy penalties. A further factor that degrades the performance of such systems is the relatively small amount of available RAM and secondary memory. This limits the complexity of the speech models which the system can store, as well as the sophistication of the recognition process it can perform.

Very little hardware that is specifically optimized for speech recognition applications, and that is accessible for research use, has been described in literature. The *In Silico Vox* project (1) is an attempt to implement speech recognition algorithms directly as hardware, by using a large FPGA. This system is limited in terms of its vocabulary size and the sophistication of its acoustic models, as it is currently only capable of recognizing up to 1000 words at a rate that is 2.2 times slower than real time. A speech recognition library has also been made available by Microchip specifically for the dsPIC30F 16-bit fixed-point family of microprocessors (2). This system is also very limited as the library only supports 100 specific words with American English pronunciation.

The aim of this project is to develop a portable high-performance microprocessor based hardware platform that can be used to completely abstract the speech recognition process from a personal computer. Hence, the system should be able to emulate the PC's peripherals, such as the mouse and keyboard, which control the input to the computer. The system, however, should not be limited to this application. We envision that a successful portable system could be employed as general voice-activated device, which can for example be used to activate household appliances for disabled persons, or in a car for hands free operation.

Finally, the system should have similar capabilities in terms of processing power and memory to a personal computer. This will allow speech recognition systems to be developed on a PC, and then easily transferred to the portable device. With these guidelines in mind, the system requirements are set out in the following section.

1.1 System Requirements

The aim of this project is to develop a stand-alone embedded system upon which state of the art processor intensive speech processing algorithms can be performed. The system would then be able to control peripheral devices through voice commands. The system requirements for the design are as follows:

1. The system must utilize a floating point central processing unit (CPU). The CPU must be able to perform 32-bit floating point calculations.
2. The system must support a minimum of 512 MB RAM.
3. The system must support a minimum of 512 MB non-volatile secondary storage memory.
4. The design must include a stereo audio analog to digital converter (ADC) and a stereo audio digital to analog converter (DAC).
5. The design must include microphone pre- amplification.
6. The design must include a universal serial bus (USB) peripheral interface with a personal computer. The design must be capable of implementing the HID keyboard and HID mouse subset of the USB2.0 HID specification.
7. The design must include a liquid crystal display (LCD).

It is clear from the above specification that the most important system component is the CPU itself. The choice of the CPU directly influences the processing power, amount of RAM, secondary storage and the number of peripheral devices that can be included in the design.

Therefore it is necessary to choose the CPU first and the following chapter discusses the process by which this was done.

2 Processor Selection

This chapter discusses the selection process of the processor that is utilized in our design. A number of currently available high-performance processors were considered and compared, before a final choice was made.

2.1 Comparison of processors

Several processors were considered for this design. An in depth comparison of these processors was performed, which is summarized in Table 2-1. The different categories used to evaluate the processors are listed in the first column and the capabilities of each processor are listed in the remaining column. An explanation for the notes and abbreviations appear at the end of the table, as do the descriptions of each of the processors themselves.

Processor	Analog Devices SHARC 21368	Analog Devices TigerSHARC TS201	Texas Instruments TMS320C6727	Microchip dsPIC30F6014	Freescale MPC7457	Renesas SH7751
Clock Speed (MHz)	400	600	300	40	1200	240
# Processing Elements	2	2	2	1	1	1
GFLOPS	2.4	3.6	1.8	No	unknown	1.7
MIPS	400-800*1	3600*2	2400	30	2900	430
Internal Register Size	32-bit	32-bit	32-bit	16-bit	32-bit	32-bit
Instruction Word Length	48-bit	128-bit	256 bit	24 bit	32-bit	16-bit
Address Bus Size	24 bit	32-bit	32-bit	No	32-bit	26 bit
Data bus Size	32-bit	64-bit	32-bit	No	64-bit	32-bit
32-bit Floating Point	Yes	Yes	Yes	No	Yes	Yes
40-bit Floating Point	Yes	Yes	Yes	No	No	No
64-bit Floating Point	No	No	Yes	No	Yes	Yes
16-bit Fixed Point	Yes	Yes	Yes	Yes	Yes	Yes
Cache	32x48bit instruct	6 x 128Kbit	32KB	No	32KB(L1)	24KB
EMIF Transfer Rate	532MB/s	1GB/s	400MB/s	No	640MB/s	640MB/s
External SDRAM Supported	384MB	1024MB	256MB	None	*3	128MB
SDRAM Speed	133MHz	125MHz	100MHz	None	Unknown	Unknown
Onboard SRAM	256KB	None	256kB	8KB	None	None
Onboard DRAM	None	3072KB	None	None	None	None
Onboard ROM	768KB	None	384KB	192KB	None	None
Flash supported	242MB	16MB	256MB*4	None	None	None
Slow Device Protocol	No	2*5	No	No	No	No
DMA channels	32	14	16	None	None	4
SPI Bus	2	None	2	2	None	None
SPORTS	8	None	16	None	None	None
SRC	Yes	No	No	No	No	No
SPDIF	Yes	No	No	No	No	No
JTAG	Yes	Yes	Yes	No	Yes	Yes
Link ports	None	4	None	None	None	None
I2C Bus	None	None	2	1	None	None
EEPROM	None	None	None	4KB	None	None
PCI	No	No	No	No	No	Yes
PCMCIA	No	No	No	No	No	Yes
Serial Communication Interface	No	No	No	No	No	Yes
Smart Card	No	No	No	No	No	Yes

Package	256 BGA	576 BGA	256 BGA	80-lead TQFP	483 CBGA	256 BGA
Price	R665	R1365	R146	R48	R194	Unknown

* see notes for explanation

1. The processor can perform 1 48-bit instruction every cycle; it can operate in SIMD mode and therefore perform the same instruction on 2 sets of different data. Therefore the maximum MIPS is 800.
2. The processor can perform 6 32-bit instructions per cycle in SIMD mode. Therefore the maximum MIPS are 3600.
3. The processor does not support a SDRAM controller directly. It can interface with a SDRAM controller via MPX or 60X bus protocols.
4. The TMS320C6727 only has enough address lines on its flash interface to support 16KB of flash memory. However, this can be expanded to support 256MB of flash through the use of general purpose input/output pins.
5. The TigerSHARC also supports a slow device protocol which can be used to interface with slower devices such as secondary memory. It supports a total of 512MB over 2 banks.

Abbreviations:

GFLOPS:	Giga-floating point operations per second.
MIPS:	Million instructions per second.
EMIF:	External memory interface.
SPI:	Serial peripheral interface. Used for interfacing to peripheral devices.
SPORTS:	Serial ports. Used for interfacing to ADCs and DACs.
SRC:	Sample rate converter.
SPDIF:	Sony/ Phillips digital interface. A digital audio interface.
JTAG:	Joint test access group IEEE 1149.1 boundary-scan standard.
I2C:	Inter-integrated component communication.

Table 2-1: Comparison of processors.

Table 2-2 compares the development boards that were available from the various processors. The categories used to evaluate the development boards are listed in the first column and the specifications of each board in the following columns. No evaluation board was available for the Microchip dsPIC30F6014. Freescale and Renesas do not manufacture their own development boards. Therefore development boards from Sandpoint X3 and Codescape were analyzed.

Evaluation kits	**Analog Devices SHARC**	**Analog Devices TigerSHARC**	**Texas Instruments**	**Sandpoint X3 (Freescale)**	**Codescape (Renesas)**
Processor	**21369**	**TS201**	**TMS320C6713**	**MPC7455**	**SH7751**
Clock speed	400MHz	600MHz	225MHz	800MHz	240MHz
SDRAM	4MB	32MB	8MB	128MB	Unknown
Flash memory	1MB	512KB	512KB	None	Unknown
Price	R3296	R6659	R2630	R25972	R18044

Table 2-2: Comparison of processor evaluation boards.

The following paragraphs describe each CPU listed in Table 2-1 in more detail.

2.1.1 Analog Devices TigerSHARC TS201

The TigerSHARC TS201 uses a static superscalar architecture. It has dual computational units, which each contain an arrhythmic logic unit (ALU), a multiplier, a register file and a communications logic unit (CLU). It also contains dual integer-ALUs which are separate to the computational units. The processor uses a 128-bit very large instruction word (VLIW) which enables it to execute between 1 and 4 32-bit instructions per cycle. The CPU can be configured to operate in single instruction multiple data mode (SIMD). In this mode the CPU can execute up to 6 32-bit instructions per cycle allowing a maximum of 6 32-bit floating point calculations per cycle or 24 16-bit fixed point calculations per cycle. It has 4 128-bit internal data busses that connect the processing elements to the 6 4-Mbit internal DRAM memory banks providing an internal bandwidth of 33GB/s.

The processor has a 64-bit external data and a 32-bit external address bus. The 32-bit address bus allows the CPU to address a 4G 32-bit word space. The onboard SDRAM controller can support a total of 1024MB of SDRAM over 4 SDRAM modules. The 64-bit external data bus can operate a maximum throughput of 1GB/s.

The external data bus also supports peripheral protocols such as the pipelined and slow device protocols which can address 512MB over two banks. This could be used to interface with secondary storage memory.

The CPU contains no support for serial peripheral devices such as analog-to-digital converters (ADC's). Therefore a field programmable gate array (FPGA) would be necessary to interface the CPU to an ADC and a digital-to-analog converter (DAC). The same FPGA could used to interface with a liquid crystal display (LCD) and to an USB controller.

The development board includes an ADC, a DAC, 32 MB SDRAM and 4-Mbit flash memory. It provides accessible connections to the external data bus, external address bus and control signals. A peripheral prototype board that connects through the external connections could therefore be designed, thereby simplifying the development of the final board that would also contain the CPU, and SDRAM modules.

2.1.2 Analog Devices SHARC 21368 Processor

This processor is based on a super Harvard architecture. It contains two computational processing elements that operate as a SIMD engine. The processor can either operate as a single processing unit or in SIMD mode where each processing element executes the same instruction but on different data. Each processing element contains a set of parallel computation units such as an ALU, multiplier, shifter and a register file.

The super Harvard architecture allows two operands to be fetched from the data memory, two operands from the program memory, and a 48-bit instruction to be fetched from cache in a single cycle.

Transfers between the internal memory and the core can be sustained at 6.4GB/s when operating at a clock frequency of 400MHz.

The processor contains 2-Mbits internal static RAM and 6-Mbits internal ROM. It supports 4 external memory devices which can be either SDRAM or asynchronous memory. A maximum of either 16 MB asynchronous memory or 128MB SDRAM can be connected on each of the 4 banks. If configured to use one bank in asynchronous mode, which would be necessary in this design, then the processor would be able to support a maximum of 384MB SDRAM.

The SHARC is designed specifically for audio applications. This is emphasized by the variety of IO interfaces it supports, such as serial ports, I2C, SPIs and UARTs.

These IO interfaces provide a simple means to interface with ADCs and DACs. However glue logic would still be required to interface with a LCD, secondary storage and USB.

2.1.3 Texas Instruments TMS320C6727

This CPU uses a 256-bit very large instruction word (VLIW). This allows it to execute between one and eight 32-bit instructions per cycle (six of which can be floating point instructions).The CPU supports double precision floating point, where as the TigerSHARC and SHARC do not, however a 64-bit floating point operation can take up to 4 cycles to complete.

The CPU has two data paths that connect to 2 processing units. Each unit has four functional units, namely a data-addressing unit, a multiplier and units responsible for arithmetic, logic and branch functions. Each processing unit contains 32 general purpose registers.

The CPU has 32 KB program cache, 256KB on-chip RAM and 384KB on-chip ROM. The memory controller supports single cycle data accesses between the CPU to ROM and RAM. It can perform up to three of the following parallel accesses to internal RAM and ROM:

- 2 64-bit accesses from the CPU.
- 1 256 bit program fetch from the core and program cache.
- 1 32-bit data access from the peripheral system.

The external memory interface has a 32-bit data bus and supports a maximum of 256MB of SDRAM. It also directly supports 16KB of flash memory, which can be extended to 256MB through the use of general purpose IO pins.

Like the SHARC, the TMS320C6727 is designed specifically for audio applications. This is emphasized by the variety of serial peripheral interfaces it contains. However, external interface circuitry would be necessary to interface with a USB controller and a LCD.

The development board supplied by Texas Instruments contains the TMS320C6713 CPU and not the higher performance TMS320C6727 CPU. This board has only 8MB onboard RAM and 18KB of L1 cache. It does however incorporate 64KB of L2 cache. These limitations may cause bottle necks if the size of program is too large to fit in the internal memory.

2.1.4 Microchip dsPIC30F6014

This CPU is a 16-bit processor and uses a 24-bit instruction word length. The core is capable of performing one data memory read, one working register read, one data memory write and one program memory read per instruction cycle. The digital signal processor (DSP) engine features a high speed fixed point 17-bit X 17-bit multiplier, a 40-bit ALU, two 40-bit saturating accumulators and a 40-bit bidirectional barrel shifter. The CPU has no support for hardware floating point operations. It also has no support for external memory such as SDRAM or Flash memory.

2.1.5 Freescale MPC7457:

The MPC7457 implements the 32-bit Power PC and uses a superscalar architecture. As many 4 instructions can be fetched from the cache simultaneously and as many as 12 instructions can be in the instruction queue. Most instructions have a single cycle execution time.

The CPU contains 11 independent execution units, consisting of 3 register files, a branch processing unit, 4 integer units, a floating point unit, 4 vector units and a load/store unit. It contains 32 KB of L1 cache, 512 KB of L2 cache and supports up to 4MB of external SRAM as L3 cache. It is capable of performing 32-bit and 64-bit floating point operations.

The processor also supports the MPX and 60X bus protocols, which allow the MPC7457 to interface with memory controllers and host bridges. Its memory management unit can address a 36bit physical address and a 52bit virtual address. The MPX and 60X bus protocol support a sustained throughput of 640MB/s via a 64-bit data bus.

The MPC7457 would therefore require a host controller that interfaces to SDRAM, it would also require external logic such as an FPGA or PLD to interface with ADC's, a USB, an LCD display and external flash memory.

2.1.6 Renesas SH7751:

Like the SHARC, the SH7751 uses a super Harvard architecture. This super scalar architecture allows the simultaneous execution of 2 instructions, reaching a maximum throughput of 2 instructions per cycle. It has a 32-bit internal data bus and it has a 16-bit instruction length.

It uses a memory management unit (MMU) to create a 32-bit virtual memory address space. Part of the address space corresponds to external memory, which is addressed by 29 address lines. This allows a total of 448MB of external memory such as SRAM, DRAM and ROM, PCMCIA and MPX to be directly attached in 7 separate areas, each with a maximum size of 64MB. An eighth area is used as reserved space (therefore the total memory that can be addressed including this reserved space is 512MB). The MMU uses address translation to convert from the virtual memory address to the physical memory address. The 7 areas can be configured to support different memory types in different ways. One is able to configure a maximum of 128MB SDRAM and a maximum of 448MB SRAM.

The CPU can execute the floating point multiply and accumulate, floating point add, floating point subtraction and floating point multiply instructions with 3 cycle latency for single precision and a 8 cycle latency for double precision. These instructions also have a pipeline delay of 1 cycle for single precision and a pipeline delay of 6 cycles for double precision.

The CPU supports a serial communication interface which would provide a simple means to interface with ADC's, USB and to LCD displays. It however would require external logic such as a PLD to interface with secondary storage memory.

2.2 Conclusion

For all of the CPU choices considered it would be necessary to use external communication logic to interface with most of the required peripheral components. The support for peripheral components is therefore not a major determining factor for the choice of an appropriate processor.

To simplify the design and prototype process, it is necessary to select a CPU for which a development board is available that includes reasonable amounts of RAM and also external connections that can be used to interface with a peripheral board. This would allow the debugging of the peripheral components to be performed, before a final prototype including the processor and system memory is developed.

The development boards available for the Renesas and Freescale CPU's are both very expensive when compared to those supplied by the other manufacturer and therefore they were not considered for this design.

No development board was available for the higher performance Texas Instruments CPU. This processor does not meet the system requirements with respect to the amount of RAM that can be supported and is therefore not considered for this design.

The Microchip dsPIC30F6014 does not meet any of the system requirements and is therefore not considered for this design.

The Analog Devices SHARC does not meet the system requirements with respect to the amount of RAM supported and is therefore not considered for the design.

The only processor that does meet the RAM system requirements is the Analog Devices TigerSHARC. The TigerSHARC development board contains 32MB of RAM, and the external interface of the development board can be used to interface with peripheral devices by means of suitable interconnection logic. The Analog Devices TigerSHARC TS201 600MHz DSP is therefore chosen for the design that is described in this thesis, and the following chapters describe the design and implementation of this system in detail.

3 System Design Overview

Our system utilizes the Analog Devices TigerSHARC 600MHz digital signal processor as the central processing unit. The TigerSHARC makes use of its external data bus, which operates at 100MHz, to interface to 512MB SDRAM, an FPGA and 512KB ROM.

The FPGA is in turn used to interface with all the peripheral components, such as an LCD display, a Compact Flash card, a multi-channel audio input and output system, a USB interface, a logic analyzer, a UART and a push-button and LED interface. A high level block diagram of our design is illustrated in Figure 3-1.

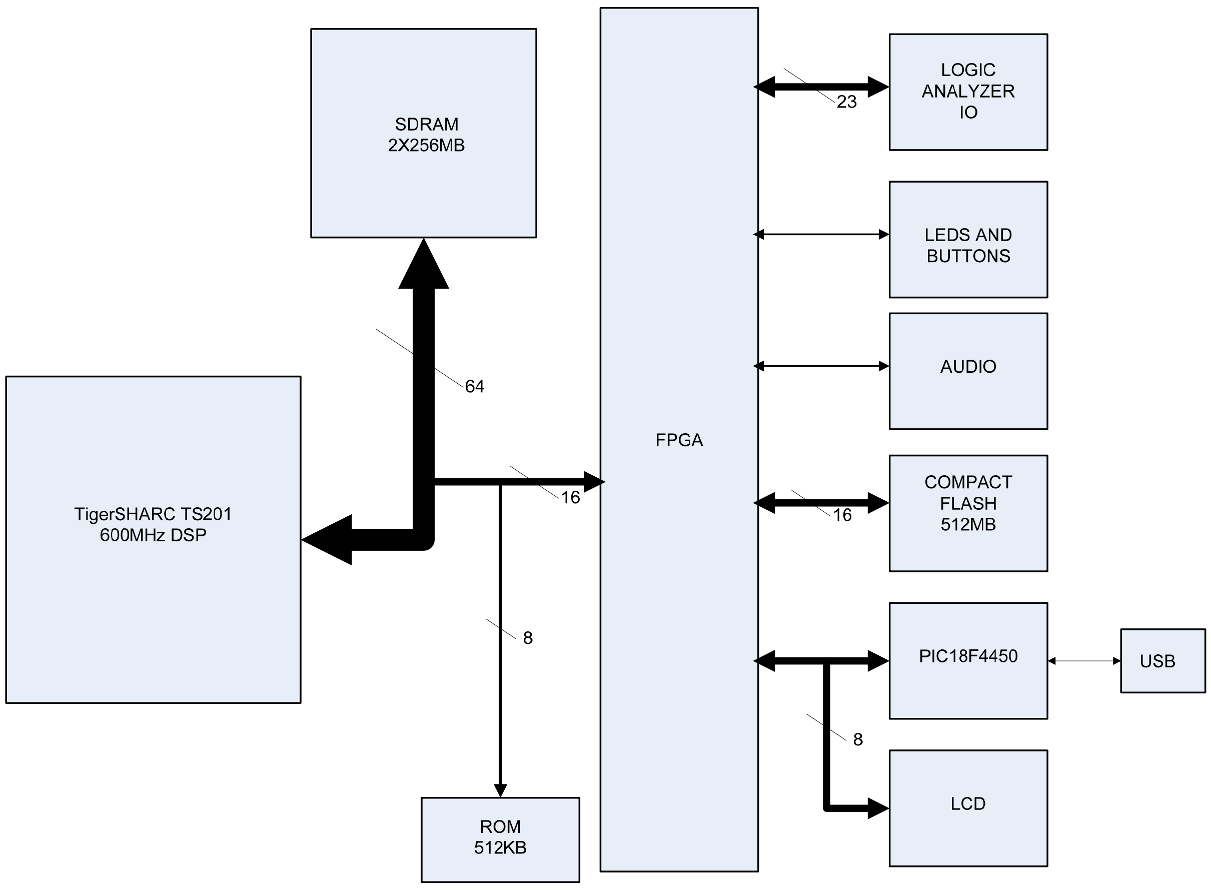


Figure 3-1: High level block diagram of the system.

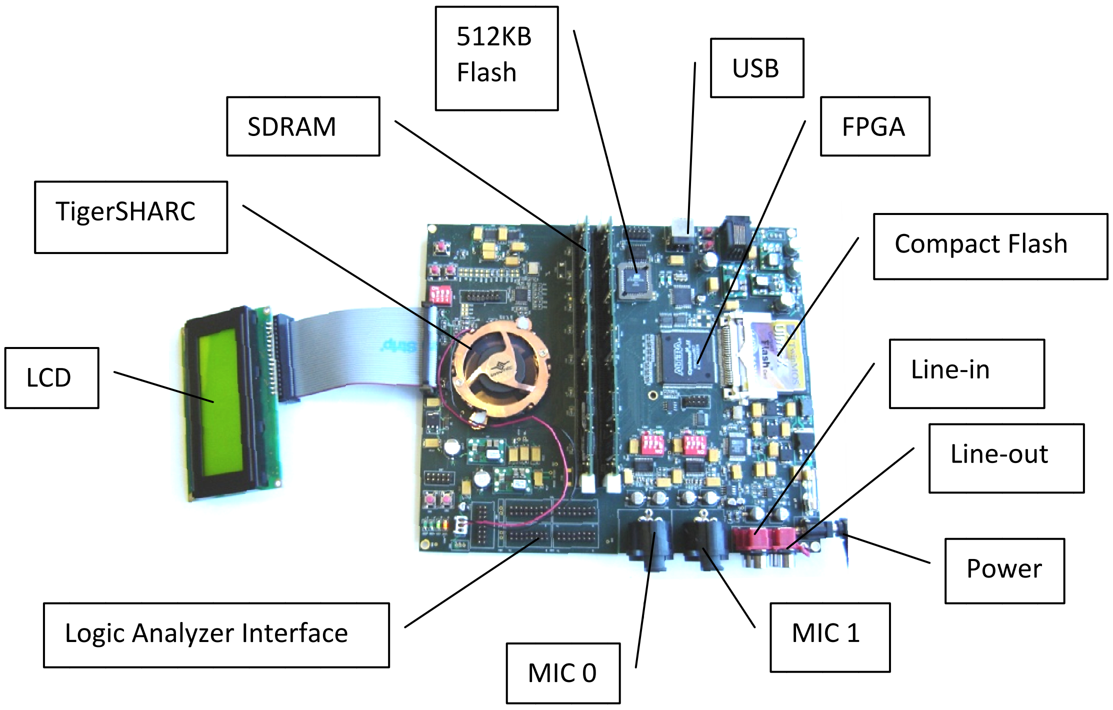


Figure 3-2: A photograph of the final prototype illustrating the different system components.

The system was designed in two phases. The initial phase involved interfacing the Analog Devices TigerSHARC TS201 development board to a peripheral board.

The design of this peripheral board involved implementing parts of the system specification, such as the secondary storage, audio, LCD, and USB. The final stage involved incorporating the peripheral board into a design which also included the TigerSHARC processor and RAM. A photograph of the final prototype is illustrated in Figure 3-2.

In designing this system, consideration needed to be taken for the clock distribution, signal integrity, system reset and the system power distribution, all of which are not illustrated in the block diagram.

This thesis documents the complete system design in detail. In order to improve the documents readability, the flow of the hardware design is broken down into the following chapters:

- Chapter 4 discusses the TigerSHARC processor.
- Chapter 5 discusses the FPGA system.
- Chapter 6 discusses the audio system.
- Chapter 7 discusses the storage system.

- Chapter 8 discusses the USB and LCD system.
- Chapter 9 discusses the UART system.
- Chapter 10 discusses the push buttons and LEDs system.
- Chapter 11 discusses the logic level translation system between various components
- Chapter 12 discusses the system reset.
- Chapter 13 discusses the SDRAM.
- Chapter 14 discusses the clock distribution and signal integrity.
- Chapter 15 discusses the power supply system
- Chapter 16 discusses the schematic capture and PCB layout.

Finally, the software that was written to demonstrate the correct working of this system is discussed in Chapter 17 and the performance of the processor is evaluated in Chapter 18.

4 The Central Processing Unit

This chapter discusses the implementation of the Analog Devices TigerSHARC TS201 600MHz digital signal processor (DSP) as the central processing unit in our design.

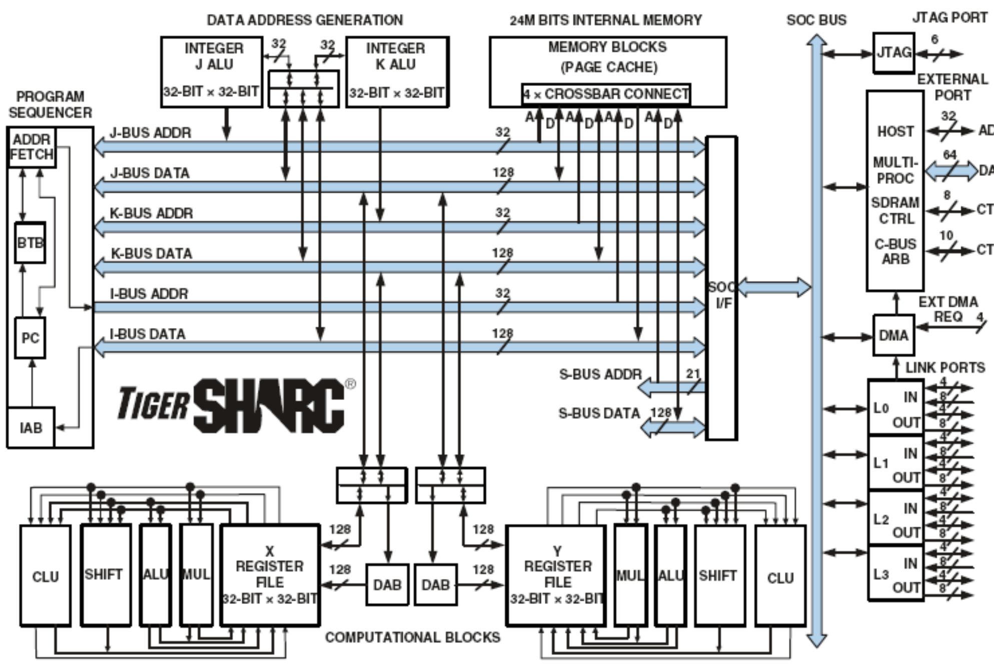


Figure 4-1: Functional Block Diagram of the TigerSHARC (reproduced from datasheet (3)).

Figure 4-1 illustrates a functional block diagram of the DSP. The following sections explain the parts of the DSP which are applicable to this design.

4.1 TigerSHARC Computational Blocks

The TigerSHARC is a static superscalar CPU. It has dual computational units, which each contain an ALU, a multiplier, a register file and a communications logic unit. It also contains dual integer-ALUs. It has a 128-bit very large instruction word (VLIW) which enables it to execute between 1 and 4 32-bit instructions per cycle. The CPU can be configured to operate in single instruction multiple data mode (SIMD) where both computational units perform the same instructions but on different data. In this mode the CPU can execute up to 6 32-bit instructions per clock cycle allowing a maximum of 6 32-bit floating point calculations per cycle or 24 16-bit fixed point calculations per cycle.

4.2 TigerSHARC internal memory

The ADSP-TS201S processor internal memory consists of 24M bits of on-chip DRAM, divided into six blocks of 4M bits that are addressable as 128K × 32-bit words. Each block—M0, M2, M4, M6, M8, and M10—can store program instructions, data, or both, so applications can configure memory to suit specific needs. Placing program instructions and data in different memory blocks, however, enables the DSP to access data while performing an instruction fetch. Each memory segment contains a 128K bit cache to enable single cycle access to internal DRAM. The internal memory space is illustrated in Figure 4-2.

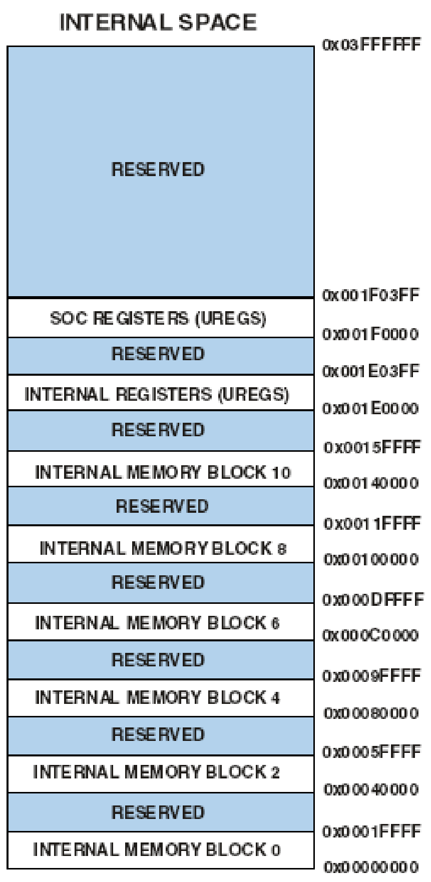


Figure 4-2: TigerSHARC internal memory space, showing blocks M0 to M10 (reproduced from datasheet (3)).

The linker descriptive file (LDF) is used to tell the compiler in which sections of memory the program code and data must be placed. Although, memory can be allocated dynamically at runtime through the use of multiple heaps, it is necessary to reserve memory for the heap in the LDF and to statically declare in which portions of memory the code and data must be placed. For example, it is possible to place multiple heaps in the internal memory or in the external memory. For more information on how to program the LDF and the compiler please refer to the Linker manual and C/C++ compiler manual (4).

4.3 External Port

The TigerSHARC TS201 processor’s external port provides an interface with off-chip memory and peripherals. The 4G 32-bit word space is included in the DSP’s unified memory map, which is illustrated in Figure 4-3.

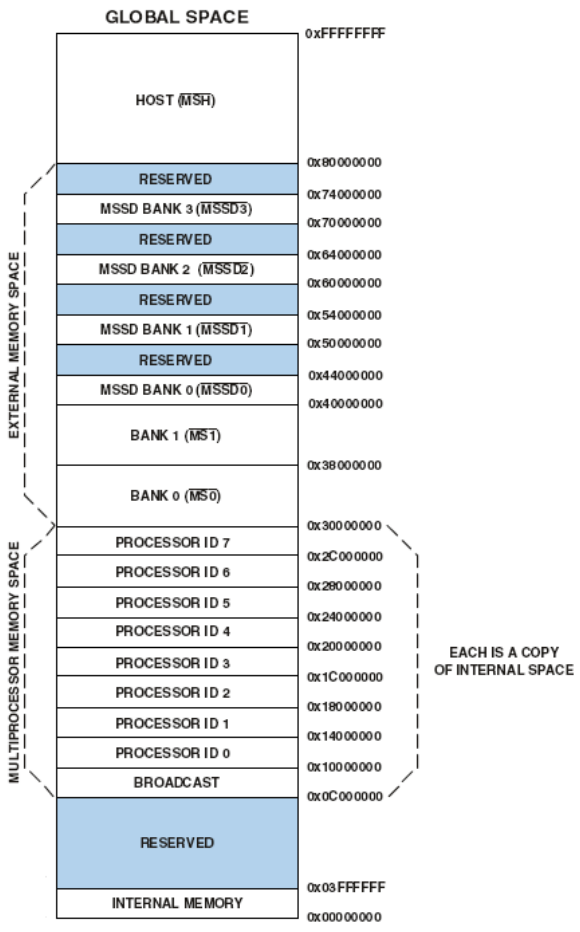


Figure 4-3: TigerSHARC unified memory map (reproduced from datasheet (3)).

The external system bus provides a single 64-bit data bus and a single 32-bit address bus. The external data bus can be configured for 32-bit and 64-bit, little endian operations. Our system is configured to operate with a 64-bit data bus. The lower 32-bits of the external data bus are connected to even memory addresses and the upper 32-bits are connected to the odd addresses. The addressing of the external memory devices and memory mapped peripherals is facilitated by the on-chip decoding of high order address lines to generate memory bank select signals.

The TigerSHARC can be configured to operate in a multiprocessor environment. Figure 4-3 illustrates how it is possible to address each of the processor's internal memory space within the unified memory map. Since our design implements 1 processor, however, the multiprocessor memory space is not used. Similarly the host memory space provides an interface with a host processor. However, since no host processor is implemented in this design, the host processor memory space is also not used.

The external port supports pipelined, slow and SDRAM protocols.

The TigerSHARC TS201 processor has two memory spaces, named MS0 and MS1, which can be used to interface with peripheral devices using the pipelined or slow device protocol. The processor also has four memory spaces which can be used to interface with SDRAM devices, named MSSD0 bank 0, MSSD1 bank 1, MSSD2 bank 2, MSSD3 bank 3.

Each memory space is 64M 32-bit words in size. Consequently the chip selects signals for these address spaces are $\overline{\mathrm{MS0}}$, $\overline{\mathrm{MS1}}$, $\overline{\mathrm{MSSD0}}$, $\overline{\mathrm{MSSD1}}$, $\overline{\mathrm{MSSD2}}$ and $\overline{\mathrm{MSSD3}}$ respectively, and Table 4-1 illustrates address decoding of these chip select signals. Address bus bits 26-0 are used to address each of the 64M x 32-bit words in each memory space.

Address Bus Bits 31-27	Chip Select Signals
00110	$\overline{\mathrm{MS0}}$
00111	$\overline{\mathrm{MS1}}$
01000	$\overline{\mathrm{MSSD0}}$
01010	$\overline{\mathrm{MSSD1}}$
01100	$\overline{\mathrm{MSSD2}}$
01110	$\overline{\mathrm{MSSD3}}$

Table 4-1: Chip select address decoding.

Depending on the configuration of the drive strength of the TigerSHARC (discussed in section 4.4.4), the control signals, address bus and data bus can take up to 4ns to become valid after the rising edge of the system clock cycle (SCLK) in which they were asserted. The signals are also held for a minimum of 1ns and disabled within 2ns after the following rising edge. Therefore all timing diagrams discussed in the following sections illustrate the address, data bus and control signals cycles with a slight delay after the rising edge of the clock.

The following section discusses the pipelined and slow devices protocols that can be used in memory space MS0 and MS1. The SDRAM controller and interface are discussed in chapter 13.

4.3.1 Pipelined protocol

The pipelined protocol is the faster of the two communication protocols that can be used in memory space MS0 and MS1. It allows for the pipelining of transactions with a throughput of one datum per cycle and a transaction latency that can be programmed to be between one and four cycles. The address and control signals of a transaction are issued in the address cycle and the data is transferred a few cycles later. The TigerSHARC processor can issue an address of a new transaction every cycle and does not need to wait for the completion of the data cycle of the first transaction before beginning the address cycle of the new transaction. Figure 4-4 illustrates the pipeline protocol read cycle with a read latency of 4, for four sequential read cycles. Figure 4-5 illustrates the pipeline protocol write cycle with a write latency of 1, for four sequential write cycles.

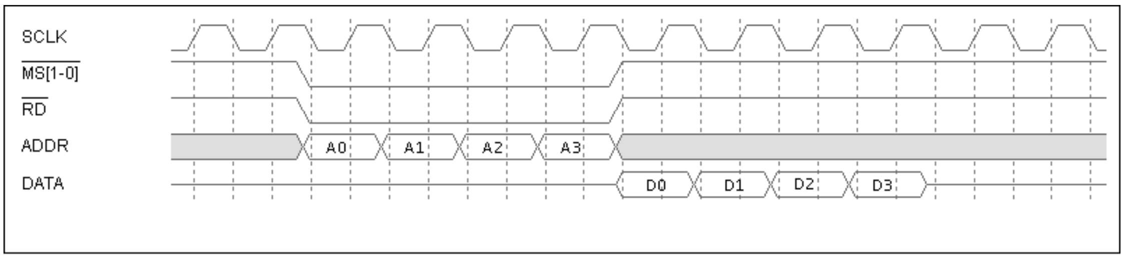


Figure 4-4: Pipelined protocol: read cycle, with a read latency of 4.

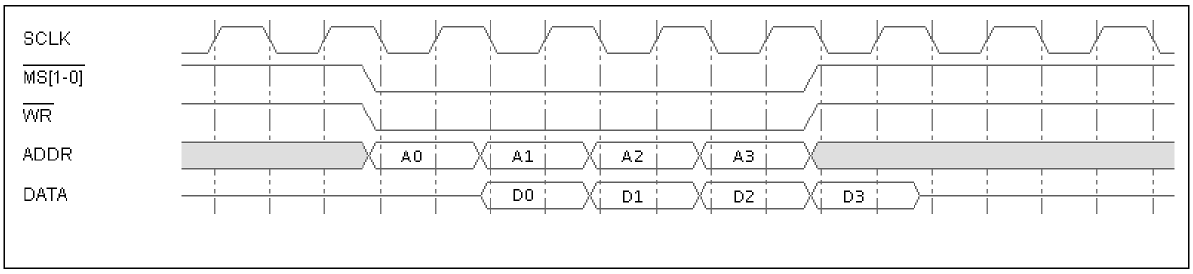


Figure 4-5: Pipelined protocol: write cycle, with a write latency of 1.

4.3.2 Slow device protocol

The TigerSHARC also supports a slow device protocol in memory spaces MS0 and MS1. The protocol allows the TigerSHARC to interface with non critical devices, and supports the insertion of wait cycles. A maximum of three internal wait cycles can be inserted, while external wait cycles can be inserted through the use of the acknowledge (ACK) signal. The slow device protocol read and write cycles are illustrated in Figure 4-6 and Figure 4-7 respectively. Figure 4-8 and Figure 4-9 illustrate the read and write cycles with external wait cycles.

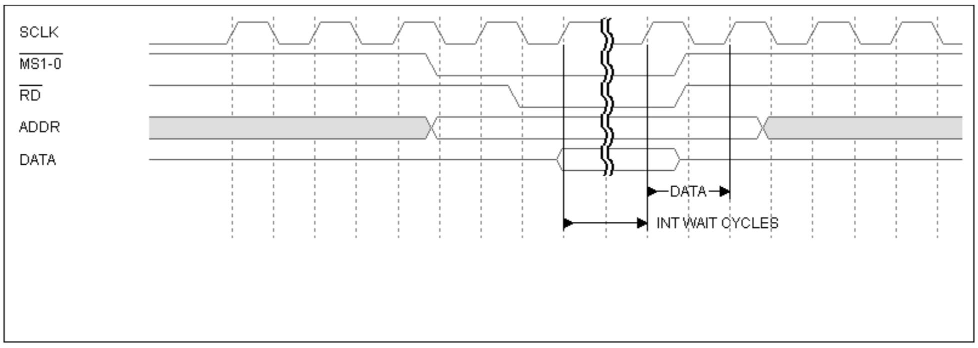


Figure 4-6: Slow device protocol: read cycle.

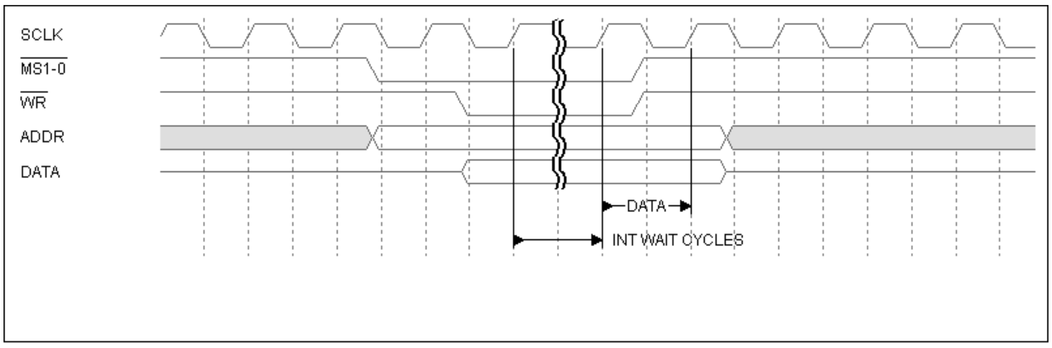


Figure 4-7: Slow device protocol: write cycle.

Figure 4-6 and Figure 4-7 illustrate that the chip selects signals $\overline{\mathrm{MS0}}$ and $\overline{\mathrm{MS1}}$ are asserted in the first clock cycle. In the next cycle the read or write control signal is asserted, after which the programmed internal wait cycles are inserted. After the completion of the wait cycles, the host or the device will sample the data bus in the data cycle.

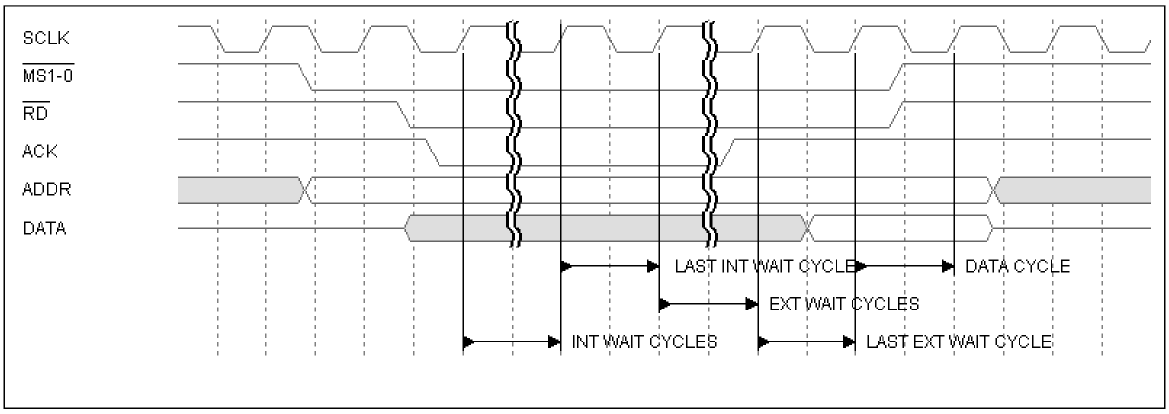


Figure 4-8: Slow device protocol: read cycle with external wait cycles.

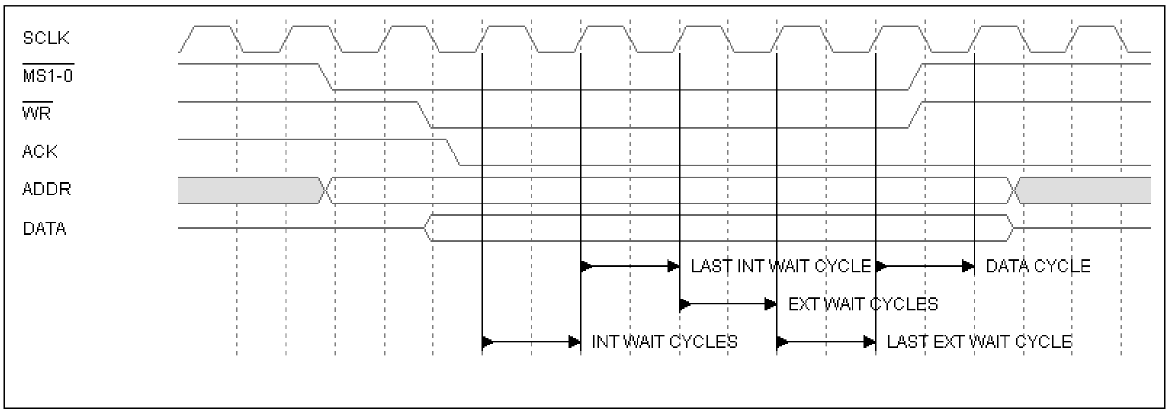


Figure 4-9: Slow device protocol: write cycle with external wait cycles.

Figure 4-8 and Figure 4-9 illustrate that the chip selects signals $\overline{\mathrm{MS0}}$ and $\overline{\mathrm{MS1}}$ are asserted in the first clock cycle. In the next cycle, the read or write control signal is asserted, after which the programmed internal wait cycles are inserted. The slave device must assert the ACK signal before the completion of the internal wait cycles. If the device asserts the ACK signal before the start of the last internal wait cycle, then the TigerSHARC will insert extra wait cycles until the slave device de-asserts the ACK signal. The TigerSHARC will then insert one more internal wait cycle. After which the host or the device will sample the data bus in the data cycle.

4.3.3 DMA Controller

The TigerSHARC has an on-board DMA controller that provides 14 zero overhead DMA channels that can function without processor intervention. The controller operates independently of the core, enabling DMA transfers to execute whilst the DSP’s core continues to execute program instructions. The DMA controller is capable of performing both one dimensional and two dimensional memory transfers. It can perform internal to internal, internal to external and external to internal memory transfers as well as DMA chaining, whereby the DMA controller can be configured to initialize another DMA transfer.

The DMA channels 0 to 3 perform transfers between external memory and internal memory. This design employs DMA channel 0 to perform memory transfers for the audio driver. DMA channel 3 is used to perform block transfers between the FPGA and internal memory and vice versa. The other DMA channels are not currently used and remain for future expansion.

4.3.4 Interrupt controller

The DSP supports nested and non nested interrupts. Each interrupt is associated with a register in the interrupt vector table. Each interrupt has an interrupt latch register and an interrupt mask register. The interrupts can be programmed to be either level or edge sensitive. The DSP has four external interrupt signals, namely IRQ3-0. This design uses IRQ0 and IRQ1. IRQ0 is employed by the audio module in the FPGA to initiate the software audio driver to perform DMA block transfers of the audio between the FPGA and the TigerSHARC and vice versa. Interrupt IRQ1 is triggered by the FPGA when one of the push buttons is depressed.

4.3.5 EPROM/FLASH Interface

The TigerSHARC can be configured to boot from up to 16MB external 8-bit EPROM or flash memory. The EPROM or flash memory interface is not mapped in the processors unified memory space, Instead the TigerSHARC uses DMA channel 0 to access the 8-bit interface using 16 wait cycles. During this operation the DMA controller packs the bytes into 32-bit instructions. Applications can also access the EPROM (to program the memory) during normal operation via the DMA.

Our system uses the ATMEL AT49BV040B (U45) 512KB flash memory to boot from (illustrated in schematic 21 in Appendix A). The AT49BV040B is housed within a PLCC32 socket and can therefore be removed from the system. Since the TigerSHARC can support a maximum of 16MB of flash memory a footprint for AMD SL29GL128P 16MB flash memory chip was included on the PCB. The SL29GL128P has the same internal configuration as the ATMEL flash, and differs only in its larger size. If this larger capacity is required in future, the ATMEL flash can be removed from its socket, the SL29GL128P mounted on the provided footprint and resistor R14 to be inserted so that the $\overline{\mathrm{BMS}}$ chip select signal will address the 16MB flash device.

4.4 TigerSHARC Configuration

The following sections discuss the hardware configuration of the TigerSHARC DSP including the clock configuration, reference voltages, identification pins, IO impedance settings, configuration pins and JTAG interface.

4.4.1 TigerSHARC Clock Domains

The TigerSHARC system clock (SCLK) can operate at frequencies of up to 125MHz. However, since the Analog Devices TigerSHARC TS201 development board operates with a system clock of 100MHz, the peripheral board was designed to operate synchronously from the same 100MHz system clock.

To ensure portability from the peripheral board design to the final prototype, the 100MHz system clock was maintained and the method used to generate the system clock source is described in section 14.1 on page 103.

The TigerSHARC external system clock input (SCLK) is used by an internal phase locked loop (PLL) to generate the core clock (CCLK). Our system uses a 600MHz CCLK signal, which is the highest core clock frequency supported by the TigerSHARC. To achieve this, a PLL multiplication ratio of 6 is programmed by connecting resistors R128, R129 and R133 to SCKLRAT 2-0 pins (illustrated in schematic 20 in Appendix A).

4.4.2 Reference Voltage Filtering

It is necessary to provide a filtered voltage supply to the analog supply pin (VDD_A) of the internal PLL circuit as well as for the clock reference (SCLK_VREF) and the IO reference (DSP_VREF). The configurations recommended by the data sheet are used and are illustrated in schematic 19 in Appendix A.

4.4.3 Processor Identification

The TigerSHARC processor can operate as part of a multiprocessing system; in which it is necessary to assign an ID to each processor. Since this system implements only one processor, the processor must be assigned an ID of 0. This is done using resistors R115, R117 and R120 which configure pins ID[2-0] to the correct ID, as is illustrated in schematic 20 in Appendix A. Since the desired processor ID is 0, it is not necessary to place the resistors because the internal pull down resistors will default the configuration to an ID of 0.

4.4.4 IO Driver Impedance Control

The TigerSHARC processor's IO drivers can operate in A/D driver or normal mode. In A/D mode, the driver impedance is continuously matched to the line impedance, and in normal mode the drive strength can be programmed to between 10% and 100% of the maximum, using pins DS[2-0]. Normal mode is the recommended mode, and is the mode in which our system is configured using resistors R131 and R143 as illustrated in schematic 20 in Appendix A.

The drive strength itself, is set using resistors R132, R135 and R136 (also illustrated in schematic 20 in Appendix A) to configure pins DS[2-0]. Our system is designed to operate with a drive strength of 70% (this will be discussed in section 14.2). However, since the internal pull up and pull down resistors of pins DS[2-0] default to 70%, it is not necessary to place these resistors.

4.4.5 Link Port Termination

The TigerSHARC has four Link ports which can be used for on-board or off-board inter-processor communication within a multi-processing environment. Since this implementation uses only a single processor, these unused input pins must be attached to IO 2.5V as specified in the datasheet. The output pins must be left unconnected. The configuration is illustrated in schematic 18 in Appendix A.

4.4.6 Configurable pins

The $\overline{\mathrm{BMS}}$, $\overline{\mathrm{BUSLOCK}}$, $\overline{\mathrm{BM}}$ and $\overline{\mathrm{TMR0E}}$ pins are connected through a dip switch (U37) via 500Ω pull-up resistors to the IO 2.5V supply. This is illustrated in schematic 20 in Appendix A. Table 4-2 illustrates the function of the switch as well as the default configuration of each pin.

Switch pin	Signal	Off (pulled-low)	On (pull-up)
1	$\overline{\mathrm{BMS}}$	Boot from EPROM (default)	Boot from host
2	$\overline{\mathrm{BM}}$	Disable interrupts at boot	Enable interrupts at boot (default)
3	$\overline{\mathrm{TMR0E}}$	1-Bit link port	4-bit link port (default)
4	$\overline{\mathrm{BUSLOCK}}$	SYSCON/ SDRCON one time programmable	Always programmable (default)

Table 4-2: Configurable pins connected to switch U37.

4.4.7 Miscellaneous pull-up resistors

The datasheet specifies that it is necessary to pull the $\overline{\mathrm{HBR}}$, $\overline{\mathrm{BOFF}}$, $\overline{\mathrm{DMAR0}}$, $\overline{\mathrm{DMAR1}}$, $\overline{\mathrm{DMAR2}}$, and $\overline{\mathrm{DMAR3}}$, up to the IO 2.5V supply using 4.7kΩ resistors. This is illustrated in schematic 17 in Appendix A.

4.4.8 Power Supply Pins and Decoupling.

The TigerSHARC requires a 1.2V supply for the core, 1.6V for the DRAM and 2.5V to for IO. Schematic 19 in Appendix A illustrates which pins are connected to these supplies. The requirements and power calculations for these supplies are discussed in section 15.1 on page 113.

The 1.2V supply requires at least 8 1nF, 4 10nF and 5 100nF decoupling capacitors, the 1.6V supply requires at least 6 1nF, 2 10nF and 4 100nF decoupling capacitors and the 2.5V supply requires at least 8 1nF, 2 10nF and 4 100nF decoupling capacitors.

All these decoupling capacitors are illustrated in schematic 27 in Appendix A.

The capacitors are placed as close as possible to the TigerSHARC supply pins, with the following priority: 1.2V, 1.6V, 2.5V and 1nF, 10nf, 100nF.

4.4.9 TigerSHARC JTAG Interface

The TigerSHARC is programmed via a Joint Test Action Group (JTAG) interface. Schematic 26 in Appendix A illustrates the JTAG header and resistors which are connected to the processor. The JTAG header must be placed within 6cm of the processor as specified by the Analog Devices JTAG Emulation Technical Reference (5). In the event that signal conditioning needs to be performed for the JTAG interface, then the series zero Ohm resistors that have been include in our design can be altered to dampen the rising edge of the signals. However, for the developed prototype, this was not necessary.

4.5 Thermal Relief

The maximum total power required by the TigerSHARC processor is given by the following equation:

$$P_{TOTAL} = P_{CORE}(\mathrm{MAX}) + P_{DRAM}(\mathrm{MAX}) + P_{IO}(\mathrm{MAX})$$

$$= I_{CORE(MAX)} \times V_{CORE} + I_{DRAM(MAX)} \times V_{DRAM(MAX)} + I_{IO(MAX)} \times V_{IO}$$

Using the maximum currents from the TigerSHARC specification (3) and the known voltages, we find

$$P_{TOTAL} = 2.95 \times 1.2 + 0.43 \times 1.6 + 0.15 \times 2.5W$$

$$= 4.6W$$

The datasheet states that the TigerSHARC has the following thermal characteristics

$T_{CASE(MAX)} = 85°\mathrm{C}$

$\theta_{JC} = 0.7°C/W$, the junction to case thermal coefficient.

$\theta_{CA} = 17°C/W$, the case to ambient thermal coefficient, with an airflow of 0m/s.

$\theta_{CA} = 10.2°C/W$ with an airflow of 1m/s.

$\theta_{CA} = 9°C/W$ with an airflow of 2m/s.

The temperature of the case is given by the following equation (6):

$$T_{CASE} = P_{TOTAL} \times (\theta_{CA} - \theta_{JC}) + T_{AMBIENT}$$

Assuming, $T_{AMBIENT} = 25°\mathrm{C}$ we find that:

$$T_{CASE} = 4.6 \times (17 - 0.7) + 25°\mathrm{C}$$

$$= 99.98°\mathrm{C}$$

Because the case temperature of 99.98°C exceeds the maximum case temperature of 85°C, a heat-sink is required. The engineer to engineer note EE-182 (6) from Analog Devices recommends heat-sink solutions from Cool Innovations and from AAVID Thermalloy. However, the Cool

Innovations heat-sinks do not specify the thermal coefficients with zero air flow. Therefore only the solutions from AAVID were analysed. The specifications for the most efficient heat sink from AAVID with part number 374524B60023G (7) is given below:

$\theta_{SA} = 16.5°C/W$, the sink to ambient thermal coefficient, with an airflow of 0m/s.

$\theta_{SA} = 5.47°C/W$ with an airflow of 1m/s

$\theta_{SA} = 4.5°C/W$ with an airflow of 2m/s

The temperature of the case using the heat sink is calculated using the following equation:

$$T_{CASE} = P_{TOTAL} \times \left(\theta_{SA} + \theta_{CS} - \theta_{JC}\right) + T_{AMBIENT} \tag{4-1}$$

Where θ_{CS} is the case to sink thermal coefficient which is equal to the thermal coefficient of the thermal grease, and is typically 0.7 to 0.9 °C/W. The calculations in following paragraphs use 0.9°C/W as the worst-case case to sink temperature coefficient.

The case temperature is calculated using equation (4-1) to be 101.8°C for no air flow and 51.08°C for an air flow of 1m/s and 46.62°C/W for an air flow of 2m/s. It is therefore clear that if a passive heat sink is used, then a system fan would also be required to introduce a constant airflow into the enclosure. As an alternative, the ball grid array (BGA) heat sink with an integrated fan solution from AAVID THERMALLOY was also evaluated. This heat sink and fan combination with part number 11-5602-51 has a case to ambient temperature coefficient of 3.7°C/W and can dissipate 15W.

Using equation (4-1) the maximum case temperature can be calculated to be

$$T_{CASE} = 4.6 \times (3.7 + 0.9 - 0.7) + 25$$

$$= 42.94°\text{C}$$

Therefore the heat sink with an integrated fan would easily meet the requirements for this design. Since this particular device could not be sourced from local suppliers the equivalent heat sink from Vantek (ICEBERQ CCB-A1C) was used instead.

The fan operates from 12V and draws 50mA. The power is supplied via header (U57) as is illustrated in schematic 17 in Appendix A. Extra power headers (U56 and U58) are included in the design, to make provision for a panel mounted fan that could be used to keep the ambient temperature low inside an enclosure.

4.6 Summary and Conclusion

This chapter described the design aspects relating directly to the operation of the Analog Devices TigerSHARC TS201. The device was configured to interface to the FPGA using the pipeline protocol for efficient data transfer, the core clock is programmed to operate at its maximum frequency of 600MHz, and the DMA controller is setup to use channels 0 and 3 for audio transfers and compact flash communication respectively. A suitable fan-assisted heat sink was determined by considering the expected operating conditions of the CPU.

The following chapters describe peripherals and system management.

5 FPGA System

This chapter discusses the implementation of the FPGA which interfaces with the TigerSHARC and the peripheral components. A general description of the design of the top level VHDL entities is given in the sections below. However, each individual peripheral module of the FPGA is discussed separately in the following chapters.

5.1 Altera Cyclone II FPGA

The Altera Cyclone II EP2C5Q208C7 FPGA (8) is used in the system as an interface between the TigerSHARC DSP and the peripheral devices. Factors that determined this choice of FPGA were the component size, speed grade, pin count, cost and the availability of the component.

Initially, VHDL code was written to determine the required FPGA size. It was found that the Altera Cyclone II EP2C5 and the Xilinx Spartan 3E XC3S250E FPGA(9) families met the size and speed grade required. A minimum of 115 IO pins (excluding those to be used as the logic analyzer interface) are required to interface with all the peripheral devices. In order to keep the complexity and cost of the PCB mounting process low, it was desirable to use the largest non ball grid array (BGA) package available. The plastic quad flat pack 208 pin (PQFP208) package of the Xilinx FPGA and the Altera FPGA provided 135 and 138 user IO pins respectively. The Altera Cyclone II EP2C5Q208C7 is priced at R117 while the Xilinx Spartan 3E XC3S250E4PQ208C is priced at R134. Finally, the Xilinx FPGA was not available in a prototyping quantity at the time. Consequently, the Altera Cyclone II EP2C5Q208C7 FPGA was chosen for use in our design.

5.2 TigerSHARC to FPGA interface

The TigerSHARC development board uses memory space MS0 to interface with the boards peripherals in the slow device protocol mode. As a precaution, it was decided to design the interface between the TigerSHARC and FPGA in such a way to allow either the pipelined protocol or the slow device protocol to be used. Then if problems were later experienced in using the pipelined protocol, the slow device protocol could be used instead. Since memory space MS0 and MS1 can each only operate either in slow protocol or in pipelined protocol mode, memory space MS1 was used to interface with the FPGA.

Figure 5-1 illustrates a high level block diagram of the communication interface between the TigerSHARC, FPGA and the internal FPGA modules. The following section describes the design of the internal modules of the FPGA.

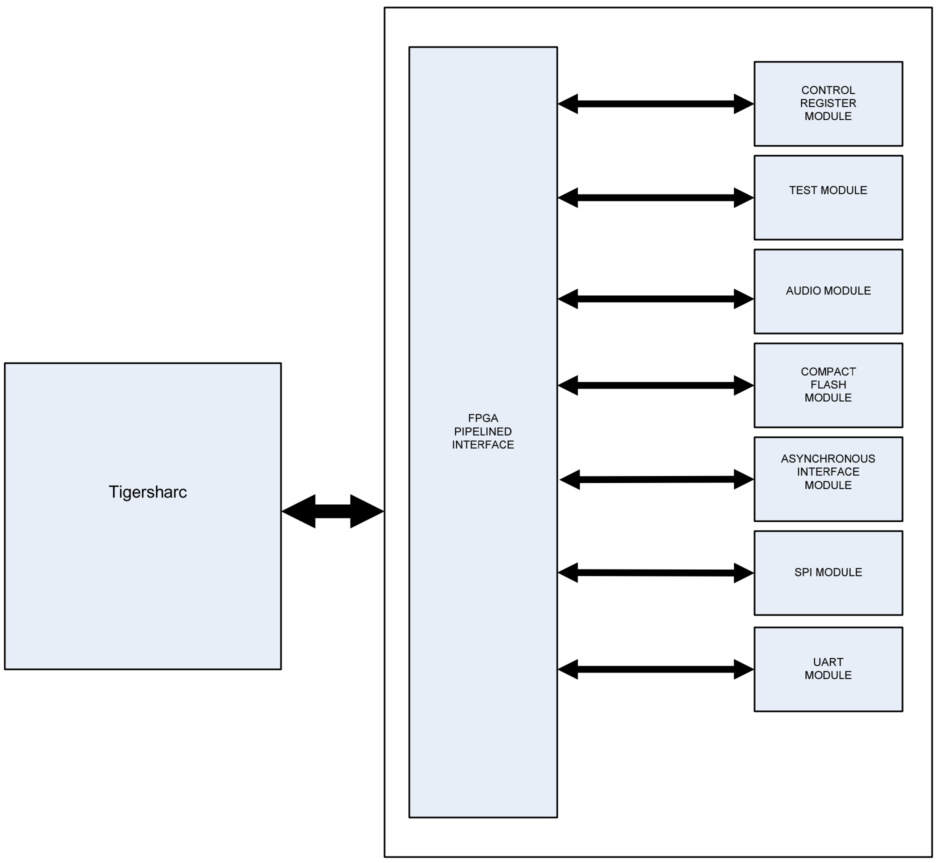


Figure 5-1: A Bock diagram of the TigerSHARC to FPGA interface.

The following signals are used to communicate to the FPGA.

Input Signals

- SYSCLK: The 100MHz system clock.
- $\overline{\mathrm{RESET}}$ The reset signal for the FPGA (active low).
- $\overline{\mathrm{MS1}}$: Memory space 1 chip select (active low).
- $\overline{\mathrm{RD}}$: Read control signal (active low).

- $\overline{\mathrm{WRL}}$: Write low control signal (active low).
- ADDRESS [7:0]: An 8-bit address bus for the FPGA internal registers.

Bidirectional Signals

- DATA_BUS[15 :0]: A 16-bit bidirectional data bus.

Output Signals

- $\overline{\mathrm{IRQ}}$[1:0]: Interrupt request to the processor (active low).
- $\overline{\mathrm{DMAR}}$[1:0]: DMA request to the processor (active low).
- ACK: Acknowledge signal.

The interface between the TigerSHARC, the FPGA pipelined module and FPGA internal modules is synchronous with the system clock. All the internal registers of the FPGA appear as memory mapped variables to the TigerSHARC. The following paragraphs describe the interface used to access the FPGA's internal registers via the pipelined protocol.

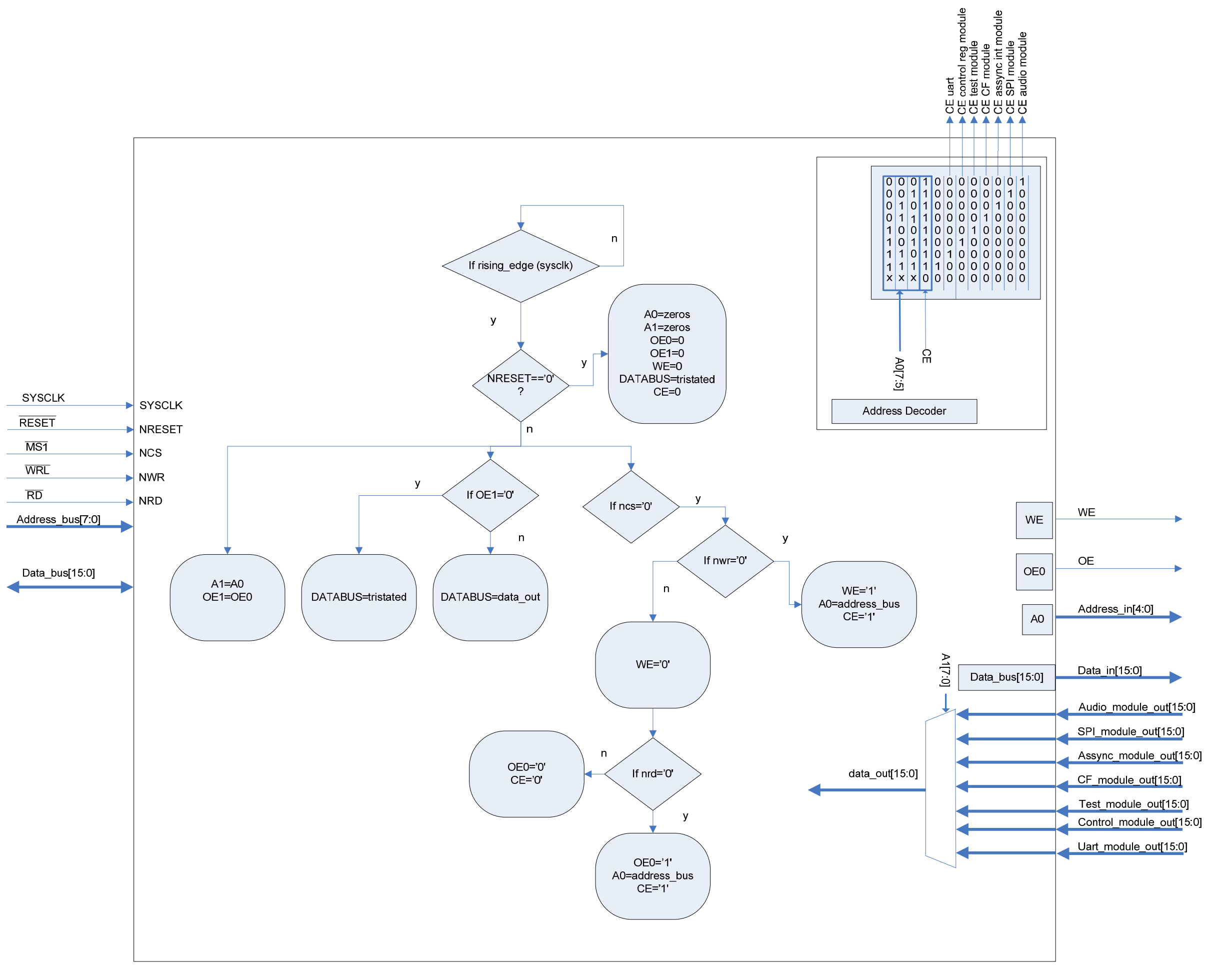


Figure 5-2: The TigerSHARC to FPGA pipelined protocol interface module.

The FPGA pipelined protocol interface module synchronizes communication between the TigerSHARC and the FPGA's internal modules and is illustrated in Figure 5-2. The FPGA pipelined protocol interface module is also responsible for generation of the chip enable (CE) of all the internal FPGA modules. As is illustrated in the top right of Figure 5-2, the bits 7:5 of the address bus are used to generate the CE signals, and the address bus bits 4:0 are passed to the FPGA's internal modules to address the internal registers.

The synchronization between the FPGA's internal modules and the TigerSHARC is illustrated by the signal flow diagram in Figure 5-2 which is evaluated at every rising edge of the system clock. The single-bit registers OE1 and OE0 are used to synchronize the output of the internal modules with the correct time slot require by the pipeline protocol. The operation of the interface is described below for read and writes cycles.

Write Cycle

As is illustrated in Figure 4-5 on page 21, the address, chip select and write control is set up in the first clock cycle (address cycle) of the pipelined protocol and the data bus is driven in the next clock cycle (data cycle). The flow diagram illustrates that, once the address cycle is detected, then the CE is decoded, WE is asserted and the address is loaded into register A0. These signals are all valid for the next clock cycle, and upon the following rising edge the FPGA's internal module, the data bus value will be stored into the correct register, as is illustrated in the timing diagram in Figure 5-4 on page 37.

Read Cycle

As is illustrated in Figure 4-4 on page 20, the address, chip select and read control signal is set up in the first clock cycle (address cycle) and the data bus is sampled by the TigerSHARC 4 cycles later (data cycle). The reason for a clock latency of 4 is described below, with respect to the flow diagram in Figure 5-2 on page 34 and the read cycle timing diagram of the internal FPGA modules illustrated in Figure 5-4 on page 37.

The flow diagram illustrates that the address bus is sampled at the rising edge of the clock following the assertion of the chip select, read control signal and address. The address is loaded into register A0, the CE is decoded, and OE is asserted. These signals are all valid for the next clock cycle, and upon the following rising edge of the clock the FPGA's internal module will sample the OE and Address_in signals. Since each of the internal module's Data_out signal is synchronized with the system clock, the output will only be valid in the following clock cycle, as is illustrated in Figure 5-4. The Data_out signal is then sampled at the next rising edge of the clock by the pipelined interface module and output on the data bus for the following clock cycle.

5.3 FPGA Internal module design

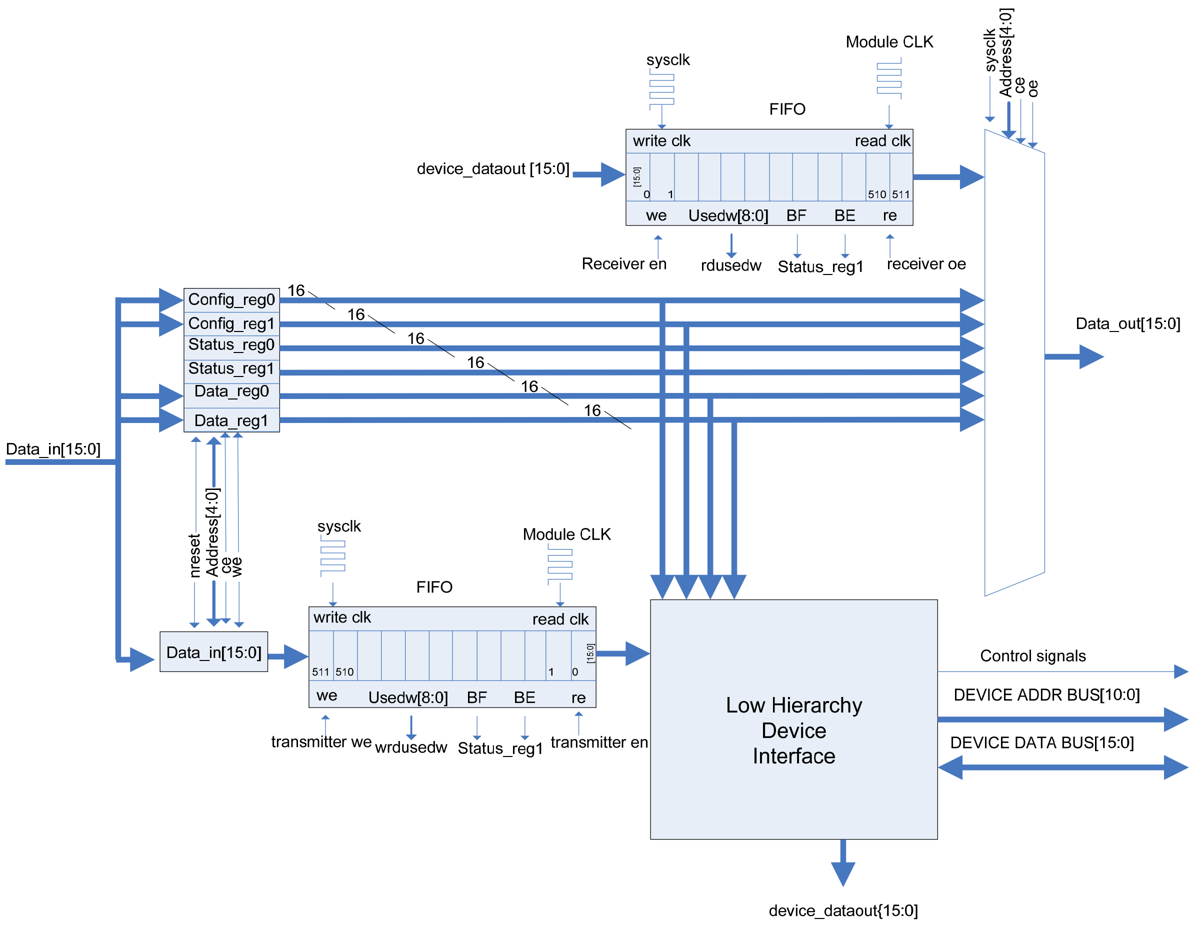


Figure 5-3: The general design of the top level entities of the FPGA modules.

Figure 5-3 illustrates the general design used for each top level entity of the FPGA modules. Each module in our design contains memory mapped configuration registers, status registers, data registers and first-in first-out buffers (FIFO). A description of the registers and the FIFOs are given in the paragraphs below. Where applicable, the modules also contain parallel processes such as state machines, shift registers, counters and lower hierarchy entities. The processes control internal and external signals which are used to interface the module to an external device.

Each module is accessed by the CPU as 16-bit memory mapped registers. Each module has the following signals:

Input signals:

- SYSCLK: The 100MHz system clock.
- MODULE_CLK: The 100MHz, 20MHz or 200MHz module clock.

- CE: The chip enable (active high).
- OE: The output enable (active high).
- WE: The write enable (active high).
- ADDRESS[4:0]: An address bus to address the modules' internal registers
- DATA_IN[15 :0]: 16-bit input data bus.
- External device input signals.

Output signals:

- DATA_OUT[15 :0]: 16-bit output data bus.
- Interrupt request and DMA controls signals to the processor.
- External device output signals.

The following paragraphs give a description of the basic building blocks of the module illustrated in Figure 5-3.

Registers

Each module design contains configuration, status and data registers. These registers are designed as 16-bit memory addressable registers. The configuration and data registers are read and write enabled whilst the status registers are read-only registers.

Figure 5-4 illustrates the read and write access to the internal FPGA registers. During a write cycle the address, CE and WE are asserted and at the next rising edge of the clock the data is written into the register DATA_REG. During a read cycle the address, CE and OE signals are asserted. At the next rising edge the registers DATA_REG's stored data is output on the DATA_OUT bus.

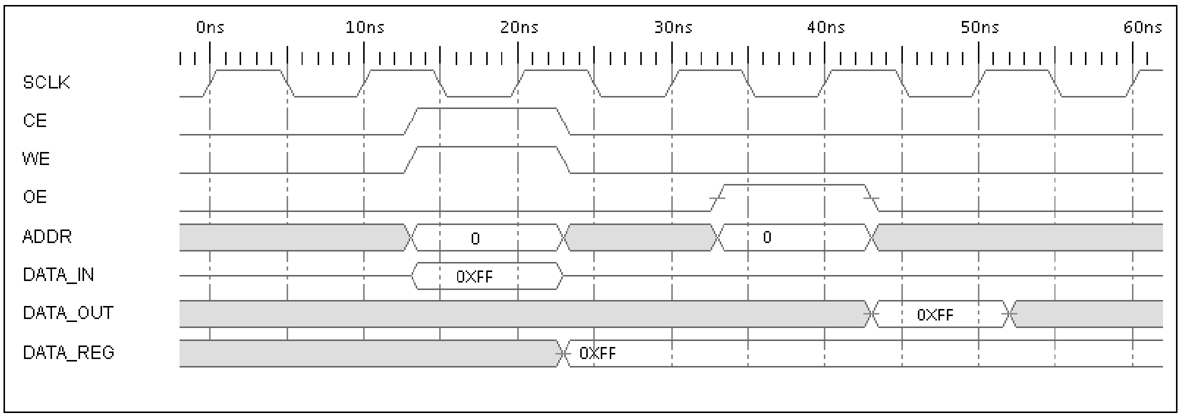


Figure 5-4: Read and write access to internal FPGA registers.

First-In First-Out Buffers

First-in first-out buffers (FIFO) are used in this design to synchronize the system-side and the device-side of the design. The FIFO supplies certain status and flag variables, namely the number of words of data stored in the FIFO and buffer full and buffer empty flags. A threshold flag is also generated by comparing the number of words in the FIFO with a threshold that can be stored into a configuration register. The flags are stored in status registers. The number of words used and the flags are available to the user as read-only memory mapped status registers.

Clocks

The system-side of each module is synchronous to the system clock (SYSCLK), which operates at 100MHz. The device-side of each module operates in synchrony to the module clock (MODULE_CLK). Depending on the module's timing requirements, a module clock of 100MHz, 200MHz or 20MHz is used. These clocks are generated by the internal PLL of the FPGA.

5.4 FPGA Control Register

The FPGA contains a control register module with one memory mapped register that is used to reset the FPGA. This register is listed in Table 5-1.

Address	Write /Read	Register	Description
38000040	W/R	Control_reg	Control register

Table 5-1: FPGA Control Registers

The control register is used in the following way:

Bit 0:	If asserted to ‘0’, all the registers, FIFOs and state machines in the FPGA will be reset. Upon power up this bit is set to 0. The output of this bit is logically ANDed with the external $\overline{\text{RESET}}$ signal. Therefore the FPGA will remain in reset after the external reset has been de-asserted. It is necessary to write to logical ‘1’ to this bit to bring the FPGA out of reset.
Bits 15-1:	unused.

5.5 FPGA Logic Analyzer Interface

The extra IO pins that are available on the FPGA are used to interface with a Tektronix Logic Analyzer. In total 23 IO signals can be connected to the logic analyzer using headers U30 and U31. Series source resistor networks are used as termination resistors on these IO pins. The resistor networks are illustrated in schematic 9 in Appendix A. The Altera Quartus II Logic Analyzer Interface Editor is used to multiplex multiple banks of internal registers to the logic analyzer IO pins. It is necessary to setup the interface before the VHDL code is synthesized. Once, the code is compiled and downloaded into the FPGA, the Logic Analyzer Interface Editor uses JTAG to determine which banks are multiplexed to the logic analyzer.

For future expansion, zero ohm resistors are included in series with the headers U30 and U31. This would allow LED’s to be connected to the headers. The zero ohm resistors are illustrated in schematic 9 in Appendix A.

5.6 FPGA configuration

The Cyclone II FPGA devices use SRAM cells to store configuration data. Since SRAM memory is volatile, the configuration data must be downloaded to the FPGA each time the device powers up.

The FPGA can be configured by active serial (AS), passive serial (PS) or directly by JTAG (Joint Test Access Group). When using JTAG, the device needs to be programmed each time at power on (this method is good for debugging purposes). PS configuration allows the FGPA to be configured by an external host such as a microprocessor or the Altera MAX II CPLD. The AS configuration scheme allows the Cyclone II FPGA to be configured using low cost serial configuration devices. This method uses the smallest number of component pins to program the FPGA. The Altera EPCS1 (U12) serial configuration device is used to configure the Cyclone II EP2C5PQ208 FPGA (U41).

JTAG indirect configuration was used to program the EPCS1 serial configuration device. This method requires the fewest additional components and the smallest number of connections to the FPGA. The serial configuration device connections, the JTAG connections and the required FPGA configuration resistors are illustrated in schematic 8 in Appendix A.

JTAG indirect configuration is performed as follows. First the synthesized FPGA programming file is loaded into the Altera Quartus programming file converter to generate a JTAG indirect programming file. Compression needs to be enabled when generating the serial configuration device configuration file in order to fit the configuration file into the EPCS1 configuration device. The JTAG programmer utility is then configured to perform the JTAG indirect programming. Once programming is initiated, the FPGA is programmed with a serial flash programming utility. The FPGA programmer then inherently uses this utility to program the serial configuration device. Once complete, the FPGA power needs to be reset. Thereafter the FPGA is configured upon power up using the configuration that is now stored in the EPCS1 serial configuration device. This configuration takes approximately 61.2ms and it is therefore necessary to keep the system in reset or to prevent it from communicating with the FPGA until the configuration is complete.

5.7 FPGA Power Requirements

The FPGA requires a 1.2V supply for the core and 3.3V for the IO. The requirements and power calculations for these supplies are discussed in section 15.1 on page 113 . Schematic 10 in Appendix A illustrates which pins are connected to these supplies. The decoupling capacitors which are placed as close as possible to the FPGA power supply pins are also illustrated in this schematic.

5.8 Summary and Conclusion

This chapter described the use of the Altera Cyclone II FPGA as the interface to the peripheral components in our design. The interface that allows communication between the FPGA and the TigerSHARC using the pipelined protocol was described here in detail, as was the internal FPGA module design, the logic analyzer interface and the FPGA control register module.

The following chapters discuss the peripheral components of the system and the FPGA's internal modules that interface with each.

6 Audio System

This chapter describes the design of the audio system. Initially an overview of the audio system is given, thereafter the implementation of each component and the FPGA modules that interface with these components is described.

6.1 Overview

The audio system requirements include stereo ADC and DAC audio channels, as well as microphone pre-amplification. After considering a number of ADC and DAC device options, the decision was made to implement 1 stereo line level output channel, 1 stereo line level input channel and to implement two microphone channels.

The Analog Devices ADI1836A codec and Texas Instruments PGA2500 digitally controlled differential analog microphone pre-amplifier were chosen to perform these tasks. A high level block diagram of the audio system is illustrated in Figure 6-1, and design and implementation of this system is discussed in the following sections.

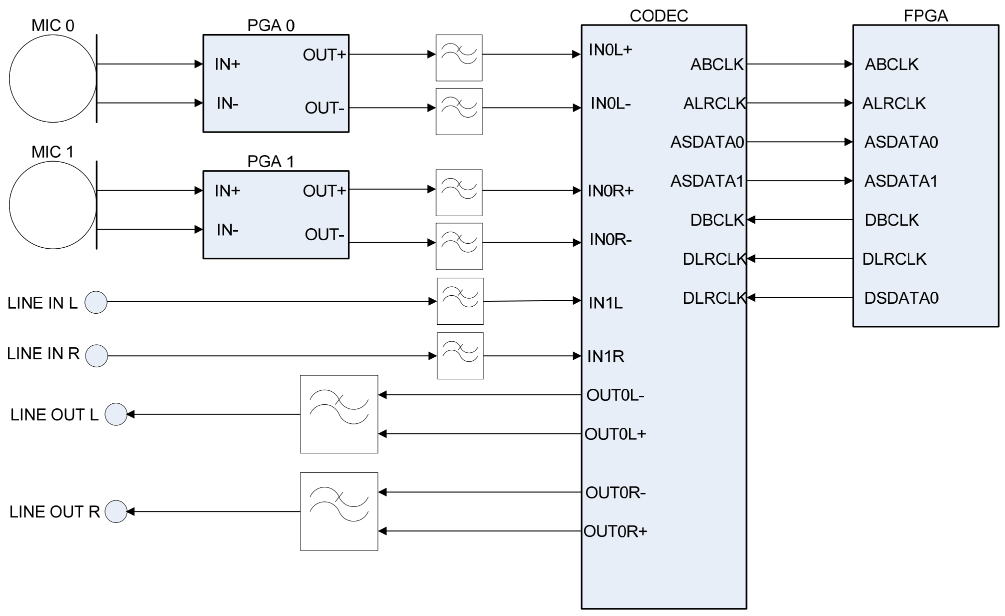


Figure 6-1: A high level block diagram of the audio system.

6.2 Analog Devices ADI1836A Codec

The ADI1836A (U40) is a high performance single chip codec that provides 6 DACs (configured as 3 stereo pairs) and 4 ADC (configured as two stereo pairs). The ADI1836A (illustrated in schematic 14 in Appendix A) uses a multi-bit sigma-delta architecture. It supports 16 and 24 bit sampling resolutions at sampling frequencies of 48 and 96 KHz. The ADC's include an on-board digital decimation filter with 120 dB stop-band attenuation and linear phase response, operating at an over-sampling ratio of 128 (for 48 kHz operation) or 64 (for 96 kHz operation).

Design Considerations

ADC Side

The primary ADC pair operates in differential input mode, while the second ADC pair can operate in either differential or in single ended mode. The best performance in terms of signal to noise ratio is achieved when the ADCs operate in differential mode. If the ADC is operated in differential mode, then the external differential amplifier must be isolated from glitches caused by the ADC's internal switching capacitors by placing a single pole anti-aliasing filter (consisting of a series connected 100 ohm resistor (R57, R58, R84 and R85) and a 1 nF capacitor (C14, C15, C80 and C92)) placed between each input and ground (these resistors and capacitors are illustrated at the output of the PGAs in schematic 3 and 12 in Appendix A. If the ADC is configured in single ended mode, it is not necessary to include the resistor because the internal programmable gain amplifier (PGA), which is only used for singled ended configurations, is internally connected to the ADC through a series 100Ω resistor. Only the capacitors (C94, C105, C106 and C129) are needed, as they are connected directly to these internal 100 Ω resistors.

It was decided to implement the primary ADC channel in differential mode and the secondary channel in single ended mode. This configuration allows the microphone programmable amplifiers to be connected directly to the ADC1L and ADC1R channel since they provide a differential output signal. The microphone PGAs are discussed in section 6.3. This configuration also allows an audio line-in channel to be connected directly to the ADC2L and ADC2R channel in single ended mode.

DAC Side

The DAC channels provide the codec with six fully differential analog outputs for improved noise and distortion performance. Each of the differential output pins have a DC offset of 2.25V and a peak-peak swing of +-1.4V for a 0dB signal. The datasheet recommends a single opamp third order low pass filter to remove high frequency noise on the output pins. The filter used is discussed in section 6.2.2. Our application requires only one stereo output channel and therefore only one of the stereo output channels is used.

Clock Signals

The codec can operate at a master clock frequency of 256, 512 or 768 times the sample rate. When operating at a 48 KHz sample rate and using a clock multiplier of 256, the required input frequency is 12.288MHz. The C-MAC CFPS-73B 12.288MHz crystal oscillator (U22) was chosen to drive the master clock and is illustrated in schematic 14 in Appendix A.

Serial Interface

The codec transfers the audio samples to the CPU via a serial port, which supports right or left justified, I2S and 128 or 256 bit packed DSP serial port modes. The default mode used by the codec is the I2S mode and this was used to transfer the audio samples from the codec to the FPGA and vice versa. The I2S interface is discussed in section 6.2.1 on page 46.

Codec Configuration

The codec's internal control registers are programmed separately using the SPI protocol. The following section discusses the codec's configuration registers. The SPI protocol is also used for inter-component communication to the microphone programmable gain amplifiers and is therefore discussed in section 6.5 on page 57.

The codec uses a 16bit SPI word format which is illustrated in Table 6-1.

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	4 bit address				RD/WR		10 bit data field									

Table 6-1: Codec's 16-bit SPI word format

- Bits 15 to 12 are used as 4 bit address.
- Bit 11 determines a read or write command. i.e. 1=read, 0=write.
- Bit 10 is reserved.
- Bit 9 to 0 serves as a 10 bit data field.

Extracts of the datasheet describing the configuration registers of the CODEC, are illustrated in Appendix D on pages 207,208 and 209.The codec is configured once initially during operation.

The DAC side is configured to operate in the following mode:

- Sampling frequency :48KHz
- Sampling resolution :16bit
- Serial mode: I2S
- DAC1L and DAC1R channels are configured on and DAC(3:2)L and DAC(3:2)R are muted.
- The DAC1L and DAC1R volume registers are set to zero attenuation.

The ADC side is configured to operate in the following mode:

- Sampling frequency :48KHz
- Sampling resolution: 16bit
- Serial mode: I2S
- ADC1 is configured in differential mode and ADC2 is configured in single ended mode.
- ADC1 and ADC2 gain is set to 0dB gain.
- The filter is set is set to high pass mode.

6.2.1 I2S interface

The I2S interface is a 3 wire interface and the format of the protocol is illustrated in Figure 6-2 .

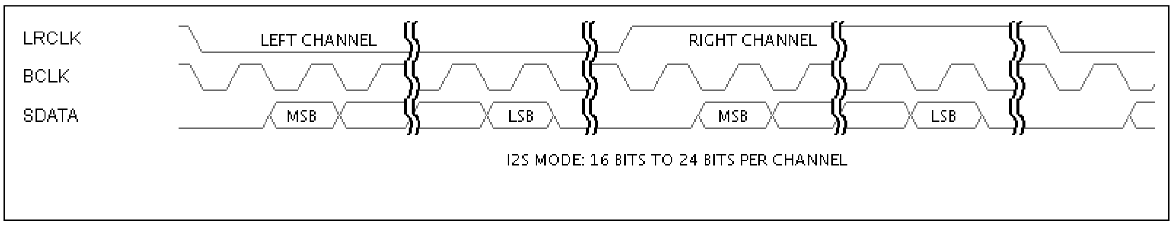


Figure 6-2: The I2S protocol.

The I2S interface consists of a bit clock (BCLK), a left-right channel clock (LRCLK) and the serial audio data (SDATA). The BLCLK clock operates at 64 times the sampling frequency and the LRCLK clock operates at the sampling frequency. As is illustrated in Figure 6-2, the audio sample bits are delayed by one cycle after the edge of the LRCLK. Figure 6-2 also illustrates that the most significant bit (MSB) of the audio sample is transmitted first. The size of the data transmitted is dependent on the configuration of the codec i.e. 16 or 24 bit.

6.2.2 Line out

The codec DAC1L and DAC1R channels are differential output signals. The datasheet recomends the the use of a 3rd order single opamp filter circuit (illustrated in Figure 6-3) to remove high frequency noise on the output pins and to convert the differential single to a single ended output. The same circuit used in the evalutation board is therefore employed in our design for the DAC1L and DAC1R channels. The Texas Instruments LMV722 dual opamp (U32) is used and the circuit which includes both channels is depicted in schematic 4 in Appendix A. The output of the filter is connected to a stereo RCA connector (U6).

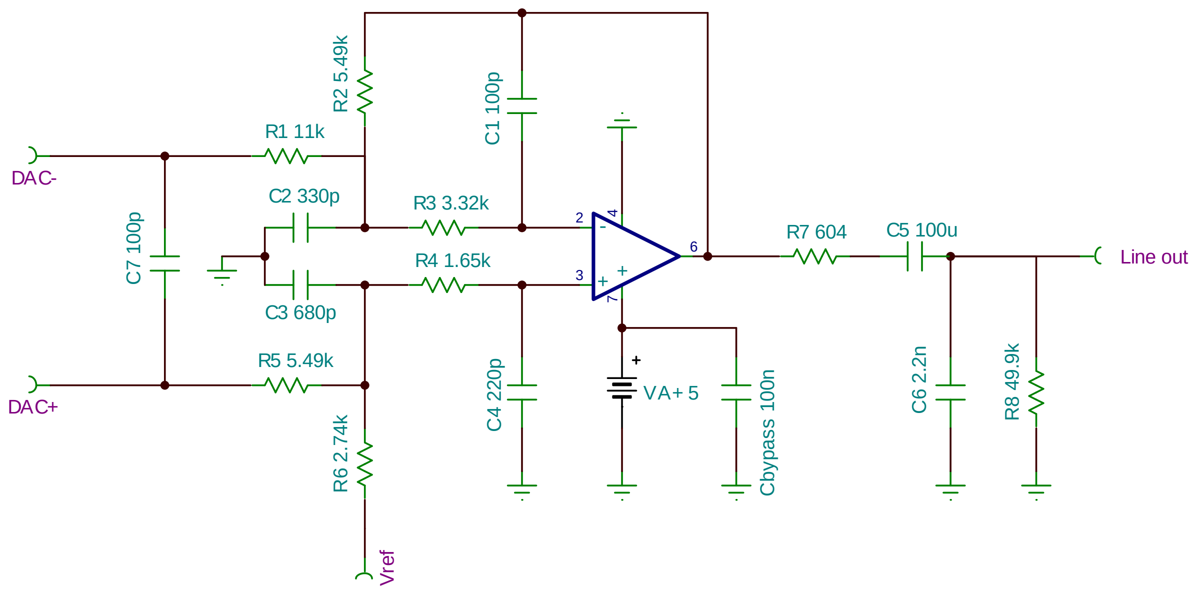


Figure 6-3: 3rd order single opamp low pass filter circuit.

An analysis of the filter was performed using the Texas Instruments TINA Spice simulation software and the frequency response is illustrated in Figure 6-4. As Figure 6-4 illustrates, the filter has -3dB frequency of 115 kHz. The cut-off frequency is high to ensure that the filter has a unity gain in the audio pass band (0-22.1kHz).

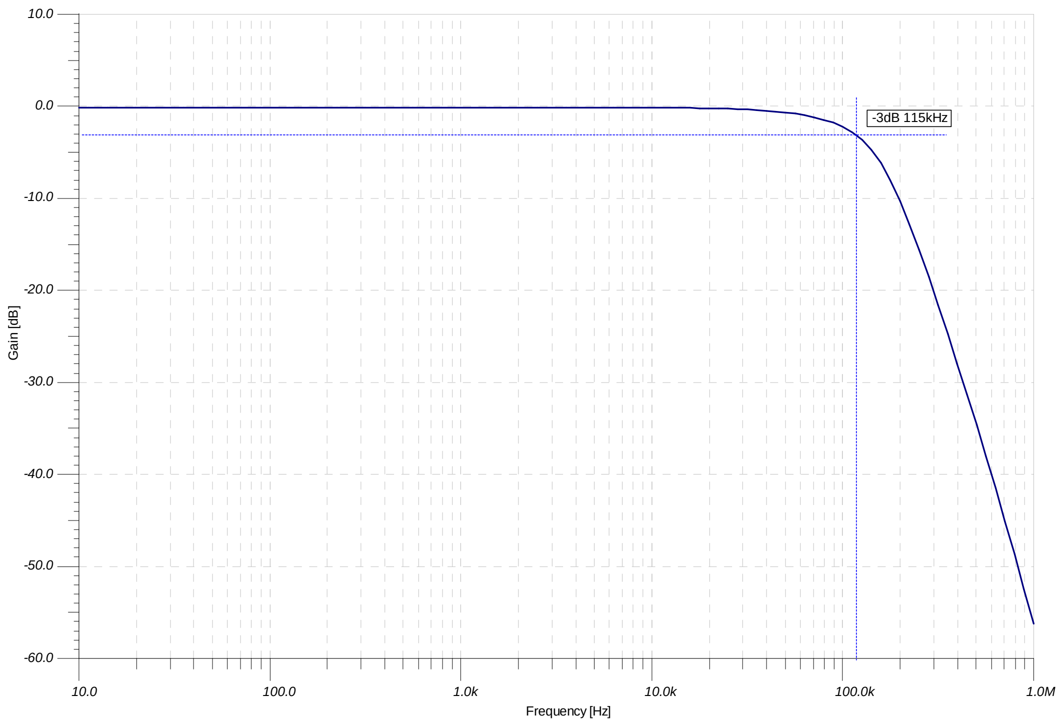


Figure 6-4: The frequency response of the DAC output LPF.

6.2.3 Codec reference voltage

The codec supplies a reference voltage (Vref), which is used to bias the external opamps and differential amplifiers to the same common mode voltage as used by the codec. Since, the maximum current that can be supplied by this output pin is 50uA, it is necessary to buffer Vref with an opamp (U33). This circuit is illustrated in schematic 14 in Appendix A.

6.2.4 Codec Power supply

The AD18836A is designed to operate from a 5V supply. To improve performance, separate analog and digital 5V power supplies have been employed. The Codec also provides a separate power supply connection for the digital output drivers that can be fed from a 3.3V or 5V source. Because the FPGA IO drivers operate at 3.3V, the digital output driver supply has been connected to a 3.3V source.

To further improve performance, separate analog and digital power and ground planes are used in this design. The analog and digital ground planes are connected together using a zero ohm resistor (R86) in schematic 14 in Appendix A. The PCB layout of the analog power and ground planes are illustrated in Appendix B.

6.3 Microphone Channels

The Texas instruments PGA2500 digitally controlled differential analog microphone pre-amplifier was chosen for this design. The PGA2500 is a differential input and differential output microphone preamplifier with low noise and a wide dynamic range. The programmable gain amplifier can be programmed to unity gain or 10dB-65dB gain in steps of 1dB. The amplifier also has four programmable general purpose CMOS outputs. The gains settings and internal functions are programmable using an SPI port.

Two PGA2500 microphone amplifiers are used in this design. One PGA (U10) is connected to the differential input of the ADC1L channel and the other PGA (U43) is connected to the differential input of the ADC1R channel. Both PGAs use the same bias circuitry and therefore only U10 (illustrated in schematic 3 in Appendix A) is discussed here. The schematic for U43 is illustrated in schematic 12 in Appendix A.

The same circuit which is recommended in the datasheet (10) is employed in our design. An XLR microphone connector (U36) is used to connect the microphone to the system. The differential output of the XLR input is connected to DC blockings capacitors (CT21 and CT22), current limiting resistors (R62 and R61), short circuit protection diodes (D5, D6, D7 and D9) and then the differential input of the PGA. The diodes used are the MBRA120TLT3 diodes from ON semiconductor, which are the same diodes as recommended in the datasheet (10).

The DIP switch (U28) is connected to pins 7, 8 and 9 of the PGA via pull-up resistors R54, R56 and R53. The functions of each pin of the switch are shown in Table 6-2.

Switch pin	PGA Pin	Off (pulled-low)	On (pull-up)
1	8	Gain is set using SPI	Sets the gain to unity gain
2	9	Zero crossing detector disable	Zero crossing detector enable
3	7	DC servo enable	DC servo disable
4	NC	-	-

Table 6-2: PGA Configurable pins connected to switch U28

The internal DC servo minimizes the DC offset present at the output. The zero crossing detector pin forces the PGA to apply gain changes on zero crossings of the input signal. Our system operates in the default configuration with both the DC servo and zero crossings detector enabled.

The general purpose output pin 1 (GPO1) is connected to the green LED3 with a 475Ω series resistor. The LED is used by the system to indicate that the PGA is on. The over voltage output register pin (OVR) is connected to the red LED4 with a 475Ω series resistor. The LED is used by the system to indicate that the input signal is clipping.

6.3.1 PGA SPI Interface

The Texas Instruments PGA2500 internal configuration register is programmed using the SPI protocol. The following section discusses the PGA2500's configuration register. The SPI protocol is also used for communication between other components and is therefore discussed in section 6.5 on page 57.

The PGA2500 uses a 16bit SPI word that is illustrated in Table 6-3:

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DC Servo Enable=0	CM Servo Enable=1	0	Overload Indicator	GPO [4:1]				0	0	G[5:0]					

Table 6-3: PGA2500 16-bit SPI word format

Bit 15:	DC servo enable (Active low). See explanation in the previous section.
Bit 14:	CM servo enable (Active high). This servo ensures that differential input and output signals have the same common mode voltage.
Bit 13:	Unused, always ‘0’
Bit 12:	Overload Indicator Bit (0=5.1Vrms, 1=4Vrms).
Bit 11-8:	GPO 4-1. General purpose outputs (active high).
Bit 7:	Unused, always ‘0’.
Bit 6:	Unused, always ‘0’.
Bit 5-0:	G[5:0] amplifier gain. N=G[5:0] dB

For N=0

Gain= 0dB

For N=1:56

Gain(dB)=9+N

For N=57:63

Gain(dB)=65

The PGA2500 supports daisy chaining of the SPI protocol. Figure 6-5 illustrates that the FPGA SDO is connected to PGA0 SDI, PGA0 SDO is connected PGA1 SDI and PGA1 SDO is connected to FPGA SDI.To program the two PGA2500s in this design a 32-bit word is transferred via the SPI protocol. The upper 16-bits are loaded into PGA1 and the lower 16-bits are loaded into PGA0. It is possible to read the previous configuration of the PGAs since this is output over the SPI protocol when the PGAs are programmed.

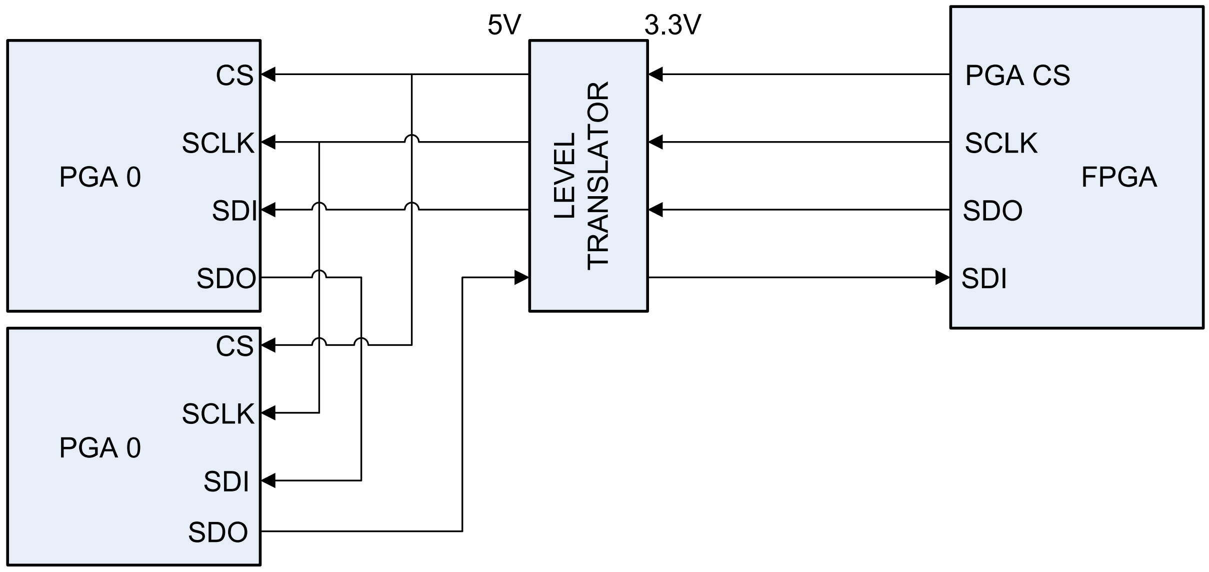


Figure 6-5: Daisy chain arrangement of the PGAs.

6.4 FPGA Audio Module

Figure 6-6 illustrates the design of the FPGA audio module which is used to interface the TigerSHARC processor to the audio codec. The design and operation of this module is discussed in the following sections.

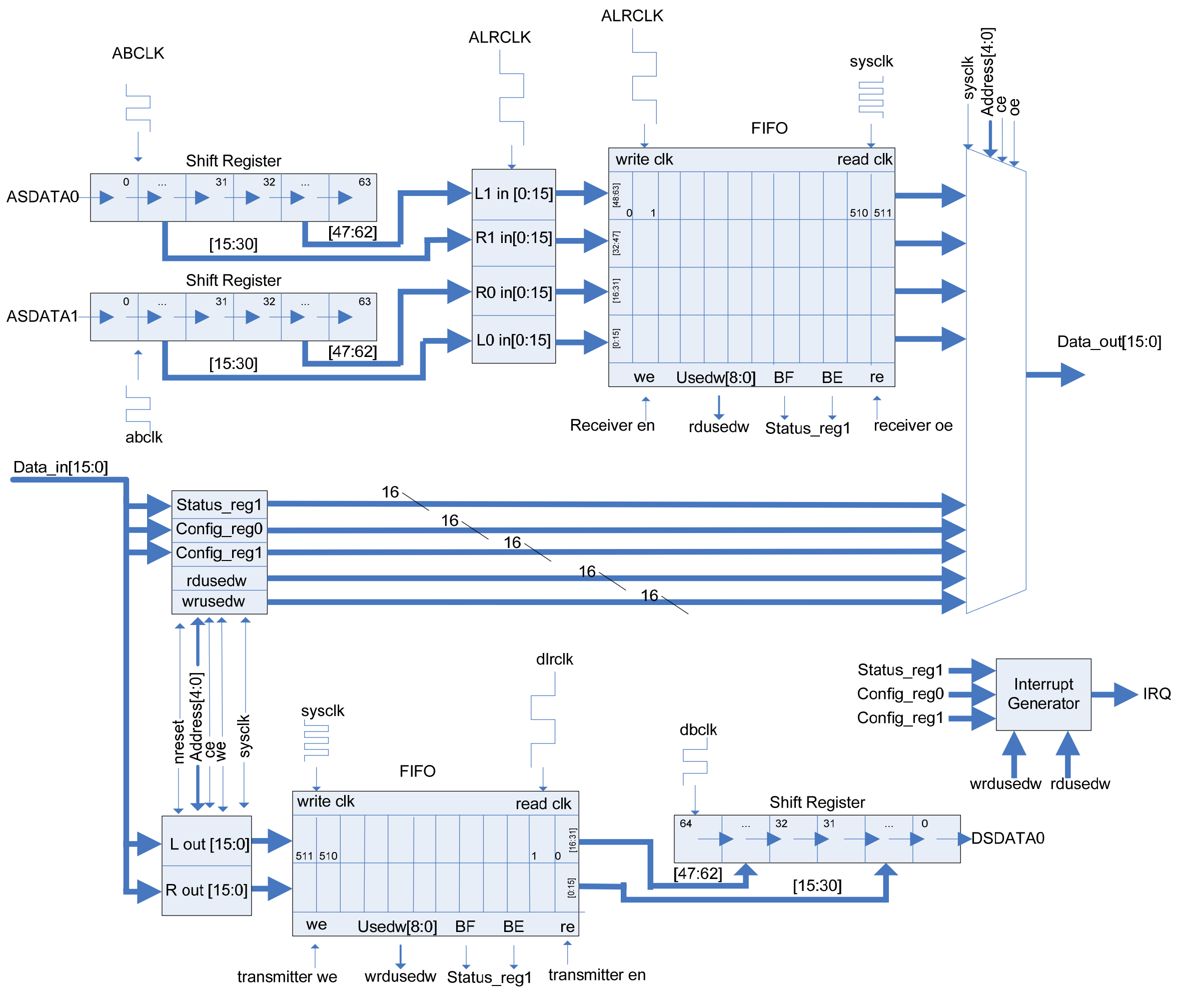


Figure 6-6: The FPGA audio module.

Multi Channel Audio Input

One 64-bit shift register per ADC stereo pair is used to receive the serial audio data for each channel from the codec according to the I2S format (discussed in section 6.2.1 on page 46). The serial data (ASDATA0 and ASDATA1) is shifted into the shift registers at each rising edge of the ABCLK clock signal. The ABCLK signal operates at 64 times the sampling frequency (Fs) and is generated by the codec. The 16-bit left and right channel data is then loaded into registers (L1_in, R1_in, L0_in and R0_in) at the rising edge of ALRCLK. The ALRCLK signal operates at Fs and is also generated by the codec. At the next rising edge of ALRCLK, L1_in, R1_in, L0_in and R0_in are loaded into a 512x64-bit FIFO. The output side of the FIFO operates from the system clock (SCLK). The 64-bit output of the FIFO appears as 4 addressable 16-bit registers as described in Table 6-4. The use of the FIFO allows the audio input to be transferred in blocks to the internal memory of the TigerSHARC DSP.

The FIFO supplies certain status and flag variables, namely the number of words of data stored in the FIFO, a buffer full flag and a buffer empty flag. A threshold flag is also generated, by comparing the number of words in the FIFO with a threshold that can be programmed into config_reg0. The flags are stored in status_reg1 and the number of words used as well as status_reg1 are available to the user as read only addressable registers.

Multi Channel Audio Output

The 16-bit left and right channel output data is written into registers L0_out and R1_out and is then loaded into a 512x32bit FIFO. The output side of the FIFO operates at DLRCLK which is an internal copy of the ALRCLK. At the rising edge of the DLRCLK, the 16-bit left and right channel data is loaded from the FIFO into the 64-bit shift register. The serial audio data (DSDATA0) is then transmitted to the codec at the rising edge of the DBCLK, which is an internal copy of the ABCLK.

The use of the FIFO allows the audio output to be transferred to the FPGA in blocks from the internal memory of the TigerSHARC DSP.

The output FIFO supplies certain status and flag variables, namely the number of words of data stored in the FIFO, a buffer full and a buffer empty flag. A threshold flag is also generated by comparing the number of words in the FIFO with a threshold that can be programmed into config_reg1. The flags are stored in status_reg1 and the number of words used, as well as status_reg1 are available to the user as read only addressable registers.

Audio Interrupt Request

The audio module can be configured to generate an interrupt request signal. This signal can be used by the CPU to initiate the transfer of a block of audio data between TigerSHARC memory and the audio module FIFOs. The interrupt request (IRQ) generator can be configured to monitor the input or the output FIFO threshold flag. If the configured flag is asserted, then the IRQ signal is asserted and remains asserted until the threshold flag is de-asserted.

Audio Module Registers

Table 6-4 illustrates the memory addressable registers within the FPGA Audio module. The first column indicates the address of the register, the second column indicates whether register is writable (W), readable (R) or both (W/R), the third column is the register's VHDL name and the fourth column is a brief description. If necessary a detailed description of the register is given below the table.

Address	Write /Read	Register	Description
3800000	W R	Left_out Right0_in	Left DAC channel data Right 0 ADC channel data
3800002	W R	Right_out Left0_in	Right DAC channel data Left0 ADC channel data
3800004	W	Right1_in	Right1 ADC channel data
3800006	W	Left1_in	Left1 ADC channel data
3800008	W/R	Config_reg0	Configuration register 0
380000A	W/R	Config_reg1	Configuration register 1
380000C	R	Status_reg1	Status register 1
3800010	R	IFIFO_usedw	Number of Input FIFO words used (max=512)
3800012	R	OFIFO_usedw	Number of Output FIFO words used (max 512)

Table 6-4: Audio module memory addressable registers.

Configuration register 0:

Bit 0: IRQ enable:

If bit 0 is set to '1' , this will enable the audio module's interrupt generator.

Bits2-1: Output-FIFO read mask:

"01": The right output channel audio data must first be written to the right_out register. When the left audio data is written to the left_out register, the output FIFO write enable is asserted. The left_out and right_out register will then be loaded into the output FIFO.

"10": The left output channel audio data must first be written to the left_out register. When the right audio data is written to the right_out register, the output FIFO write enable is asserted. The left_out and right_out register will then be loaded into the output FIFO.

Bits 6-3: Input-FIFO read mask:

"0001": The current input audio data, except input channel right 0 must be read first. When the right 0 audio data is read, the input FIFO read enable is asserted. The FIFO will then output the next set of audio samples.

"0010": The current input audio data, except input channel left 0 must be read first. When the left 0 audio data is read, the input FIFO read enable is asserted. The FIFO will then output the next set of audio samples.

"0100": The current input audio data, except input channel right 1 must be read first. When the right 1 audio data is read, the input FIFO read enable is asserted. The FIFO will then output the next set of audio samples.

"1000": The current input audio data, except input channel left 1 must be read first. When the left 1 audio data is read, the input FIFO read enable is asserted. The FIFO will then output the next set of audio samples.

Bits 15-7:	Input FIFO threshold. 0<integer<512 If the number of words in the FIFO is equal or greater than this number then the threshold flag is asserted.

Configuration register 1:

Bits 0-2:	Unused
Bit 3:	Audio module enable. Setting this bit to ‘1’ will enable the audio module.
Bits 6-4:	Unused
Bits 15-7:	Output FIFO threshold 0<integer<512 If the number of words in the FIFO is less than this number then the threshold flag is asserted.

Status register 1:

Bit 0:	Output FIFO buffer empty flag. ‘1’=empty
Bit 1:	Output FIFO buffer full flag synchronized with the write clock. ‘1’=full
Bit 2:	Input FIFO buffer empty flag. ‘1’=empty
Bit 3:	Input FIFO buffer full flag. ‘1’=full
Bit 4:	Unused.
Bit 6:	Input FIFO buffer threshold flag. When high, the number of stored words is greater or equal to the receiver threshold.
Bit 6:	Output FIFO buffer threshold flag. When high, the number of stored words is less than the receiver threshold.
Bits 15-7:	Unused.

6.5 Serial Peripheral Interface

The Serial peripheral interface (SPI) is a 4 wire serial interface. The interface consists of a chip select signal (CS), a serial clock signal (SCLK), a serial data out signal (SDO) and a serial data in signal (SDI). Various implementations of the protocol exist because different manufacturers use different data word sizes and may choose to receive and transmit data on the rising edge or on the falling edge of the serial clock. The components used in this project are designed to operate as slave devices while the host operates as the master. Each of the components receives data on the rising edge and transmits data on the falling edge of SCLK. A timing diagram of the SPI protocol is illustrated in Figure 6-7.

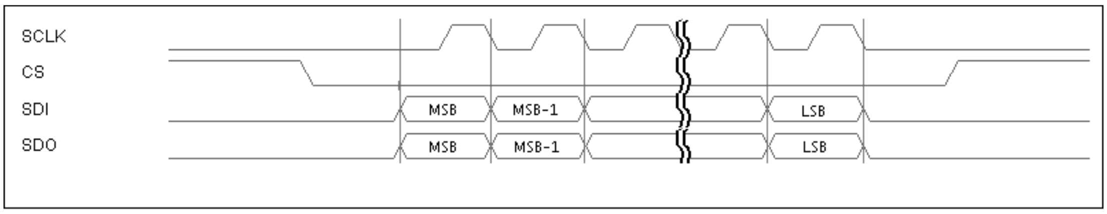


Figure 6-7: A timing diagram of the SPI protocol.

The SPI interface is used to write to and read from the configuration registers of the two programmable microphone amplifiers and the audio codec. A high level block diagram of this system is illustrated in Figure 6-8.

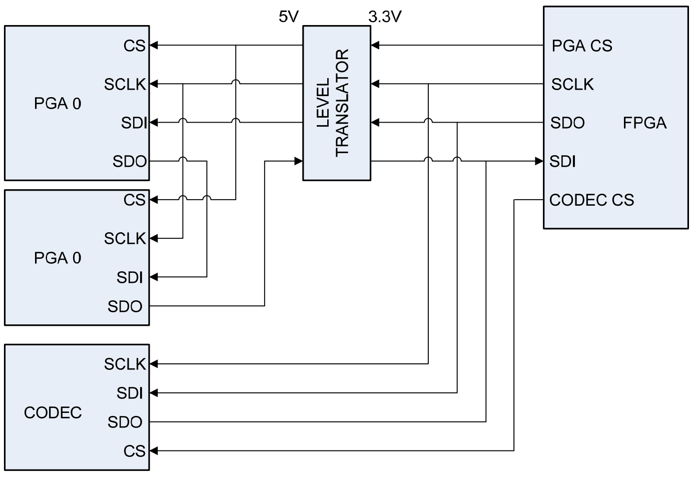


Figure 6-8: A high level block diagram of the SPI system.

As is illustrated in Figure 6-8, the SPI system requires a level translator as the PGA operates at 5V and the audio codec and FPGA operate a 3.3V. The MAX3002E bidirectional level translator is used to perform this function. Since it is used in multiple instances, the MAX3002E is described in chapter 11.

6.6 The FPGA SPI module

A high level block diagram of the SPI top level VHDL module is illustrated in Figure 6-9. The module contains memory addressable registers and a lower hierarchy module called SPI Interface. The module was initially written with a 32-bit system interface, however it was found that there were too few pins available on the FPGA to accommodate a 32-bit interface, and hence a 16-bit interface was used instead. This was accomplished by allowing the lower and upper 16-bits of each register to be accessed separately. The accomplished module itself however operates as a 32-bit system.

A high level block diagram of the SPI interface module is illustrated in Figure 6-10 and a description of the registers is given in Table 6-5. The core of the SPI interface consists of a 128-bit shift register and a state machine. As is illustrated in Figure 6-10, the internal state machine controls the operation of the shift register, internal signals and output signals.

The operation of the module is described after the description of the registers.

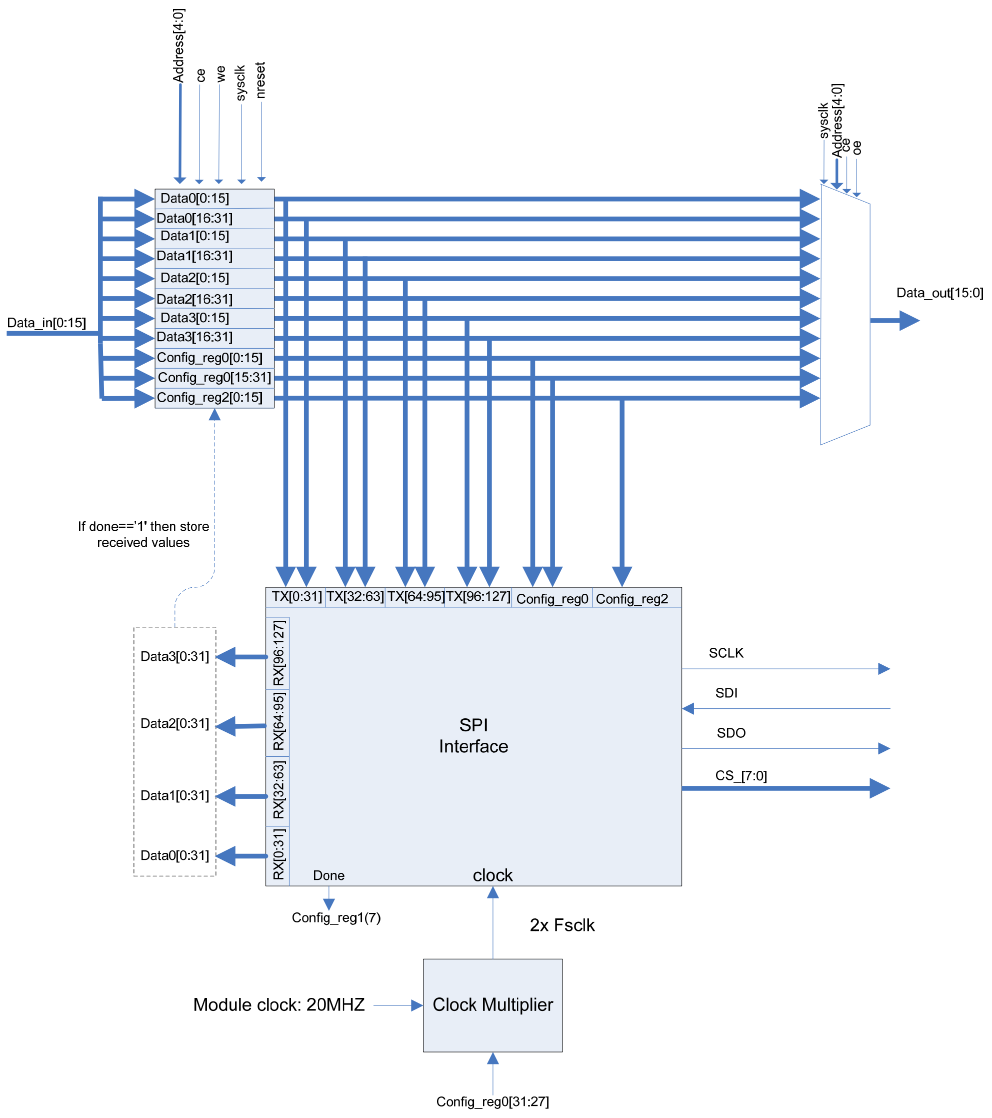


Figure 6-9: A high level block diagram of the SPI top level VHDL module.

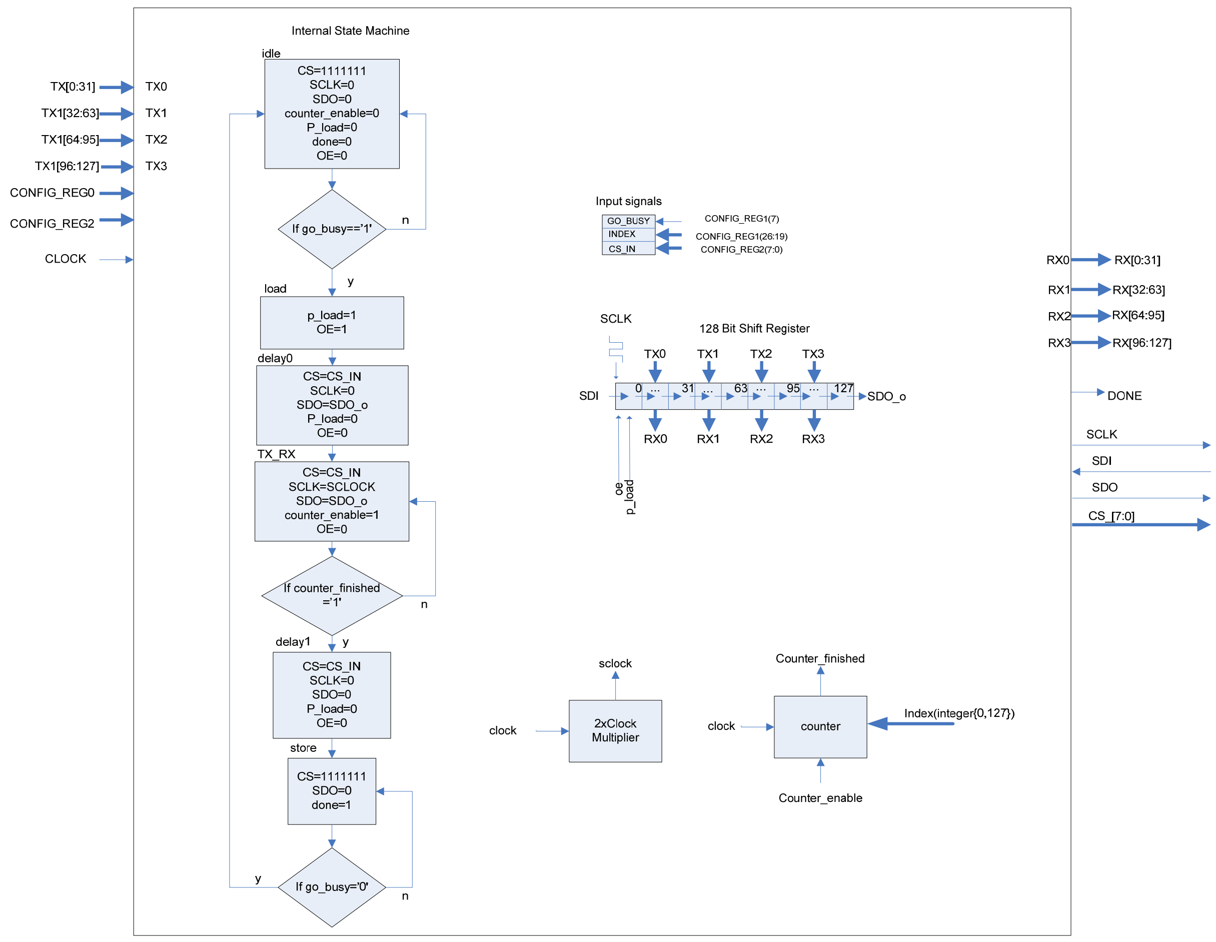


Figure 6-10: A high level block diagram of the lower hierarchy SPI interface module.

Address	Write /Read	Register	Description
38000020	W/R	Data_reg0[15-0]	Data register 0 lower 16-bits
38000022	W/R	Data_reg0[31-16]	Data register 0 higher 16-bits
38000024	W/R	Data_reg1[15-0]	Data register 1 lower 16-bits
38000026	W/R	Data_reg1[31-16]	Data register 1 higher 16-bits
38000028	W/R	Data_reg2[15-0]	Data register 2 lower 16-bits
3800002A	W/R	Data_reg2[31-16]	Data register 2 higher 16-bits
3800002C	W/R	Data_reg3[15-0]	Data register 3 lower 16-bits
3800002E	W/R	Data_reg3[31-16]	Data register 3 higher 16-bits
38000030	W/R	Config_reg0[15:0]	Configuration register 0 lower 16-bits
38000032	W/R	Config_reg0[31:16]	Configuration register 0 higher 16-bits
38000034	W/R	Config_reg2[15:0]	Configuration register 2

Table 6-5: SPI module memory addressable registers.

Table 6-5 illustrates the memory addressable registers within the SPI top level VHDL module. The first column indicates the address of the register, the second column indicates whether register is writable (W), readable (R) or both (W/R), the third column is the register's VHDL name and the fourth column is a brief description o the register. A detailed description of the registers is given below:

Configuration register 0:

Bits 6-0:	Reserved
Bit 7:	Busy-or-go flag. If asserted to '1', the SPI will transmit and received the payload. The busy flag will remain asserted to '1' until the operation is complete. Once the busy flag is deasserted to '0', then the received data will appear in the data registers[3:0].
Bits 16-8:	Reserved
Bit 17:	Clock enable: A logical '1'enables the SPI serial clock is enabled
Bit 18:	Reserved
Bits 26-19:	Number of bits to be transmitted. (Maximum=128).
Bits 31 -27:	SPI SCLK clock counter (count)

Configuration register 2:

Bits 7-0:	SPI_CS(7:0)

SPI Initialization

Upon reset, all bits in configuration register 0 are set to zero and all bits configuration register 2 are asserted to ones. The SPI internal clock is therefore disabled and the CS registers are all set to '1'. The internal state machine is also reset into its idle state.

It is necessary to program the SPI clock (set in bits 31-27 of configuration register 0) according to the following equation:

$$F_{sclk} = \frac{20}{2\times(2\times count+1)} MHz$$

Once the clock has been programmed it may be enabled by asserting bit 17 in configuration register 0.

SPI Operation

The SPI module is designed to transmit and receive a maximum of 128-bits of data. The user must first poll the busy flag to determine whether the SPI module is busy. If the module is busy then the user must wait until this is no longer the case. The transmission payload can be loaded into data registers 3 to 0 (the transmission occurs from the most significant bit (MSB) in data register 3 to the least significant bit (LSB) in data register 0). The user may then program the CS signals in configuration register 2 and the number of bits which must be transmitted in configuration register 0.

The user must then assert the busy-or-go flag to ‘1’. The internal state machine, which operates at twice the SCLK frequency, then proceeds from its idle state to the load state, where the data is loaded into the shift register. The state machine then transits to the delay0 state, where the programmed CS is asserted and a delay of one SCLK cycle is inserted. The SDO and SCLK signals remain ‘0’ whilst in this delay. The state machine then transits to the TX_RX state. In this state the data is transmitted serially via SDO and received via SDI while the SCLK signal output. A counter controls the number of bits that are received and transmitted according to bits 26-19 of configuration register 0. Once the correct number of bits have been transmitted and received, the state machine transits to the delay1 state where the CS select remains asserted and the SCLK and SDO are set to ‘0’. The state machine then transits to the store state, where the CS is de-asserted and the received data is placed into data registers 3-0 (receive occurs from the LSB in data register 0 to the MSB in data register 3). Once the received data has been stored, the busy-or-go flag is de-asserted. The SPI module can then be loaded with the next payload.

6.7 Summary and Conclusion

This chapter described the design of the multi-channel audio system. This audio system incorporated the Analog Devices ADI1836A codec and the Texas Instruments PGA2500 digitally controlled differential analog microphone pre-amplifiers, to give the TigerSHARC access to a stereo line-in audio channel, a stereo line-out audio channel and two microphone channels with up to 65dB gain each. The FPGA audio module is efficiently used to transfer all of the channels audio data in DMA block transfers and therefore minimizes the processing time required to transfer the slow IO of the audio. Finally, an FPGA SPI module is used to program the gain of the microphone pre-amplifiers and the configuration registers of the codec.

7 Storage System

This chapter discusses the implementation of the Compact Flash storage card as the secondary storage medium and describes the design of the FPGA Compact Flash interface module in the following subsections.

7.1 Compact Flash Storage Medium

Communication to or from the Compact Flash storage card is performed using the task file registers that are integrated into the card. These registers provide all the necessary control and status information related to this storage medium.

The Compact Flash specification 4.0 (11) specifies three modes which can be used to interface with all Compact Flash cards:

1. PC CARD ATA using memory mode.
2. PC CARD ATA using I/O mode.
3. True IDE mode.

In memory mode, the Compact Flash registers are accessed via memory references. The registers are accessed asynchronously in a 2KB memory space. The internal 512-byte data FIFO's contents are individually addressable within in this memory space. To operate in this mode an asynchronous interface would need to be designed that conforms to the timing mode specifications of the Compact Flash card.

In I/O mode the standard PC-AT disk I/O memory spaces 1F0H-1F7H, 3F6-3F7h (primary) or 170H-177H, 376-377h (secondary) are used to access the registers. In this mode access is performed through interrupt requests and optional DMA transfers. The internal 512-byte data FIFO's individual contents are not addressable in this mode. To operate in this mode an interface would need to be designed that conforms to the DMA and Interrupt controller specifications.

The IDE Mode allows a computer to access the Compact Flash as an IDE hard drive. In this mode access is performed through interrupt requests and DMA transfers. The internal 512-byte data

FIFO's individual contents are not addressable in this mode. To operate in this mode an interface would need to be designed that conforms to the DMA and interrupt controller specifications.

It was decided to implement the Compact Flash card interface face in memory mode because this is the only mode where internal contents of the Compact Flash Card's data FIFO can be addressed separately in the 2K memory space. This would allow the host to manipulate the data within the Compact Flash instead of transferring the entire contents of the FIFO to the host, manipulate the data and transfer the data back to the Compact Flash.

Memory mode requires the following connections to interface with the card.

Input signals:

Address Bus [10:0]: The address bus addresses the internal registers.

$\overline{\text{REG}}$:	High, accesses common memory, while low accesses configuration memory. These memory areas are described in the test below.
$\overline{\text{CE1}}$	Low indicates an access to an even byte.
$\overline{\text{CE2}}$	Low indicates an access to an odd byte.
$\overline{\text{OE}}$	Output enable (active low).
$\overline{\text{WE}}$	Write enable (active low).
RESET	If high then the card will be reset.

Bidirectional signals:

Data Bus [15:0]

Power Connections:

VCC: This can be connected to a 3.3V or 5V supply. In this design VCC is connected to a 3.3V as this is the IO voltage used by the FPGA.

Output Signals:

$\overline{\text{CD1}}$, $\overline{\text{CD2}}$:	These signals are grounded internally and can be used by the host to determine whether a card is connected.
$\overline{\text{WAIT}}$:	This signal is driven low by the card to delay completion of the current IO cycle.
READY:	If high, the card is ready to accept a new data transfer operation.

The Compact Flash host interface and connections are illustrated in schematic 5 in Appendix A. The required pull-up resistors for the unused pins as well as the decoupling capacitors are also illustrated.

A TwinMOS Ultra-X 512MB 140X Compact Flash card was chosen for this design. This card is able to operate with the fastest 80ns read and write cycles. The 3M 7E50 horizontal Compact Flash card connector is used to connect the card to the hardware.

Access to a Compact Flash card can be performed in 8-bit or 16-bit mode using $\overline{\mathrm{CE1}}$ and $\overline{\mathrm{CE2}}$. Our design accesses the Compact Flash card in 16-bit mode to improve data throughput. Therefore $\overline{\mathrm{CE1}}$ and $\overline{\mathrm{CE2}}$ are connected together and are considered one signal $\overline{\mathrm{CE}}$.

The Compact Flash card contains two areas of memory, namely the configuration memory and the common memory.

The configuration memory contains several configuration registers, which are listed in Appendix D on page 210. The registers can be accessed using 300ns read cycles and 250ns write cycles as depicted in Appendix D on pages 210 and 211. The configuration registers default into memory mapped mode. Therefore it is not necessary to access the configuration registers (this would have been necessary to enable the other two modes).

The common memory area can be accessed using 250ns, 120ns, 100ns and 80ns read and write cycle modes. These modes are depicted in Appendix D on pages 212 and 213. By default, all Compact Flash cards are required to support read and write cycles of 250ns. The faster modes are vendor dependent.

The common memory area contains the task file registers. These registers are depicted in Appendix D on page 214 while an extract of the CF4.0 specification given in Appendix D on pages 215 to 220 lists the definitions of these registers. The CF-ATA command set uses the task file registers to issue commands to the Compact Flash card. The CF-ATA command set is displayed in Appendix D on pages 221 to222. Only explanations for those commands used in this design are listed in this appendix on pages 223 to 225. For details on other commands please refer to the CF4.0 specification (11).

Our design includes an interface that can access the Compact Flash card in all of its speed grades. To do this, the host is first configured to run at a 250ns cycle time. The speed grade information is then downloaded from the Compact Flash card, after which the interface can be configured to operate in the fastest speed mode supported by the card.

7.2 FPGA Compact Flash Module

The Compact Flash top level entity is illustrated in Figure 7-1. The figure illustrates the memory addressable registers and FIFOs that can be accessed by the TigerSHARC. A detailed explanation for each is given in Table 7-1. The configuration registers control the Compact Flash module and the lower hierarchy Compact Flash interface module. The data that is received from the Compact Flash via CF_dataout, is inserted into the data input FIFO. The address and the read or write command is loaded into the CA_in register which is then loaded into CAFIFO and the data that is written to the Compact Flash is loaded into the Data_reg register which is then loaded into the data output FIFO. The lower hierarchy Compact Flash interface module is used to produce the asynchronous signals that are used to interface with the Compact Flash card, and is discussed separately in section 7.2.

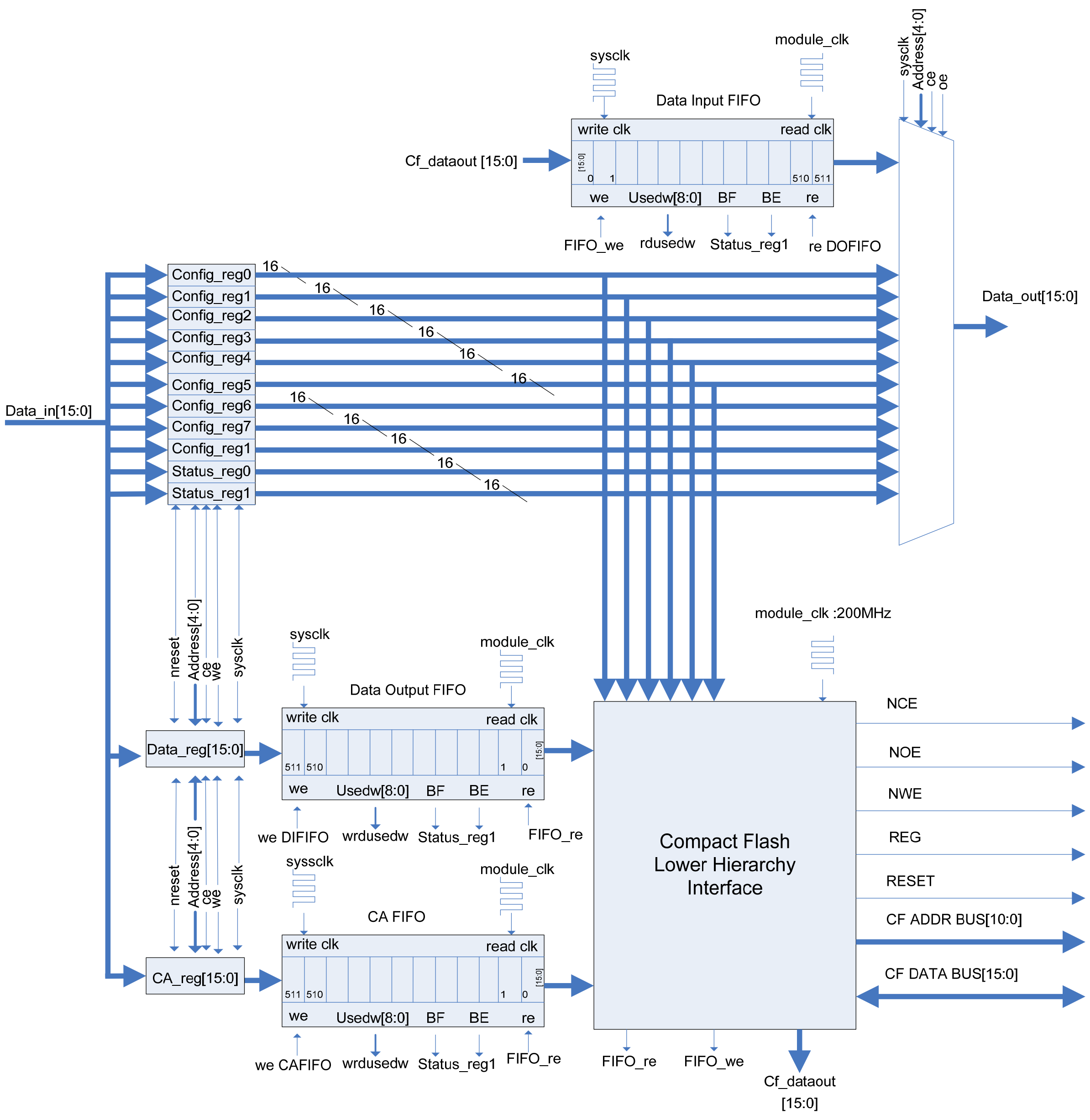


Figure 7-1: Compact Flash top level entity.

Address	Write /Read	Register Name	Description
38000060	W/R	Config_reg0	Configuration register 0
38000062	W/R	Config_reg1	Configuration register 1
38000064	W/R	Config_reg2	Configuration register 2
38000066	W/R	Config_reg3	Configuration register 3
38000068	W R	Data_reg	Data Output FIFO Data Input FIFO
3800006A	W	CA_FIFO	Command and address FIFO
3800006C	W R	Data_reg(direct) Status_reg0	Data Output FIFO. CA_FIFO is loaded with address of Compact Flash data register. Status register 0
3800006E	R	Status_reg1	Status register 1
38000070	R	Status_reg2	Status register 2
38000072	R	CA_FIFO usedw	Number of FIFO words used (max=512)
38000074	R	Data_OFIFO usedw	Number of FIFO words used (max=512)
38000076	R	Data_IFIFO usedw	Number of FIFO words used (max=512)
38000078	W/R	Config_reg4	Configuration register 4
3800007A	W/R	Config_reg5	Configuration register 5
3800007C	W/R	Config_reg6	Configuration register 6
3800007E	W/R	Config_reg7	Configuration register 7

Table 7-1 Compact Flash module memory addressable registers

Table 7-1 illustrates the memory addressable registers within the FPGA Compact Flash module. The first column indicates the address of the register, the second column indicates whether register is writable (W), readable (R) or both (W/R), the third column is the register's VHDL name and the fourth column is a brief description o the register. If necessary a detail description of the register s is given below:

Data Output FIFO: This FIFO is loaded with data that is sent to the Compact Flash card. The FIFO depth is 512x16Bits.

Data Input FIFO: This FIFO is loaded with data that is received from the Compact Flash card. The FIFO depth is 512x16Bits.

Data Command and Address FIFO:

This FIFO is loaded with the address and read or write command that is sent to the Compact Flash card. The FIFO depth is 512x14Bits. It is important to write to the data output FIFO first and then to the command and address FIFO, as the data is removed from these two FIFOs only if there is data available in the latter.

Bit 13: **'1'=** Read Command.

Bit 12: **'1'=**Write Command.

Bit 11: $\overline{\text{REG}}$ signal. When bit 11 is asserted to '0', the Compact Flash configuration registers can be accessed.

Bit 10-0: Compact Flash Address Bus.

The configuration registers that are described below are used to store the count values that are used in the Compact Flash interface module to generate the necessary asynchronous signals. Each register holds two count values. In each case the VHDL signal name is given, this is then followed by a value (eg. C1 or C10), which represents the VHDL signal in the flow chart in Figure 7-2. The count values can be related to the time in nano-seconds by multiplying the count value by 5ns (the period of the 200MHz module clock).

Configuration register 0:

Bits 6-0: read_cycle_time (C13): Read cycle time
(C12=C13-1) Time when FIFO_re is asserted
Bits 13-7: read_time_oe_L(C8): Time when OE is asserted to '0'

Configuration register 1:

Bits 6-0: read_time_oe_H (C9): Time when OE is de-asserted to '1'
Bits 13-7: reserved

Configuration register 2:

Bits 6-0: write_cycle_time (C7): Write cycle time

(C3=C7-1) Time when FIFO_re is asserted

Bits 13-7: write_time_we_L (C5): Time when WE is asserted to ‘0’

Configuration register 3:

Bits 6-0:write_time_we_H (C6): Time when WE is de asserted to ‘1’

Bits 13-7: write_time_data_out (C1): Time when data bus is driven

Configuration register 4:

Bits 6-0: write_time_data_Z (C2): Time when data bus is tri stated

Bits 13-7: read_time_CE_H (C10): Time when CE is de asserted to ‘1’

Configuration register 5:

Bits 6-0: write_time_CE_H (C4) : Time when CE is de asserted to ‘1’

Configuration register 6:

Bits 8-0:	Output FIFO threshold 0<integer<512 If the number of words in the FIFO is equal or greater than this number then the threshold flag is asserted.

Configuration register7:

Bits 8-0:	Input FIFO threshold 0<integer<512 If the number of words in the FIFO is equal or greater than this number then the threshold flag is asserted.

Status register 0:

Bit 0: Data output FIFO buffer empty flag.	‘1’=empty
Bit 1: Data output FIFO buffer full flag.	‘1’=full
Bits 11-3: Data output FIFO number of words used.	‘1’=full

Status register 1:

Bit 0: Command/Address output FIFO buffer empty flag.	‘1’=empty
Bit 1: Command/Address output FIFO buffer full flag.	‘1’=full
Bits 11-3: Command/Address output FIFO number of words used.	‘1’=full

Status register 2:

Bit 0: Data input FIFO buffer empty flag.	‘1’=empty
Bit 1: Data input FIFO buffer full flag.	‘1’=full
Bits 11-3: Data input FIFO number of words used.	‘1’=full

The Compact Flash lower hierarchy interface module operates at 200MHz. At this frequency the clock period is 5NS, which is required in order to create an asynchronous design that will be fast enough to operate the Compact Flash interface with 80ns read and write cycles.

The core of the design consists of a counter. The counter is used to generate the timing of the asynchronous signals which are used to interface with the Compact Flash card. The counter is initialized to zero at reset. In this state all the Compact Flash signals remain de-asserted and the data bus is tri-stated. When the count is zero, the CAFIFO buffer empty flag is constantly monitored. If the buffer empty flag is no longer asserted, the counter starts to count. Depending on whether the CAFIFO is loaded with a read or write command, the system starts processing either a read or write.

At each count the counter register is compared with the values (C0-13) that are stored in the configuration registers. Figure 7-2 presents a flow chart which indicates the signal values for a given value of the counter. Table 7-2 illustrates the appropriate count values for the 80NS read and write cycles.

If a read command is processed, the data that is received is inserted into the data input FIFO. The read cycle completes when the counter reaches the read cycle time, and the write cycle completes when the counter reaches the write cycle time. The counter is then reset and a new command can be processed. Table 7-2 illustrates the values that are used as the count values along with the signal timing values that are derived from the Compact Flash specification. Figure 7-3 illustrates the timing diagram for the write cycle and Figure 7-4 illustrates the timing diagram for the read cycle.

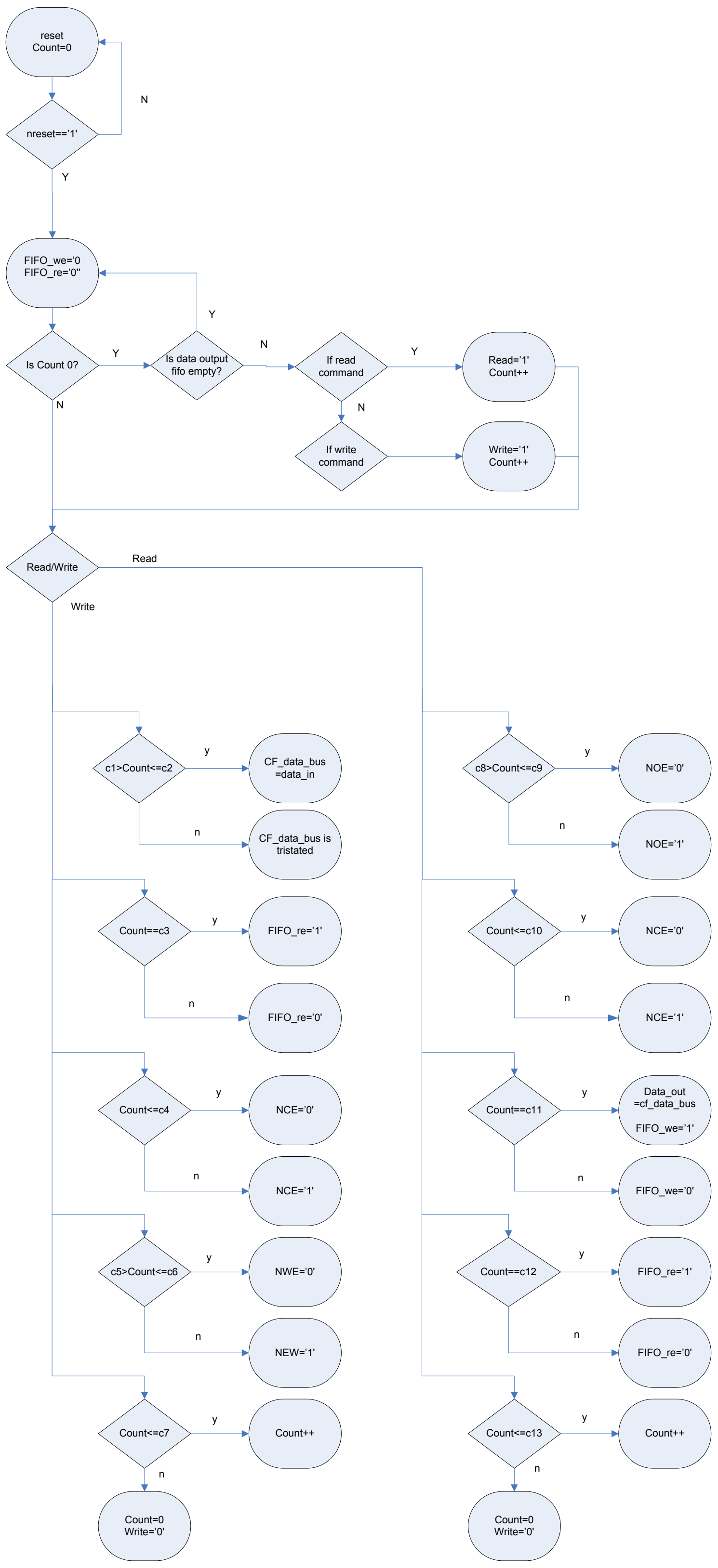


Figure 7-2: Compact Flash interface flow diagram.

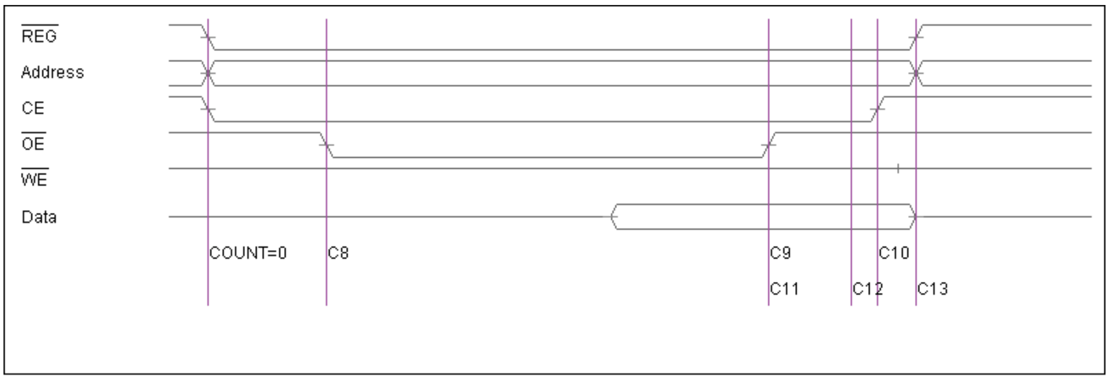


Figure 7-3: Compact Flash read cycle.

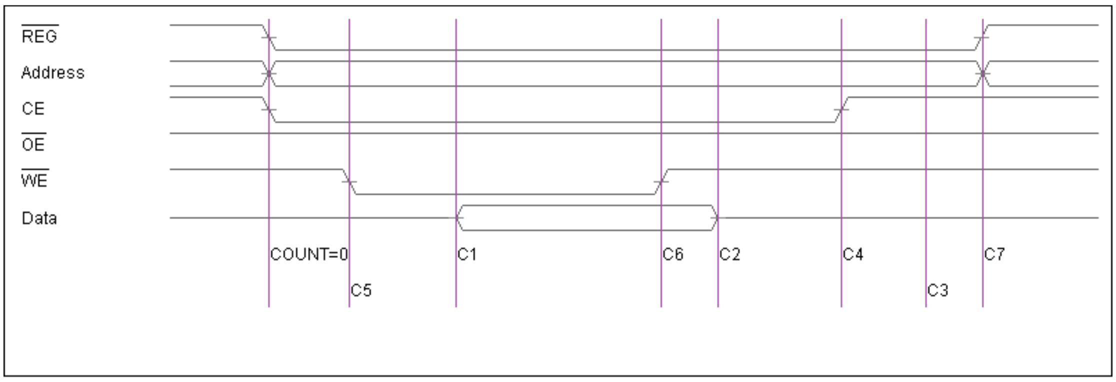


Figure 7-4: Compact Flash write cycle.

Count	Count value	VHDL Signal name	Time [ns]
C1	3	write_time_data_out	15
C2	11	write_time_data_Z	55
C3	14	FIFO_we_time	70
C4	12	Write_time_ce_H	60
C5	5	write_time_we_L	1
C6	9	write_time_we_H	45
C7	15	write_cycle_time	75
C8	1	read_time_oe_L	5
C9	13	read_time_oe_H	55
C10	13	Read_time_ce_H	65
C11	13	Read_time_data	55
C12	14	FIFO_re_time	70
C13	15	read_cycle_time	75

Table 7-2: The Compact Flash count values used for 80ns read and write cycles.

7.3 Summary and Conclusion

This chapter described the interface that is used to communicate with a Compact Flash storage card. This interface is consistent with the Compact Flash 4.0 specification. In conjunction with the FAT32 file system, our system can support a Compact Flash card with a capacity up to 32GB operating in the fastest transfer mode with read and write cycles of 80ns. Transfers between the FPGA Compact Flash interface and the TigerSHARC have been optimized to transfer an entire 256X 16-bit sector within the Compact Flash card via DMA.

8 LCD and USB System

This chapter describes the design of LCD and Universal Serial Bus (USB) system. It also describes the FPGA asynchronous interface that communicates to the Microchip PIC18F4550, which implements the USB protocols, and to the LCD.

8.1 FPGA Asynchronous Interface Module

The FPGA asynchronous interface module is used to communicate between the TigerSHARC and PIC18F4550 and between the TigerSHARC and the LCD screen. The LCD screen and PIC18F4550 share the same 8-bit data bus but have separate control signals. A high level block diagram of the asynchronous interface VHDL module is illustrated in Figure 8-1. The design consists of configuration registers, status registers, input and output FIFOs, and a lower hierarchy entity called Asynchronous Interface.

A detailed description of the memory mapped registers, input FIFOs and output FIFOs is given in Table 8-1. The lower hierarchy asynchronous interface entity consists of a state machine which is used to generate the asynchronous signals which are required to interface with the LCD screen and PIC18F4550. The algorithmic state machine (ASM) chart is illustrated in Figure 8-2. The internal flow of the LCD write and LCD read states are in turn illustrated in Figure 8-3. The operation of the asynchronous interface module will be explained using read and writes to the LCD and PIC18F4550 as examples.

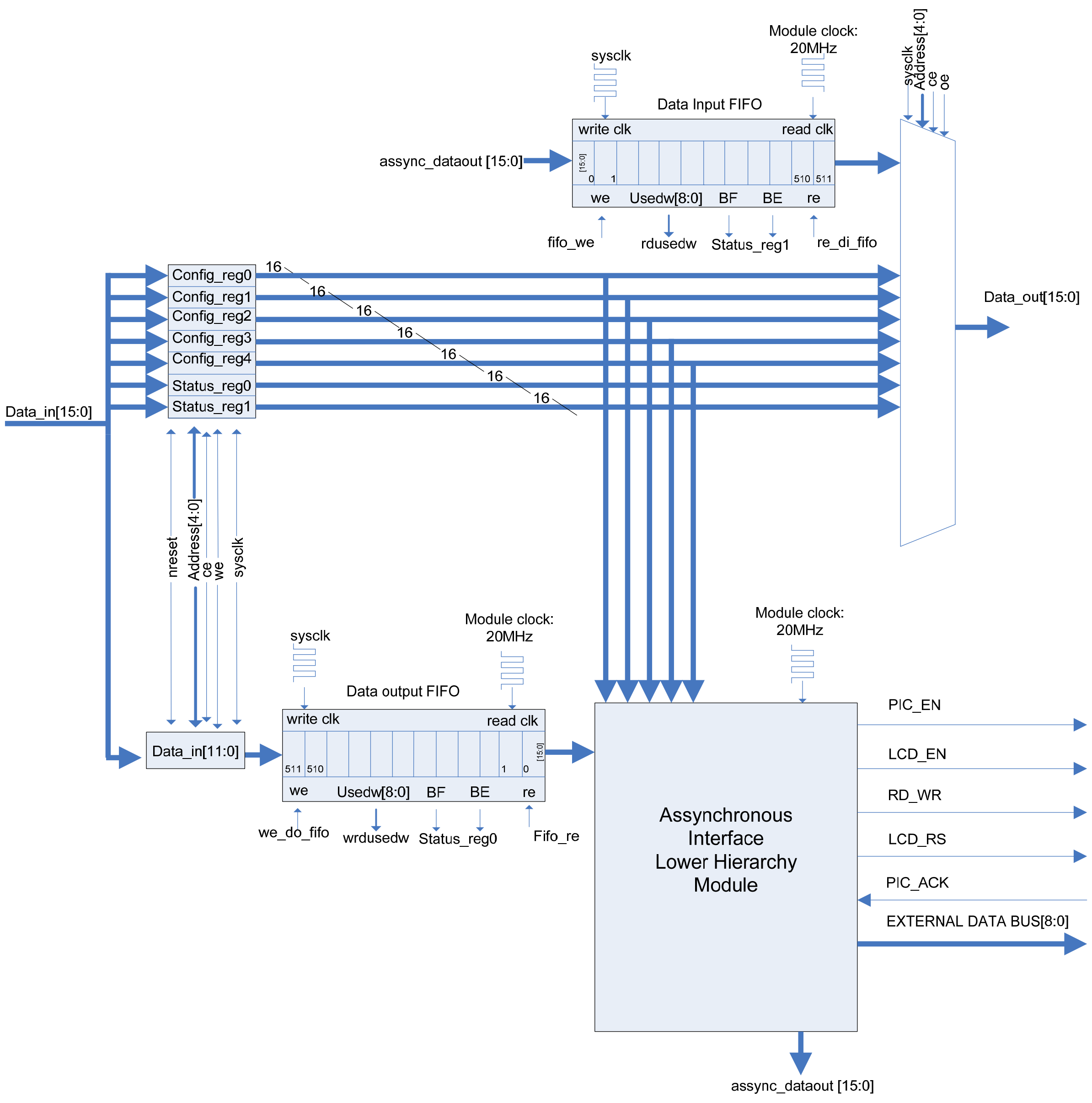


Figure 8-1: A block diagram of the asynchronous interface top level entity.

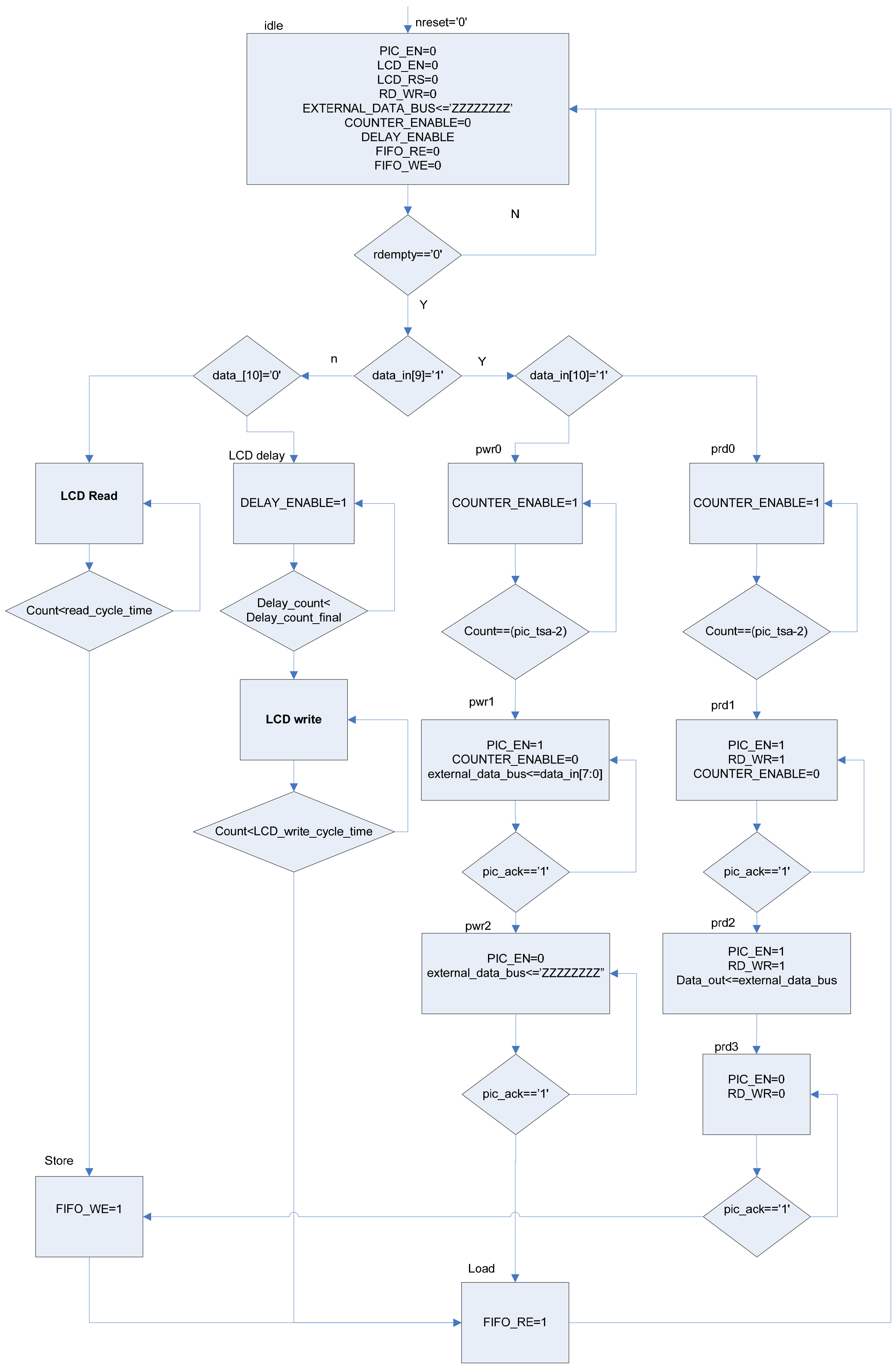


Figure 8-2: Asynchronous interface state machine.

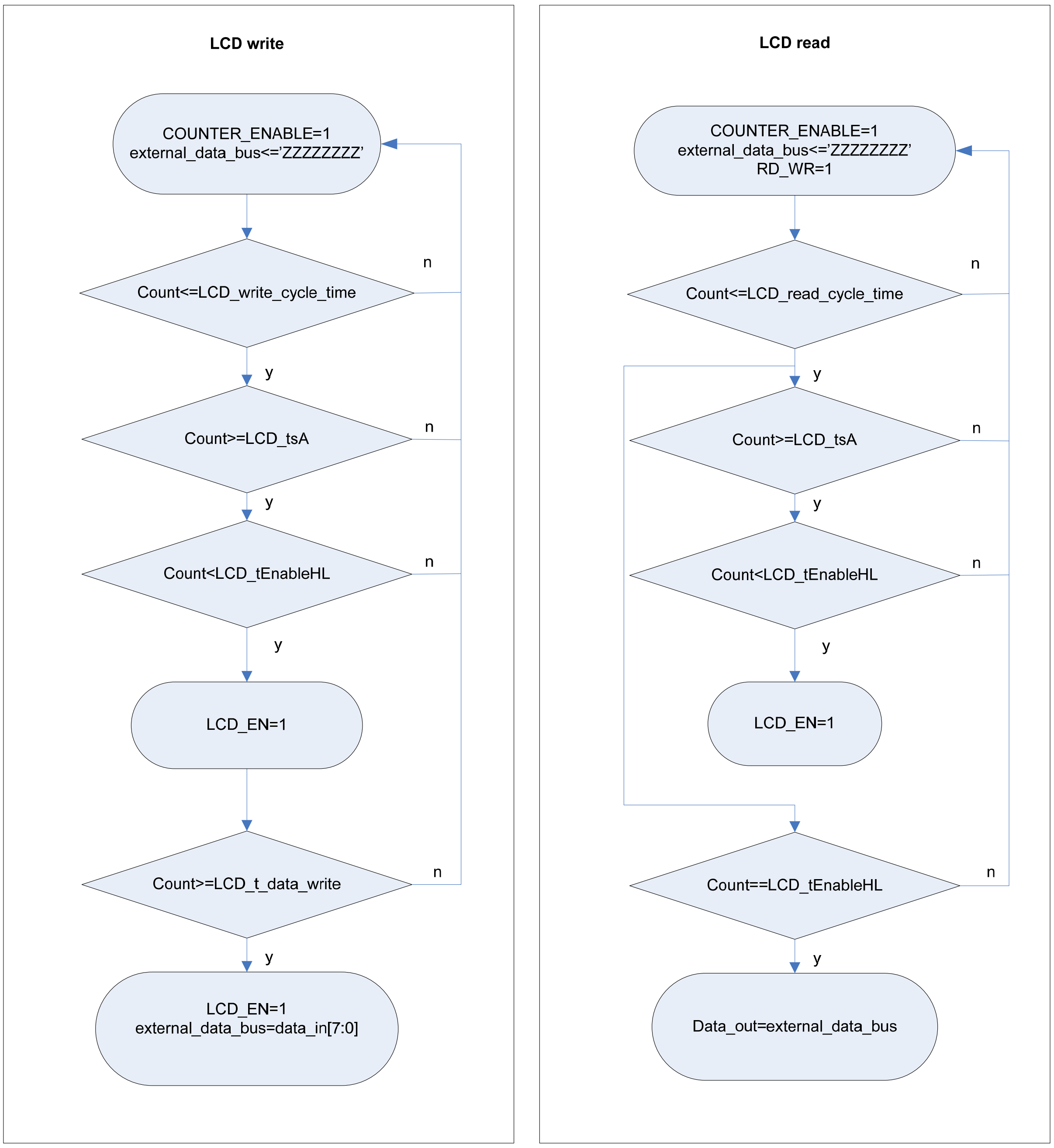


Figure 8-3: Asynchronous interface LCD write and LCD read state.

Address	Write /Read	Register	Description
38000040	W/R	Control_reg0	Control register 0
38000042	W/R	Control_reg1	Control register 1
38000044	W/R	Control_reg2	Control register 2
38000046	W/R	Control_reg3	Control register 3
38000048	W/R	Control_reg4	Control register 4
3800004A	W R	Data-command Output FIFO Data Input FIFO	Data and command Output FIFO Data Input FIFO
3800004C	R	Status_reg0	Status register 0
3800004E	R	Status_reg1	Status register 1
38000050	R	Status_reg2	Status register 2

Table 8-1: Asynchronous interface memory addressable registers.

Table 8-1 illustrates the memory addressable registers within the asynchronous interface top level VHDL module. The first column indicates the address of the register, the second column indicates whether register is writable (W), readable (R) or both (W/R), the third column is the register's VHDL name and the fourth column is a brief description o the register. If necessary a detail description of the register is given below:

Control register 0:

Bits 11-0: Unused

Control register 1:

Bits 5-0: PIC_tsA : The initial delay, in module clock cycles, that is inserted before communication with PIC81F4550 starts. This is programmed by the TigerSHARC to 3 clock cycles.

Bits 11-6: Reserved

Control register 2:

Bits 5-0: Reserved

Bits 11-6: LCD_read_cycle_time: The LCD read cycle time in module clock cycles. This programmed by the TigerSHARC to 50 clock cycles.

Control register 3:

Bits 5-0: LCD_write_cycle_time: The LCD write cycle time in module clock cycles. This programmed by the TigerSHARC to 50 clock cycles.

Bits 11-6: reserved

Control register 4:

Bits 5-0: LCD_tEnableHL: The time, in module clock cycles, when LCD_CE is de-asserted to '0'. This programmed by the TigerSHARC to 21 clock cycles.

Bits 11-6: LCD_t_data_write: The time, in module clock cycles, when the data bus is driven. This programmed the TigerSHARC to 14 clock cycles.

Bits 15-13: reserved

Data and command output FIFO (OFIFO):

Bits 7-0:	8-bit data.
Bit 8:	reserved
Bit9:	'1'= Microchip interface.
	'0'= LCD interface.
Bit 10:	'1' =read command.
	'0'= write command.

Bit 11: reserved

Status register 0:

Bit 0: Data output FIFO buffer empty flag.	'1'=empty
Bit 1: Data output FIFO buffer full flag.	'1'=full

Bits 11-3: Data output FIFO number of words used (0 to 255).

Status register 1:

Bit 0: Data input FIFO buffer empty flag. ‘1’=empty

Bit 1: Data input FIFO buffer full flag. ‘1’=full

Bits 11-3: Data input FIFO number of words used (0 to 255).

Asynchronous Interface Initialization

Upon reset, all bits of control registers 0, 1 and 2 are set to zero and all the bits of control register 3 and 4 are asserted to one. The FIFOs are cleared and the internal state machine is reset to its idle state. It is necessary to program the correct timing values into the control registers as stated above.

Asynchronous Interface Operation

The asynchronous interface uses input and output FIFOs to synchronize the system side with the state machines which control the communication between the PIC18F4550 and the LCD screen.

The 12 bit data and command FIFO is used to send the 8-bits of data and a 4-bit command which tells the state machine whether it is communicating with the PIC18F4550 or the LCD. The output of this FIFO is passed to the lower hierarchy asynchronous interface module. Within this module it is referred to Data_in. Unless otherwise stated, Data_in refers to the output of the FIFO in the following text.

The operation of the asynchronous interface is best explained with example read and writes to the PIC18F4550 and the LCD and with reference to the ASM flow diagrams in the following paragraphs. It is assumed that the interface has been initialized:

PIC18F4550 write

The TigerSHARC writes to the data and command output FIFO. As is illustrated in Figure 8-2, if bit 9 of Data_in bit is high and bit 10 is low then the state machine transits to state pwr0. Once the initial delay that is programmed into control register 1 is complete, the state machine transits to state pwr1.

The CE signal is asserted by the state machine and the data bus is driven with the data. Once the PIC18F4550 acknowledge signal is asserted, the CE is de asserted and the data bus is tri-stated in state pwr2. Once the acknowledge signal is de asserted the state machine proceeds to load the next command and data from the FIFO and then returns to the idle state.

PIC18F4550 read

To read from the PIC18F4550, the user must first ensure that the data output FIFO is empty to ensure that no other read commands are in the pipeline. This can be done by polling the output FIFO buffer empty flag in status register 0.

The data is then written to the data and command output FIFO. As illustrated in Figure 8-2, if bit 9 of Data_in is a high and bit 10 is a high then the state machine transits to state prd0. Once the initial delay that is programmed into control register 1 is complete, the state machine transits to state prd1.

Now the CE and RD_WR signals are asserted. Once the PIC18F4550 acknowledge signal is asserted, the data is sampled from the data bus in state prd2. The CE and RD_WR signals are then de asserted in state prd3. Once the acknowledge signal is de-asserted, the state machine proceeds to write the data to the data input FIFO. The next command and data is then loaded from the FIFO in the load state and then returns to the idle state.

The user must poll the data input FIFO buffer empty flag in status register 1 to see if the data has been stored. Once the flag is asserted the data can be removed from the FIFO.

LCD write

The TigerSHARC writes to the data and command output FIFO. As is illustrated in Figure 8-2, if bit 9 of Data_in is low 0 and bit 10 is low then the state machine transits to the LCD_delay state.

The LCD operates at 270 kHz. At this frequency command processing within the LCD unit can take up to 43us. Therefore a 50us delay is inserted using a counter in the LCD delay state. By inserting this delay it is no longer necessary to poll the LCD to ensure that the command is complete.

Once the delay is complete, the state machine transits to the LCD_write state. This write state uses a counter to generate the timing of the asynchronous signals required to interface to the LCD screen as is illustrated in Figure 8-3.

Once the counter reaches the LCD write cycle time, the state machine proceeds to the load state to load the new command. Finally, the state machine returns into the idle state.

LCD read

The user must first ensure that the data output FIFO is empty to ensure that no other read commands are in the pipeline.

The TigerSHARC writes to the data and command output FIFO. As is illustrated in Figure 8-2, if bit 9 of Data_in is high and bit 10 is low then the state machine transits to the read state.

The LCD read state uses a counter to generate the timing of the asynchronous signals required to interface to the LCD screen, as is illustrated in Figure 8-3.

Once the counter reaches the LCD write cycle time, the state machine proceeds to the store state to store the new sampled data and then to the load state to load the new command. Finally, the state machine returns to the idle state.

8.2 Microchip PIC18F4550

The Microchip PIC18F4550 is used in this design to interface with a personal computer via USB. The PIC18F4550 is USB V2.0 compliant and is capable of operating at low speed (1.5Mb/s) as well as full speed (12Mb/s). The PIC18F4550 supports control, interrupt, isochronous and bulk transfers.

The following sections describe the hardware necessary to interface the TigerSHARC to the PIC18F4550 via the FPGA asynchronous.

The PIC8F4550 (U19) is illustrated in schematic 6 in Appendix A. The PIC18F4550 operates from a 5V supply. It is therefore necessary to perform logic level translation since the FPGA IO operates from 3.3V. This is performed using the Maxim MAX3002EUP bidirectional level translator (discussed in section 11.1).

The FPGA's external asynchronous interface bus is connected via the MAX3002EUP (U50) to the PIC18F4550's 8-bit port (PORTD). The asynchronous interface control signals are connected via a second MAX3002EUP (U51) to PORTB of the PIC18F4550.

Microchip PIC18F4550 USB Interface

The USB differential connections of the PIC18F4550 (pins 43 and 42) are connected to the type B USB connector (U5). The differential impedance of the USB bus is 90Ω. Therefore the differential connections are routed with a 0.129mm trace separation and 0.129mm trace width to result in a differential impedance of 90 Ω (this was designed by simulation using Mentor Graphics Hyperlynx software package).

The USB 3.3V bus power is connected to pin 1 of the USB connector. The PIC18F4550 RB0 pin is connected through R40 to the USB 3.3V power. This pin is used to detect whether the PIC18F4550 is connected to the bus. If it is connected, pin RB0 will be pulled up to the USB 3.3V, and if it is not, pin RB0 will be pulled down to ground via R41.

Microchip PIC18F4550 Programming

The PIC8F4550 is programmed used the Microchip MPLAB ICD 2 programmer. The RJ45 connector (U44) is used to connect the PIC18F4550 to the programmer and is illustrated in schematic 6 in Appendix A.

Microchip PIC18F4550 LEDs and switches

The PIC8F4550 memory clear (PIC_MCLR) signal is connected in parallel through a 100Ω resistor (R43) to a switch (SW1), which is pull-up with R42 to 5V. Depressing the switch will reset the PIC18F4550. This is useful for debugging purposes.

Switch (SW4) is connected to the PIC18F4550 pin RB0. This switch is useful for debugging purposes and is illustrated in schematic 15 in Appendix A.

The PIC8F4550 IO pins RA0 and RA1 are connected to LEDs through series 470Ω resistors (R72 and R71). These LEDs are used by the USB software to illustrate the status of the USB connection. The LEDS are illustrated in schematic 15 in Appendix A. The LEDs , when on, will be biased at 9.1mA. This calculated using the following equation: $I_{LED} = \frac{V_{CC\,5V} - V_{LED(on)}}{R} = \frac{5-0.7}{470}$=9.1mA.

Microchip PIC18F4550 frequency control

The PIC18F4550 operates from a Murata Ceraloc CSTCC 8MHz ceramic resonator (U18). The resonator does not require external capacitors in parallel with input and output to ground as it has internal built in capacitors.

The internal PLL of the PIC18F4550 is configured using software, to generate a 48MHz clock from the external 8MHz input.

8.3 LCD Screen

The PowerTip PC2004LRU-AWA-B LCD (12) is used in this design. The LCD screen has a 20 character by 4 line display. The LCD operates at 270 kHz and from 5V. It therefore requires logic level translation to interface to the FPGA asynchronous interface as the FPGA IO operates at 3.3V. The LCD is connected to PCB via header U20 (illustrated in schematic 6 in Appendix A) which leads to the 5V side of the asynchronous interface's external 8-bit data bus, which is in parallel with the PIC18F4550. The required LCD control signals (LCD_EN, RD_WR and LCD_RS) that are generated by the asynchronous interface are connected via the MAX3002EUP (U51) to the LCD header (U20). These connections are illustrated in schematic 12 in Appendix A.

The LCD contrast is set using the voltage divider formed by R16 and R17. The LCD backlight is powered through R119, a 3.3Ω current limiting resistor that is connected to 5V through jumper (U71). The backlight requires 4.2V and typically operates at 260mA, depending on the brightness of the LEDS. The 3.3Ω resistor will limit the LCD backlight current to 242mA which is calculated using the following equation:

$$\mathrm{I} = \frac{\mathrm{V_{CC}} - \mathrm{V_{LED}}}{\mathrm{R119}} = \frac{5-4.2}{3.3} = 242\mathrm{mA}$$

Removing the jumper will disable the backlight.

8.4 Summary and Conclusion

This chapter described the design of the system that incorporates a USB peripheral device that can be used to communicate with a PC at 12Mbps as well as an interface to an LCD display. The FIFO buffers, which efficiently implement the asynchronous interface module in the FPGA, allows the TigerSHARC to continuously send data to the USB peripheral and to the LCD screen without having to wait for these slower devices to process the data.

9 UART Interface System

A UART has been included to assist in system debugging by allowing the system to communicate with a PC terminal to display debug information. The following section discusses the design of this interface.

9.1 FPGA UART module

The UART top level entity is illustrated in Figure 7-1. The figure illustrates the memory addressable registers and the output data FIFO that can be accessed by the TigerSHARC. A detailed explanation for each register is given in Table 9-1. Only the transmission side of the UART has been implemented. The receiver signal is connected to the FPGA but is not implemented. The operation of the UART interface is explained after Table 9-1.

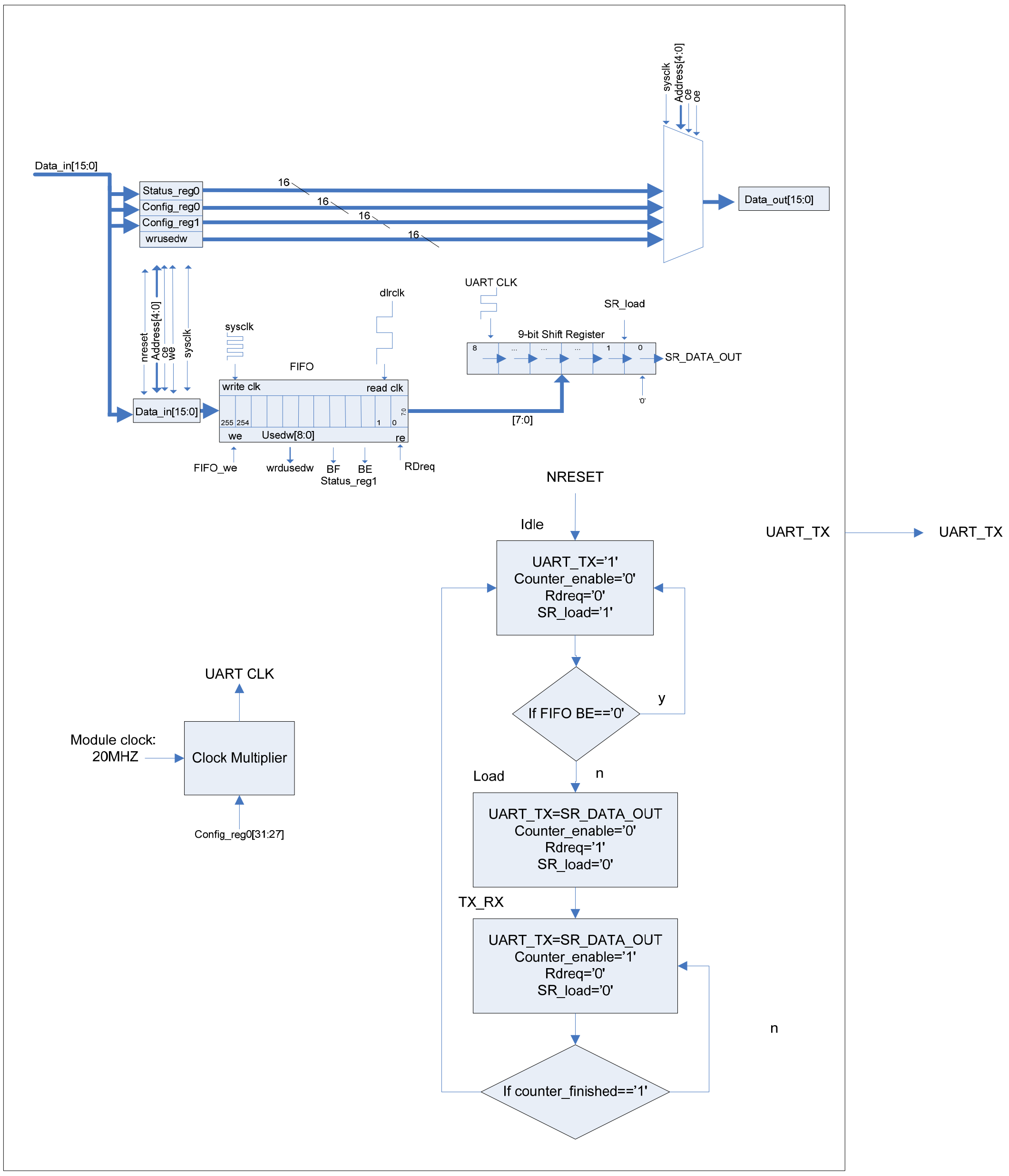


Figure 9-1: The FPGA UART module.

Address	Write /Read	Register	Description
380000C0	W	Data register1	8-bit TX data output FIFO
380000D0	W/R	Config_reg0	Configuration register 0
380000D2	W/R	Config_reg1	Configuration register 1
380000D4	R	USEDW	Number output FIFO words used, Max=256
380000D6	R	Status_reg0	Status Register 0

Table 9-1: UART memory addressable registers

Table 9-1 illustrates the memory addressable registers within the UART interface top level VHDL module. The first column indicates the address of the register, the second column indicates whether register is writable (W), readable (R) or both (W/R), the third column is the register's VHDL name and the fourth column is a brief description of the register. Detailed descriptions of the registers are given below:

Configuration register 0:

Bits 15-0: UART clock period:

$$\textbf{Output frequency} = \frac{\mathbf{20}}{(\mathbf{2}\times\mathbf{Confg\_reg0}+\mathbf{1})}\,\mathbf{MHz} \qquad \text{9-1}$$

Configuration register 1:

Bit 0: If asserted to '1', the internal UART clock is enabled. The clock must be enabled only after the clock period has been programmed.

Bits 1: If asserted to '1', the UART data is inverted. This bit must be set as the MAX232 inverts the data externally.

UART Initialization

Upon reset, all bits of configuration register 0 are set to ones and all bits of configuration register 1 are asserted to zero. The UART internal clock is therefore disabled and the internal state machine is also reset to the idle state.

It is therefore necessary to program the UART clock according to equation 2-3. Our system sets configuration register 0 to 260, which enables the UART to operate at 39000 BAUD. Once the clock has been programmed it may be enabled by asserting bit 0 in configuration register 1. It is necessary to set bit 1 in configuration register 1 to invert the clock as the external RS232 driver inverts the data.

UART Operation

The UART module is designed to transmit 8-bits of data. The user must first poll the FIFO buffer full flag. If the FIFO is full, the user must wait until the FIFO is no longer full before proceeding. The transmission payload can be loaded into data register 1 for transmission.

The internal state machine controls the transmission of the data. The flow diagram shown in Figure 9-1, illustrates that when the state machine is in the idle state, it will continuously monitor the buffer empty flag to see if there is data in the FIFO. If there is data in the FIFO then the state machine will transit to the LOAD state. In the LOAD state the 8- bit data is loaded into bits 8 to 1 of the shift register. A zero is loaded into bit 0, as this is the start bit required for RS232 communication. The state machine then transit to the TX_RX state where the data is transmitted. A counter is used to control the number of bits that are transmitted. Once the counter is finished and the transmission is complete, the state machine returns to the idle state.

9.2 RS232 Line Driver

The Texas Instruments MAX3232E multichannel RS232 line driver is used to translate the 5V UART signal to the +12V and -12V logic levels used in RS232 communication. The MAX3232E (U17) operates at 5V, therefore a MAX3002EUP (U51) is used to convert the FPGA 3.3V signals to these levels. The external circuitry required by the MAX3232E is illustrated in schematic 7 in Appendix A. Capacitors C119, C39, C120 and C118 are used by the internal charge pump to generate the +12V and -12V logic levels. The second channel of the MAX3232E is connected to PIC18F4550's UART. However, the PIC18F4550 has an anomaly and is not able to invert the UART signal which is required communicate with a PC.

9.3 Summary and Conclusion

This chapter described the design of the system that incorporates a UART interface which can be used to communicate with a PC terminal. An efficient implementation using FIFO buffers in the FPGA UART interface module allows the TigerSHARC to continuously send data to the PC terminal without having to wait for this slow communication interface.

10 FPGA Test Registers, LEDs and Push-Buttons system

This chapter describes the system that is used as interface with the FPGA tests register, and the LEDs and push-buttons which are connected to the FPGA.

10.1 FPGA Test Registers and Buttons Interface

The test registers and buttons interface top level entity is illustrated in Figure 7-1. The figure illustrates the memory addressable registers, the switches, LEDs and the IRQ signal that are connected to the module. The memory addressable registers are described in Table 10-1.

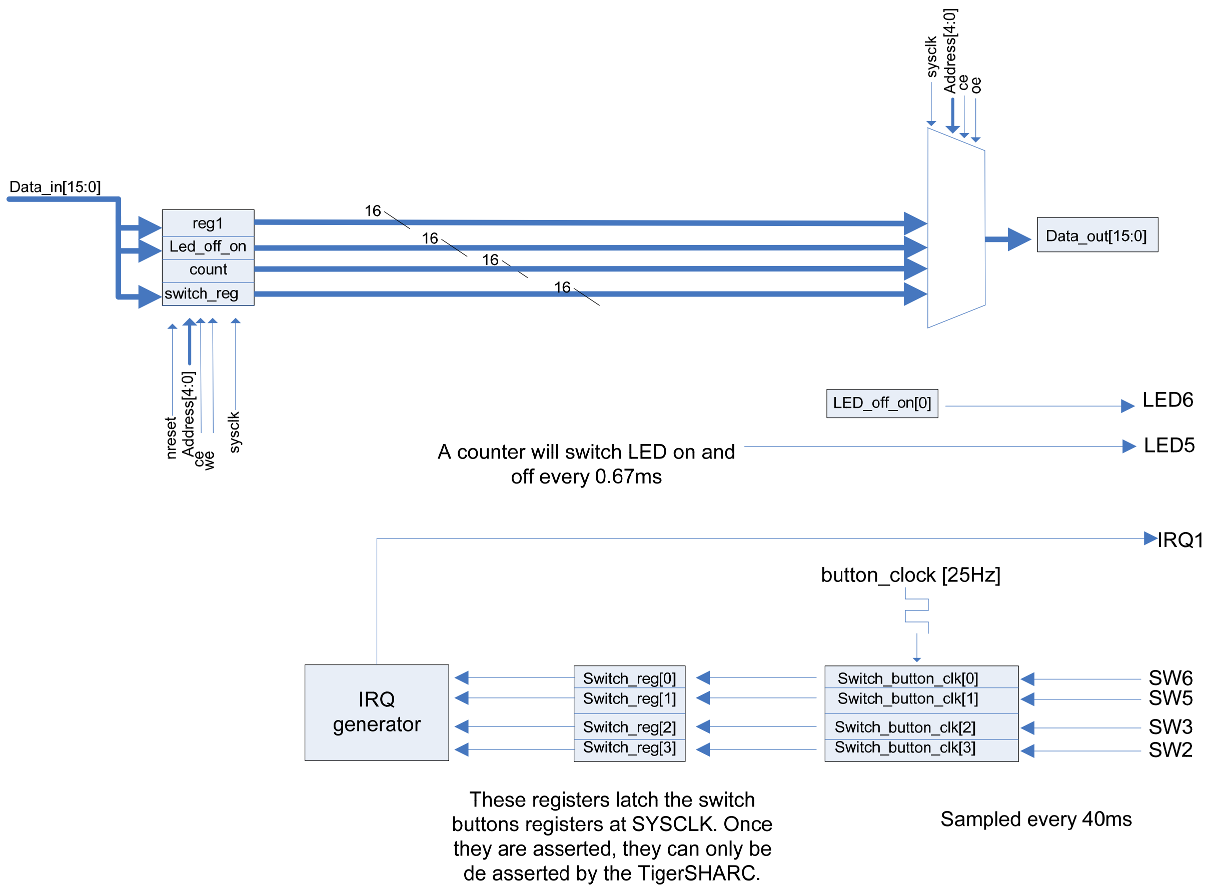


Figure 10-1: The test registers and buttons Interface top level entity.

Address	Write /Read	Register's VHDL Name	Description
38000040	W/R	Data reg1	Data register 1. This register is used for debugging purposes.
38000042	W/R	LED_off_on	LED6 on/off register
38000044	R	Count register	Count register
38000046	w/R	Switch_reg	Push buttons register

Table 10-1: Test registers and push-button interface memory addressable registers.

Table 10-1 illustrates the memory addressable registers within the test registers and push buttons interface top level VHDL module. The first column indicates the address of the register, the second column indicates whether register is writable (W), readable (R) or both (W/R), the third column is the register's VHDL name and the fourth column is a brief description of the register. If necessary a detailed description of the registers is given below:

LED register:

Bits 0: If asserted to '1' then LED6 = on. If asserted to '0' then LED6 = off.

Bits 15-1: Unused.

Count register:

Bits 7-0:	Value of Count2. Count2 is initialized at reset to 0. After a read from Count2, it is incremented by 1. The register is used for debugging purposes.
Bits 15-8:	Unused.

Push-button register:

Bit 0:	If bit 0 ='0' then switch6 has been pressed.
Bit 1:	If bit 0 ='0' then switch5 has been pressed.
Bit 2:	If bit 0 ='0' then switch3 has been pressed.
Bit 3:	If bit 0 ='0' then switch2 has been pressed.

Bits 3-0 need to be set to '1' to acknowledge that the button has been pressed. Only then will the next button press be registered.

Push-Button Operation

The FPGA external switch connections are pulled-up to VCC 3.3V using 10kΩ resistors. This is illustrated in schematic 15 in Appendix A. If the external switches (SW2, SW3, SW5 and SW6) are depressed, the corresponding input to the FPGA will be connected to ground. The external switches are sampled by a 25Hz clock, called button_clock, and are stored in the 4-bit register switch_button_clk. The 25Hz clock is generated by an internal counter. The switches are sampled at this low frequency to ensure that the switch is properly depressed when the value is sampled. If any of the switch_button_clk registers contain a logical zero (i.e. the switch has been pressed) then it is latched into the corresponding bit of the switch_reg register, which operates at SYSCLK. The switch register's value will not change from a logical zero when the switch is released. To reset logic one must be written into the corresponding bit of the switch_reg register.

The output of the switch_reg is used to generate an interrupt. If any of the switch_reg bits are zero, then the IRQ1 signal is pulled low. The signal will remain low until all of the switch_reg bits are set to logical one. The TigerSHARC IRQ1 interrupt is set to a level sensitive interrupt and it is therefore necessary to set all the switch_reg bits to logical one in order for a new interrupt to be triggered.

LED 5

The FPGA output to LED5 is connected to an internal counter that operates from 100MHz system clock. The counter counts to $2^{25} - 1$. When it reaches this value, it inverts the signal that is connected to LED5 and resets the counter value to zero and continues on counting. LED5 will therefore flash on and off continuously at 0.745Hz. LED5 is used to give an indication whether the FPGA system clock is operational.

LED 7

LED7 (illustrated in schematic 15 in Appendix A) is connected to the FPGA pin 67. The LED is not used in our design and could be used for future work.

Power LED

The red LED8 is connected to VCC 3.3V and is used to indicate that the system is powered.

LED Headers

All the LEDs that are used in this system are connected in parallel to header U62 (illustrated in schematic 15 in Appendix A). It is therefore possible to remove the surface mount LEDs and replace them with panel mounted LEDs that connect to header U62.

Switch Headers

The switches (SW2, SW3, SW5, SW6 and SW9) are connected in parallel to header U47 (illustrated in schematic U47). It is therefore possible to connect panel mounted switches in parallel with these switches.

10.2 Summary and Conclusion

This chapter described the design of the system that can be utilized to turn the LEDS on and off, and the method in which the push-buttons are interfaced to the TigerSHARC. In both cases the LEDs and push-buttons are seen as memory mapped registers by the TigerSHARC.

11 Logic Level Translation

This chapter discusses the component that is utilized to perform logic level translation between the components that operate at different voltage levels.

11.1 Maxim MAX3002EUP

The Maxim MAX3002EUP 8-bit bidirectional logic level translator is used in our design to perform level translation between the FPGA 3.3V IO and the components that operate at 5V. The MAX3002EUP is designed to be bidirectional without the need for a directional pin and is guaranteed to operate up to 20Mbps (13). The fastest communication within this design that uses the MAX3002EUP (U39), is the SPI protocol which operates at 1Mbps. This is therefore within the capabilities of the MAX3002EUP. The MAX3002EUP is also used to perform logic level translation between the FPGA and LCD (U50), and the FPGA and PIC18F4550 (U51). The biasing circuitry for U50 is illustrated in schematic 6 in Appendix A.

11.2 Summary and Conclusion

The use of the MAX3002EUP significantly simplified the design process. It allows communication to take place between components that operate at different voltage levels without the conventional use of the uni-directional buffers that would have required extra logic to control the direction of communication.

12 System Reset

It is necessary to provide a global system reset, to insure that each component is held in reset while the respective power supplies stabilize. This chapter discusses the component that is utilized to generate the system reset.

12.1 Texas Instruments TPS3836 Reset Manager

The Texas Instruments TPS3836 reset manager (U38) is used in our design to generate an active low system reset and is illustrated in schematic 24 in Appendix A. The TPS3836 can be configured to generate a reset pulse of 10ms by connecting CT (pin1) to ground or it can configured to generate a reset pulse of 200ms by connecting CT (pin1) to VCC.

The TPS3836 is configured to generate an active low reset for 200ms upon power up and also upon a manual reset triggered by depressing the external reset switch (SW9). This is sufficient time to allow for power stabilization and power supply sequencing of the system (discussed in section 15.3 on page 113).

12.2 Summary and Conclusion

The Texas Instruments TPS3836 reset manager is successfully implemented in our system to provide a 200ms reset pulse that enables all the power supplies to stabilize and for the components to initialize themselves to their internal reset state.

13 SDRAM System

This chapter discusses the Analog Devices SDRAM controller and the SDRAM memory modules that are utilized in our design.

13.1 SDRAM Controller and Memory Modules

The Analog Devices TigerSHARC TS201 DSP has an onboard SDRAM controller. The controller can support a total of 1024MB of single data rate SDRAM arranged in 4 memory banks of 64M x32bit words each, and which are selected by the $\overline{MSSD}$[3-0] (memory select SDRAM) select lines. The SDRAM controller is synchronous with the system clock and can be configured to interface with standard SDRAM devices.

To create large memory systems, it is necessary place many SDRAM devices in parallel. This can be accomplished by using industry standard dual in line memory modules. However, as more devices are connected in parallel, so the total capacitive load increases. For example the total load on the address bus for 2x256MB single rank unregistered SDRAM modules is approximately 68pF. The TigerSHARC IO drivers cannot supply enough output current to drive these address lines and it is therefore necessary to buffer the address lines. This can be accomplished by using industry standard registered single rank SDRAM modules. Registered SDRAM modules buffer the address lines with registers that are synchronous with the SDRAM clock. The loading generated by 2x256MB registered single rank SDRAM modules is approximately 16pF. This is within the load capabilities of the TigerSHARC and hence this configuration was employed in our design.

To select the appropriate registered SDRAM modules to interface with the TigerSHARC, the following factors need to be taken into consideration:

- Operating Voltage.
- Maximum operating frequency.
- Maximum supported memory size.
- The number of internal banks in which the SDRAM is arranged.

- Column Address Strobe (CAS) latency: the delay, in number of system clock cycles (SCLK), between the time that the SDRAM detects the read command and the time that it provides the data at its output pins.
- The refresh rate of the SDRAM.
- Precharge (PRC) to Row Address Strobe (RAS) delay: the required delay, in number of system clock cycles (SCLK), between the time the SDRAM controller issues a PRE command and the time it issues an ACT command.
- Row Address Strobe (RAS) to Precharge (PRC) delay: the required delay, in number of system clock cycles (SCLK), between the time the SDRAM controller issues an ACT command and the time it issues a PRE command.
- The SDRAM page size.
- Initialization sequence.

Table: 13-1 illustrates the capabilities of the of the TigerSHARC SDRAM controller with regards to the above criteria. It also illustrates the specifications of the of the Kingston memory chosen for this design.

SDRAM Features	**SDRAM Controller**	**Kingston 256MB Module**
Voltage	2.5V or 3.3V	3.3V
Max Frequency	125MHz	133
Max Memory Size	256MB	256MB
Number of internal banks	2 or 4 banks	4 banks
CAS Latency	1 to 3 cycles	3 cycles
Refresh Rate	32 and 64 ms	64 ms
PRC to RAS delay	2 to 5 cycles	3 cycles
RAS to PRC delay	2 to 8 cycles	6 cycles
Page size	256,512 and1024 Words	1024 Words
Init sequence	MRS-> REF or REF-> MRS	MRS-> REF or REF-> MRS

Table: 13-1 SDRAM controller and SDRAM module capabilities.

The following TigerSHARC IO Pins are required to interface with SDRAM:

Output pins:

$\overline{\text{RAS}}$: Row addresses strobe. Indicates that a row address is valid during a read or write of SDRAM.

$\overline{\text{CAS}}$:	Column addresses strobe. Indicates that a column address is valid during a read or write of SDRAM.
LDQM:	Low word SDRAM data mask. LDQM tri-states the SDRAM data mask buffers of the lower 32-bits of the 64-bit data bus when performing 32-bit writes to odd addresses.
HDQM:	High word SDRAM data mask. HDQM tri-states the SDRAM data mask buffers of the higher 32-bits of the 64-bit data bus when performing 32-bit writes to odd addresses.
SDCKE:	SDRAM clock enable.
$\overline{\text{SDWE}}$:	SDRAM write enable.
Address bus [10:1]:	Connected to SDRAM address [9:0]
SDA10:	Connected to SDRAM address bit 10 pin.
Address bus [14:12]:	Connected to SDRAM address [12:11].
Address bus [15:14]:	Connected to SDRAM Bank Address BA[1:0].
$\overline{\text{MSSD0}}$:	Memory select SDRAM 0. Connected to SDRAM module 0 chip select 0 and 2 ($\overline{s0}$ and $\overline{s2}$).
$\overline{\text{MSSD1}}$:	Memory select SDRAM 1. Connected to SDRAM module 1 chip select 0 and 2 ($\overline{s0}$ and $\overline{s2}$).
$\overline{\text{MSSD2}}$:	Memory select SDRAM 2. Connected to SDRAM module 0 chip select 1 and 3 ($\overline{s1}$ and $\overline{s3}$).
$\overline{\text{MSSD3}}$:	Memory select SDRAM 3. Connected to SDRAM module 1 chip select 1 and 3 ($\overline{s1}$ and $\overline{s3}$).

Bidirectional pins:

Data bus [63:0]: 64-bit data bus.

The $\overline{\text{MSSD2}}$ and $\overline{\text{MSSD3}}$ memory chip select signals are connected to the $\overline{\text{s1}}$ and $\overline{\text{s3}}$ pins of the DIMM connector so that the system could in future interface with 2x512MB dual rank modules. A dual rank module essentially consists of 2 single rank modules arranged onto 1 DIMM. However, the system has not been tested in this configuration and the support thereof cannot be guaranteed.

13.2 SDRAM Controller Configuration Register

The lower 16-bits of the TigerSHARC SDRCON 32-bit register are used to configure the SDRAM controller. The register is setup with values supplied in the Kingston (14) and Nanya(15) datasheets (Nanya are the manufactures of the memory components used on the Kingston memory module):

Bit 0:	‘1’ SDRAM controller enabled
Bits2-1:	$\overline{\mathrm{CAS}}$ latency: “10” = 3 cycle latency.
Bit 3:	‘1’ Pipeline enabled. This is necessary since the memory modules are registered and therefore operate with an extra latency cycle.
Bits 5-4:	“10”: 1024 word page size.
Bits8-7:	“10”: Refresh once every 2200 cycles. The TigerSHARC will issue a refresh command to the SDRAM controller every 2200 SOC cycles. This is necessary to ensure a maximum refresh time of 64ms. The SDRAM needs to be refreshed every $\frac{SOC\ clock \times t_{ref}}{rows}$ = $\frac{300MHz \times 64ms}{8192\ rows}$ =2343 cycles. Therefore refreshing every 2200 SOC cycles will meet this requirement.
Bits10-9:	“01”: Precharge to RAS delay: 3 cycles.
Bits 13-11:	“100”: RAS to precharge delay: 6 cycles.
Bit 14:	‘1’: This sets the initialization sequence of the SDRAM controller to sequentially issues: a PRE command, eight auto refresh cycles, and an MRS (Mode Register Set) command upon initialization.
Bit 15:	‘0’: Only used for 2.5V mobile RAM.

13.3 Summary and Conclusion

This chapter described the utilization of TigerSHARC SDRAM controller to interface with two Kingston 256MB Registered SDRAM modules, providing a total of 512MB system memory, which is expandable to 1024MB. The registered memory modules are utilized to bring the address bus loading within acceptable values.

14 Clock Distribution and Signal Integrity

This chapter describes the clock distribution of our design and the techniques that are utilized to guarantee signal integrity of these high speed signals.

14.1 Clock Distribution

The Analog Devices TigerSHARC TS201 development board operates with a system clock of 100MHz. The peripheral board was designed operate synchronously from the same 100MHz system clock. To ensure portability from the peripheral board to the final prototype design, it was decided to keep the system clock at 100MHz.

The TigerSHARC TS201 requires an external clock source with a maximum jitter of 500ps. The FPGA PLL requires an external clock source with a maximum jitter of 500ps.The SDRAM external clock source jitter is not specified.

The SDRAM configuration is designed to possibly support two dual rank modules (i.e 2x256MB banks per DIMM). Two clock sources are required by a dual rank module as opposed to one clock source for a single rank module. This system therefore requires four clock sources for the SDRAM modules. The FPGA requires two clock sources (one for the system side and one for the PLL to generate the 20MHz and 200MHz internal FPGA clocks). The TigerSHARC requires 1 clock source. The total number of clocks required is therefore seven.

When designing a synchronous system it is necessary to route the various clock traces with the same length so that each component sees the rising edge of the clock at the same time. The TigerSHARC datasheet (3) requires that the clock traces of the system be routed with a trace length tolerance of +-15.86mm (+-125mils).

However, the SDRAM modules add extra delay to the SDRAM clock. This is due to the extra trace length between the SDRAM DIMM connector and the components on the module. The SDRAM modules also have a first order resistor-capacitor clock damping filter at the end of the trace. This filter adds a further delay of 400 ps to the clock.

It is therefore necessary to compensate for the extra delay on the SDRAM clock nets in order to ensure that the setup and hold requirements of all the components are met. This can be performed in two ways:

1. Extending the length of the FPGA and TigerSHARC clock trace lengths to delay their clocks by a corresponding amount.
2. Using a clock skew manager to induce this clock delay.

A board simulation of the SDRAM modules was performed using Mentor Graphics' Hyperlynx Software to determine the delay that is added by the SDRAM module. It was found that a delay of approximately 600ps is induced by the SDRAM module. However, to increase the hold margin of the system, the decision was made to induce a delay of 1000ps. This would require 25cm to be added to the length of the FPGA and TigerSHARC clock traces. To achieve this, the trace would have to be routed from one side of the PCB to the other and back. Since the design is constrained for space, it was decided to use a clock skew manager to induce the clock delay.

The Cypress RoboClock CY7B9550 (U18) was chosen for this function. This component uses an external clock source to generate 4 output clock pairs of independently configurable clock skew. When operating at 100MHz, the clock manager can induce a clock skew of between zero and 7.5ns in steps of 625ps. The output clock jitter is 50ps, which well within the requirements of the receiver components. The RoboClock uses three-level logic pins for configuration. The resistors used to configure each pin are illustrated in schematic 25 in Appendix A. The clock skew manager is configured to the follow settings:

- All SDRAM clocks are set to zero skew.
- The FPGA, FPGA PLL and DSP clock are configured to lag the SDRAM clock by 1.25ns.

The following method was used to determine the required configuration of the clock manager. The FPGA and DSP clock traces were routed to the same length with a +-15.86mm tolerance. The four SDRAM clocks were routed to the same length, also with a tolerance of +-15.86mm. Hyperlynx was then used to determine the skew between the clock on the pin of the TigerSHARC and the SDRAM modules. This delay was determined to be 600ps as illustrated in Figure 14-1. In order to increase the hold margin of the SDRAM component's, a clock delay of 1.25ns was introduced for the DSP and FPGA clocks. The TigerSHARC and FPGA will therefore lag the SDRAM clock by 0.65ns.

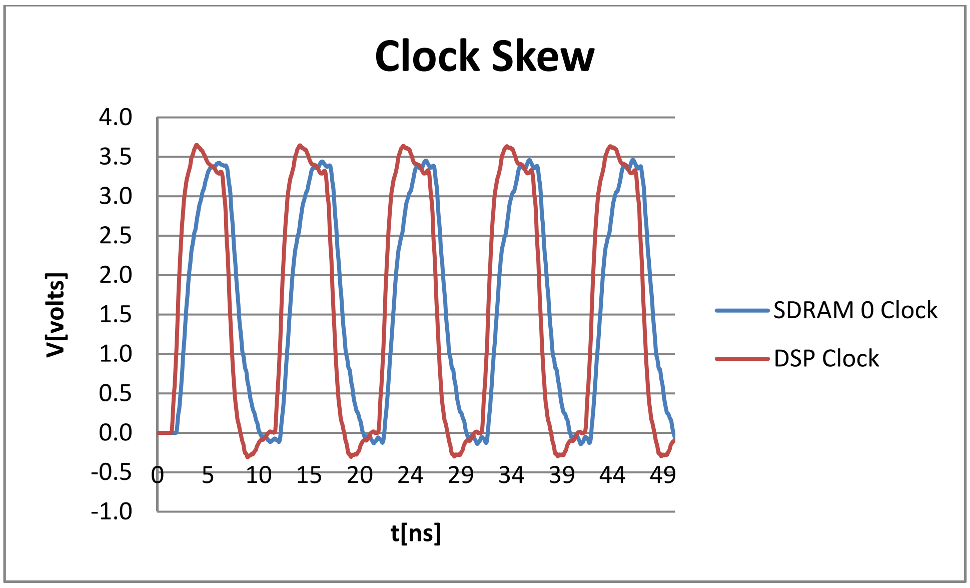


Figure 14-1: Clock skew (600ps) of DSP clock (leading) and SDRAM clocks (lagging).

The TigerSHARC will hold the data bus for a minimum of 1ns after the following clock cycle's rising edge (3). By inducing the clock skew, the hold margin is increased to 1.65ns. The hold margin is therefore improved as the SDRAM requires a minimum hold time of 0.8ns (15). The system timing budget is illustrated in Table 14-1. The first column indicates components between which the communication is taking place. The second column shows the time that the data takes to become valid at the output pin of the driving component after the rising edge of the clock. The third column is the maximum propagation delay of all the traces. The fourth column lists the clock skew between the external clocks input of components. The fifth column lists the minimum setup time that is required by each component. The sixth column lists the resulting minimum period and the sixth column is the maximum frequency at which communication can operate, which is calculated using the following equation: $F_{max} = \frac{1}{t_{output\ valid}+t_{trace}+t_{clock\ skew}+t_{setup\ time}}$

Communication Direction	$t_{output\ valid}$ [ns]	t_{trace} [ns]	$t_{clock\ skew}$ [ns]	$t_{setup\ time}$ [ns]	Period [ns]	F_{max} [MHz]
TigerSHARC to FPGA	4	1.94	0	1.326	7.552	125
FPGA to TigerSHARC	4.85	1.94	0	1.5	8.29	120
TigerSHARC to SDRAM	4	1.94	0.65	1.5	8.89	112
SDRAM to TigerSHARC	5.4	1.94	0.65	1.5	9.49	105

Table 14-1: System timing budget

Table 14-1 illustrates that the maximum frequency that this system can reliably operate at is 105MHz. The system clock of 100 MHz is therefore within this limit.

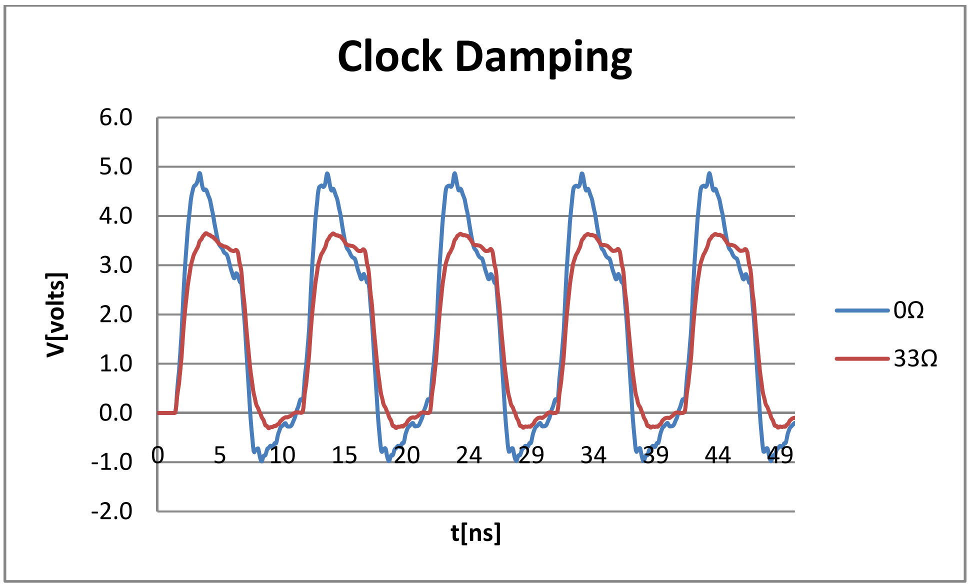


Figure 14-2: Clock damping with and without a series source resistor.

Finally, it is necessary to insert a series source resistor as close as possible to the clock driver to dampen the overshoot of the clock and to condition the waveform of the clock signal seen by the receiver component. Figure 14-2 illustrates the DSP clock of 100MHz simulated with and without a 33Ω series source resistor. Simulations using Hyperlynx were employed to determine the correct value of each of the resistors required for every clock source. The values used for each resistor are illustrated in schematic 25 in Appendix A.

14.2 Signal Integrity

This section describes the method used to design the system to ensure signal integrity when operating the external data bus at 100MHz.

The response of any system of conductors to an incoming signal depends greatly on whether the system is smaller than the effective length of the fastest electrical feature of the signal, or vice versa (16). The length of a rising edge in a signal is given by:

$$l = \frac{T_r}{D} \qquad 14\text{-}1$$

Where T_r is the rise time in ps, D is the delay in ps/cm (typically 70 ps/cm for FR4 laminate (16)) and l is the length of the rising edge in cm. The rise time of the TigerSHARC IO driver, with a drive strength of 70%, is 450ps. This value was determined from the TigerSHARC's IBIS simulation file. Therefore, using these parameters, the length of the rising edge is calculated to be

$$l = \frac{T_r}{D} = \frac{450}{70} = 6.42\ cm$$

A system can be considered to be a lumped system if the length of the conductor is less than one- sixth the length of the rising edge (16). Therefore this system can be treated as a lumped system if the length of the conductors connecting the components can be kept less than $\frac{l}{6} = \frac{6.42}{6} = 1.07\ cm$. However, since this is practically not possible, it is necessary to treat this system as a distributed system.

It is therefore necessary to treat the conductors between the components as transmission lines. Two types of transmission lines used in PCB design are Microstrip and asymmetric Stripline. Microstrip transmission lines are used on the outer layers of the PCB and asymmetric Stripline transmission lines are used in the internal layers.

The PCB layer stack-up used for this design (illustrated in Figure 14-3) is a six layer stack-up. The top and bottom layers use Microstrip and the internal routing layers use Stripline. The system's design complexity requires the use of a six layer PCB. The layer stack-up is a standard stack-up used by the manufacturers of the PCB (TRAX). TRAX can produce PCBs with a minimum conductor width of 0.127mm (5 mils). Due to the system's design complexity, it was indeed necessary to use a conductor trace width of 0.127mm. The characteristic impedance of the Microstrip transmission line for the outer layers with a conductor width of 0.127mm is solved with Hyperlynx to be 60.9Ω.

The characteristic impedance of the asymmetric Stripline transmission line for the inner layers with a conductor width of 0.127mm is solved with Hyperlynx to be 58.7Ω.

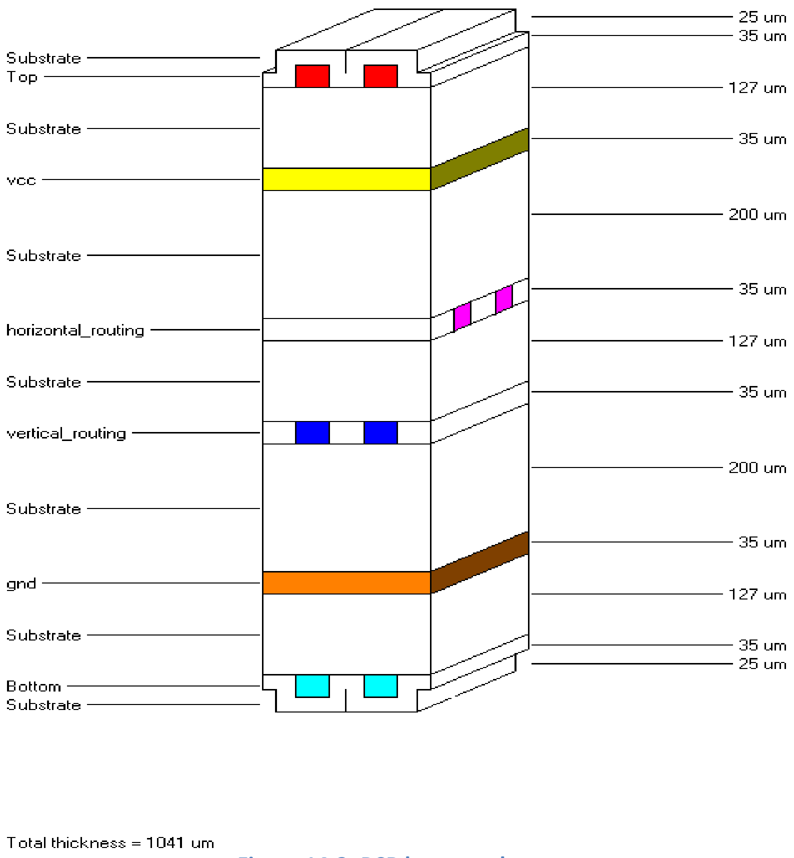


Figure 14-3: PCB layer stack-up.

The resulting impedance mismatch results in a reflection coefficient given by the following equation (16):

$$reflection\ coefficient = \frac{Z_S - Z_M}{Z_S + Z_M}$$

where Z_S is the impedance of the Stripline and Z_M is the impedance of the Microstrip. The impedance mismatch will result in 1.8% of the signal to be reflected.

The JEDEC JESD21-C (17) specification for the manufacturing of PC133 SDRAM modules requires that the impedance control be kept between 50Ω and 65Ω. The JEDEC specification therefore allows for a maximum of 11% of the signal to be reflected. This system's impedance mismatch is therefore well within the JEDEC specification.

It is necessary to terminate a distributed system using either source or end termination (16). However LVTTL, drivers typically cannot supply enough current for an end termination (16) and this is the case with the TigerSHARC IO drivers (this was verified through simulation using Mentor Graphics Hyperlynx). It is therefore necessary to use series source termination resistors.

It is necessary to place the series resistors as close as possible to the IO driver so that the resistor and IO driver combination can be considered as a lumped circuit. The analysis performed in the following paragraph uses data bus bit 7 as an example.

The output impedance of the TigerSHARC IO driver (at 70% drive strength) is calculated using Hyperlynx to be 8.2Ω. To match the driver impedance to the effective trace impedance (which is calculated by Hyperlynx to be 59.6Ω), a series resistor of 51.4Ω is therefore required. However, the driver is too heavily loaded by the attached components to drive the data bus when the 51.4Ω resistor is included.

A solution was found by decreasing the value of the series resistor to 10Ω. A 10Ω resistor does not slow the rise time down as much as the 51.4Ω resistor, but the value of the resistor is still large enough to reduce the overshoot of the signal and prevent damage of the component. Figure 14-4 illustrates the simulated signal waveform for bit 7 of the data bus signal that is driven by the TigerSHARC and received by the FPGA with a 0 Ω, 10 Ω and 51.4Ω series source resistor. The dotted lines are the decision levels at 0.7V and 1.8V.

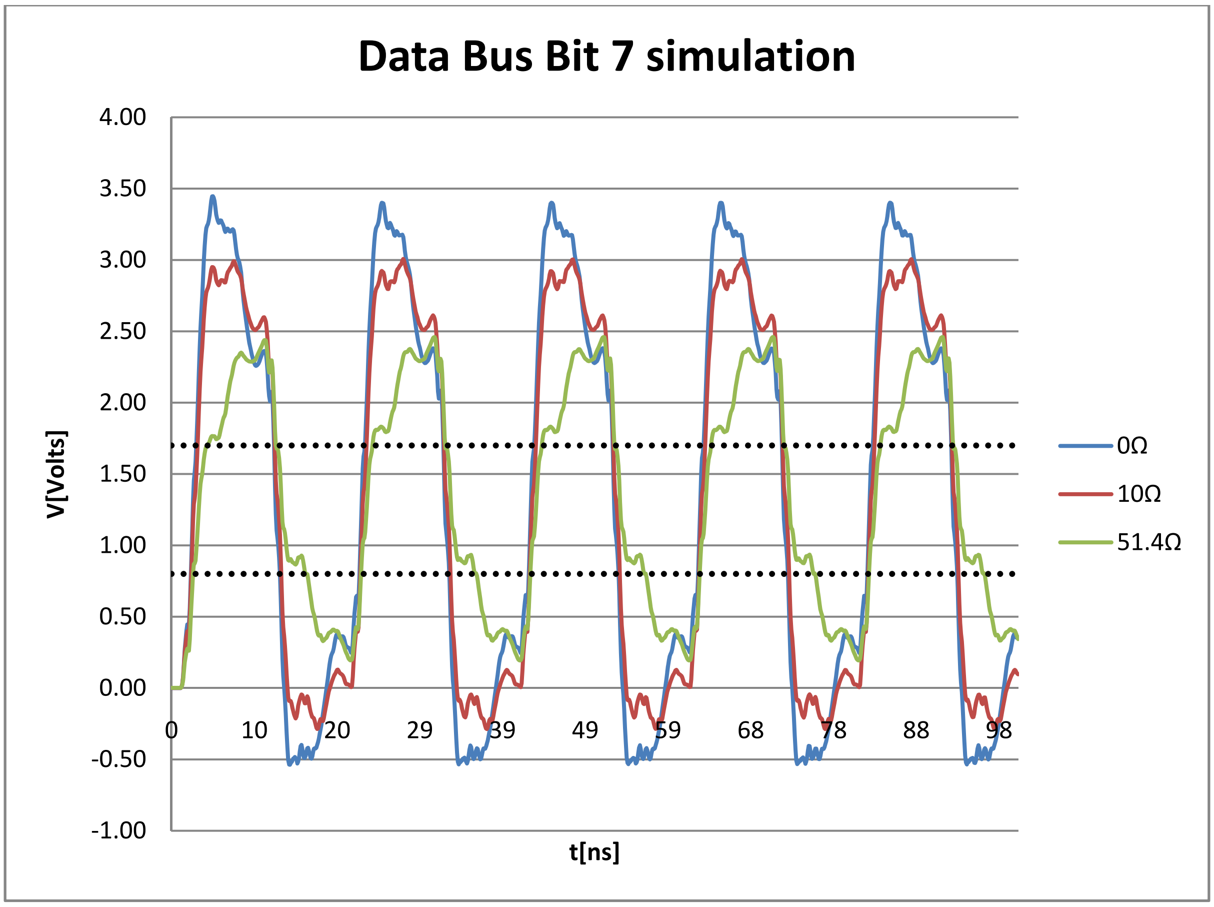


Figure 14-4: Simulation of bit 7 of the data bus for three choices of series resistors.

The various high speed signals of the TigerSHARC experience different loading, yet the driver current drive strength of each signal is the same. It is therefore necessary to uses different resistor values to condition each of these signals to ensure signal integrity. The method used to determine the ideal resistor values is described in the following paragraph.

The heaviest loaded signal was identified. The current drive strength setting of the TigerSHARC was then adjusted so that the signal integrity could be guaranteed. A current drive strength of 70% was found to be sufficient. An analysis of each of the signal output waveforms was then performed to determine the value of the source resistors that would ensure signal integrity.

Figure 14-5 illustrates the simulation of data bus bit 63. This signal is only loaded by the two SDRAM modules, whilst data bus bit 7 (illustrated in Figure 14-4) is loaded by the 512KB flash memory, FPGA and the two SDRAM modules. The final resistor value chosen for data bus bit 63 is 18Ω.

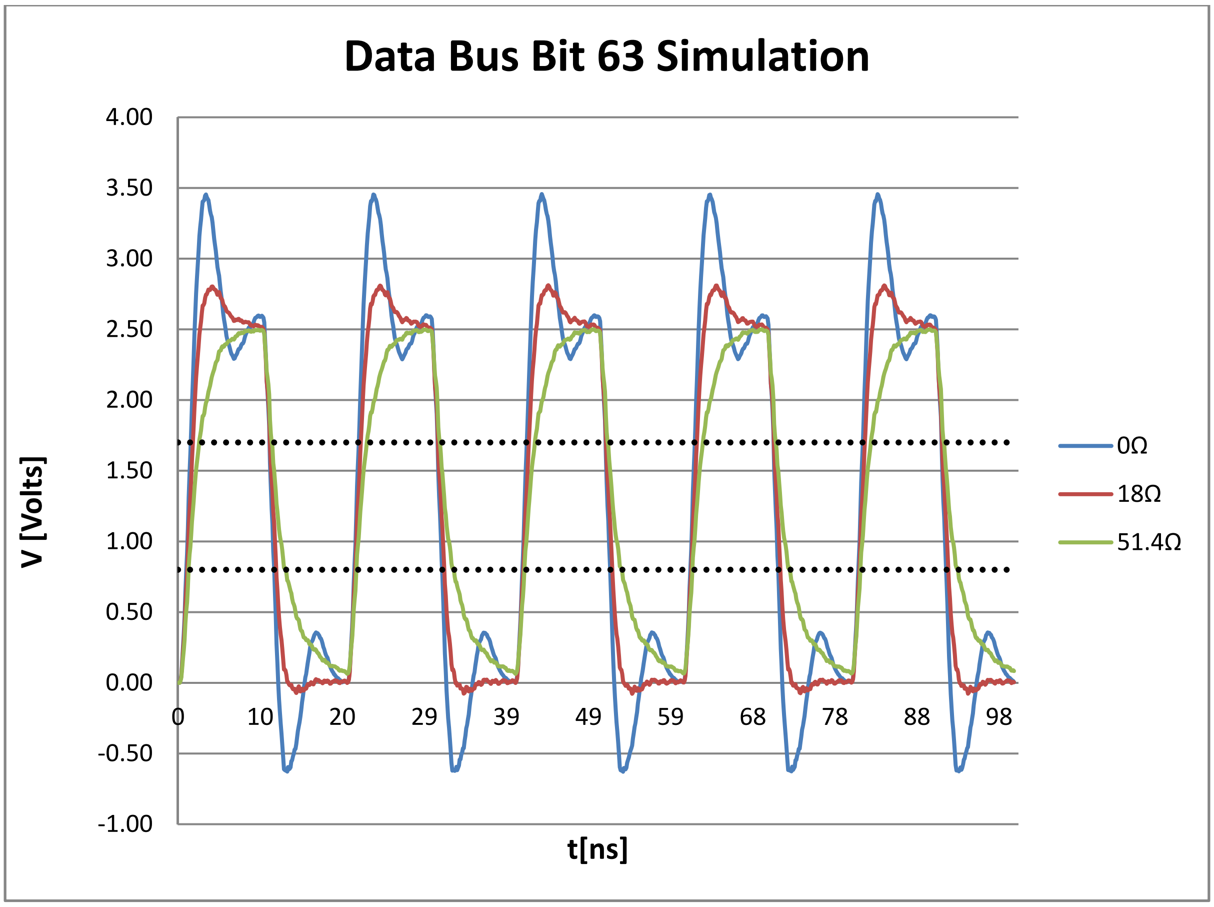


Figure 14-5: Simulation of Data bus bit 63.

This design uses resistor networks as series resistors in order to minimize the component size and count. The resistor network values for the TigerSHARC are illustrated in schematic 16 in Appendix A.

This same method was used to determine the values of the FPGA series resistor networks. The drive strength of the FPGA data bus is set to 16mA. The SDRAM modules already contain series resistors and simulations using models of 256MB SDRAM modules from MICRON were performed to ensure signal integrity. Simulations were also performed on all the other high speed signals in this design using Hyperlynx in order to guarantee signal integrity on the receiver components.

The crosstalk between two PCB traces can be calculated using the following equation (16):

$$Crosstalk \approx \frac{K}{1+\left(\frac{D}{H}\right)^2}$$

where $K < 1$, D is the distance between the centre of the traces and H is the height above the ground plane.

The following equation gives an upper bound on the crosstalk:

$$Crosstalk < \frac{1}{1+\left(\frac{D}{H}\right)^2} \qquad (14\text{-}2)$$

In this system, all the high speed signals are routed with a trace-trace separation of 2.54mm (20mils). The distance between the centres of the traces is therefore 2.67mm (25mils) and the height above the ground plane is 0.127mm (5mils). The maximum percentage crosstalk that can be induced by two parallel traces is calculated from equation (2-5), is 3.8%.

Hyperlynx is capable of performing a crosstalk simulation on adjacent traces. The program will report a crosstalk violation if a voltage of 150mV or greater is induce on the trace. This is well below the 0.7V logic low decision level used by the components in this design. The simulation reported no violations due to crosstalk and therefore routing the high speed signals with a trace-trace separation of 2.54mm (20mils) is deemed to be acceptable.

14.3 Summary and Conclusion

This chapter described the clock management techniques and the high-speed signal requirements that enable our system to synchronously interface at 100MHz with 512MB SDRAM via the SDRAM protocol and at with the FPGA using the TigerSHARC pipelined protocol. In order to achieve the clock signal synchronization that falls within the TigerSHARC and the memory module's specification, use had to be made of a clock skew manager. Furthermore, series source resistor damping was used to avoid ringing on all the clock signals. Finally, series source resistor networks were used to condition the waveform of the data bus, address bus and control signals in order to guarantee signal integrity.

15 Power Supply

This chapter discusses the power supply configuration of the final prototype. The following subsections discuss the design of the power distribution network, the power supply protection, filtering and power supply sequencing.

15.1 Power Distribution Network

In total, 11 separate voltage sources are required for the design. This is a complex challenge and the following paragraphs describe the method used to overcome this problem.

An analysis was performed to group the different components into the required voltage sources. The criteria used to group each voltage supply are as follows:

- The current required by the component.
- The placement of the component on the PCB.
- Does the regulator supply analog or digital components?

The analysis revealed that the following voltage supplies could be categorized and presented as a distribution network, which is illustrated in Figure 15-1.

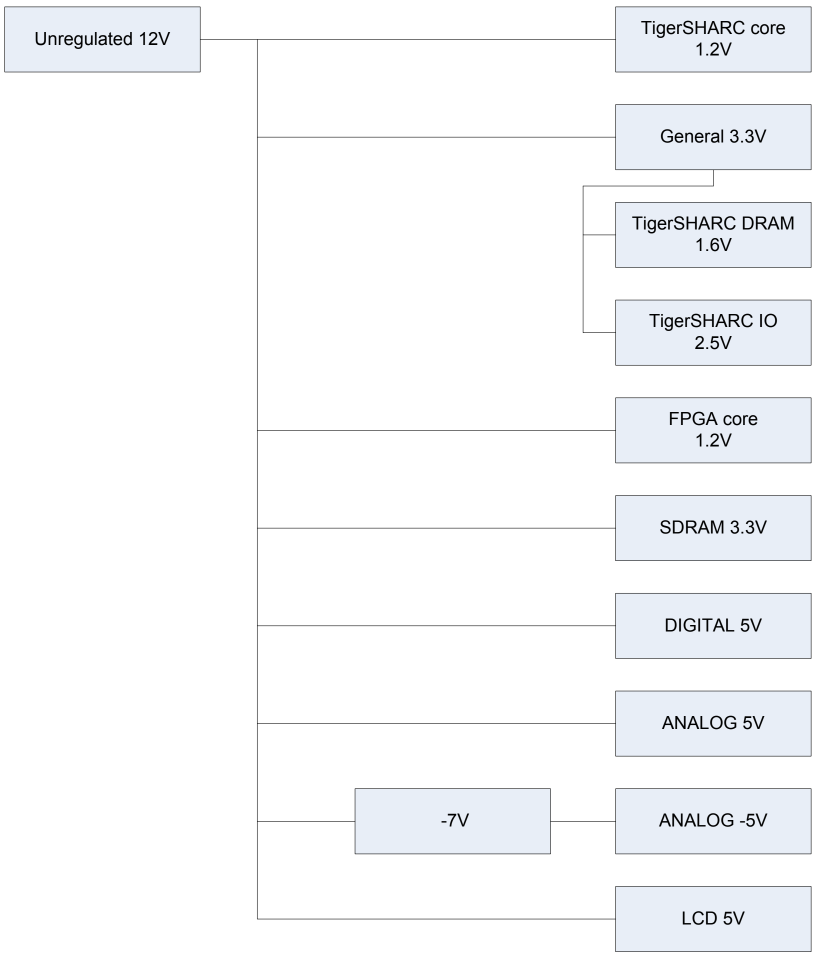


Figure 15-1: Power distribution network.

The system was designed so that the power supply regulation could be tested before the power supply is connected to the PCB power planes and components. To achieve this, a link is used between the output of each regulator and its power plane to isolate the components from the regulator. This scheme is illustrated in Figure 15-2. The figure illustrates that the minimal components, such as the regulator’s input decoupling capacitor, output decoupling capacitor and biasing circuitry, are placed on the PCB before the link. Further bulk capacitors are placed on the power plane itself to provide additional decoupling as is required by the components.

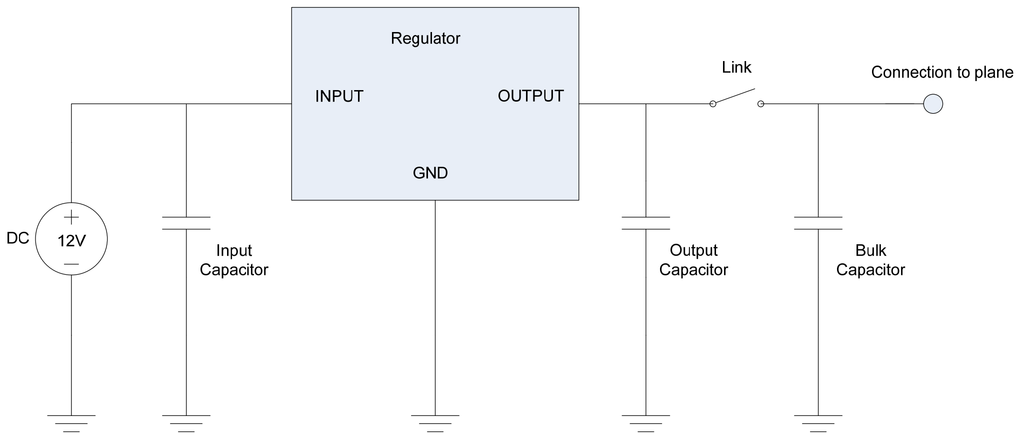


Figure 15-2: Regulator connection scheme.

The following paragraphs detail the power consumption of the components within each group and detail the choice of regulator for each voltage supply:

TigerSHARC core 1.2V

When operating at 600MHz, the TigerSHARC core requires a maximum current of 2.95 A at 1.2V. Due to the high current consumption, a switching regulator is used to supply the 1.2V. A switching regulator that can supply a minimum current of 4A would be sufficient for this purpose. This would ensure that the switching regulator can operate within its limits to ensure reliability. The TigerSHARC requires a maximum ripple of 60mV on its 1.2V supply.

The Texas Instruments PTH12000LAH (U61) point-of-load switching regulator was chosen to supply the 1.2V. The power supply operates from 12V, can supply up to 6A and has maximum output ripple of 20mV. The power supply is illustrated in schematic 23 in Appendix A. Resistor R_{19} is used to program the output voltage, although this resistor is omitted as the regulator will default to 1.2V.

The input voltage required by the regulator is 12V. The regulator requires a 100uF input capacitor (CT53), for which a 100uF electrolytic capacitor with 50V rating was chosen. An output capacitor of minimally 100uF is recommended by the datasheet (18). The datasheet recommends the use of tantalum capacitors with a minimum equivalent series resistance (ESR) of 7mΩ. The AVX TPS series 100uF capacitor (CT53) with voltage rating of 16 V and an ESR of 60mΩ was chosen for this design.

The TigerSHARC engineering EE-179 note (19) recommends a minimum bulk capacitor value of 470uF with an ESR less than 25mΩ. The AVX TPM series 680uF tantalum capacitor CT55 with a voltage rating of 6.3V and ESR maximum of 23mΩ is used after the link as a bulk capacitor for the 1.2V plane. A second capacitor (CT56) is placed in parallel to provide further decoupling if needed.

The efficiency of the regulator for a load of 4A is given by the data sheet as 85%. The datasheet states that when the regulator operates at 1.2V, it will be able to perform reliably within its full current range of 0-6A at maximum ambient temperature of 85°C. It is therefore necessary to keep the ambient temperature below 85°C.

TigerSHARC DRAM 1.6V

When operating at 600MHz, the TigerSHARC DRAM requires a maximum current of 0.43 A. Due to the low current consumption, a thermally efficient low drop-out linear regulator is used to supply the 1.6V. The TigerSHARC requires a maximum voltage ripple of 60mV. The TigerSHARC requires that, upon power-up, the DRAM voltage be applied only after the core and IO voltages are stable. It is therefore necessary to use the regulator that can be controlled by a voltage supervisor that monitors theses voltage supplies.

The Texas Instruments programmable output TPS77801 regulator was chosen for this purpose. The regulator has an enhanced power pad under the IC which can be connected to the in the internal ground plane to enhance heat dissipation. Its maximum load regulation is 3%. At 1.6V this results in a maximum ripple of 48mV which is below the 60mV required by the TigerSHARC. The regulator (U60 illustrated in schematic 23 in Appendix A) can supply a maximum or 750mA. It has an active low enable pin (pin 5) which is connected to the output of the voltage sequencer (the voltage sequencing is discussed in section 15.3). Resistors R_{104} and R_{91} are used to program the output voltage to 1.6V as follows:

$$V_O = 1.1834 \times \left(1 + \frac{R_{104}}{R_{91}}\right)$$

By choosing:

R_{91}=110kΩ

the value of R_{104} is calculated by:

$$R_{104} = \left(\frac{V_O}{1.1834} - 1\right) \times R_{91}$$

R_{104}= 38.7kΩ

The closest 1% resistor value is 39kΩ. With this value the output value will be:

V_O=1.603V

The voltage input is connected to the general 3.3V supply. The efficiency of the supply can therefore be calculated by the following equation:

Efficiency=$\frac{V_O}{V_{in}}$= 1.6/3.3= 48.4%

Therefore 51.6% of the power P_{in} will be dissipated as heat. The total power is given by:

$$P_{in} = V_{in} \times I_{in}$$

Where

$$I_{in} \approx I_{out}$$

Therefore

$$P_{HEAT} = 0.516 \times V_{in} \times I_O = 0.516 \times 3.3 \times 0.43 = 0.732W$$

The datasheet (20) states that the regulator can dissipate 2.9W at 25°C, 1.9W at 70°C and 1.5W at 85°C, all with zero airflow. A 0.732 W heat dissipation is therefore well within this regulator's capabilities.

A 10uF tantalum capacitor (CT40) with a voltage rating of 10 V is used is an input capacitor. The output tantalum capacitor is the AVX TPS series 100uF capacitor (CT61) with voltage rating of 16V and a ESR of 60mΩ. The TigerSHARC engineering EE-179 note (19) recommends a minimum bulk capacitor value of 47uF with an ESR less than 100mΩ to decouple the power plane. The AVX TPM series 470uF tantalum capacitor CT39 with a voltage rating of 6.3V and ESR maximum of 23mΩ is used for this purpose.

TigerSHARC IO VCC 2.5V

When operating at 600MHz, the TigerSHARC IO requires a maximum current of 0.15 A. Due to the low current consumption, a thermally efficient low drop-out linear regulator is required to supply the 2.5V. The TigerSHARC requires a maximum voltage ripple of 130mV. The Texas Instruments fixed output TPS77825 regulator was chosen for this supply. The regulator has enhanced power pad under the component which can be connected to the in the internal ground plane to enhance heat dissipation. Its maximum load regulation is 3%. At 2.5V this results in a maximum ripple of 75mV which is below the 130mV required by the TigerSHARC. The regulator can supply a maximum of 750mA. The regulator (U55) is illustrated in schematic 23 in Appendix A. It has an active low enable pin (pin 5) which is connected to GND, as it is always enabled.

The voltage input is connected to the general 3.3V supply. The efficiency of the supply can therefore be calculated by the following equation:

Efficiency=$\frac{V_O}{V_{in}}$= 2.5/3.3= 75.7%

Therefore 24.3% of the power P_{in} will be dissipated as heat. The total power is given by:

$$P_{in} = V_{in} \times I_{in}$$

Where

$$I_{in} \approx I_{out}$$

Therefore

$$P_{HEAT} = 0.243 \times V_{in} \times I_O = 0.243 \times 3.3 \times 0.15 = 0.120W$$

The datasheet (20) states that the regulator can dissipate 2.9W at 25°C, 1.9W at 70°C and 1.5W at 85°C all with zero airflow. A 0.12 W heat dissipation is therefore well within this regulator's capabilities.

A 10uF tantalum capacitor (CT38) with a voltage rating of 10 V is used as an input capacitor. The output tantalum capacitor is the AVX TPS series 100uF capacitor (CT37) with voltage rating of 16V and an ESR of 60mΩ. The TigerSHARC engineering EE-179 note (19) recommends a minimum bulk capacitor value of 100uF with an ESR less than 100mΩ to decouple the power plane. The AVX TPM series 470uF tantalum capacitor (CT60) with a voltage rating of 6.3V and ESR maximum of 23mΩ is used after the link as a bulk capacitor for the 2.5V power plane.

Analog VA 5V

The devices that are connected to the analog 5V supply require approximately 200mA. Due to this low current consumption, a linear regulator can be used to supply the 5V. The Texas Instruments fixed output TLV1117-50CKVUR linear regulator (U14) was chosen for this purpose. The regulator package is enhanced to dissipate the heat effectively and its' maximum voltage ripple is

15mV. The regulator can supply a maximum or 800mA and is illustrated in schematic 13 in Appendix A.

The voltage input is connected to the general 12V supply. The efficiency of the supply can therefore be calculated by the following equation:

Efficiency=$\frac{V_O}{V_{in}}$= 5/12= 41.6%

Therefore 58.4% of the power P_{in} will be dissipated as heat. The total power is given by:

$$P_{in} = V_{in} \times I_{in}$$

Where

$$I_{in} \approx I_{out}$$

Therefore

$$P_{HEAT} = 0.584 \times V_{in} \times I_O = 0.584 \times 12 \times 0.2 = 1.4\text{W}$$

The datasheet (20) states that the regulator can dissipate 3.3W at 25°C , 1.8W at 70 °C and 1.3W at 85°C, all with zero airflow. A 1.4 W heat dissipation is therefore well within this regulator's capabilities if the ambient temperature is kept low.

The regulator uses a 10uF input capacitor (CT26) with a 25V rating. An output capacitor of minimally 100uF is recommended by the datasheet (21). Accordingly, an AVX TPS series 100uF capacitor (CT25) with voltage rating of 16 V is used on the output. A series shunt inductor (L5), with an impedance of 600Ω at 100MHz is used to further filter the power supply. The inductor also connects the power supply to the 5V analog plane, and a further 100uF tantalum capacitor (CT31) is placed on the analog power plane as a bulk capacitor.

The datasheet recommends the use of a diode between the output and input pins to protect the regulator if the input is instantaneously shorted to ground. The datasheet recommends using the 1N4002 diode. However, this diode is a through-hole component, therefore the equivalent surface mount diode, the US1B (22) is used in our design.

LCD 5V

The total current required by the LCD is approximately 260mA. Due to the low current consumption, a linear regulator can be used to supply the 5V. The Texas Instruments fixed output TLV1117-50CKVUR regulator was chosen for this supply. This is the same device used to supply the analog +5V described in the previous section. The regulator (U9) is illustrated in schematic 13 in Appendix A.

Since the voltage input is again the general 12V supply, the efficiency is also 41.6%.The total power dissipated by heat is given by:

$$P_{HEAT} = 0.584 \times V_{in} \times I_O = 0.584 \times 12 \times 0.26 = 1.8\text{W}$$

The datasheet states that the regulator can dissipate 3.3W at 25°C, 1.8W at 70°C and 1.3W at 85°C all with zero airflow. A 1.8 W heat dissipation is therefore well within this regulator's capabilities if the ambient temperature is kept low.

The same input (CT23) and output (CT24) capacitors chosen for the analog 5V configuration are again employed here, as recommended in the datasheet (21). An input short protection diode is also included as before.

Digital Vcc 5V

The total current required for the digital +5V supply is approximately 200mA. As in the previous two sections, the Texas Instruments fixed output TLV1117-50CKVUR linear regulator (U13) was chosen for this for this purpose. The regulator is illustrated in schematic 13 in Appendix A and is again fed from the general 12V supply and hence the efficiency is again 41.6%.

Therefore 58.4% of the power P_{in} will be dissipated as heat. The total power dissipated by heat is given by:

$$P_{HEAT} = 0.584 \times V_{in} \times I_O = 0.584 \times 12 \times 0.2 = 1.4W$$

The datasheet (20) states that the regulator can dissipate 3.3W at 25°C , 1.8W at 70°C and 1.3W at 85°C all with zero airflow. A 1.4 W heat dissipation is therefore well within this regulator's capabilities if the ambient temperature is kept low.

The same input (CT23) and output (CT24) capacitors chosen for the analog 5V and the LCD 5V configuration are again employed here, as recommended in the datasheet (21). An input short protection diode is also again included. A further capacitor (CT32) with voltage rating of 16V is used after the link as a bulk capacitor for the VCC 5V power plane.

Negative -7.4V

A voltage supply of less than -7V is required to supply the analog -5V regulator. The Texas Instruments PTN78000A point-of-load switching regulator (U26) was chosen to provide the -7.4V supply. The regulator operates from 12V, can supply up to 1.5A and has a maximum output ripple of 10mV. The power supply is illustrated in schematic 13 in Appendix A. Resistor R_{18} is used to program the output voltage to be -7.4V. The regulator uses a 100uF electrolytic capacitor (CT53) with a 50V rating in parallel with two 4.7uF ceramic capacitors (C204 and C205) as input capacitors. An output capacitor (CT43) of minimally 100uF is recommended by the datasheet (23) and hence, a 100uF electrolytic capacitor with 50V rating.

A series shunt inductor (L9) with an impedance of 600Ω at 100MHz is used to further filter the power supply and to connect the -7.4V to input of the -5V analog regulator.

The efficiency of the supply is given by the datasheet (23) as 85%. The datasheet states that when the regulator operates at -7.4V, then the regulator will be able to operate reliability when supplying the required 50mA to the -5V regulator at maximum ambient temperature of 85°C.

Analog VA -5V

The total current required for the Analog -5V supply is approximately 50mA. Due to this low current consumption, the Texas Instruments fixed output µA79M05CKTPR linear regulator (U21) is used. The regulator package is enhanced to dissipate the heat more effectively and the regulator's maximum output voltage ripple is 50mV. The regulator can supply a maximum of 500mA and is illustrated in schematic 13 in Appendix A.

The regulator input is connected to the -7.4V supply. The efficiency of the supply can therefore be calculated by the following equation:

Efficiency=$\frac{V_O}{V_{in}}$= 5/7.4= 67.5%

Therefore 32.5% of the power P_{in} will be dissipated as heat. The total power is given by:

$$P_{in} = V_{in} \times I_{in}$$

Where

$$I_{in} \approx I_{out}$$

Therefore

$$P_{HEAT} = 0.325 \times V_{in} \times I_O = 0.325 \times 7.4 \times 0.05 = 0.12W$$

The datasheet (24) states that the regulator can dissipate 3.57W at 25°C, 1.96W at 70°C and 1.42W at 85°C all with zero airflow. A 0.12 W heat dissipation is therefore well within this regulator's capabilities.

The regulator uses a 10uF input capacitor (CT29) with a 25V rating. An output capacitor of minimally 100uF is recommended by the datasheet (24). The AVX TPS series 100uF capacitor (CT30) with voltage rating of 16 V is used on the output. A series shunt inductor (L6) with an impedance of 600Ω at 100MHz is used to further filter the power supply. The inductor is also used to connect the power supply to the -5V analog plane. A further 100uF tantalum capacitor (CT32) is placed on the analog power plane as a bulk capacitor.

SDRAM VCC 3.3V

The Kingston memory modules used for this design consume 4.323W each (14). Therefore a 3.3V volt regulator that can supply a minimum of power 8.646W and a current of 2.62A is required. The Texas Instruments PTH12000WAH point-of-load switching regulator (U53) was chosen for this purpose because it can operate from 12V and can supply 6A with a maximum output ripple of 20mV. The power supply is illustrated in schematic 23 in Appendix A. Resistor R_{76} is chosen as 2kΩ to program the output voltage to 3.3V. The datasheet (18) states that by choosing $R_{76} = 2k\Omega$, the output will be set to 3.3V. The required input capacitor (CT35) is a 100uF electrolytic capacitor with a 50V rating. An output capacitor of minimally 100uF is recommended by the datasheet as well as the use of tantalum capacitors with a minimum equivalent series resistance (ESR) of 7mΩ. The AVX TPS series 100uF capacitor (CT36) with voltage rating of 16 V and an ESR of 60mΩ were therefore chosen as the output capacitor. The AVX TPM series 470uF tantalum capacitor (CT49, 50, 57, 59) with a voltage rating of 6.3V and ESR maximum of 23mΩ are used as bulk capacitors for the 3.3V plane. These capacitors are optional and are not soldered on the PCB.

The efficiency of the supply for a load of 4A is given by the datasheet as 85%. The datasheet states that when the regulator operates at 3.3V, it will be able to operate reliably within the current range of 0-4A at maximum ambient temperature of 85°C. The maximum ambient temperature then decreases linearly to 65°C for a maximum load of 6A. It is therefore necessary to keep the ambient temperature below 85°C when using this regulator.

General VCC 3.3V

It is estimated that approximately 1A would be required for the general 3.3V supply. However, to operate from a 12V supply with a linear regulator is only 27.5% efficient. Therefore the Texas Instruments PTH08080WAH point of load switching regulator (U29) was chosen to supply the general 3.3V. This regulator operates from 12V, can supply up to 2.25A and has maximum output ripple of 15mV. The power supply is illustrated in schematic 10 in Appendix A. Resistor R_{50} , chosen to be $1.8k\Omega$ (18), is used to program the output voltage to 3.38V. The input voltage required by the regulator is 12V. The input capacitor (CT12) is a 100uF electrolytic capacitor with 50V rating. An output capacitor of minimally 100uF is recommended by the datasheet, as well as the use of tantalum capacitors with a minimum equivalent series resistance (ESR) of 7mΩ. The AVX TPS series 100uF capacitor (CT42) with voltage rating of 16 V and a ESR of 60mΩ was chosen for this design. The AVX TPM series 470uF tantalum capacitor (CT13) with a voltage rating of 10V and ESR maximum of 23mΩ is used as bulk capacitor for the 3.3V plane.

The efficiency of the supply for a load of 2A is given by the data sheet as 92%. The datasheet states that, when the regulator operates at 3.3V, it will be able to operate reliably within the current range of 0-2.25A at maximum ambient temperature of 85°C.

FPGA VCC 1.2V

The Altera Excel-based PowerPlay Power Estimator was used to estimate that the maximum current required for the FPGA core is approximately 0.41A. To operate from a 12V input, a switching regulator is required. The Texas Instruments PTH12000LAH point-of-load switching regulator (U25) was chosen to supply the 1.2V because it can operate from 12V can supply up to 6A and has maximum output ripple of 20mV. The power supply is illustrated in schematic 10 in Appendix A. Resistor R_{113} is used to program the output voltage, however the resistor can be omitted as the regulator will default to 1.2V. The regulator uses a 100uF electrolytic capacitor with 50V rating (CT14) as input capacitor. An output capacitor of minimally 100uF is recommended by the datasheet (18) as well as the use of tantalum capacitors with a minimum equivalent series resistance (ESR) of 7mΩ. The AVX TPS series 100uF capacitor (CT15) with voltage rating of 16 V and a ESR of 60mΩ was chosen for this design. The AVX TPM series 680uF tantalum capacitor (CT47 and CT48) with a voltage rating of 6.3V and ESR maximum of 23mΩ is used after the link as a bulk capacitor for the 1.2V plane.

The efficiency of the supply for a load of 4A is given by the data sheet as 85%. The datasheet states that when the regulator operates at 1.2V, the regulator will be able to operate reliability within its full current range of 0-6A at maximum ambient temperature of 85°C. It is therefore necessary to keep the ambient temperature below 85°C.

15.2 Power Supply Protection and Filtering

This section describes the method used to connect the 12V unregulated power plane on the PCB to an external Power supply. First it is necessary to determine the total power required by the board. The peripheral board consumed a total of 600mA from its 12V supply. The total power consumed by the peripheral board is therefore 7.2W.

The total power consumed can be calculated by summing the peripheral board power and the power required by the voltage supplies of the final prototype that were not included on the peripheral board. This calculation is illustrated in Table 15-1.

Voltage Supplies	Voltage [V]	Current [A]	Power [W]
TigerSHARC 1.2V	1.2	2.95	3.54
TigerSHARC 1.6V (coupled to 3.3V)	3.3	0.43	1.419
TigerSHARC 2.5V (coupled to 3.3V)	3.3	0.15	0.495
SDRAM 3.3V	3.3	1.31	4.323
Fan1	12	0.05	0.6
Fan2	12	0.05	0.6
Peripheral board	12	0.6	7.2
Total			18.17

Table 15-1: Power requirement of various elements of the final prototype.

The final prototype would therefore consume a maximum of 18.17W. Therefore a 12V power supply that can supply 1.6A would be required. However, to ensure that such a power supply would operate reliably and to allow for a margin of error in the calculation of the power requirements, a 12V power supply that can supply a minimum of 3A is used

The power supply connectors, protection and filtering are illustrated in schematic 2 in Appendix A. A DC power connector (U48) with a current rating of 3.5A is used to connect an external power source to the PCB. The +12V output of the connector is connected in series to a panel mounted switch (U64). The switch is not yet implemented on the prototype, however.

The output of the switch is connected to 20mm fuse connector (U7). The fuse used is rated at 3.5A. The output of the fuse is connected through a ST STPS745G power Schottky rectifier diode (U15), which ensures that the power can only be connected with the correct polarity to the board. The diode has an average forward current rating of 7.5A and a maximum forward voltage of 0.57V (25).

The output of the diode is connected to one coil of a common mode choke (U49), the output of which is then connected to the PCB +12V plane. The ground connection of the DC connector is connected to the second coil of the common mode choke and the output which is then connected to the PCB digital ground plane.

The common mode choke filters high frequency noise from the power supply. The choke has an impedance of 700Ω at 100MHz and a current rating of 5A (26). The choke used has the highest impedance, with the appropriate current rating, out of the range of chokes available from the supplier.

The total current required consumed by the final prototype was measured and found to be 1.4A. The total power consumed is therefore 16.8W.

15.3 Power Supply Sequencing

The TigerSHARC DSP requires that, upon power up, the DRAM 1.6 volts must be turned on only once the core 1.2V and IO 2.5V are stable. As is illustrated in Figure 15-1, the 2.5V and 1.6V are coupled to the general 3.3V supply.

The PTH08080 3.3.V regulator, used for the general supply, has an internal soft start mechanism to limit the in rush current at start up. Upon power up, the PTH08080 3.3V power supply will take approximately 5ms to stabilize once the regulator input voltage reaches 4.5V. The 1.6V and 2.5V regulators are coupled to the 3.3V supply; these regulators will stabilize within 0.8ms after the enable pins are asserted. Since the 2.5V supply is always enabled, the 2.5V regulator will essentially switch on and stabilize at the same time as the 3.3V supply.

The PTH12000LAH 1.2.V regulator has an internal soft start mechanism to limit in rush current at start up. Upon power up the PTH12000LAH 1.2.V power supply will take approximately 12.5ms to stabilize once the regulator input reaches 10.8V.

The MAXIM MAX6898AALT adjustable voltage sequencer is used to sequence the activation of the 1.6V regulator. The MAX6898AALT (U52) can operate from a wide power supply range (1.5V to 5.5V) and has an adjustable capacitor delay. The MAX6898AALT (illustrated in schematic 23 in Appendix A) is configured to operate from the general 3.3V and will begin to operate 2ms after the power supply voltage reaches 1.5V.

From the above explanations, it is clear that the 3.3V and 2.5V will stabilize first, approximately 6ms after power up. It is therefore sufficient to monitor only the 1.2V supply, since this will take significantly longer (12.5ms) to stabilize.

The MAX6898AALT has an internal threshold 0.5V, when the monitoring voltage decreases to 5mV below the 0.5V threshold, then the MAX6898AALT will de assert the $\overline{\text{enable}}$ output.

The MAX6898AALT is configured to monitor the 1.2V source. The resistor divider sets the monitoring voltage to 76% of the 1.2V source. The minimum input voltage is 0.66V (approximately 0.5x1.2V). The 1.2V (when stable) will be seen as 0.92V on the monitoring input pin. This allows 0.4V threshold between the input threshold of the MAX6898AALT.

It is unknown how long it will take for the 1.2V output to reach 0.66V. The worst case scenario would be that it reaches 0.66V instantaneously. Then the timing margin would be reduced by at the most 12.5ms. It is therefore necessary to compensate for this unknown factor.

A 6.9nF capacitor (C138) is chosen to introduce a delay of 27.6ms. The delay is calculated using the following equation in the datasheet (27):

$$t_{DELAY} = (C_{138} \times 4 + 40 \times 10^{-3})ms$$

A worst case scenario, where the 1.2V switches on instantaneously, there is still a 27.6ms-12.5ms=15.1ms margin in the timing.

To summarise, the timing of the voltage sequencing will be as follows:

- If the 12V unregulated source has reached 4.5V, then the 3.3V and 2.5V will stabilize after 6ms.
- If the 12V unregulated source has reached 10.8V, then the 1.2V will stabilize after 12.5ms.
- If the 3.3V source has reached 1.5V and the 1.2V source has reached 0.66V, the $\overline{\text{enable}}$ output will be asserted after 27.6ms.

Upon power up, the reset manager (described in section 12.1) will assert a system reset pulse of 200ms duration after a stable 3.3V supply has been achieved. The power sequencing will therefore be complete before this pulse terminates.

15.4 Summary and Conclusion

This chapter described the complex task of supplying power to all the components of the design. In total 11 regulators were required. The final prototype operates from one 12V source and utilizes approximately 16.8W.

16 Schematic Capture and PCB layout

This chapter discusses the schematic capture and the design of the PCB layout in the following subsections for the final prototype system.

16.1 Schematic Capture

The Mentor Graphics DX Designer software package was used in this design to capture all schematics. The project files are included on the DVD that accompanies this thesis. The schematics are illustrated in Appendix A, and a higher resolution PDF document is included on the DVD.

The bill of materials used in our design is illustrated in Appendix C on page 201. The final component count is 581. The majority of the components are decoupling capacitors and resistors used in the design.

16.2 PCB Layout

The PCB layout was designed using the Mentor Graphics Pads Layout software package. Although, the package is very powerful, it does not have any built-in component footprint libraries and it was therefore necessary to either generate the footprints or to import them from other PCB layout software packages.

It is possible to simulate the entire PCB design with the Mentor Graphics Hyperlynx board simulation software package. This was used for our design to ensure the successful operation of the PCB before it was manufactured. In total, only two PCBs were designed, one for the peripheral board and one for the prototype board, both of which worked correctly the first time. Only the final prototype's PCB manufacturing layers are included in Appendix B on page 190. The original Gerber files as well as the project files for both PCBs are included on the DVD that accompanies this thesis.

The routing of a PCB design, with a high speed 64-bit data and a 24-bit address bus, 13 high speed control signals and 7 high speed clock signals is a very challenging task, which is complicated not only by the fact that you have to adhere to the high speed signalling requirements (as described in section 14.2), but also by the fact that components often do not align the pins of related signals. The task can be made easier by increasing the number of routing layers, but the cost of manufacturing the PCB increases dramatically as more layers are used. To keep the costs of this design down the absolute minimum of 6 layers is used in the final prototype. By comparison, the Analog Devices development board uses a 14-layer PCB.

It is essential to route the components with the same scheduling and to keep the stubs to each component as short as possible to ensure that each component experiences a similar degree of loading on the trace. If one does not adhere to this, unwanted effects such as ringing can occur, which can lead to multiple crossings of the logic decision levels and therefore signal integrity problems.

There is a constant trade-off between the arrangement of the components on the PCB and complexity of routing the bus signals. Through an iterative process, involving numerous routing attempts followed by simulations, an optimal component placement was found that would guarantee the successful operation of the final prototype.

The optimal placement of the high speed components such as the TigerSHARC, SDRAM modules 0 and 1, and the FPGA, as well as the peripheral components, is illustrated in

Figure 16-1. This arrangement allows the high speed signals to be scheduled in the following manner: the TigerSHARC to the SDRAM module 0, the SDRAM module 0 to the SDRAM module 1, the SDRAM module 1 to the FPGA, and finally the FPGA to the Flash memory.

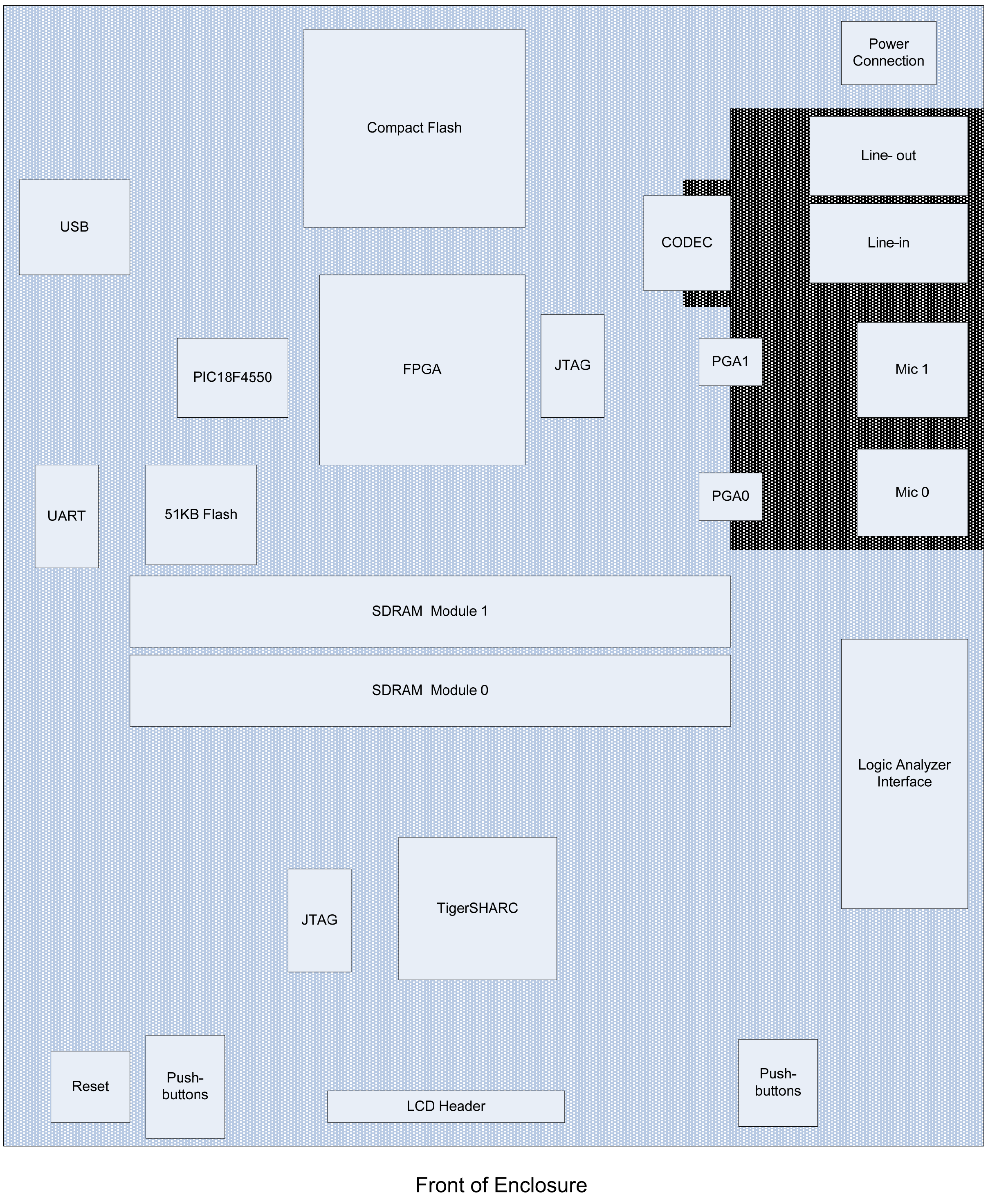


Figure 16-1: The PCB component arrangements.

The arrangement of the peripheral components is determined by the system they belong to. The audio components are placed on the right, the power connections are placed at the top right, the Compact Flash at the top of the board to allow ease of removal, the USB interface is placed on the left, the LCD and push buttons interface are placed at the bottom (at the front of the enclosure).

All the techniques with regards to the high-speed signalling (discussed in chapter 14) have been employed in the PCB design. All high speed signals have been routed with a 0.127mm (5 mills) width, to ensure a trace impedance of 59Ω, and a trace separation of 2.54mm (20mils) to nullify the influence of cross-talk. All the clock traces have been routed with a trace length tolerance of +-15.86mm (125mils).

The low speed signals are routed with a 0.127mm (5 mills) trace width and separation. The internal routing layers are routed in perpendicular directions to nullify any unwanted effects such as crosstalk between these layers.

The signal traces use vias with a pad size of 0.55mm and a drill size of 0.25mm. Larger vias with a pad size of 1.6mm and a drill size of 0.95mm are used to connect the regulators to the internal power planes. The traces that are connected to the regulators are routed with a 0.762mm (30mils) trace width to account for the higher currents. At this width, for a current of 3A, the trace temperature will be 30°C (28). The traces that connect the power supply pins of the various components are routed with a width of 0.254mm (10mils). At this width, for a current of 1.2A ampere, the trace temperature will be 30°C (28). All of the above current ratings are well above the maximum currents that are required by each of the component supply pins.

Separate power planes are used for each supply. This reduces the possibility of ground loops, and the capacitive coupling of the power and ground planes improve the power supply decoupling. In particular, separate digital and analog ground planes and supplies are used for the audio system. Only the analog signal routing and components are placed above or underneath the analog planes to ensure that no noise from the digital switching components is induced in the sensitive audio components. The analog ground plane is connected to the digital ground at one point beneath the codec through a zero ohm resistor.

All the tolerances used for this PCB are within the manufacturers (TRAX) capabilities. The final PCB was manufactured with a gold finish. This is necessary as TigerSHARC is a 576 pin ball grid array (BGA) device, and by using a gold finish the reliability of the soldering process for this device improves.

Figure 16-2 illustrates the PCB layout of final prototype (with a component count of 581) with all the routing layers super-imposed as one layer. For clarity, the ground and power planes are not shown in this figure.

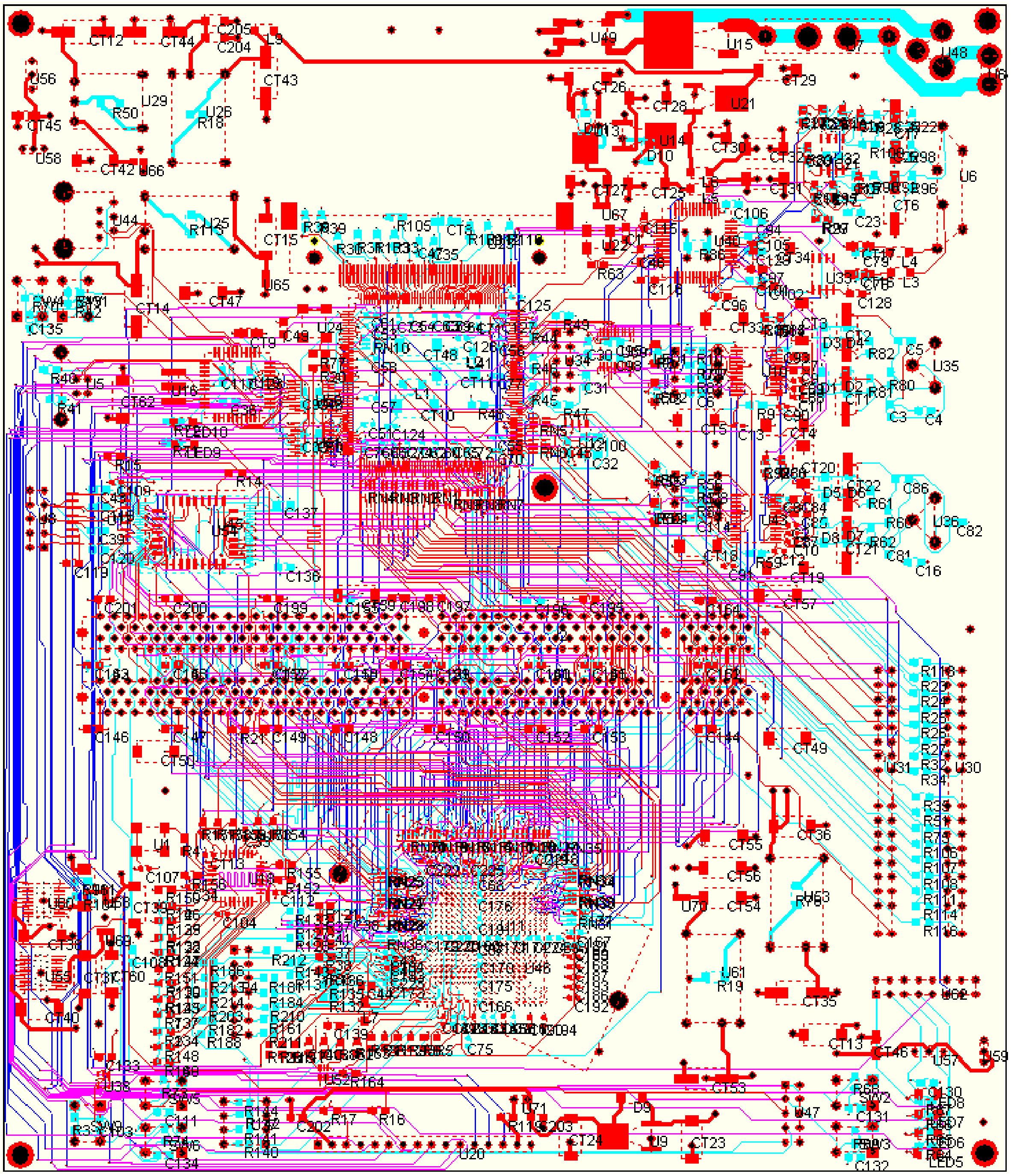


Figure 16-2: The four signal layers superimposed to illustrate the overall PCB design.

A photograph of the final prototype is illustrated in Figure 16-3. The audio line–out is at the top-right, the audio line-in is bellow the line-out, the microphone inputs are below the RCA connectors, the USB is on the left of the PCB, the Compact Flash card is at the top, the power connection is at the top right, the SDRAM is in the middle and the TigerSHARC is underneath the Fan.

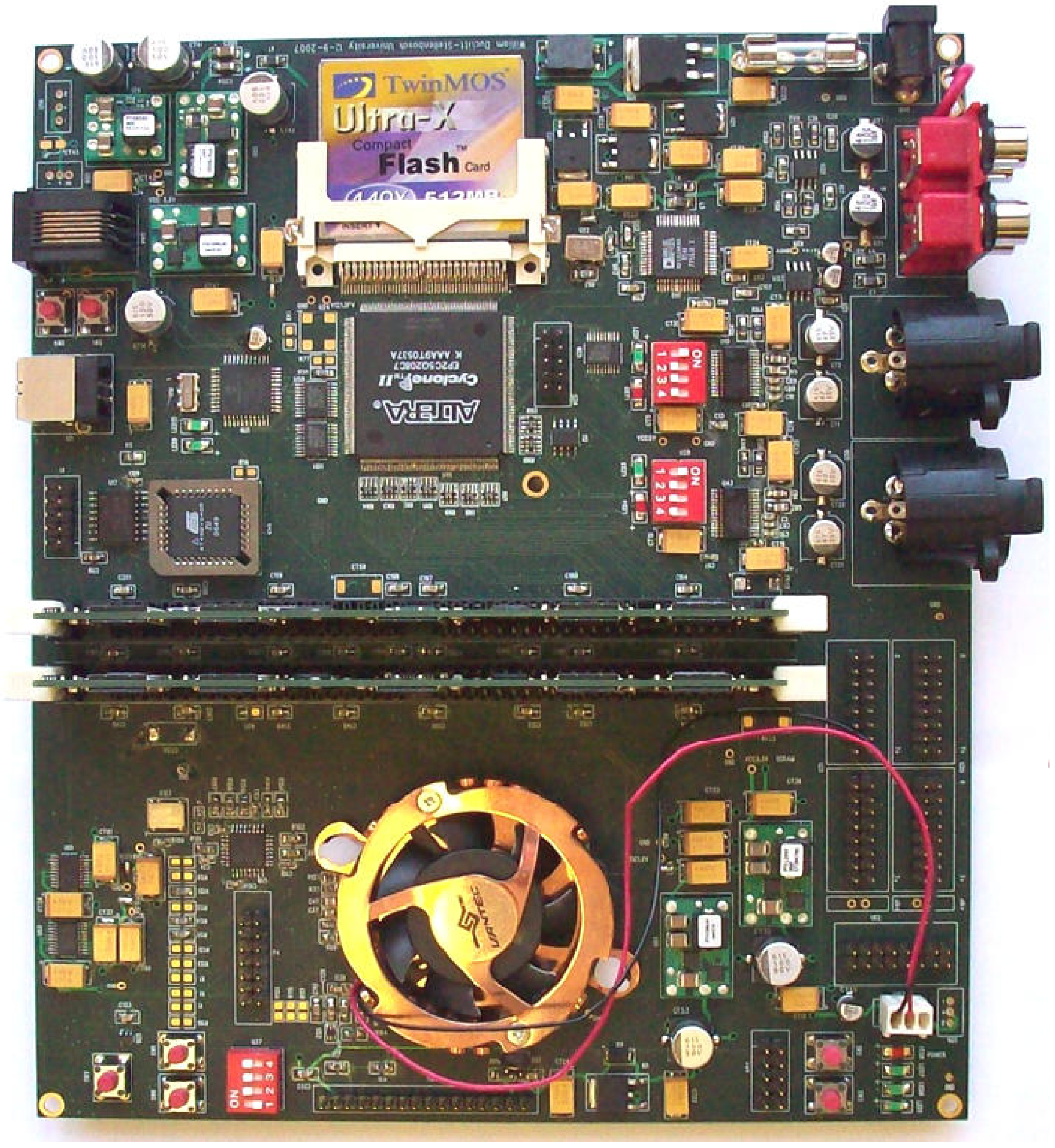


Figure 16-3: The final prototype.

A photograph of the underneath side of the PCB is illustrated in Figure 16-4 on the following page.

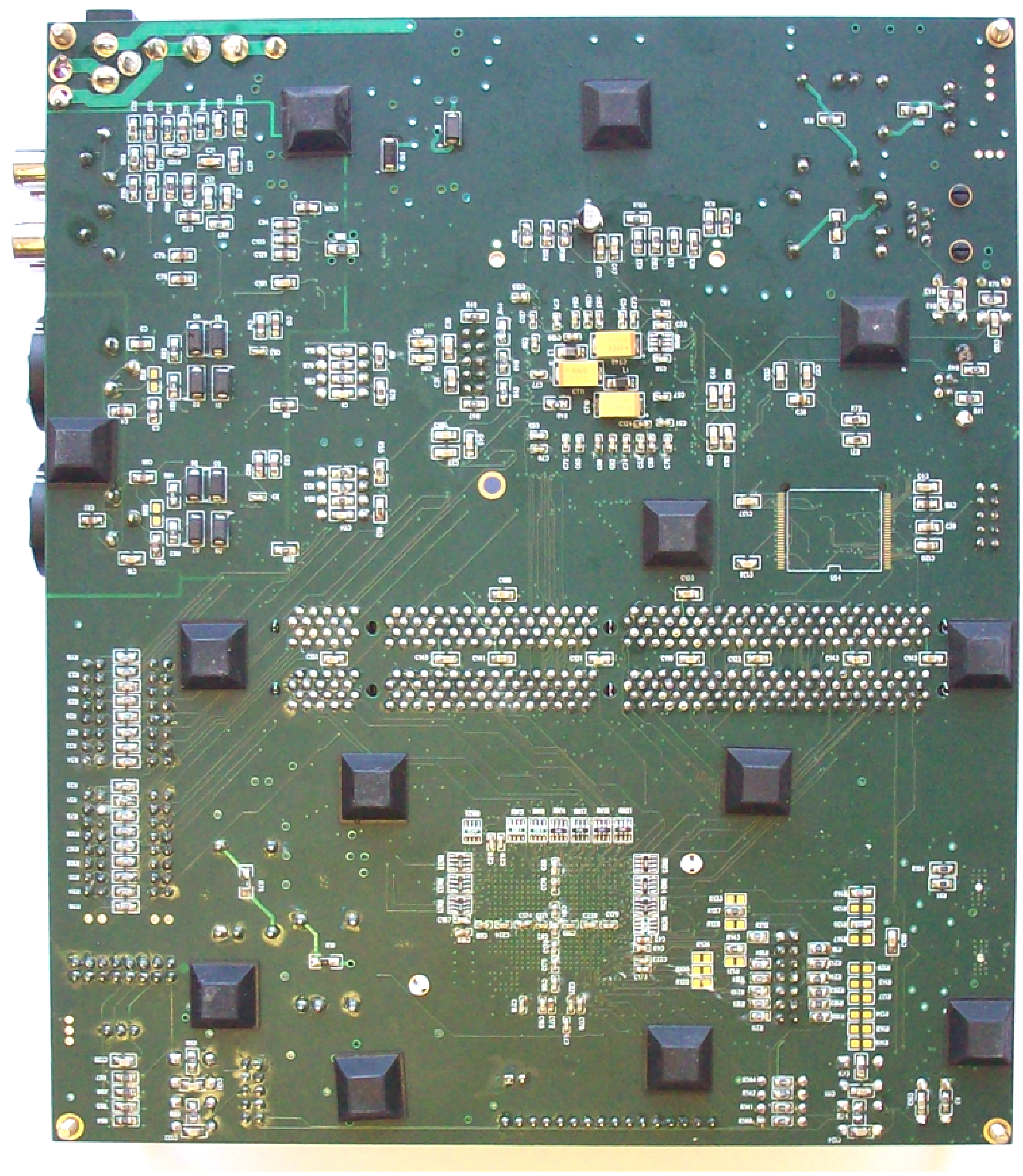


Figure 16-4: The bottom view of the final prototype.

16.3 Summary and Conclusion

This chapter described the techniques used to successfully develop the initial peripheral board as well as the final prototype board. This PCB layout was one of the most challenging aspects of this design. It was complicated not only by the fact that the routing required the adherence to high speed design guidelines, but also by the fact that the system uses a 64-bit data bus, a 24-bit address bus, 13 high speed control signals, 7 high speed clock signals as well all the other components signals. The final prototype PCB had a total of 6 layers, which include a ground layer, a power layer and four signal layers.

17 Software

This chapter describes the software drivers and code that have been written to demonstrate the operation of the prototype system. The software is described by first presenting the lower level functions, and then the higher level functions. All the software has been written in ANSI C and is included on the DVD that accompanies this thesis.

17.1 Communication with the FPGA

Communication with the memory mapped registers within the FPGA can be achieved by means of the following functions. All the FPGA memory mapped registers are defined in the 'FPGA_def.h' header file.

Function: read_fpga

Prototype:	unsigned int read_fpga(unsigned int address)
Location:	"mainA.c"
Description:	Reads the data from the specified address within the FPGA as defined in the "FPGA_def.h" header file.

Function: write_fpga

Prototype:	void write_fpga(unsigned int address, unsigned int data)
Location:	"mainA.c"
Description:	Writes to the specified address within the FPGA as defined in the "FPGA_def.h" header file.

The software that is described in the following sections makes use of these two procedures to communicate to the FPGA.

17.2 FPGA SPI module

Communication with the PGA and CODEC configuration registers occurs by means of the following functions. All the PGA and CODEC registers are defined in the ‘FPGA_def.h’ header file and the functions stated below are in ‘spi.c’.

Function: PGA_write

Prototype:	void PGA_write(unsigned int data[2])
Location:	“spi.c”
Description:	Writes 2x16-bits (data[0] and data[1)] to PGA0 and PGA1.

Function: codec_ spi_write

Prototype: void codec_spi_write(unsigned int address, unsigned int data)

Location: “spi.c”

Description: Writes the 10-bit data to the internal configuration registers within the codec. The 4 bit address is the address of the internal register defined in the codec datasheet (29).

Function: codec_ spi_read

Prototype: unsigned int codec_spi_read(unsigned int address)

Location: “spi.c”

Description: Reads the 10-bit data from the internal configuration registers within the codec. The 4 bit address is the address of the internal register defined in the codec datasheet (29).

Function: codec_setup

Prototype: void codec_setup(void)

Location: “spi.c”

Description: This function initializes the internal registers of the codec to operate in the following mode:

Sampling Frequency: 48KHz

Sampling resolution: 16-bit

Serial mode: I2S

DAC1L and DAC1R channels are configured as on, and DAC(3:2)L and DAC(3:2)R are muted.

The DAC1L and DAC1R volume registers are set to zero attenuation.

ADC1 is configured in differential mode and ADC2 is configured in single ended mode.

ADC1 and ADC2 gain is set to 0dB gain.

The filter is set is set to high pass mode.

Function: setup_pgas

Prototype: void setup_pgas (void)

Location: “spi.c”

Description: This function the initializes PGAs.

Both PGAS are configured to operate in the following mode:

Gain: 40dB
Zero crossing detector enabled.

DC servo enabled.

GPO0: on, i.e. the green LED is switched on.

17.3 Audio Driver

This section describes the operation of the Audio Driver.

Function: audio_setup

Prototype: void audio_setup(void)

Location: "audio.c"

Description: This function initializes the audio driver. It calls the codec initialization, configures the FPGA audio module and sets up the audio module interrupt service routine, which is described next. The FPGA audio module is configured to interrupt when 256 samples have been received.

Function: audio_dma

Prototype: void audio_dma(void)

Location: "audio.c"

Description: This function is the audio module interrupt service routine. It initializes the DMA0 channel to perform a 2 dimensional chained DMA transfer of the audio. The DMA transfer will transfer 4 input audio channels to internal memory from the FPGA. It will then transfer the 2 output audio channel data to the FPGA. It is configured to transfer 256 samples per channel. Once the DMA is complete it will trigger an interrupt. This interrupt is serviced by the audio driver interrupt service routine.

Function: audio_driver

Prototype: void audio_driver(void)

Location: "audio.c"

Description: This function is the audio DMA interrupt service routine. This function is responsible for mapping the audio channels to the output channels. This function would be a good starting point for future work as routines such as a fast Fourier transform could be performed here after the data has audio arrived.

17.4 PIC18F4550 and LCD Asynchronous Interface

Function: init_assync_inteface

Prototype: void init_assync_inteface (void)

Location: “assync_inteface.c”

Description: This function initializes the FPGA asynchronous module with the timing values as defined “FPGA_def.H“ and as described in section 8.1.

Function: write_pic

Prototype: void write_pic(int data)

Location: “assync_inteface.c”

Description: This function writes 8-bit data to the PIC18F4550. Communication to the PIC18F4550 occurs by sending two packets of 8-bit data. The function needs to be called twice. The first call must send the address and the next call must send the data.

Function: read_pic

Prototype: int data read_pic(int address)

Location: “assync_inteface.c”

Description: This function reads 8-bit data from the PIC18F4550 from the internal variable specified by the address.

Function: write_LCD

Prototype: void write_LCD(int data)

Location: “assync_inteface.c”

Description: This function writes 8-bit data the LCD display DRAM.

Function: read_LCD

Prototype: int read_LCD(void)

Location: “assync_inteface.c”

Description: This function reads 8-bit data from the LCD display DRAM.

Function: read_LCD_RS

Prototype: int read_LCD_RS(void)

Location: “assync_inteface.c”

Description: This function reads the LCD busy flag.

Function: write_LCD_RS

Prototype: int write_LCD_RS(int data)

Location: “assync_inteface.c”

Description: This function writes 8-bit data to the LCD’s internal registers.

17.5 Push-Buttons

Function: push_buttons

Prototype: void push_buttons(void)

Location: “audio.c”

Description: This is the interrupt service routine for the push buttons. This function in its current implementation changes the mapping of the audio outputs when a button is pressed.

17.6 Compact Flash Interface and File System

Function: init_cf_80

Prototype: void init_cf_80(void)

Location: “compactflash.c”

Description: This function initializes the FPGA Compact Flash module to operate with a 80ns read and write cycle.

Function: init_cf_250

Prototype: void init_cf_250(void)

Location: “compactflash.c”

Description: This function initializes the FPGA Compact Flash module to operate with a 250ns read and write cycle.

Function: write_cf_common

Prototype: void write_cf_common(int address, int data)

Location: “compactflash.c”

Description: This function writes the data to the specified common memory address within the Compact Flash as described in section 7.1.

Function: read_cf_common

Prototype: int read_cf_common(int address)

Location: “compactflash.c”

Description: This function is reads the data from the specified common memory address within the Compact Flash as described in section 7.1.

Function: write_cf_attr

Prototype: void write_cf_attr(int address, int data)

Location: "compactflash.c"

Description: This function is writes the data to the specified attribute memory address within the Compact Flash as described in section 7.1 (the interface must be in 250ns mode).

Function: read_cf_attr

Prototype: int read_cf_common(int address)

Location: "compactflash.c"

Description: This function is reads the data from the specified attribute memory address within the Compact Flash as described in section 7.1 (the interface must be in 250ns mode).

File System

An implementation of a FAT32 file system that is available from Analog Devices has been used in this project. The file system provides the following standard ANSI C file commands that are located in "filelib.c":

- fopen()
- fread()
- fwrite()
- fseek()
- fclose()
- frename()
- fdelete()
- fsearch()
- fcloselist()
- fcreatedir()

File system reads are sector based, and the current 512 byte sector is buffered internally. Since the TigerSHARC does not support byte operations in the external SDRAM, it is necessary to place the sector buffer (called "buffers") in the internal memory. The buffer is defined in the "global.h" header file and is place in the "data1" internal memory location. The file system also keeps a copy of the FAT table in memory. This buffer, called "fat_chain" is placed in the SDRAM bank 0. The buffer is defined in "fat32base.c".

The file system hardware layer is located in "ide_access.c". The Compact Flash readsectors() command (see extract in Appendix D) can transfer up to 256 sectors to the TigerSHARC. The file system, however, reads a single sector each time and the maximum transfer using this method is only 2.5MB/s. The interface was improved by reading all 256 sectors from the Compact Flash and placing them in a buffer. The hardware layer is modified to first check whether the desired sector is located within this buffer. If it is not then the required sector and the following 255 sectors are transferred into the buffer. This results in an improved transfer rate of 12.5MB/s for reads that involve consecutive sectors.

The following two functions are used by the file system to transfer data between the Compact Flash and buffers.

Function: fnIDE_Buffer256Sector

Prototype: int fnIDE_Buffer256Sector(WORD *buf,DWORD LBALocation)

Location: "ide_access.c"

Description:	This function first checks to see if the required 512 byte sector is within the 256x512 byte internal buffer, if not, then the required sector and the following 255 sectors are transferred into the internal buffer. The required sector is transferred into the buffer which is pointed to by *buf. This results in an improved transfer rate of 12.5MB/s. Transfers between the FPGA and the buffer occur via DMA to improve the data throughput. DMA3 is used for this purpose. The logic block address (LBA) is used by the file system to define the location of the sector. This function will return a non-zero value if an error has occurred.

Function: fnIDE_WriteBufferSector

Prototype: int fnIDE_WriteBufferSector(WORD *buf,DWORD LBALocation,BYTE count)

Location: “ide_access.c”

Description: This function writes the internal sector to the Compact Flash. It has not been optimized. The logic block address (LBA) is used by the file system to define the location of the sector. “count” is the number of sectors to be transferred (max=256). However the file system only transfers one sector at a time. This needs to be improved for future work.This function will return a non-zero value if an error has occurred.

Function: fnSystemInit()

Prototype: void fnSystemInit(void)

Location: “filelib.c”

Description: This function initializes the file system by loading the Compact Flash parameters and file allocation table from the Compact Flash.

17.7 TigerSHARC and SDRAM Initialization

Function: inita

Prototype: void inita(void)

Location: “inita.c”

Description: This function initializes the TigerSHARC system configuration (SYSCON) and the SDRAM configuration (SDRCON) registers as described in chapter 13.

Function: TEST_SDRAM0

Prototype: int TEST_SDRAM0(void)	
Location: “SDRAM_test.c”	
Description:	This function tests the SDRAM bank 0 SDRAM module as follows: Write incrementing values and then verify incrementing values at each address. Write “FFFFFFFF” (all bits high) and then verify ”FFFFFFFF” at each address. Write “AAAAAAAA” ” (bits alternate high-low) and then verify ”AAAAAAAA” at each address. Write “55555555” (bits alternate low-high) and then verify “55555555” at each address.

Function: TEST_SDRAM1

Prototype: int TEST_SDRAM1(void)	
Location: “SDRAM_test.c”	
Description:	This function tests the SDRAM bank 1 SDRAM module in the same manner as TEST_SDRAM0.

17.8 USB Keyboard and Mouse

The PIC18F4550 is used to act as a composite USB device that in turn implements a USB human interface device (HID) keyboard and an HID mouse (30). The mouse HID class driver, which is included onboard the PIC18F4550, sends a three byte report descriptor via the USB protocol to the PC containing the relative position and button information for a mouse. The report descriptor is illustrated in Table 17-1. Column 1 is the byte number, column 2 is the bit field, column 3 is the “C” variable name within the PIC18F4550 code and column 4 gives a description of the data.

Byte	Bit	Variable name	Description	
0	0	Mouse_buffer0	Button 1	‘1’=depressed
	1		Button 2	‘1’=depressed
	2		Button 3	‘1’=depressed
	3 to 7		Reserved	
1	0 to 7	Mouse_buffer1	X displacement {-127,+127}	
2	0 to 7	Mouse_buffer2	Y displacement {-127,+127}	

Table 17-1: HID Mouse Report Descriptor.

The keyboard HID class driver, which is included onboard the PIC18F4550, sends an 8 byte report descriptor via the USB protocol to the PC containing the relative information for a keyboard. The report descriptor is illustrated in Table 17-2. Column 1 is the byte number, column 2 is the “C” variable name and column 3 gives a description of the data.

Byte	Variable name	Description
0	Keyboad_buffer[0]	Modifier keys
1	Keyboad_buffer[1]	Reserved
2	Keyboad_buffer[2]	Key Code1
3	Keyboad_buffer[3]	Key Code2
4	Keyboad_buffer[4]	Key Code3
5	Keyboad_buffer[5]	Key Code4
6	Keyboad_buffer[6]	Key Code5
7	Keyboad_buffer[7]	Key Code6

Table 17-2: HID Keyboard Descriptor.

Once the mouse and keyboard report descriptor have been updated, it is necessary to tell the USB driver that there is new data that can be transmitted to the PC. This is done by using the following two functions:

Function: Emulate_Keyboard

Prototype: void Emulate_Keyboard(void)	
Location: "usermouse.c" in microchip C code.	
Description:	This function tells the USB driver that there is new data keyboard available to transmit to the PC. The USB driver will then transmit the keyboard report descriptor to the PC.

Function: Emulate_Mouse

Prototype: void Emulate_Mouse(void)	
Location: "usermouse.c" in microchip C code.	
Description:	This function tells the USB driver that there is new mouse data available to transmit to the PC. The USB driver will then transmit the mouse report descriptor to the PC.

The function **assync_interface**() located in "Main.c" in the microchip code is used to receive data and commands from the TigerSHARC and to send data back to the TigerSHARC. The communication between the TigerSHARC occurs in two byte packets. The first is the command and the second is the data byte. Table 17-3 illustrates the decoding of each command.

R/W	Command	Description
W	0xff	Disable Mouse and Keyboard communication
W	0x11	Disable Mouse communication
W	0x22	Enable Mouse communication
W	0x33	Enable Keyboard communication
W	0x44	Disable Keyboard communication
W	0x0	Load keyboard_buffer[0] with data packet
W	0x1	Load keyboard_buffer[1] with data packet
W	0x2	Load keyboard_buffer[2] with data packet
W	0x3	Load keyboard_buffer[3] with data packet
W	0x4	Load keyboard_buffer[4] with data packet
W	0x5	Load keyboard_buffer[5] with data packet
W	0x6	Load keyboard_buffer[6] with data packet
W	0x7	Load keyboard_buffer[7] with data packet
W	0x8	Execute Emulate_Keyboard();
W	0x50	Load mouse_buffer[0] with data packet
W	0x51	Load mouse_buffer[1] with data packet
W	0x52	Load mouse_buffer[2] with data packet
W	0x58	Execute Emulate_Mouse();
R	0xf6	Send 0xA5 to TigerSHARC
R	0xf6	Send 0x5A o TigerSHARC

Table 17-3: The commands that are decoded by the PIC18F4550 assync_interface() funtion.

The following functions that are located in TigerSHARC "C" code file "assync_interface.c" and can be used by the TigerSHARC to send data to the mouse and keyboard (these functions use the read_pic() and write_pic() functions). Only the function prototypes are given below:

```
void update_keyboard_0(int key)
void update_keyboard_2(int key)
void update_keyboard_3(int key)
void update_keyboard_4(int key)
void update_keyboard_5(int key)
void update_keyboard_6(int key)
void update_keyboard_7(int key)
void transmit_keyboard(void)
void update_mouse(int buttons, int dx, int dy)
```

Function: BlinkUSBStatus

Prototype: void BlinkUSBStatus (void)

Location: “usermouse.c” in microchip “C” code.

Description: This function uses the PIC18F4550 LEDs 9 and 10 to display the status of the USB as follows:

Both LEDS off:	The device is detached.
LED10 on, LED9 off:	The device is powered.
LED10off, LED9 on:	The device is in the configuration state.
LED10 flashing, LED9 off:	The device is in the address state.
LED10 flashing, LED9 flashing:	The device is configured and operating correctly.

17.9 Prototype Demonstration Software

Figure 17-1 illustrates a flow diagram of the demonstration software that executes inside the TigerSHARC. The software demonstrates the correct working of every component on the prototype PCB and a description of the flow diagram is given after the figure. The software plays wav files that are stored on the Compact Flash, writes out audio bar graphs to the LCD and writes data to the USB keyboard and mouse. By pressing the push buttons the different audio input channels are mapped to audio line output. A detailed description of the flow diagram is given after the figure.

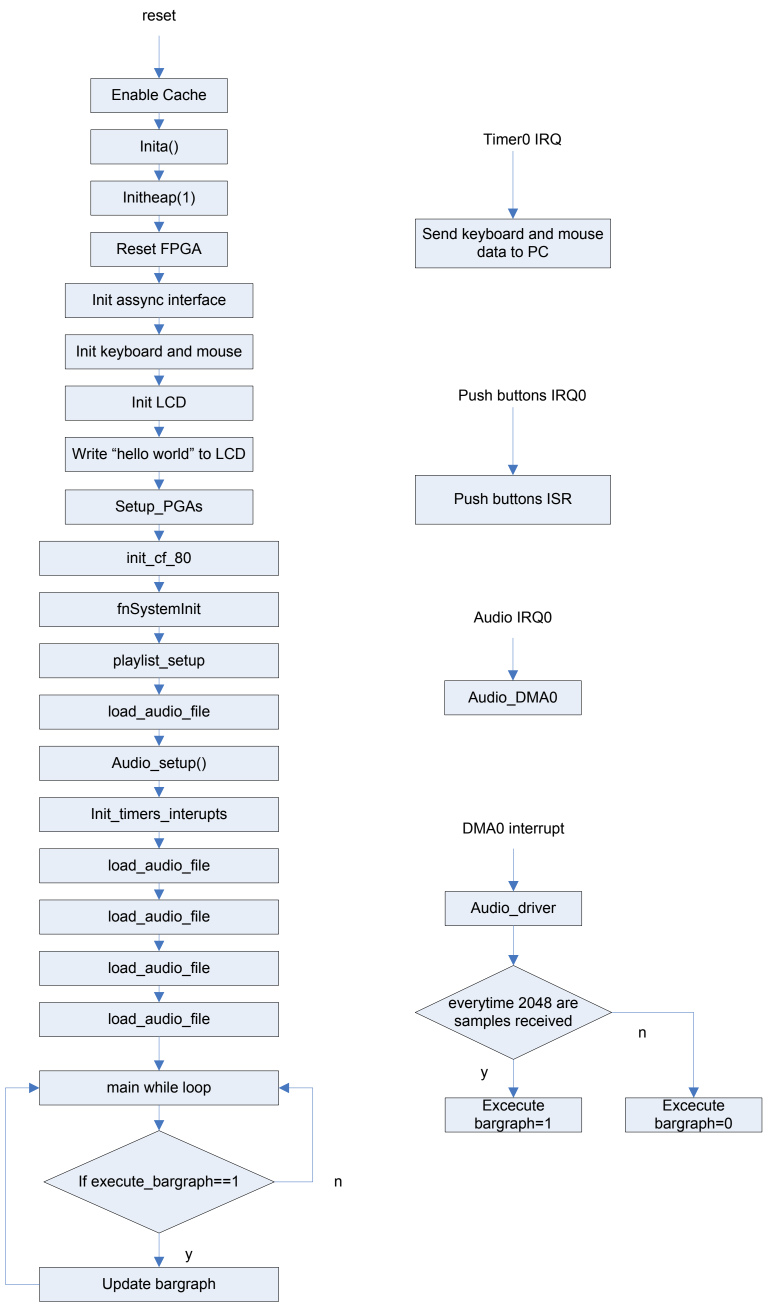


Figure 17-1: Flow diagram of the prototype software demonstration.

Main routine:

1. After a system reset, the TigerSHARC's cache is enabled.
2. The SDRAM and SYSTEM configuration registers are programmed.
3. The heap in side SDRAM Bank 1 is initialized.
4. The FPGA is reset.
5. The asynchronous interface is initialized.
6. The USB keyboard and mouse is initialized.
7. The LCD is initialized.
8. The PGAs are initialized and the PGA green LEDs will switch on.
9. The Compact Flash interface is initialized. Next, the file system is initialized. Next, the wave playlist is initialized.
10. An audio file is loaded into the heap in SDRAM bank 1 and inserted into the playlist.
11. The audio system is initialized.
12. The audio file will now start to play.
13. The timer 0 is enabled.
14. Another four audio files are loaded into the heap and are inserted into the playlist.
15. Once this is complete a bar graph will start to display on the LCD screen.

Timer0 IRQ interrupt service routine:

The timer0 interrupts every 100ms. It then runs a routine which moves the mouse in a circle and sends keyboard characters to the PC.

Push-buttons interrupt service routine:

1. When the SW5 is pressed then audio will skip to the next wave file.
2. When SW6 is pressed the microphone channels will be mapped to the audio output channels.
3. When SW2 is pressed the wave file will continue playing.
4. When SW3 is pressed the line audio channels will be mapped to the line out channels.

Audio IRQ0 interrupt service routine:

The audio IRQ0 interrupt is configured to initiate a DMA transfer of the audio samples between the FPGA and TigerSHARC.

DMA0 interrupt service routine:

The DMA0 interrupt service routine calls the audio driver which is responsible for the mapping of the audio channels to the line out and for playing the wave files. After every 2048 samples that are received, the audio driver triggers a flag. This flag in turn tells the bar graph routine that it must update the bar graph on the LCD

17.10 Summary and Conclusion

This chapter describes the software routines and drivers that have been written for our design.

Routines are provided for communication with the FPGA and for communication with the SPI bus to access the audio codec's configuration registers and the PGA's control register.

DMA based routines are provided for the bulk transfer of audio data. Routines are presented for communication with the PIC18F4550 (USB interface), the LCD, the push buttons, as well for interfacing with the Compact Flash FAT32 file system.

Finally, testing software is described which verifies the correct working of all the components in our system and is therefore the final component required for our design to be a success.

18 Processor Performance

This chapter compares the performance of the Analog Devices TigerSHARC TS201 600MHz processor to well known Intel processors.

To do this, use is made of the diagonal covariance Gaussian probability density function (PDF) which is typically used in speech processing applications. The log-likelihood of the diagonal covariance Gaussian PDF is defined as follows (31):

$$\ln p(x|\mu,\sigma^2) = -\frac{1}{2}\sum_{i=1}^{D}\frac{(x_i-\mu_i)^2}{\sigma_i^2} - \frac{D}{2}\ln(2\pi) - \frac{1}{2}\sum_{i=1}^{D}\ln(\sigma_i^2) \qquad (18\text{-}1)$$

Where D is the dimension, x is a $D \times 1$ feature vector, μ is the $D \times 1$ vector variances corresponding to the diagonal of the covariance matrix.

The computation of (18-1) can be reduced to

$$\ln p(x|\mu,\sigma^2) = \sum_{i=1}^{D}(x_i-\mu_i)^2 \times y_i + C \qquad (18\text{-}2)$$

Where

$$y_i = -\frac{1}{2}\sigma_i^{-2}$$

And $C = -\frac{D}{2}\ln(2\pi) - \frac{1}{2}\sum_{i=1}^{D}\ln(\sigma_i^2)$

The values of y_i and C can be pre-computed and do not need to be determined at run time, thereby reducing the computational complexity.

Equation (7) is used to benchmark the performance of the TigerSHARC processor in the following manner. The log-likelihood of 200 39-dimensional feature vectors (D=39) is calculated. This computation is repeated 1 million times. The average time to calculate the log-likelihood of a single 39 dimensional feature is then determined and used to compare the performance of the TigerSHARC 600MHz processor to an Intel 1.8GHz Core2Duo processor running Linux, a 700MHz Intel Pentium 4 running Linux and a 3GHz Intel Pentium D also running Linux. An extract of the C code used to benchmark the processors is illustrated in the box below (the full code is included in the DVD):

```
float test_gaussian(void){
        int i;
        int n;
        float ans=0;
        float temp=0;
        for (n=0; n<200;n++){
                for(i=0;i<39;i++){
                        temp=(x[n][i]-u[i]);
                        temp=temp*temp;
                        temp=temp*y[i];
                        ans+=temp;
                }
                ans+=C;
        }
        return ans;
}
```

The C code is compiled using the GCC compiler for the Intel processors and the results are listed in Table 18-1.

Processor	**Time [ns]**
700MHz Pentium 4	384ns
3GHz Pentium D	101.5ns
Intel 1.8GHz Core2Duo processor	89 ns
Analog Devices TigerSHARC TS201 600MHz	98.86 ns

Table 18-1: Benchmark of the processors.

As is illustrated in Table 18-1, the Analog Devices TigerSHARC TS201 processor running at 600MHz performs very similarly to the Intel 1.8 GHz Core2Duo processor.

18.1 Summary and Conclusion

This chapter is utilized to demonstrate the computational power of the TigerSHARC processor. A benchmark using multi-dimensional Gaussian log-likelihood computations, illustrated that the TigerSHARC 600MHz CPU performs similarly to an Intel Core2Duo processor operating at 1.8GHz. The TigerSHARC processor is therefore a very powerful computational unit which will suit speech recognition applications.

19 Conclusion

Our system successfully meets every aspect of the system requirements set out in Chapter 1, and incorporates the Analog Devices TigerSHARC TS201 600MHz digital signal processor to provide a high performance platform which has been shown to have computational performance comparable to an Intel Pentium 1.8GHz Core2Duo system. Our system therefore provides an excellent platform upon which speech recognition algorithms can be implemented.

The design integrates 512 MB RAM, which is expandable to 1024MB. It can support up to 32GB Compact Flash storage card, which is the largest size supported by the FAT32 file system.

Multi-channel audio input and output is provided, including two microphone channels with up to 65dB programmable gain each. The multiple audio channels allow the future implementation of techniques such as noise cancellation and beam-forming to improve the signal-to-noise ratio of the audio inputs.

The system demonstrates the ability to communicate with a PC via USB by successfully emulating a composite USB mouse and keyboard. The system is however not limited to this interface method, since the extra IO available on the FPGA could be used to control and interface to other electronic devices.

The efficient use of the FPGA's internal memory as FIFOs allows the processor to perform data transfers in blocks via DMA, which reduces the time the processor will have to spend waiting for slow IO devices such as the LCD, audio and USB, and will therefore allow the system to spend maximum time performing speech recognition computations.

All components of the prototype system have been verified to be working correctly. The system is therefore ready to be used as a vehicle for speech and audio research, which was the overall initial aim.

20 References

1. *A 1000-Word Vocabulary, Speaker-Independent,Continuous Live-Mode Speech Recognizer Implemented in a Single FPGA.* Lin, Edward C., et al. New York : ACM, 2007. 978-1-59593-600-4.

2. Microchip. dsPIC30F Speach Recogintion Libraries User's Guide. 2004.

3. Analog Devices. *Analog Devices TigerSHARC Emebedded Processor ADSP-TS201S.* 2005.

4. Analog Devices. *VisualDSP++5.0 Linker and Utilities Manual.* 2007.

5. Analog Devices. *Engineer to Engineer note EE-68: Analog Devices JTAG Emulation Technical Reference.* 2004.

6. Analog Devices. *Engineer to Engineer Note EE182: Thermal Relief Design for ADSP-TS201S TigerSHARC Processors.* 2004.

7. AAVID THERMALLOY. [Online] 2007. http://www.aavidthermalloy.com/.

8. Altera. Cyclone II Device Handbook, Volume 1. 2006.

9. Xillinx. Spartan-3E FPGA Family: Complete Data Sheet. 2006.

10. Texas Instruments. *PGA2500 Digitally Controlled Microphone Preamplifier.*

11. Compact Flash Association. www.compactflash.org. *Compact Flash Specification 4.0.* 2006.

12. PowerTip Technology Corporation. *PC2004-B.* [Online] http://www.powertip.com.tw/.

13. Maxim. *+1.2V to 5.5V,+-15kV ESD-Protected, 0.1uA, 35Mbps, 8-Channel Level Translators.* 2005.

14. Kingston. *Value Ram Memory Module Specification KVR133X72RC3L/256.* 2002.

15. Nanya. *256Mb Synchronous DRAM.* 2006.

16. Johnson, Howard and Graham, Martin. *High-Speed Digital Design: A Handbook of Black Magic.* New Jersey : Prentice Hall, 1993. 0-13-395724-1.

17. JEDEC. JESD21-C PC133 Registered DIMM Specification. [Online] 2007.

18. Texas Instruments. *PTH12000W/L 12-V Input.* 2004.

19. Analog Devices. *Engineer-to-Engineer Note EE-179: ADSP-TS20xS TigerSHARC System Design Guidelines.* 2006.

20. Texas Instruments. TPS77801Fast Transient Low Dropout Regulators. 2004.

21. Texas Instruments. *TLV1117 Adjustable and Fixed Low-Dropout Voltage Regulator.* 2006.

22. General Semiconductor. *US1A THRU US1M.* 2006.

23. Texas Instruments. *PTN78000A 1.5A, WIDE-INPUT ADJUSTABLE BUCK-BOOST SWITCHING REGULATOR.* 2006.

24. Texas Instruments. *uA79M00 Series Negative Voltage Regulators.* 2005.

25. ST . *STPS745G Power Schottky Rectifier.* 2003.

26. TDK. *Common Mode Filters(SMD) For Power Line.* 2006.

27. MAXIM. *Ultra-Small, Adjustable Sequencing/Supervisory Circuits.* 2006.

28. TRAX. Conductor Sizing Curves. [Online] www.trax.co.za.

29. Analog Devices. *AD1836A Multichannel 96kHz Codec.* 2003.

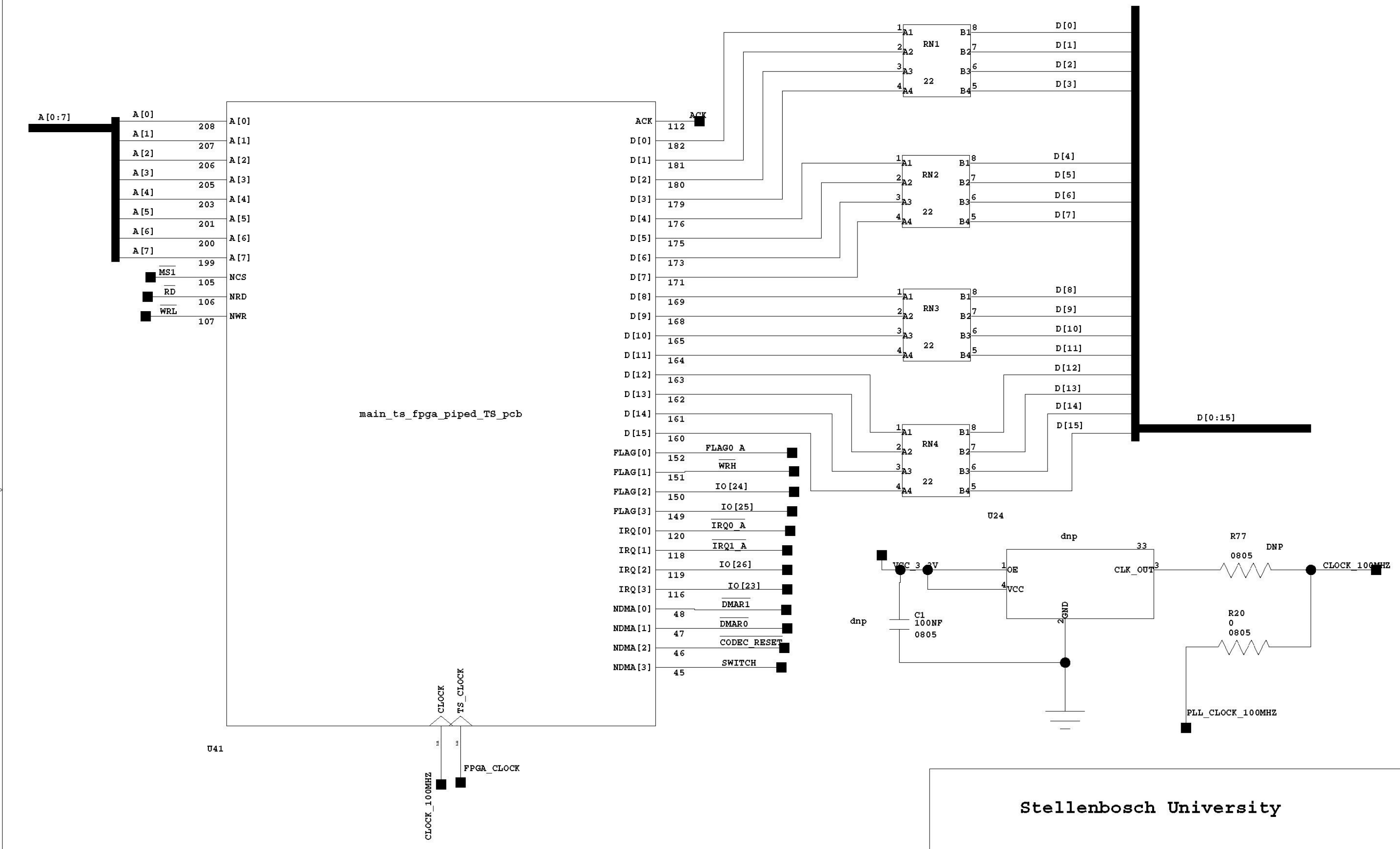
30. USB Implementers' Forum. *Universal Serial Bus Device Class Definition for Human Interface Devices (HID).* 2001.

31. Ben Gold, Nelson Morgan. *SPEECH AND AUDIO SIGNAL PROCESSING Processing and Perception of Speech and Music.* New York : John Wiley & Sons, 2000. ISBN 0-471-35154-7.

Appendix A

Schematics

The schematic layouts are displayed in the following pages. A higher resolution PDF and the original project files are included on the DVD.

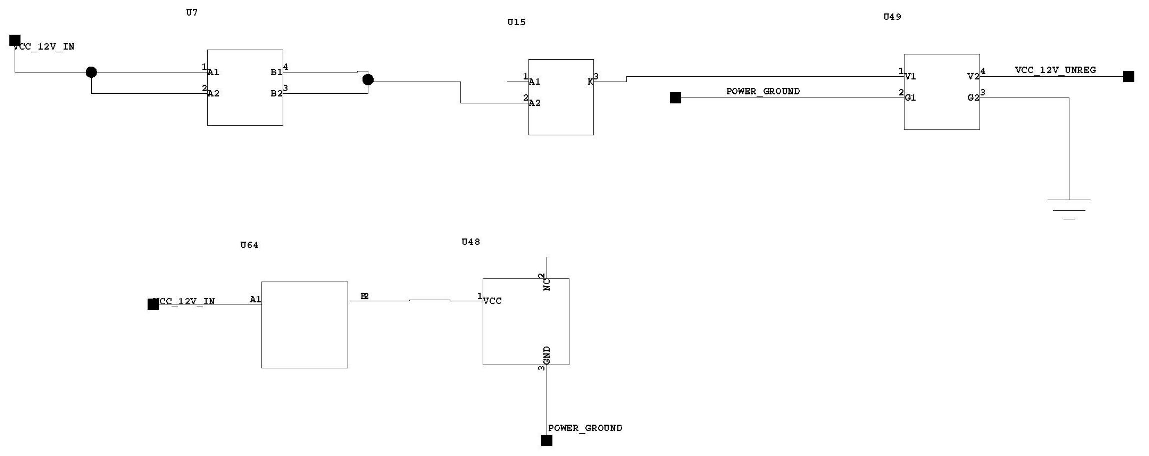


Stellenbosch University

TITLE
TigerSHARC to FPGA interface

SIZE B DWG NO REV 0

DRAWN BY WILLIAM DUCKITT SCALE SHEET 1 of 29 12-9-2007



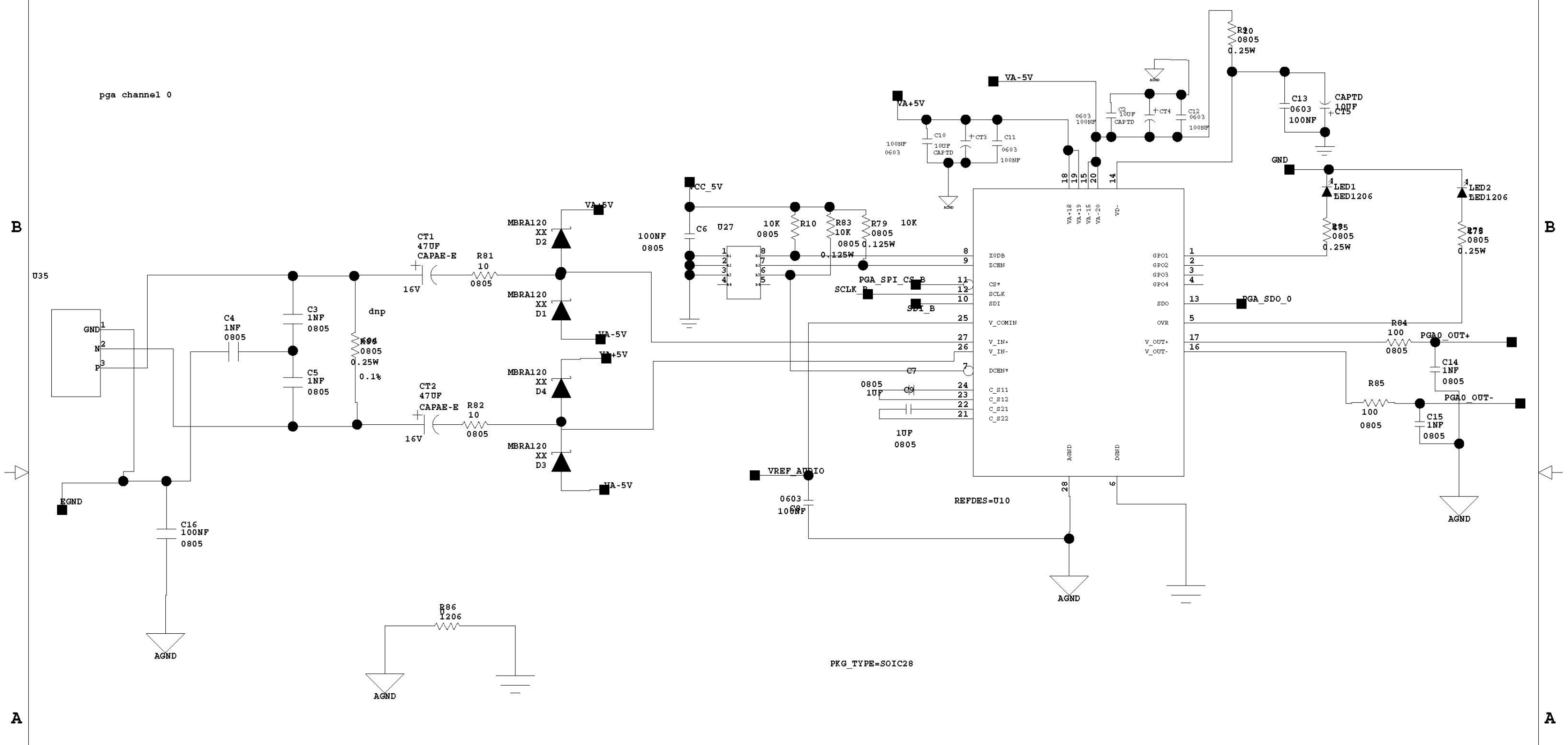
Stellenbosch University

TITLE

Power Connections

SIZE	DWG NO	REV
B		0

DRAWN BY	SCALE	SHEET		of	
WILLIAM DUCKITT		2	of	29	12-9-2007



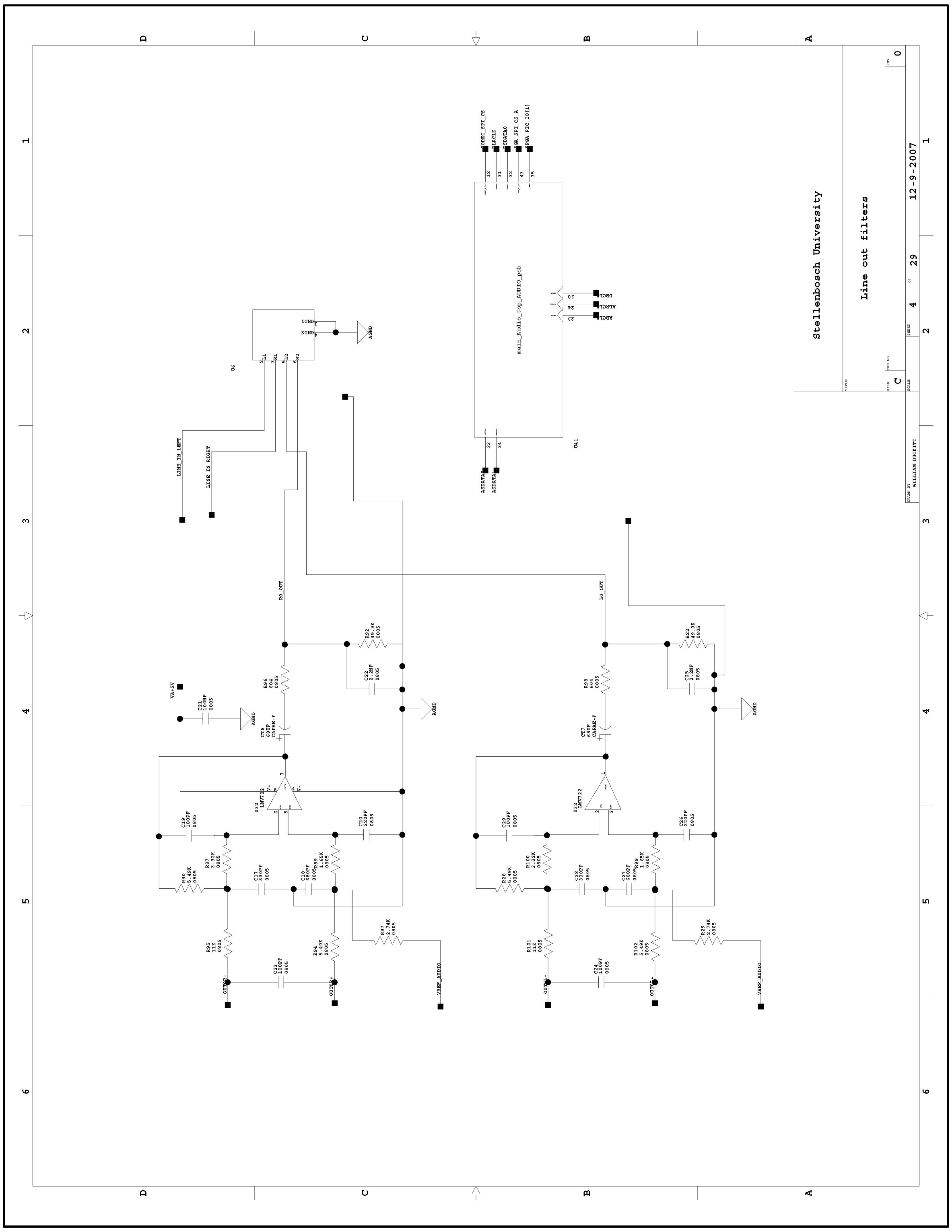
Stellenbosch University

TITLE

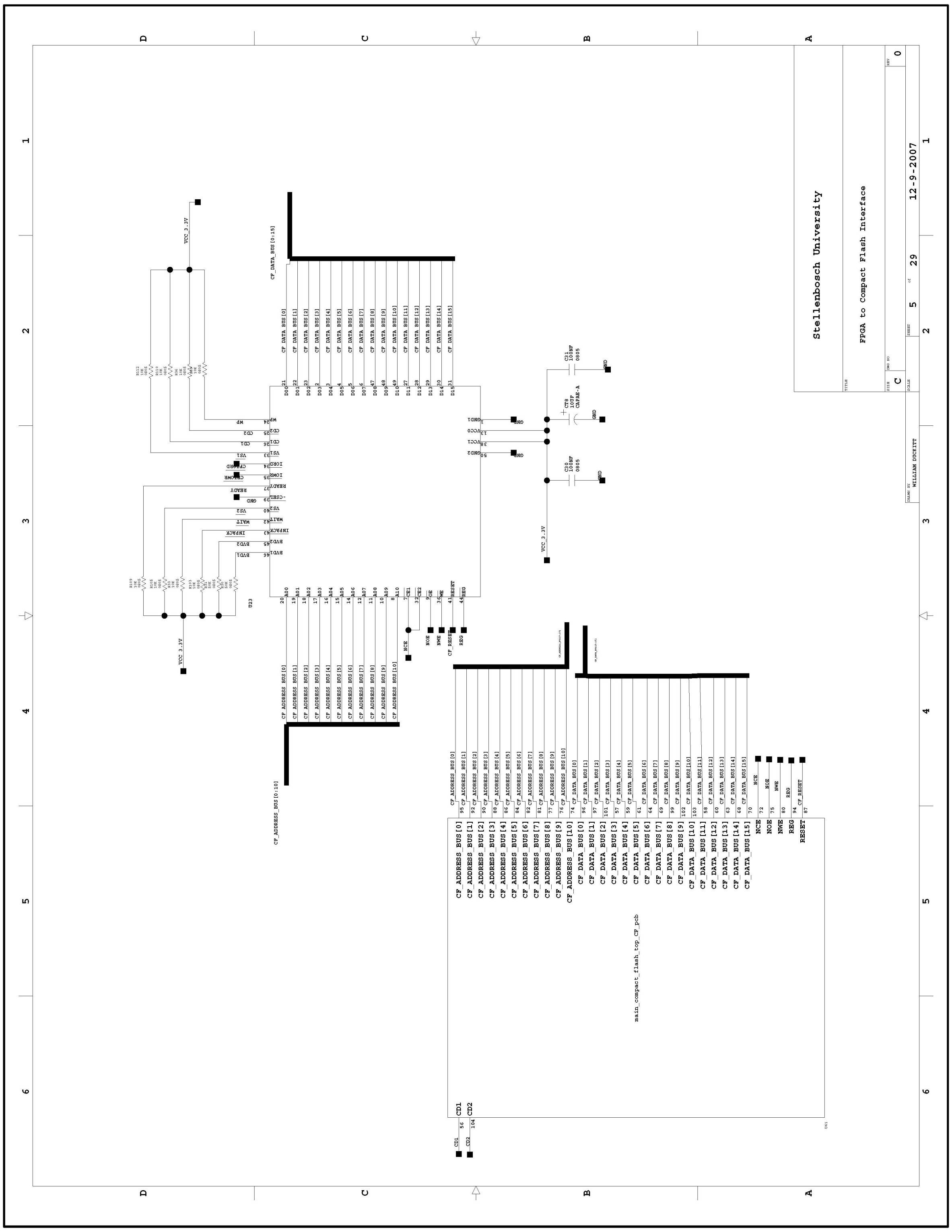
PGA Channel 0

SIZE	DWG NO	REV
B		0

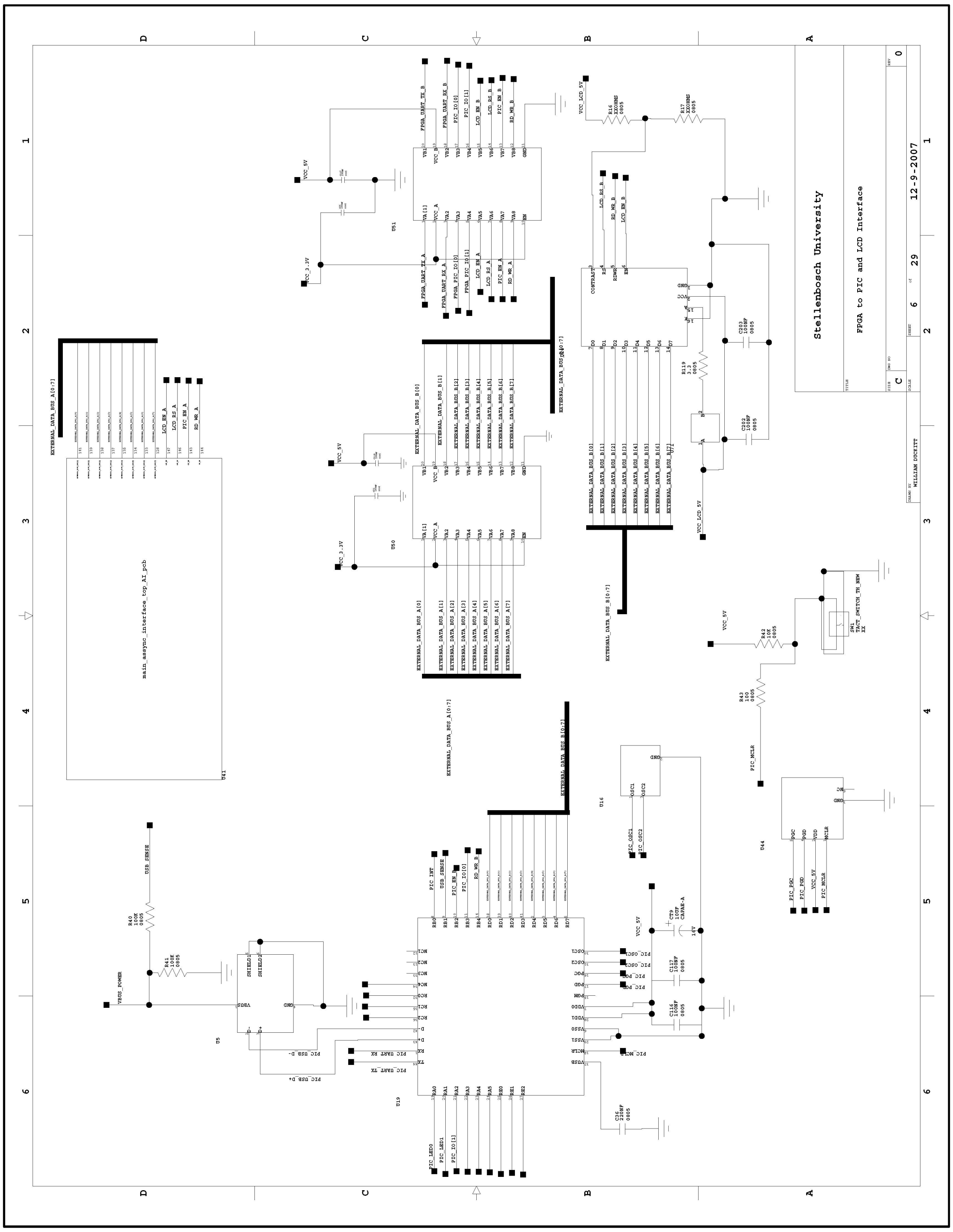
DRAWN BY	SCALE	SHEET	of	
WILLIAM DUCKITT		3	29	12-9-2007

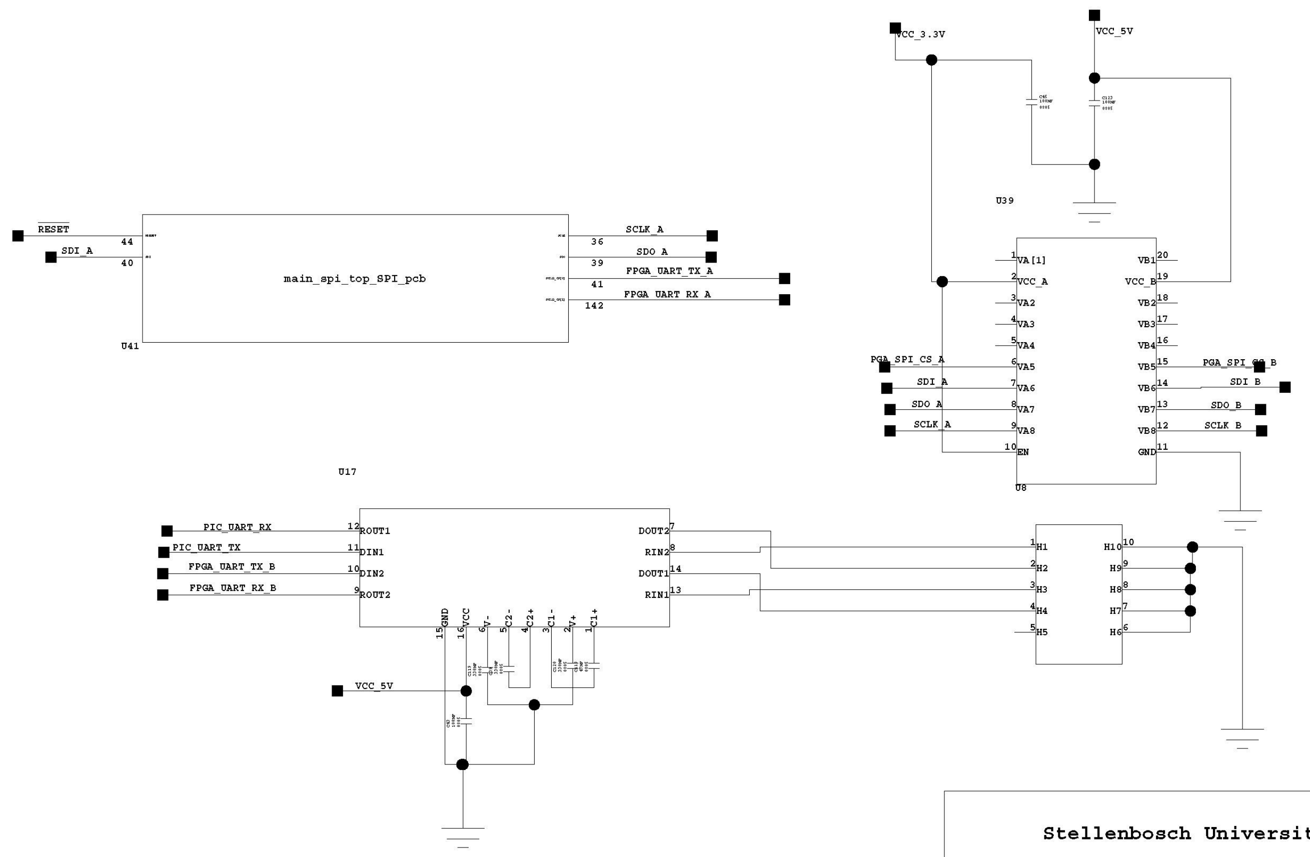


Stellenbosch University
TITLE
Line out filters
SIZE C
DWG NO
REV 0
SCALE
SHEET 4 of 29
12-9-2007
DRAWN BY WILLIAM DUCKITT
main_Audio_top_AUDIO_pcb
U41
U6
GND1
GND2
AGND
LINE_IN_LEFT
LINE_IN_RIGHT
RO_OUT
LO_OUT
VA+5V
LMV722
VREF_AUDIO
ASDATA
DBCLK
ALRCLK
ABCLK
CODEC_SPI_CS
LRCLK
SDATA0
PGA_SPI_CS_A
PGA_PIC_IO[1]
R92 49.9K 0805
R22 49.9K 0805
C22 2.2NF 0805
C25 2.2NF 0805
R96 604 0805
R98 604 0805
CT6 68UF CAPAE-P
CT7 68UF CAPAE-P
C21 100NF 0805
C19 100PF 0805
C29 100PF 0805
C20 220PF 0805
C26 220PF 0805
C17 3300PF 0805
C28 3300PF 0805
C18 680PF 0805
C27 680PF 0805
R90 5.49K 0805
R28 5.49K 0805
R97 2.74K 0805
R29 2.74K 0805
R95 11K 0805
R101 11K 0805
R94 5.49K 0805
R102 5.49K 0805
C23 100PF 0805
C24 100PF 0805

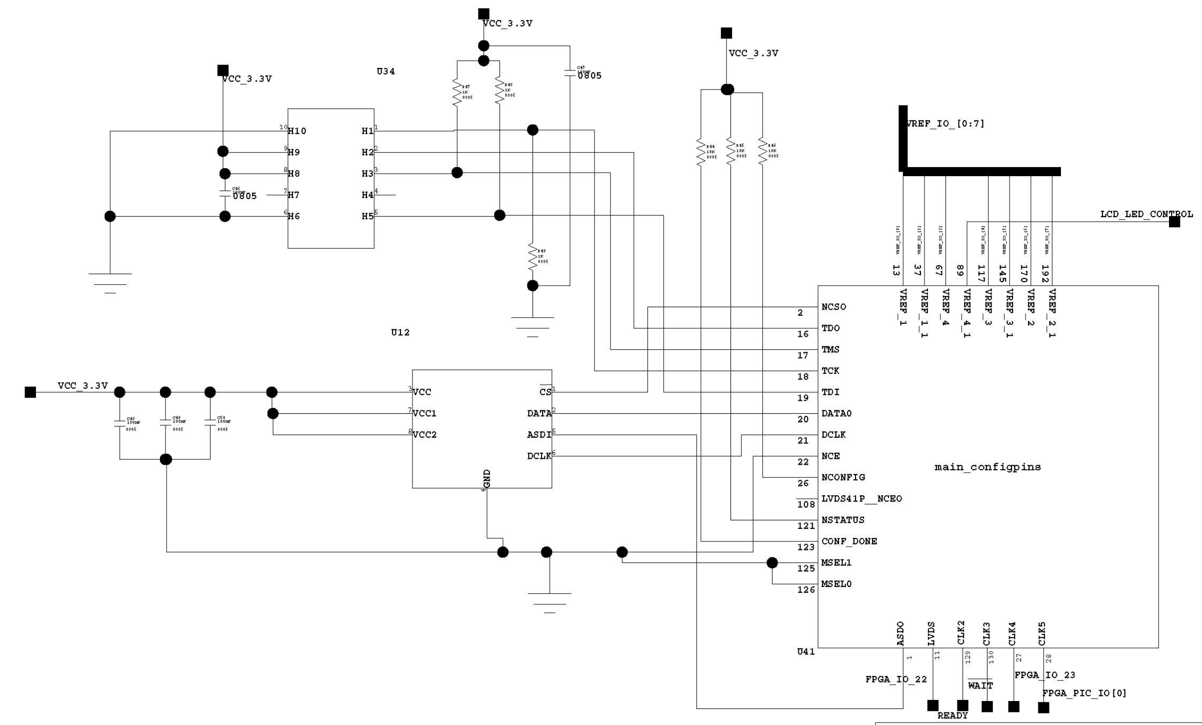


Stellenbosch University
FPGA to Compact Flash Interface
12-9-2007
WILLIAM DUCKITT
5 of 29
C
0
CF_DATA_BUS[0:15]
CF_ADDRESS_BUS[0:10]
VCC_3.3V
GND
main_compact_flash_top_CF_pcb
CD1
CD2
NCE
NOE
NWE
REG
RESET
CF_RESET
WP
VS1
VS2
IORD
IOWR
READY
WAIT
INPACK
BVD1
BVD2
-CSEL
U23
C10 100NF 0805
C11 100NF 0805
C78 10UF CAPAE-A


Stellenbosch University
FPGA to PIC and LCD Interface
12-9-2007
6 of 29
WILLIAM DUCKITT
main_async_interface_top_A1_pcb
EXTERNAL_DATA_BUS_A[0:7]
EXTERNAL_DATA_BUS_B[0:7]
TACT_SWITCH_TH_NEW
VCC_5V
VCC_3.3V
VCC_LCD_5V
VBUS_POWER
USB_SENSE
PIC_MCLR
U19
U50
U51
U44
U5



Stellenbosch University					
TITLE	SPI Level Translation				
SIZE B	DWG NO				REV 0
DRAWN BY WILLIAM DUCKITT	SCALE	SHEET 7	of	29	12-9-2007



Stellenbosch University

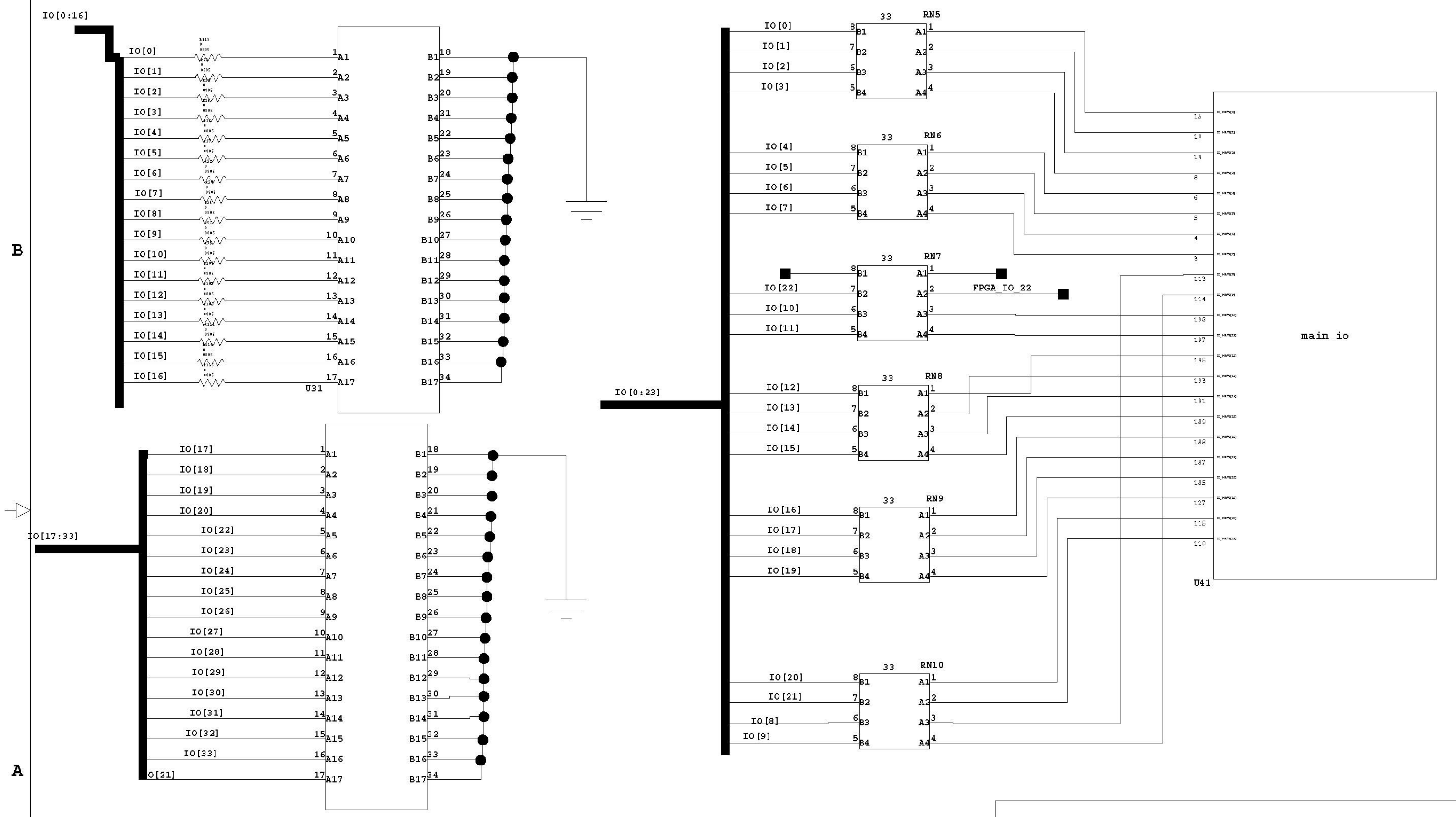
TITLE

FPGA CONFIGURATION

SIZE B DWG NO REV 0

DRAWN BY WILLIAM DUCKITT

SCALE SHEET 8 of 29 12-9-2007



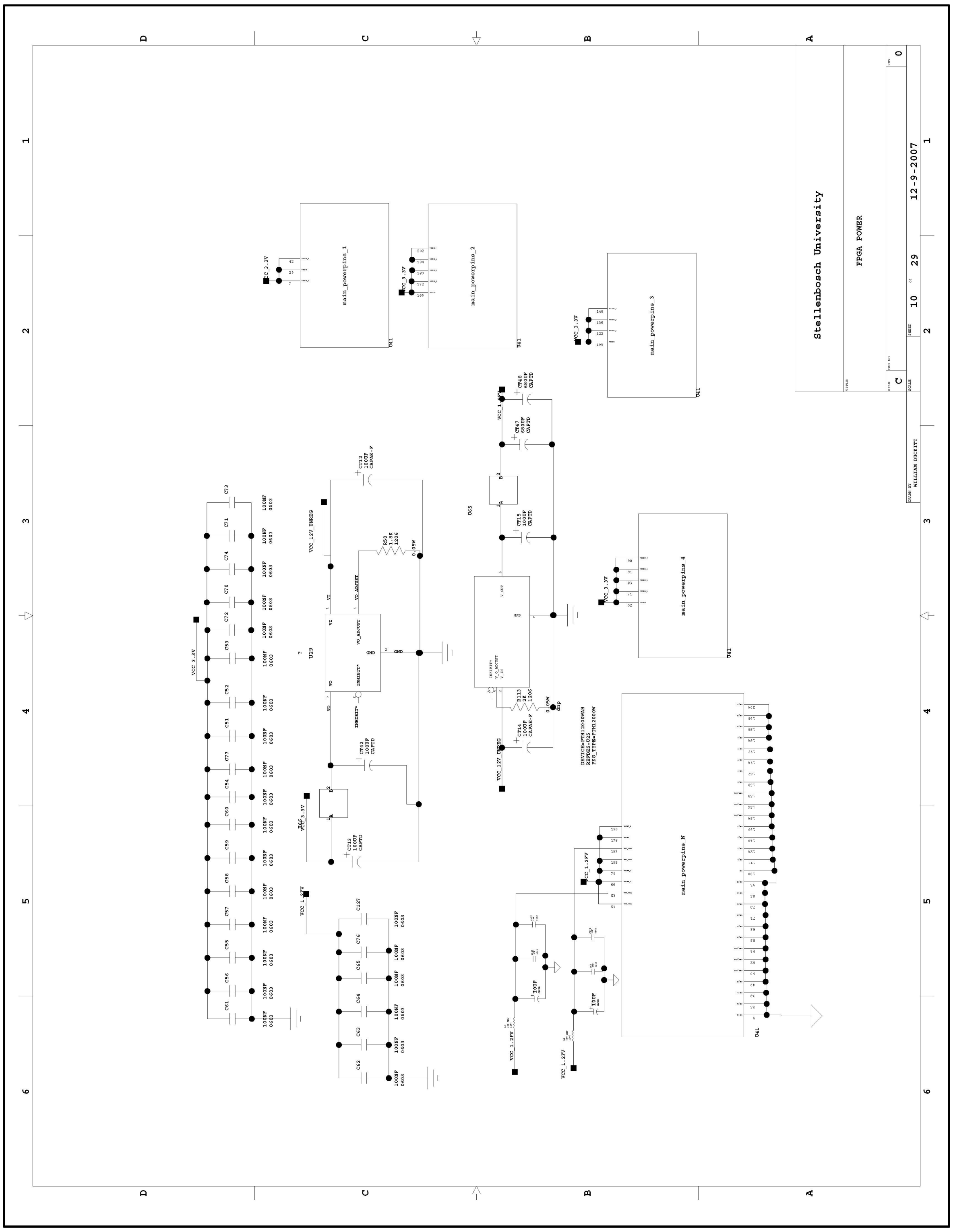
Stellenbosch University

TITLE

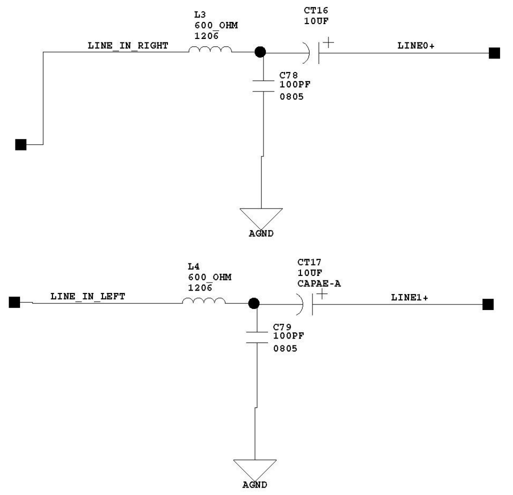
LOGIC ANALYZER IO

SIZE	DWG NO	REV
B		0

DRAWN BY	SCALE	SHEET		
WILLIAM DUCKITT		9 of 29		12-9-2007



Stellenbosch University
FPGA POWER
12-9-2007
10 of 29
WILLIAM DUCKITT
main_powerpins_1
main_powerpins_2
main_powerpins_3
main_powerpins_4
main_powerpins_N
U41
U29
U65
VCC_3.3V
VCC_1.2FV
VCC_12V_UNREG
VO_ADJUST
INHIBIT
GND
100NF 0603
DEVICE=PTH12000WAH

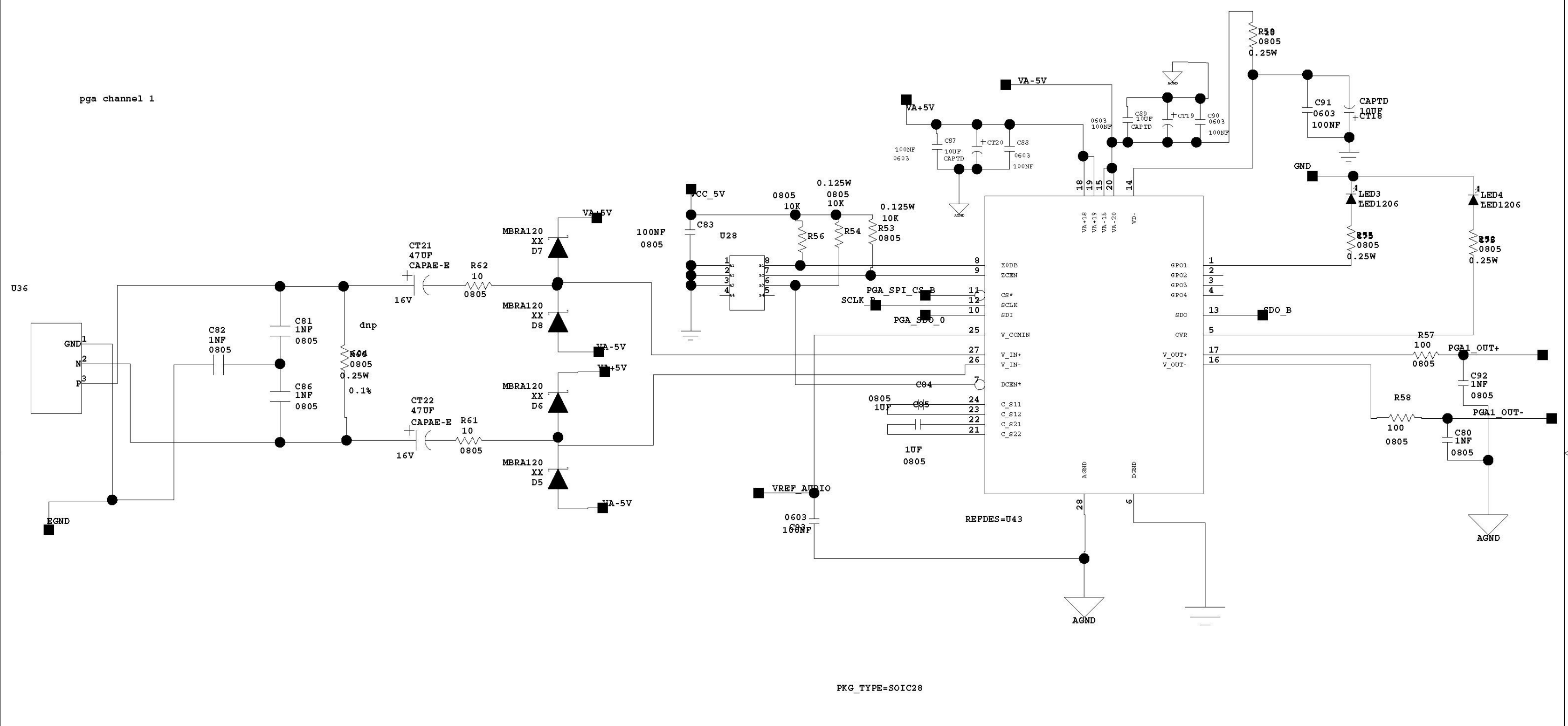


Stellenbosch University

TITLE AUDIO LINE IN

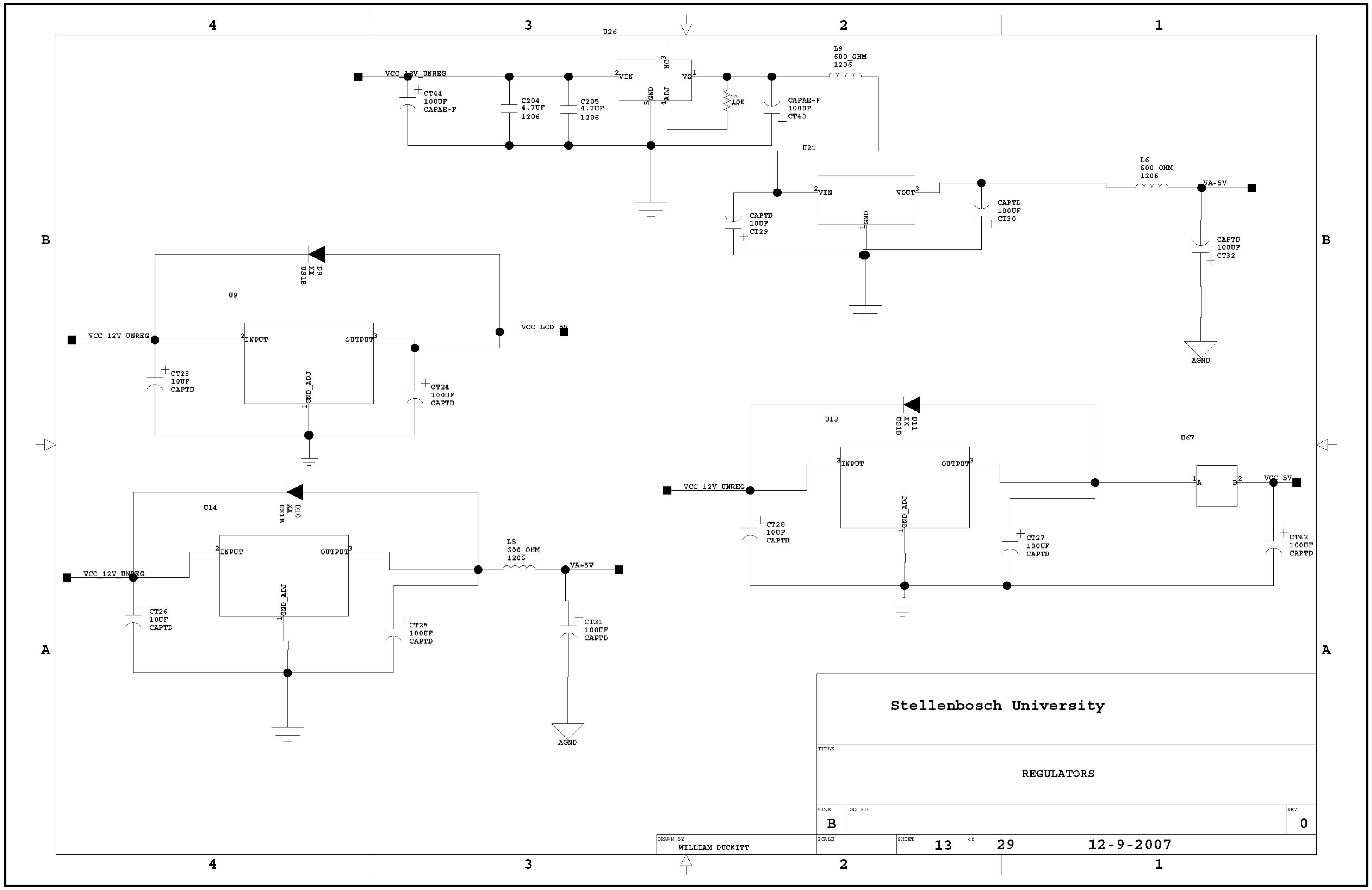
SIZE B	DWG NO	REV 0
SCALE	SHEET 11 of 29	12-9-2007

DRAWN BY WILLIAM DUCKITT



Stellenbosch University

TITLE	PGA CHANNEL 16		
SIZE B	DWG NO		REV 0
DRAWN BY WILLIAM DUCKITT	SCALE	SHEET 12 of 29	12-9-2007



Stellenbosch University

TITLE

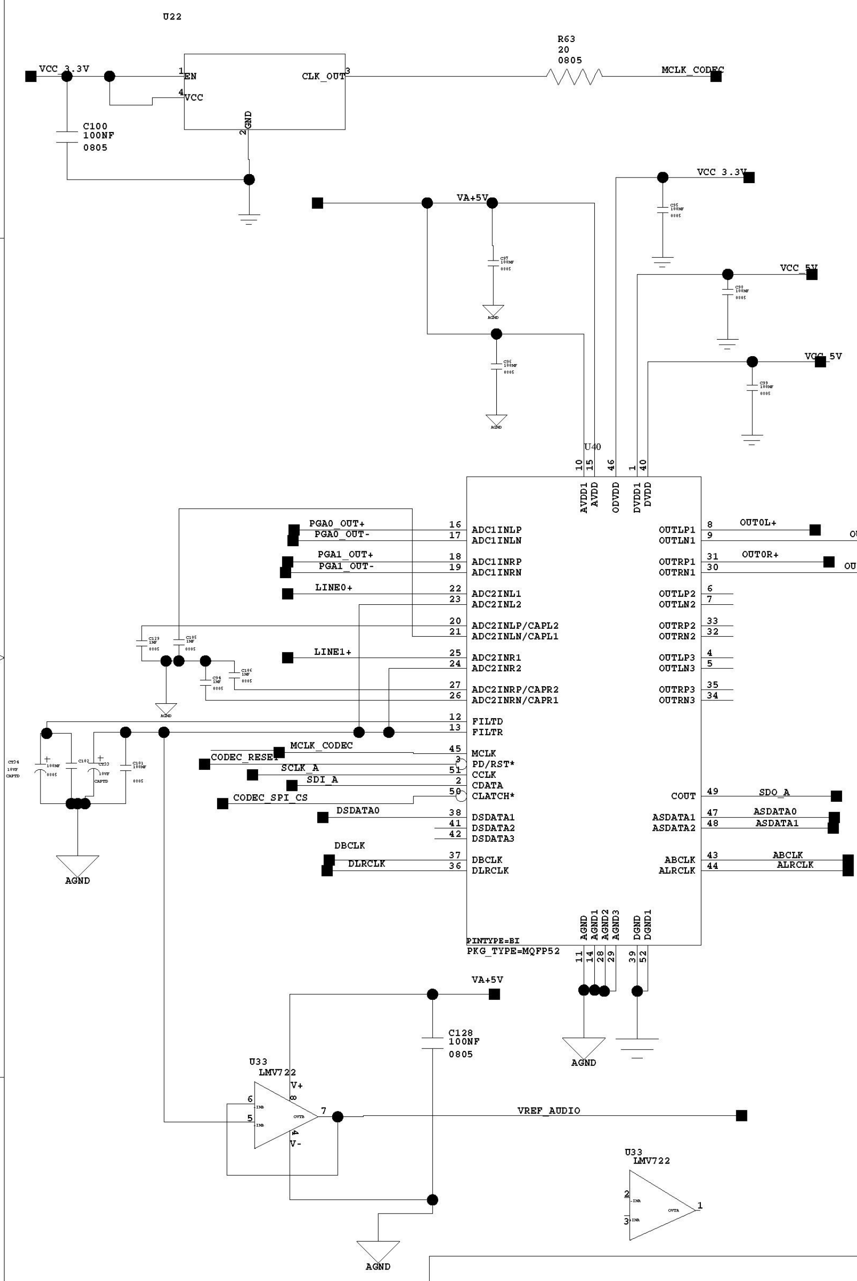
REGULATORS

SIZE	DWG NO	REV
B		0

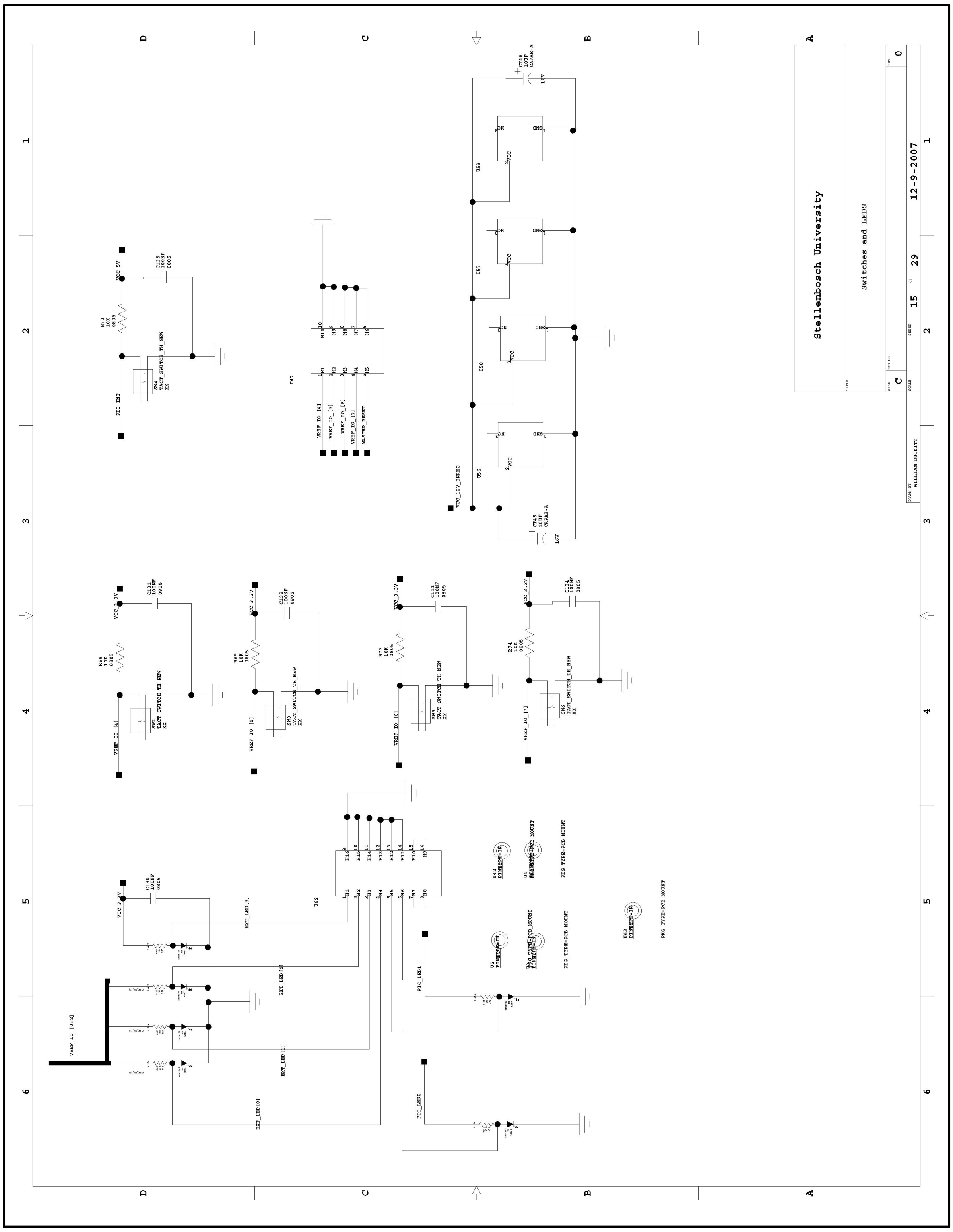
DRAWN BY WILLIAM DUCKITT

SCALE

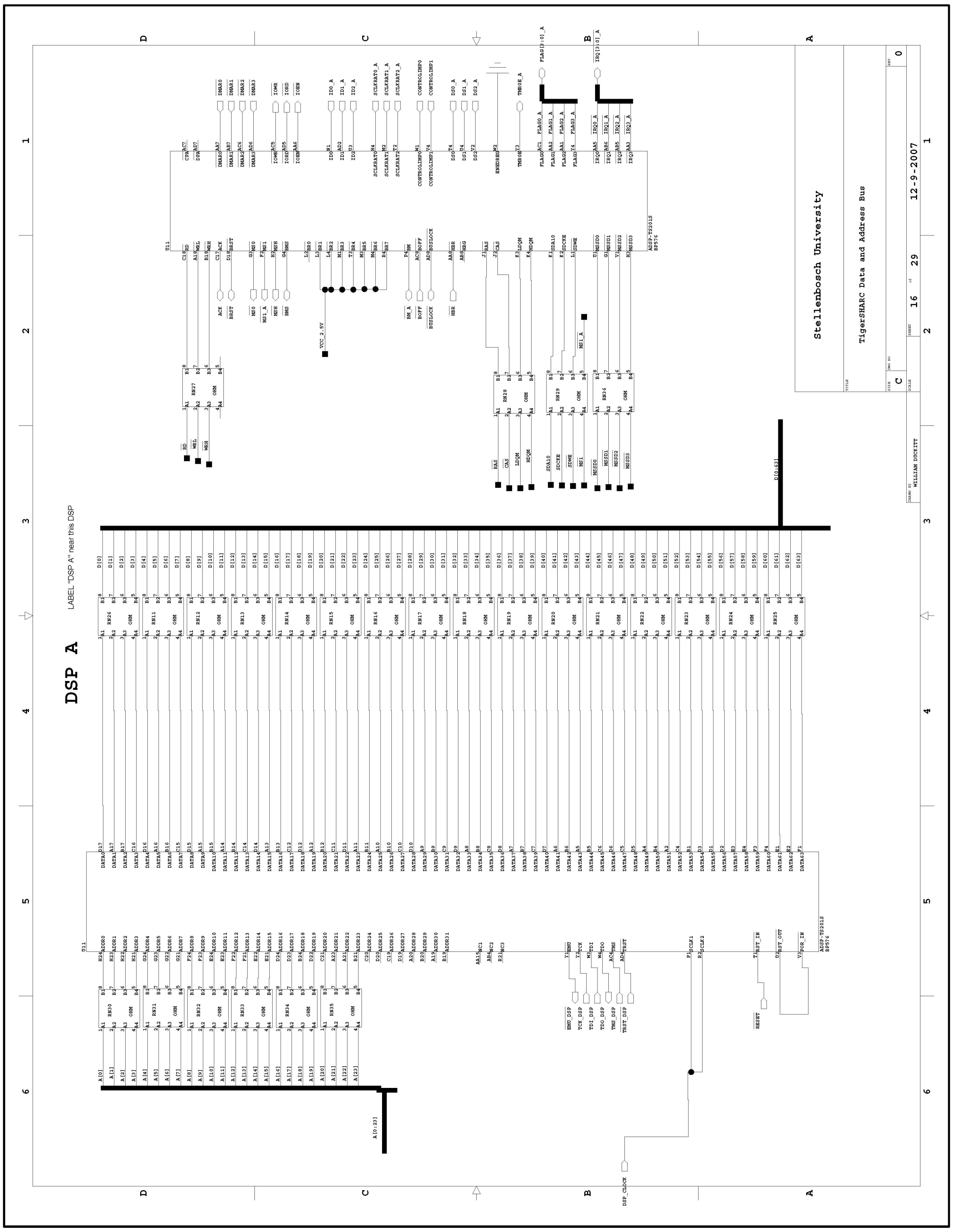
SHEET 13 of 29 12-9-2007



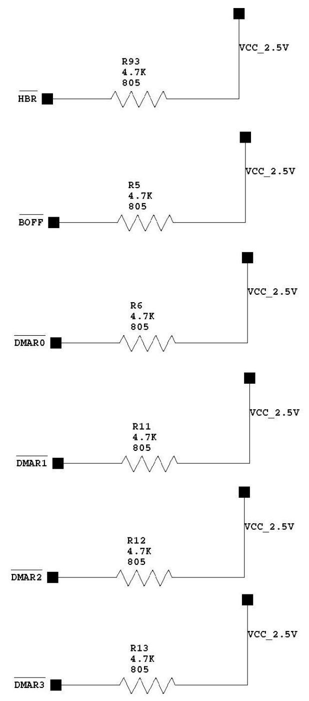
Stellenbosch University	
TITLE	CODEC
SIZE B	DWG NO
REV	0
DRAWN BY	WILLIAM DUCKITT
SCALE	
SHEET	14 of 29
	12-9-2007



Stellenbosch University
Switches and LEDS
12-9-2007
15 of 29
SIZE C
REV 0
WILLIAM DUCKITT
VCC_5V
VCC_3.3V
VCC_12V_UNREG
PIC_INT
TACT_SWITCH_TH_NEW
VREF_IO_[4]
VREF_IO_[5]
VREF_IO_[6]
VREF_IO_[7]
MASTER_RESET
VREF_IO_[0:3]
EXT_LED[0]
EXT_LED[1]
EXT_LED[2]
EXT_LED[3]
PIC_LED0
PIC_LED1
U47
U62
U56
U57
U58
U59
CT45
CT46
PKG_TYPE=PCB_MOUNT



Stellenbosch University
TigerSHARC Data and Address Bus
12-9-2007
16 of 29
WILLIAM DUCKITT
DSP A
LABEL "DSP A" near this DSP
ADSP-TS201S BP576
U11
D[0:63]
A[0:23]
DSP_CLOCK
VCC_2.5V
FLAG[3:0]_A
IRQ[3:0]_A

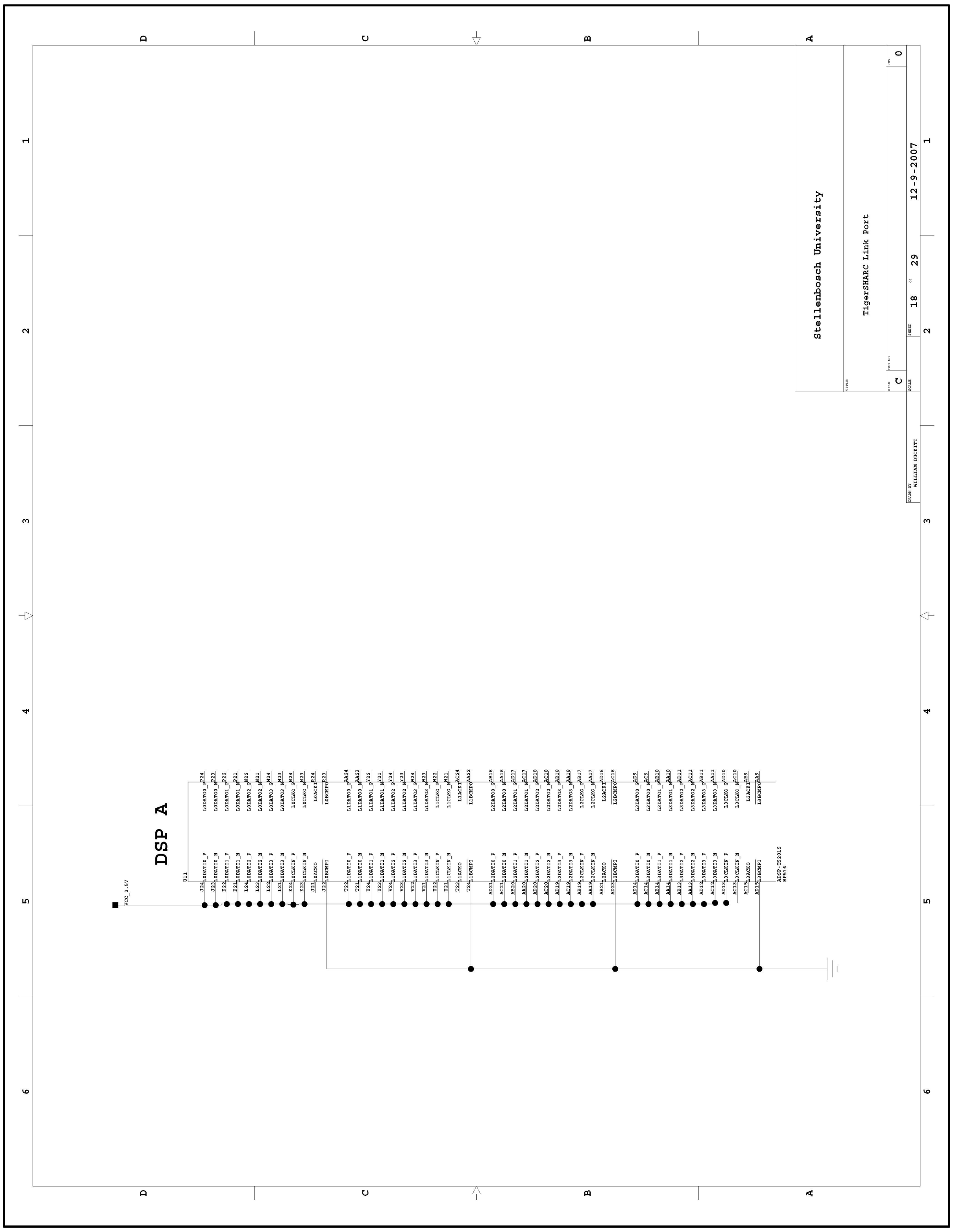


Stellenbosch University

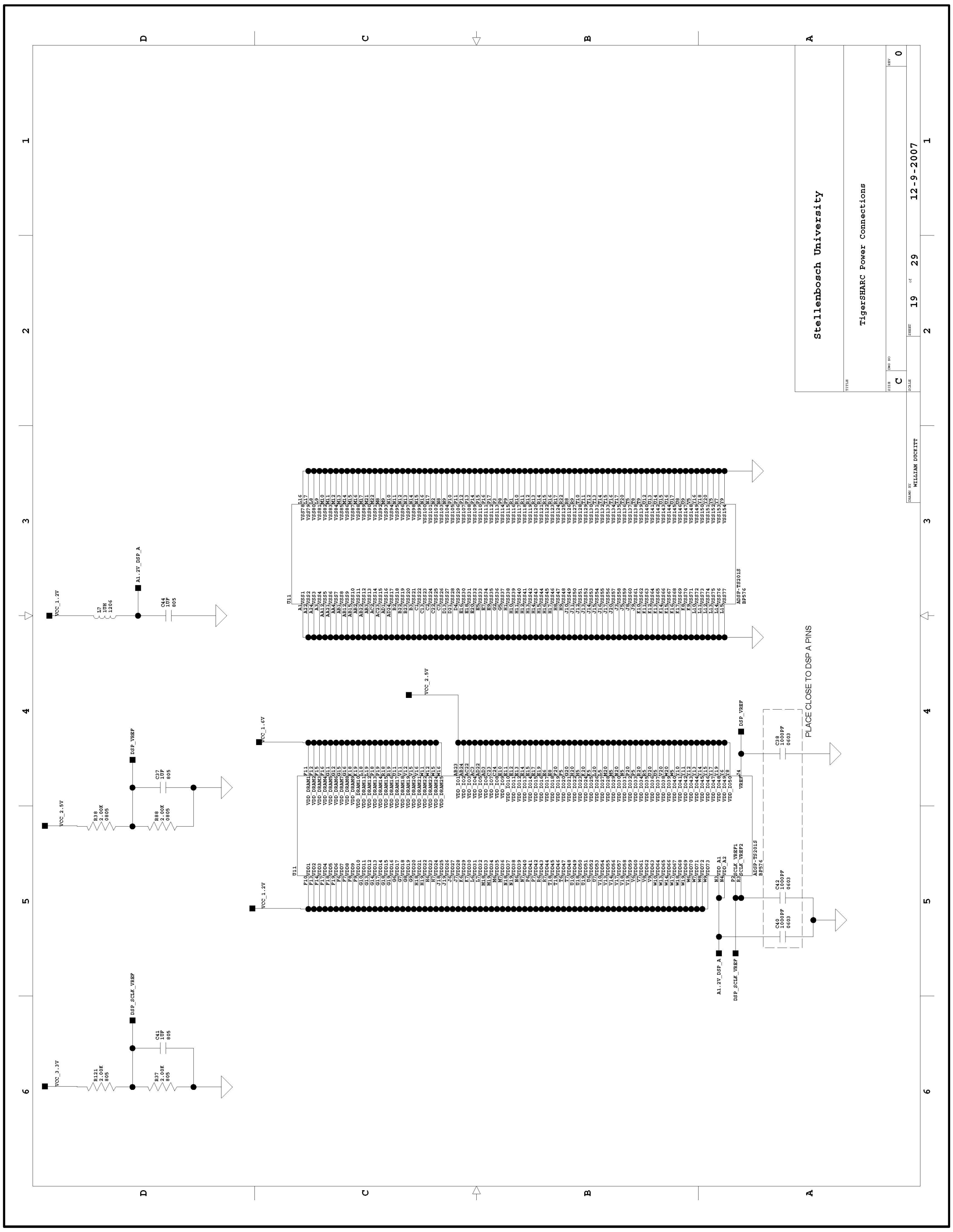
TITLE
Pull-Up Resistors

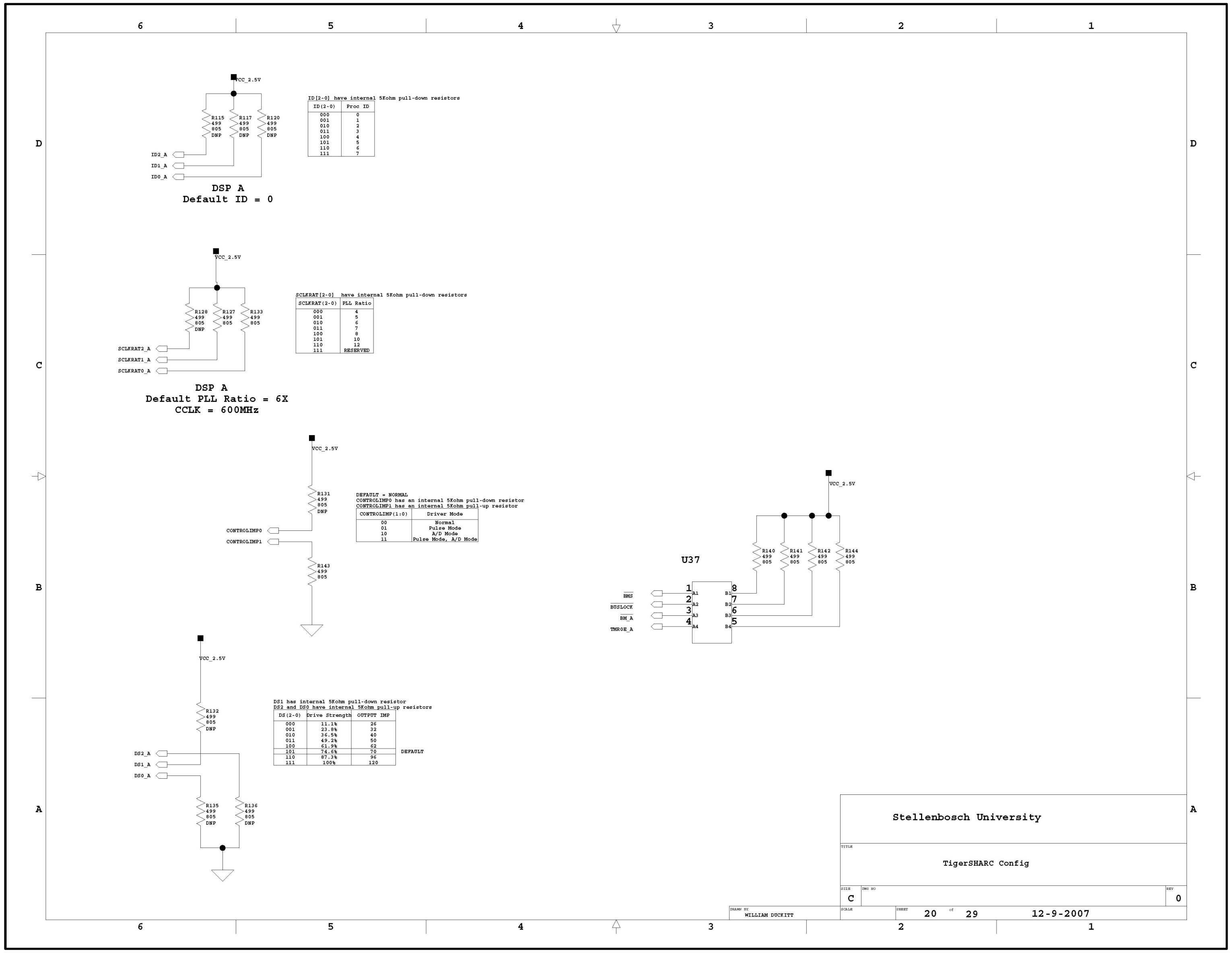
SIZE B DWG NO REV 0

DRAWN BY WILLIAM DUCKITT SCALE SHEET 17 of 29 12-9-2007



Stellenbosch University
TigerSHARC Link Port
12-9-2007
18 of 29
C
0
WILLIAM DUCKITT
DSP A
U11
VCC_2.5V
ADSP-TS201S


Stellenbosch University
TigerSHARC Power Connections
WILLIAM DUCKITT
C
19 of 29
12-9-2007
0
VCC_1.2V
VCC_1.6V
VCC_2.5V
VCC_3.3V
A1.2V_DSP_A
DSP_VREF
DSP_SCLK_VREF
L7
C44
C37
C41
C38
C40
C42
R38
R88
R121
R37
U11
ADSP-TS201S
BP576
PLACE CLOSE TO DSP A PINS



ID[2-0] have internal 5Kohm pull-down resistors

ID(2-0)	Proc ID
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

SCLKRAT[2-0] have internal 5Kohm pull-down resistors

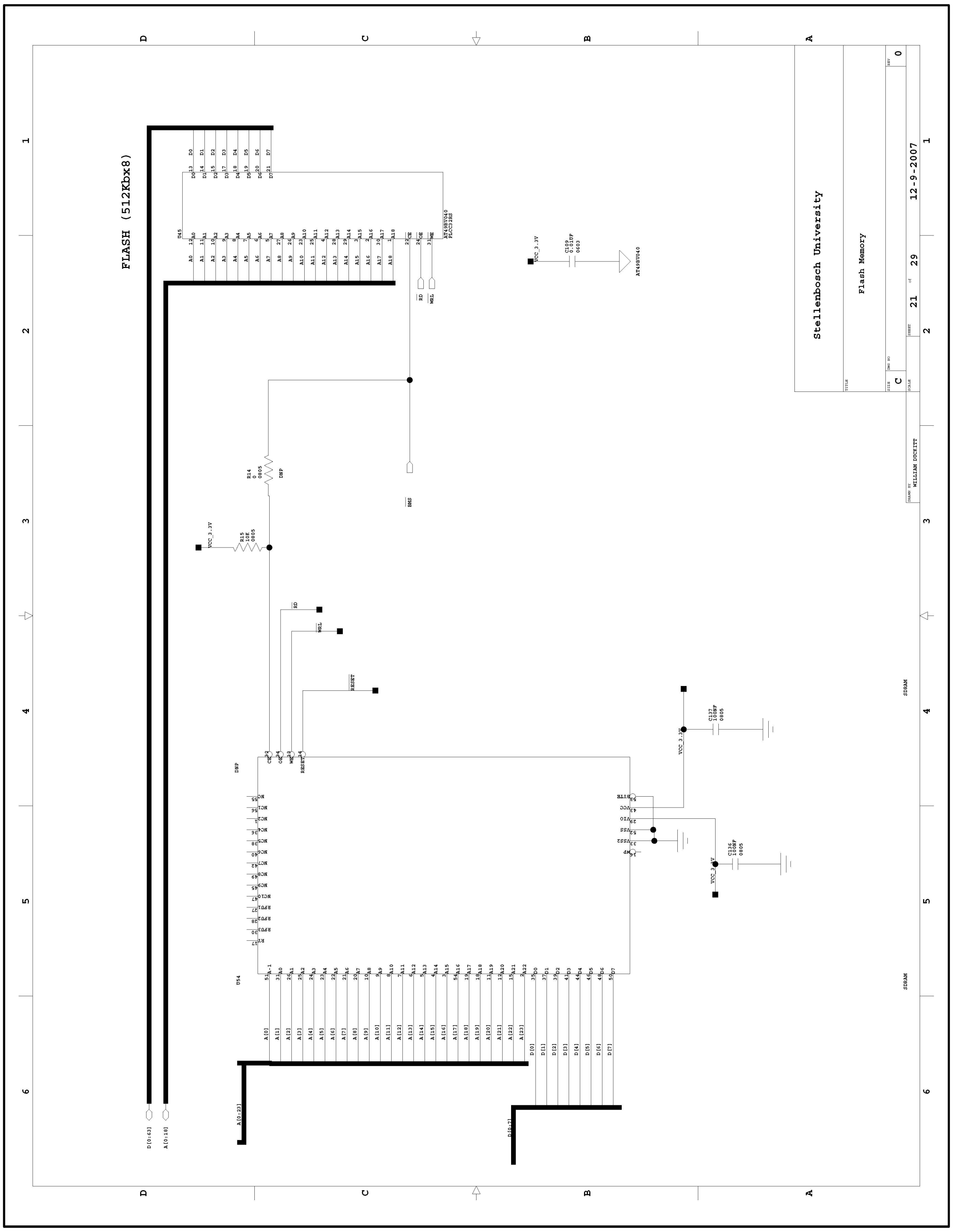
SCLKRAT(2-0)	PLL Ratio
000	4
001	5
010	6
011	7
100	8
101	10
110	12
111	RESERVED

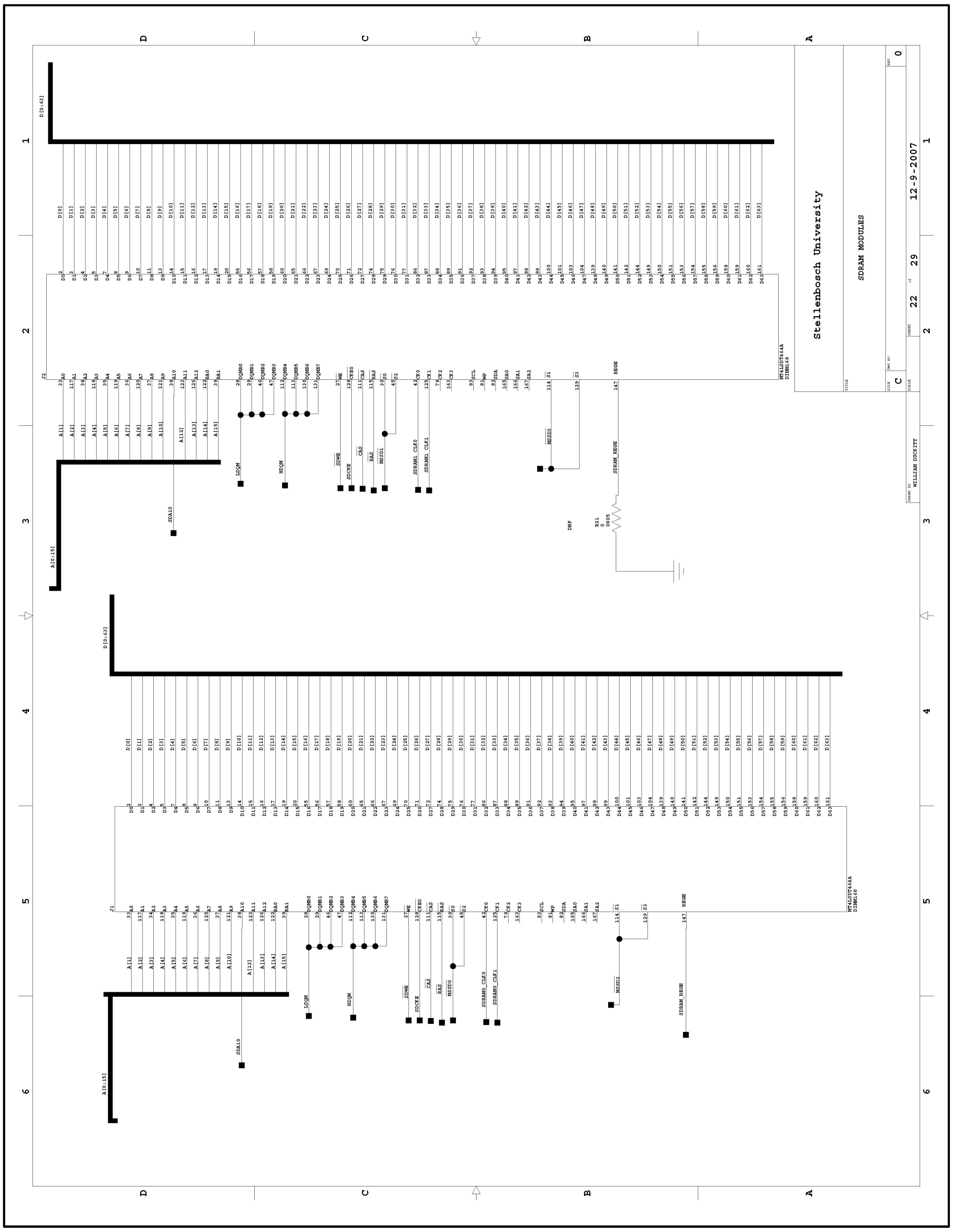
DEFAULT = NORMAL
CONTROLIMP0 has an internal 5Kohm pull-down resistor
CONTROLIMP1 has an internal 5Kohm pull-up resistor

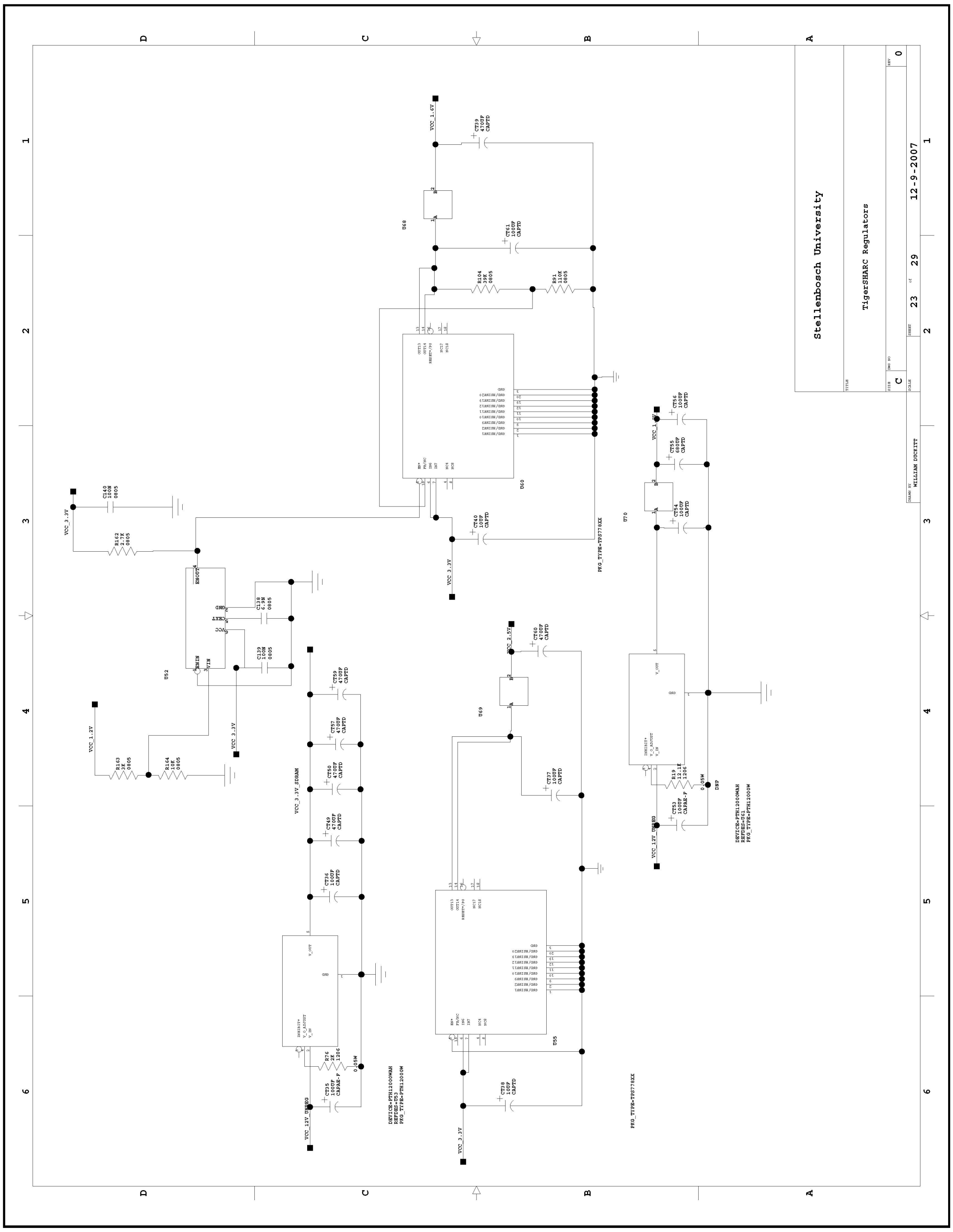
CONTROLIMP(1:0)	Driver Mode
00	Normal
01	Pulse Mode
10	A/D Mode
11	Pulse Mode, A/D Mode

DS1 has internal 5Kohm pull-down resistor
DS2 and DS0 have internal 5Kohm pull-up resistors

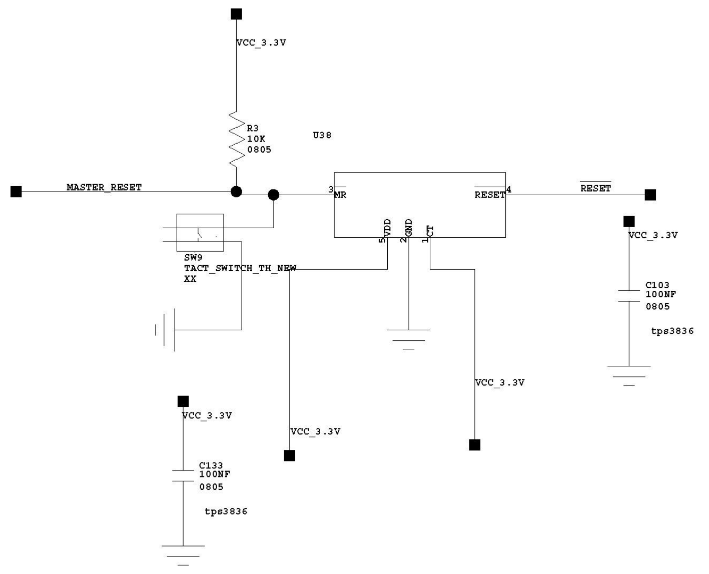
DS(2-0)	Drive Strength	OUTPUT IMP	
000	11.1%	26	
001	23.8%	32	
010	36.5%	40	
011	49.2%	50	
100	61.9%	62	
101	74.6%	70	DEFAULT
110	87.3%	96	
111	100%	120	


FLASH (512Kbx8)
Stellenbosch University
Flash Memory
12-9-2007
21 of 29
WILLIAM DUCKITT
AT49BV040
VCC_3.3V
R14
R15
C109
C136
C137
U45
U54
D[0:63]
A[0:18]
A[0:23]
D[0:7]
RD
WEL
RESET
DNP
SDRAM


Stellenbosch University
SDRAM MODULES
22 of 29
12-9-2007
WILLIAM DUCKITT
C
0
MT4LSDT464A DIMM168
J1
J2
D[0:63]
A[0:15]
SDRAM_REGE
R21 0 0805
DNP
LDQM
HDQM
SDA10
SDWE
SDCKE
CAS
RAS
MSSD0
MSSD1
MSSD2
MSSD3
SDRAM0_CLK0
SDRAM0_CLK1
SDRAM1_CLK0
SDRAM1_CLK1



Stellenbosch University
TigerSHARC Regulators
12-9-2007
23 of 29
C
0
WILLIAM DUCKITT
U52
U55
U60
U68
U69
U70
VCC_1.6V
VCC_3.3V
VCC_1.2V
VCC_2.5V
VCC_3.3V_SDRAM
VCC_12V_UNREG
PKG_TYPE=TPS778XX
DEVICE=PTH12000WAH
PKG_TYPE=PTH12000W
DNP



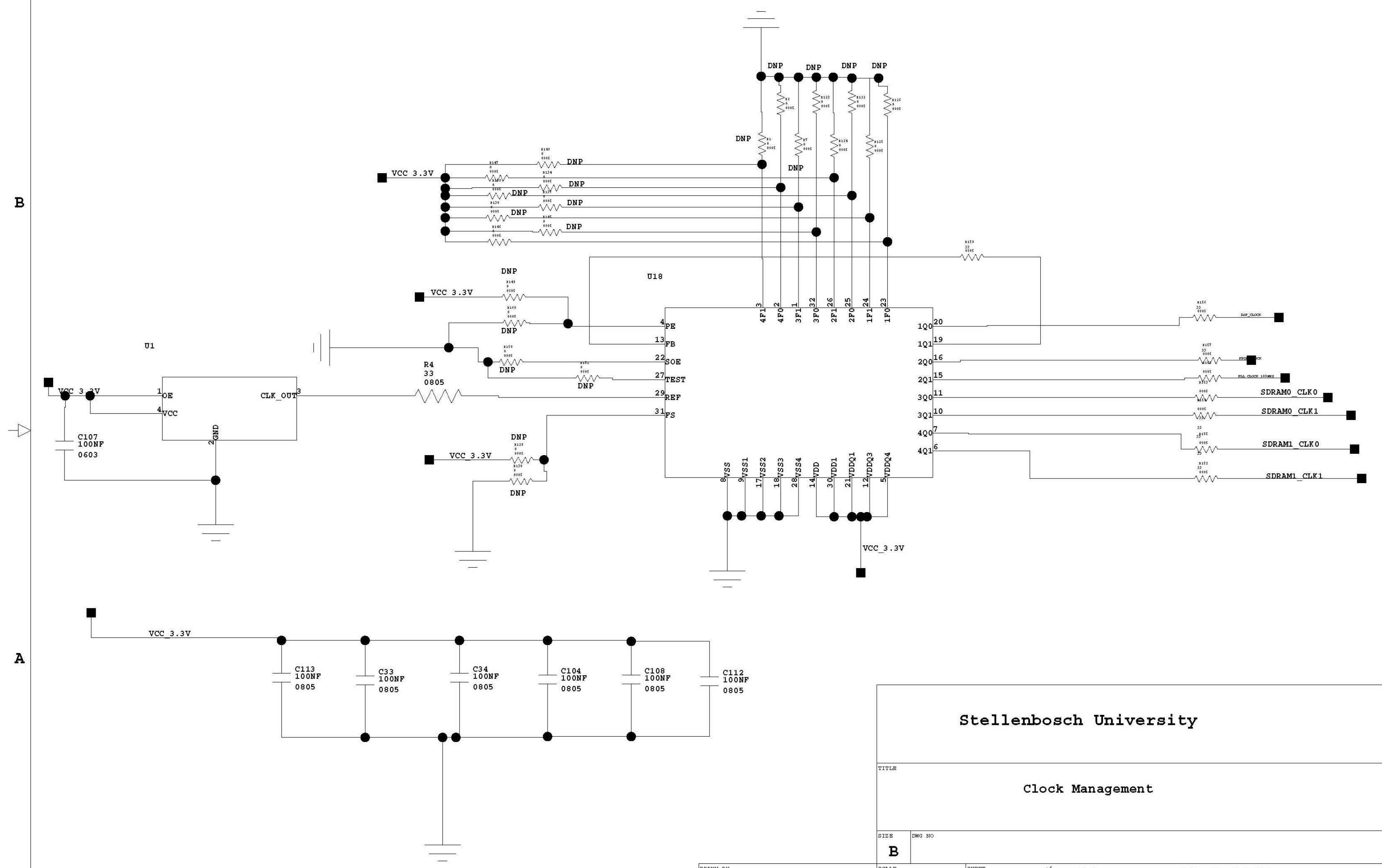
Stellenbosch University

TITLE

Reset Manager

SIZE	DWG NO	REV
B		0

DRAWN BY	SCALE	SHEET	of	
WILLIAM DUCKITT		24	29	12-9-2007



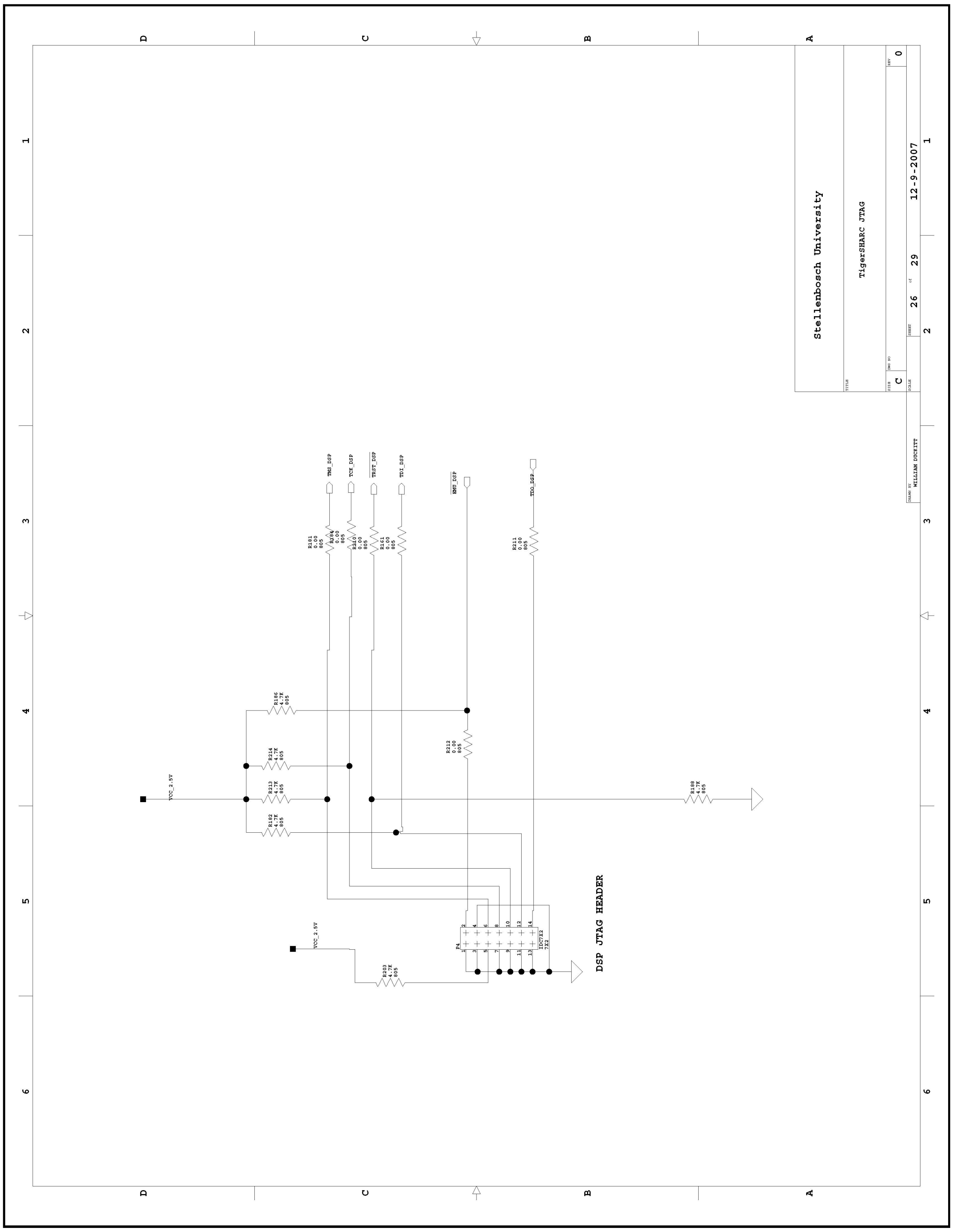
Stellenbosch University

TITLE

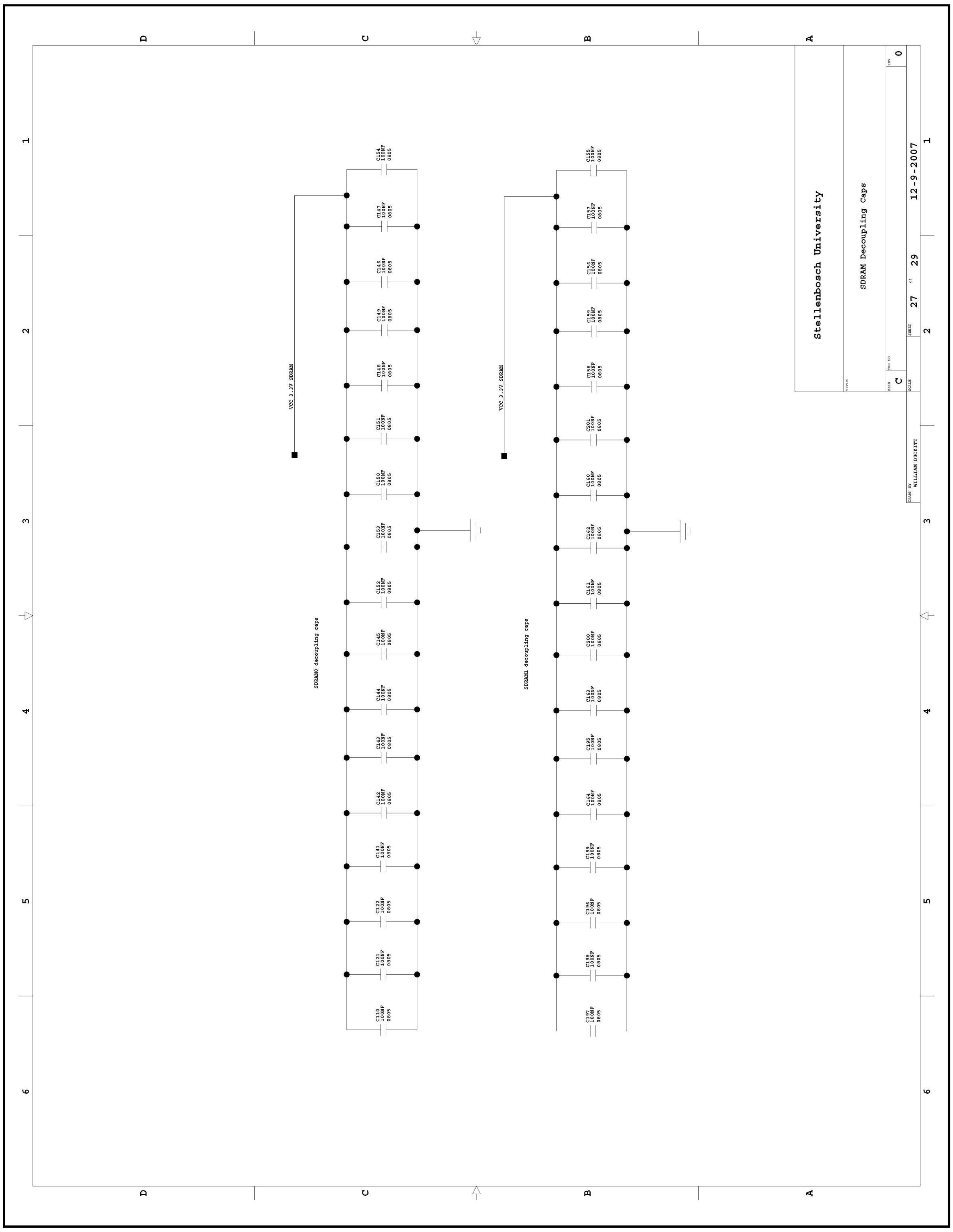
Clock Management

SIZE B | DWG NO | REV 0

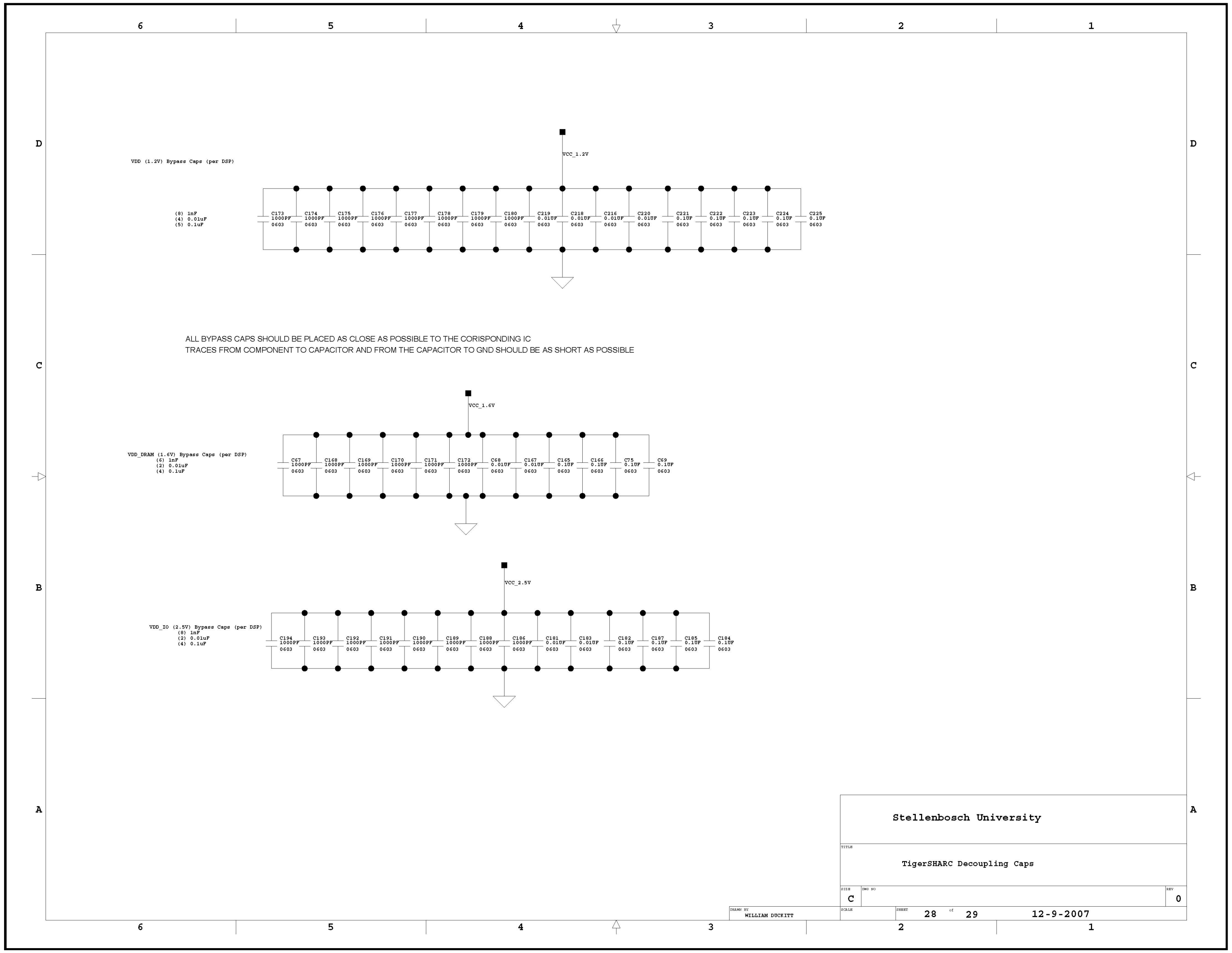
DRAWN BY WILLIAM DUCKITT | SCALE | SHEET 25 of 29 | 12-9-2007

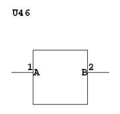


Stellenbosch University
TigerSHARC JTAG
12-9-2007
26 of 29
WILLIAM DUCKITT
DSP JTAG HEADER
TMS_DSP
TCK_DSP
TRST_DSP
TDI_DSP
EMU_DSP
TDO_DSP
VCC_2.5V
R181 0.00 805
R184 0.00 805
R210 0.00 805
R161 0.00 805
R211 0.00 805
R212 0.00 805
R186 4.7K 805
R214 4.7K 805
R213 4.7K 805
R182 4.7K 805
R188 4.7K 805
R203 4.7K 805
P4
IDC7X2
7X2



Stellenbosch University
TITLE
SDRAM Decoupling Caps
SIZE
C
DWG NO
REV
0
SCALE
SHEET
27
of
29
12-9-2007
DRAWN BY
WILLIAM DUCKITT
VCC_3.3V_SDRAM
SDRAM0 decoupling caps
SDRAM1 decoupling caps
C154 100NF 0805
C147 100NF 0805
C146 100NF 0805
C149 100NF 0805
C148 100NF 0805
C151 100NF 0805
C150 100NF 0805
C153 100NF 0805
C152 100NF 0805
C145 100NF 0805
C144 100NF 0805
C143 100NF 0805
C142 100NF 0805
C141 100NF 0805
C122 100NF 0805
C121 100NF 0805
C110 100NF 0805
C155 100NF 0805
C157 100NF 0805
C156 100NF 0805
C159 100NF 0805
C158 100NF 0805
C201 100NF 0805
C160 100NF 0805
C162 100NF 0805
C161 100NF 0805
C200 100NF 0805
C163 100NF 0805
C195 100NF 0805
C164 100NF 0805
C199 100NF 0805
C196 100NF 0805
C198 100NF 0805
C197 100NF 0805
A
B
C
D
1
2
3
4
5
6


6
5
4
3
2
1
D
C
B
A
VDD (1.2V) Bypass Caps (per DSP)
(8) 1nF
(4) 0.01uF
(5) 0.1uF
VCC_1.2V
C173 1000PF 0603
C174 1000PF 0603
C175 1000PF 0603
C176 1000PF 0603
C177 1000PF 0603
C178 1000PF 0603
C179 1000PF 0603
C180 1000PF 0603
C219 0.01UF 0603
C218 0.01UF 0603
C216 0.01UF 0603
C220 0.01UF 0603
C221 0.1UF 0603
C222 0.1UF 0603
C223 0.1UF 0603
C224 0.1UF 0603
C225 0.1UF 0603
ALL BYPASS CAPS SHOULD BE PLACED AS CLOSE AS POSSIBLE TO THE CORISPONDING IC
TRACES FROM COMPONENT TO CAPACITOR AND FROM THE CAPACITOR TO GND SHOULD BE AS SHORT AS POSSIBLE
VDD_DRAM (1.6V) Bypass Caps (per DSP)
(6) 1nF
(2) 0.01uF
(4) 0.1uF
VCC_1.6V
C67 1000PF 0603
C168 1000PF 0603
C169 1000PF 0603
C170 1000PF 0603
C171 1000PF 0603
C172 1000PF 0603
C68 0.01UF 0603
C167 0.01UF 0603
C165 0.1UF 0603
C166 0.1UF 0603
C75 0.1UF 0603
C69 0.1UF 0603
VDD_IO (2.5V) Bypass Caps (per DSP)
(8) 1nF
(2) 0.01uF
(4) 0.1uF
VCC_2.5V
C194 1000PF 0603
C193 1000PF 0603
C192 1000PF 0603
C191 1000PF 0603
C190 1000PF 0603
C189 1000PF 0603
C188 1000PF 0603
C186 1000PF 0603
C181 0.01UF 0603
C183 0.01UF 0603
C182 0.1UF 0603
C187 0.1UF 0603
C185 0.1UF 0603
C184 0.1UF 0603
Stellenbosch University
TITLE
TigerSHARC Decoupling Caps
SIZE C
DWG NO
REV 0
DRAWN BY WILLIAM DUCKITT
SCALE
SHEET 28 of 29
12-9-2007



Stellenbosch University				
TITLE	FAN			
SIZE B	DWG NO			REV 0
DRAWN BY WILLIAM DUCKITT	SCALE	SHEET 29	of 29	12-9-2007

Appendix B

PCB Layout

The PCB manufacturing layers are displayed in the following pages. Please note that the layers are not in a 1:1 scale. A higher resolution PDF and the original manufacturing files are included on the DVD.

1. Figure B-1 illustrates the top layer silk screen.
2. Figure B-2 illustrates the top layer solder mask.
3. Figure B-3 illustrates the top routing layer.
4. Figure B-4 illustrates the internal power plane.
5. Figure B-5 illustrates the internal horizontal routing layer.
6. Figure B-6 illustrates the internal vertical routing layer.
7. Figure B-7 illustrates the internal ground plane.
8. Figure B-8 illustrates the bottom routing layer.
9. Figure B-9 illustrates the bottom solder mask.
10. Figure B-10 illustrates the bottom layer silk screen.

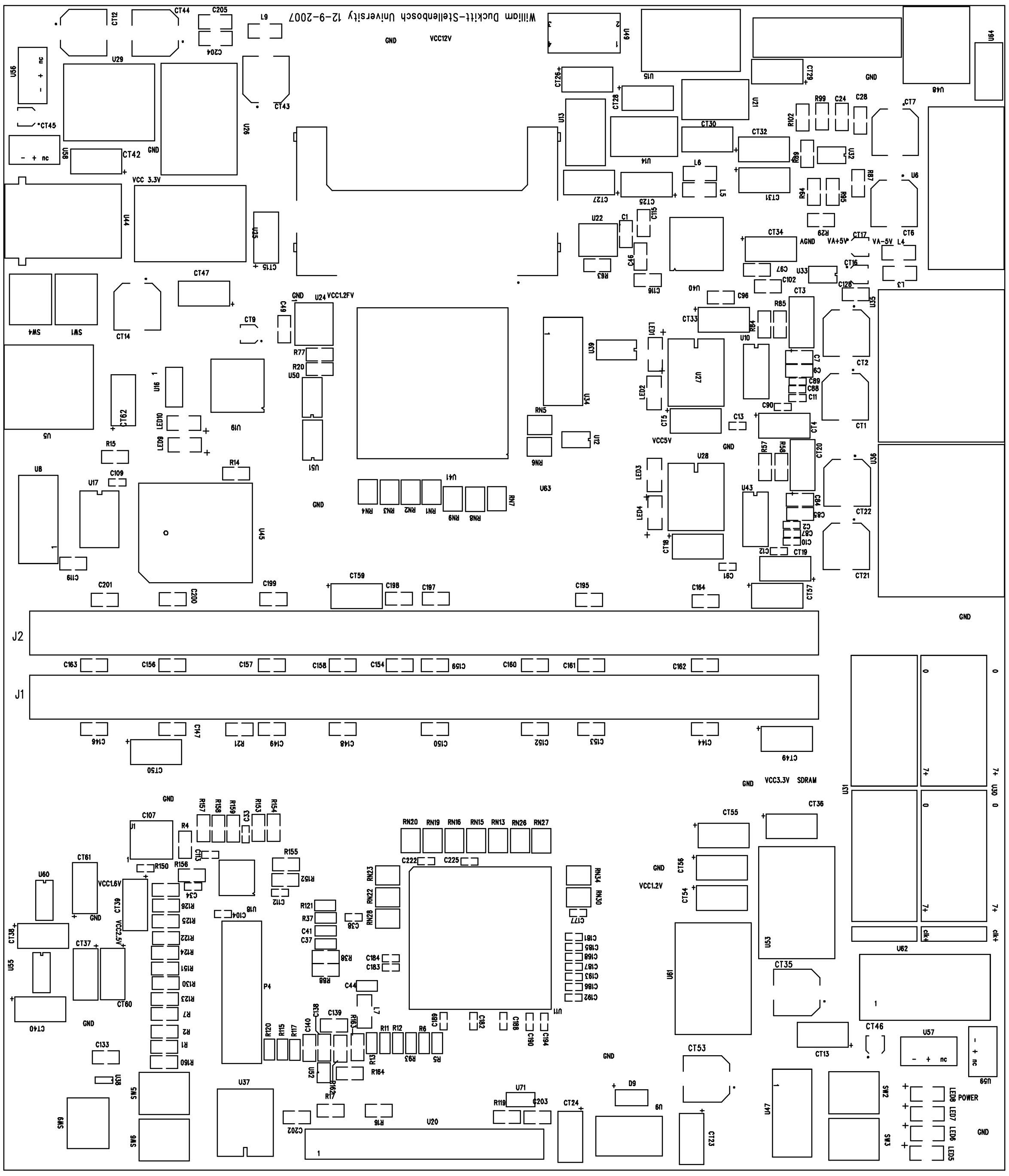


Figure B-1: Top layer silk screen.

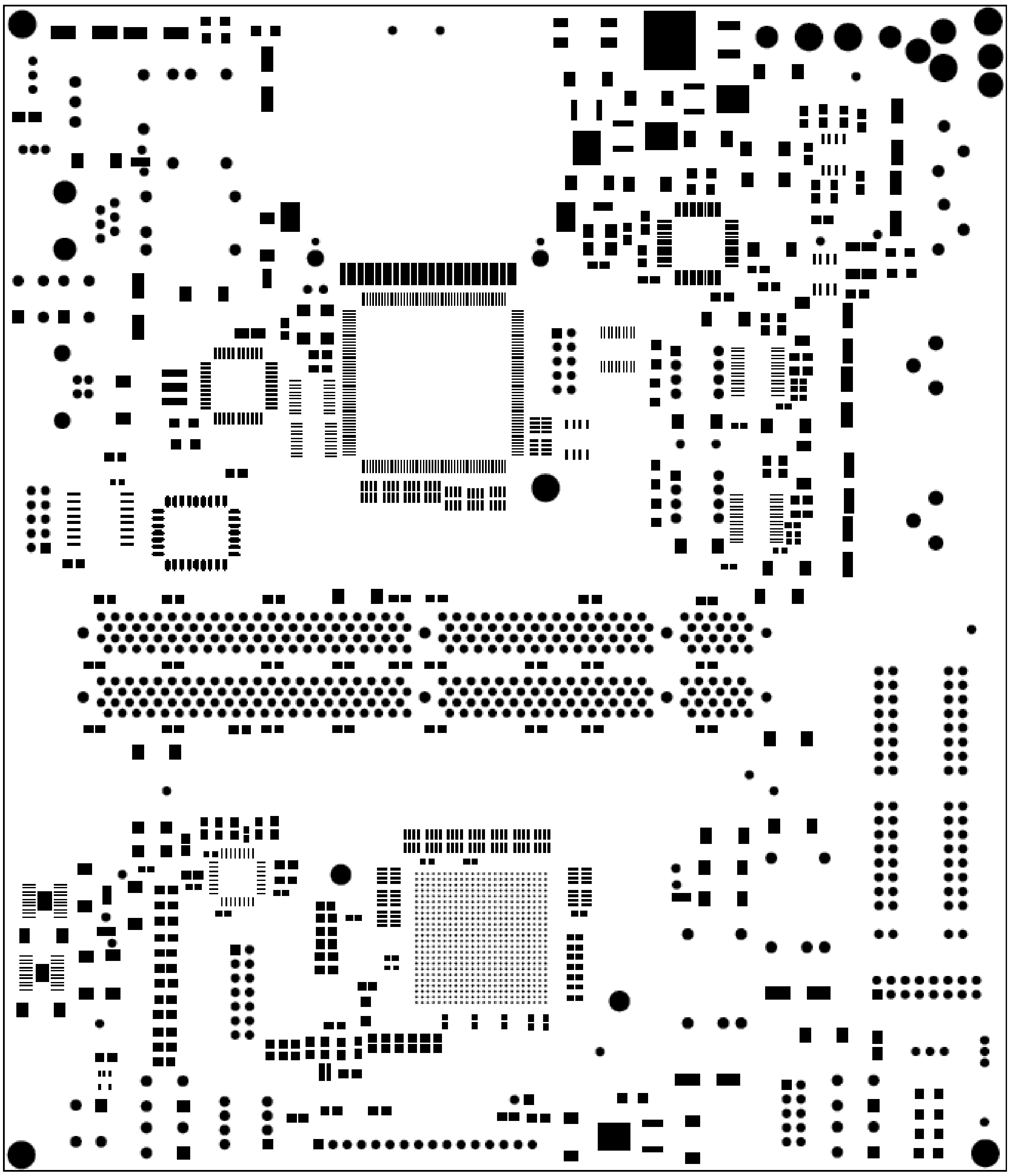


Figure B-2: Solder mask top.

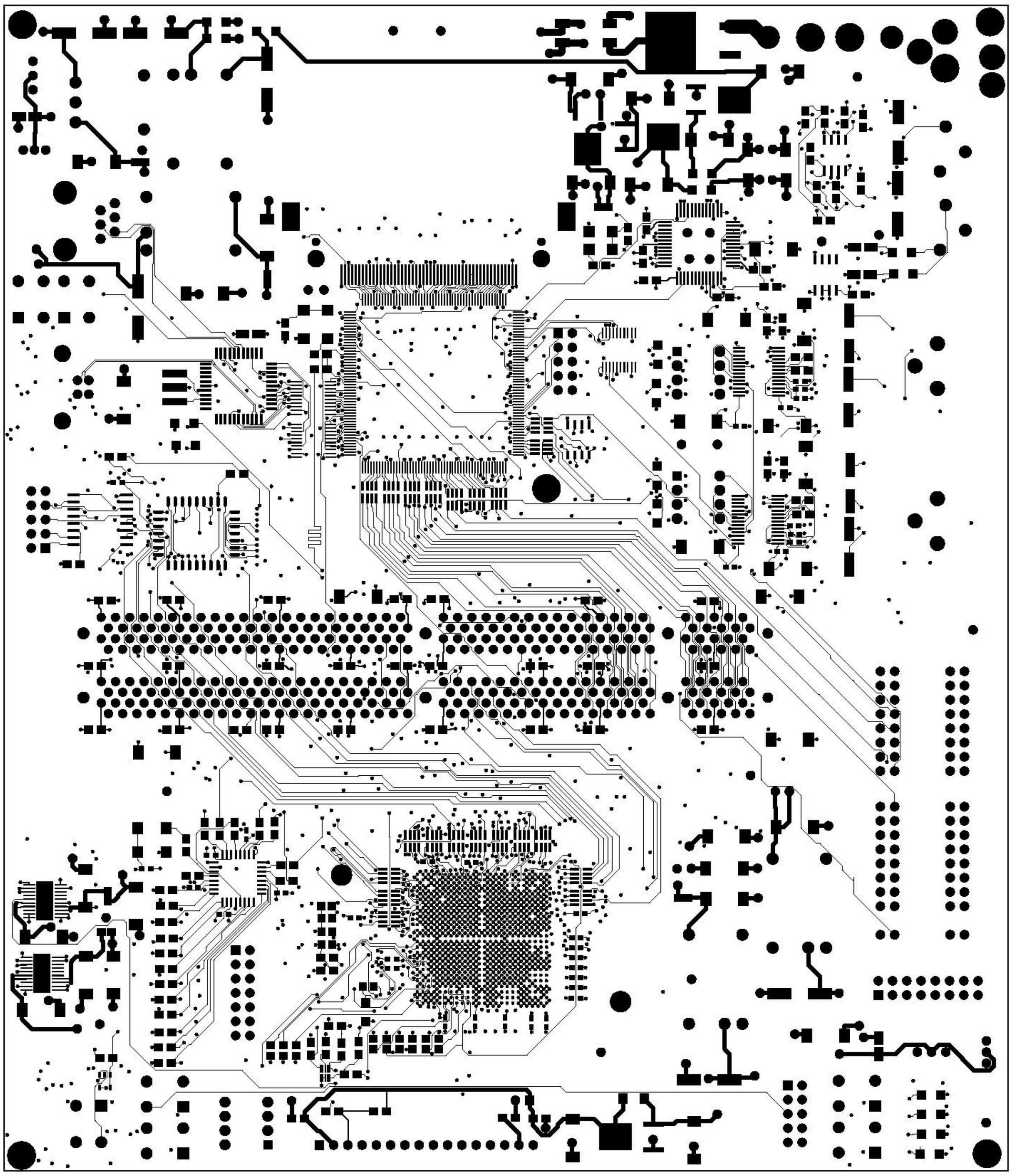


Figure B-3: Top layer routing

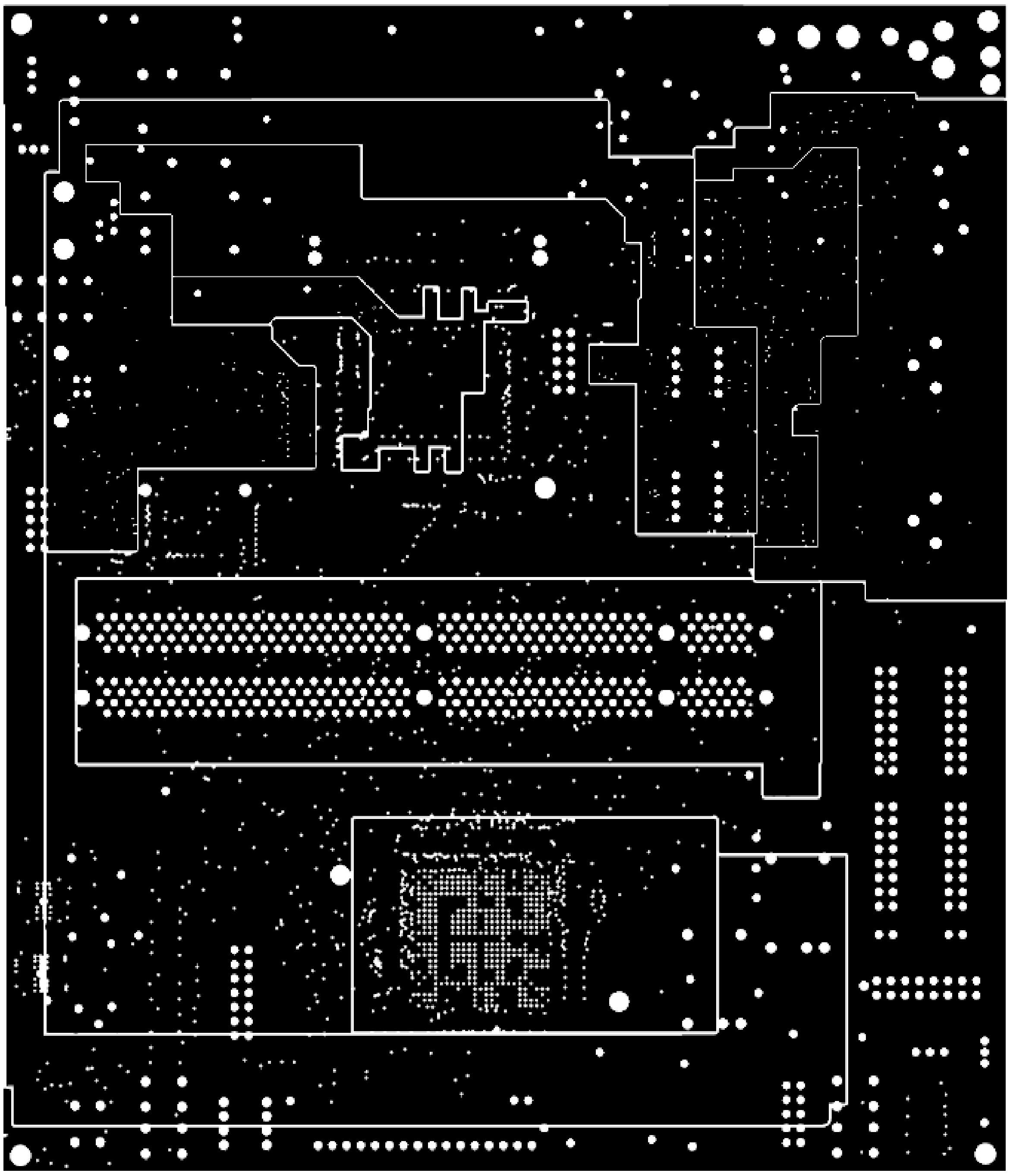


Figure B-4:Power layer.

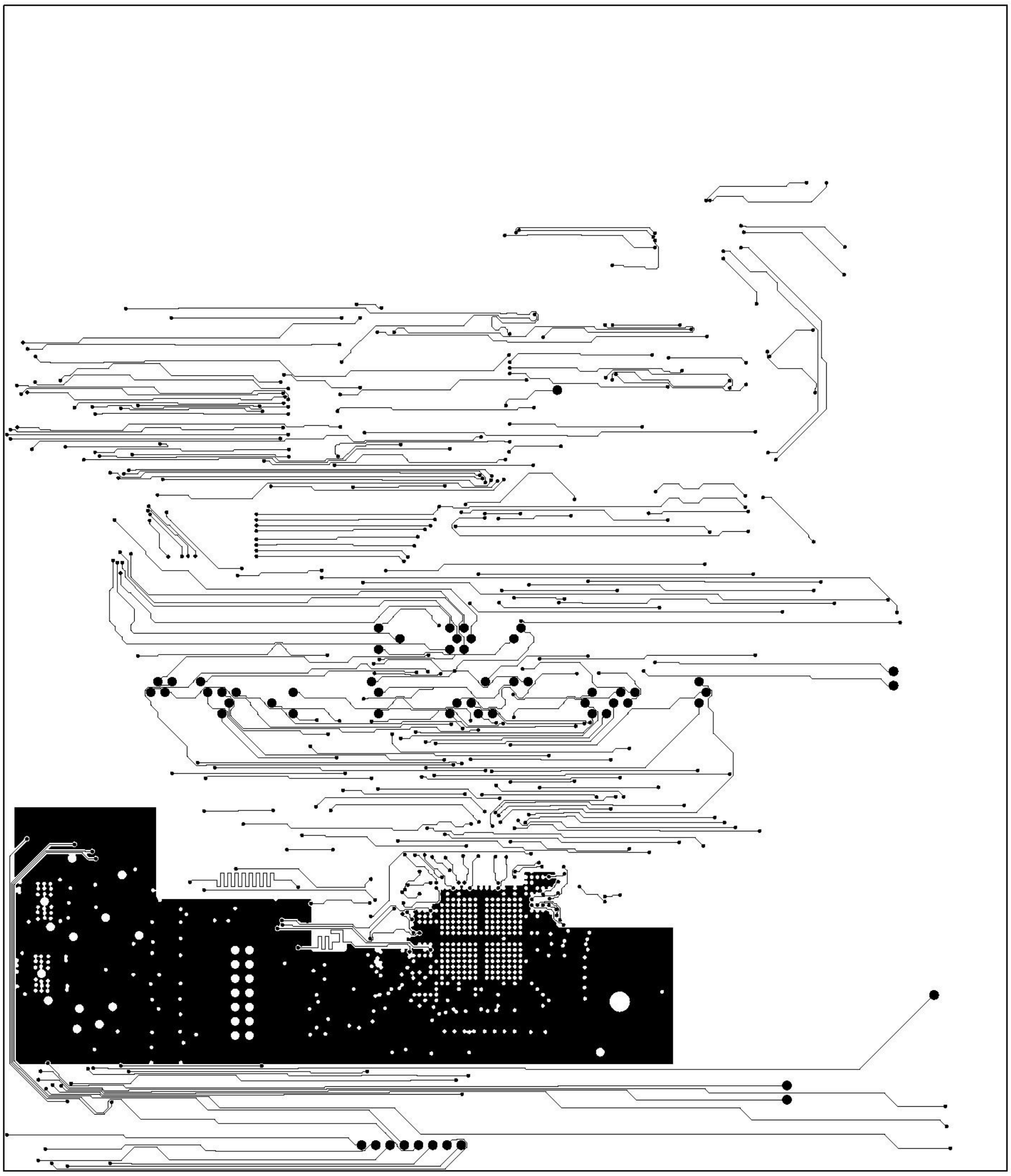


Figure B-5: Horizontal routing layer.

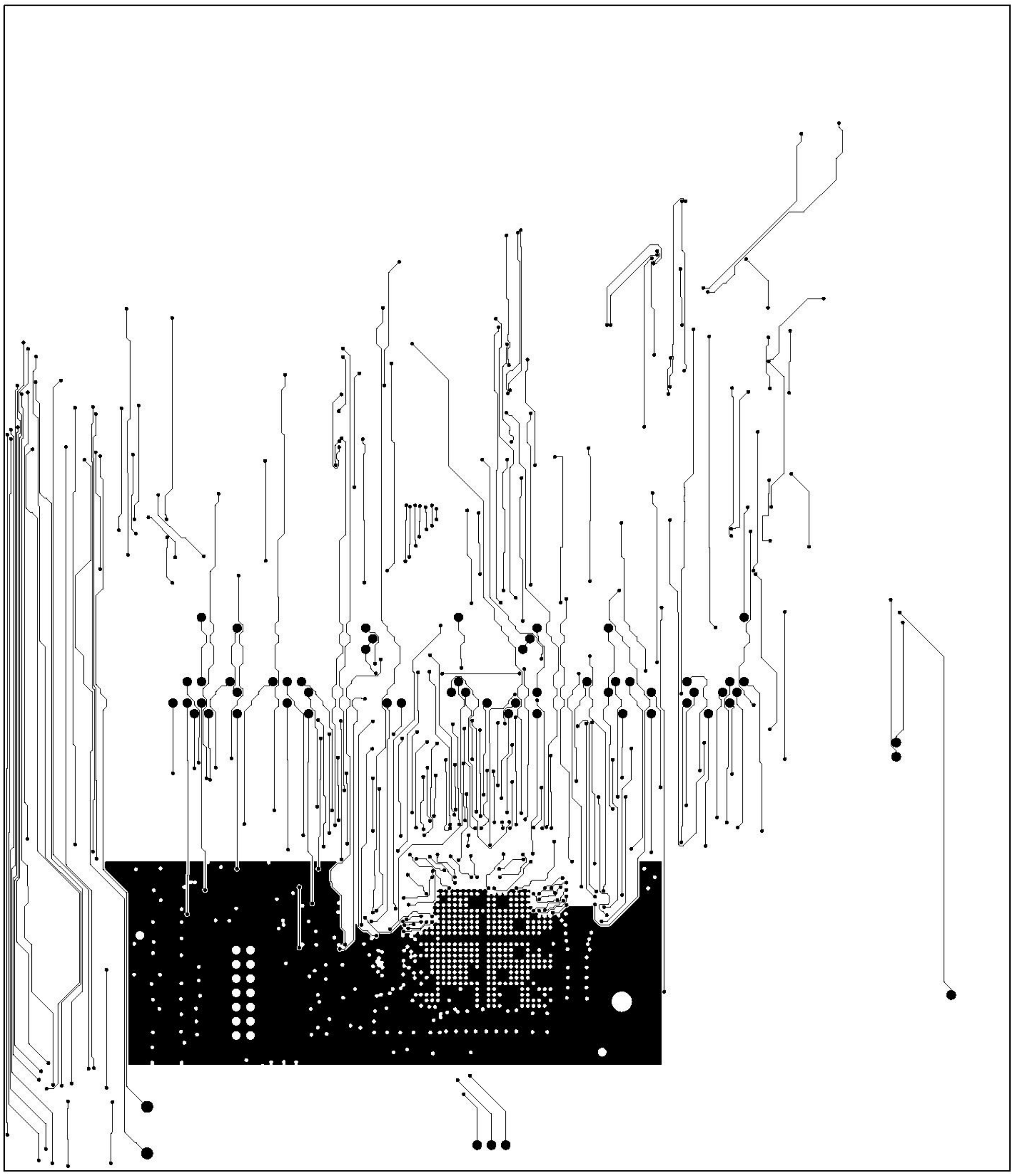


Figure B-6: Vertical routing layer.

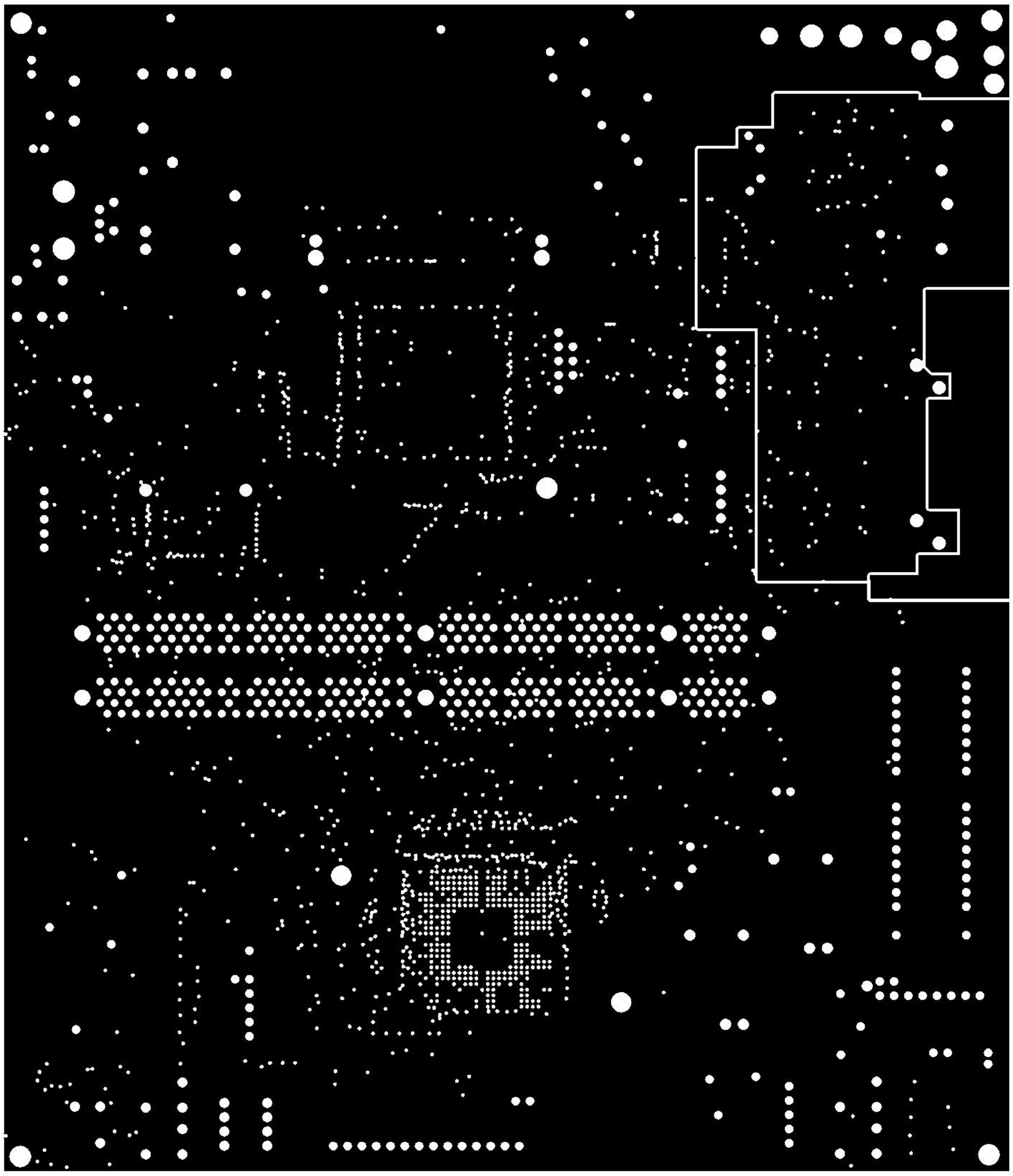


Figure B-7: Ground layer.

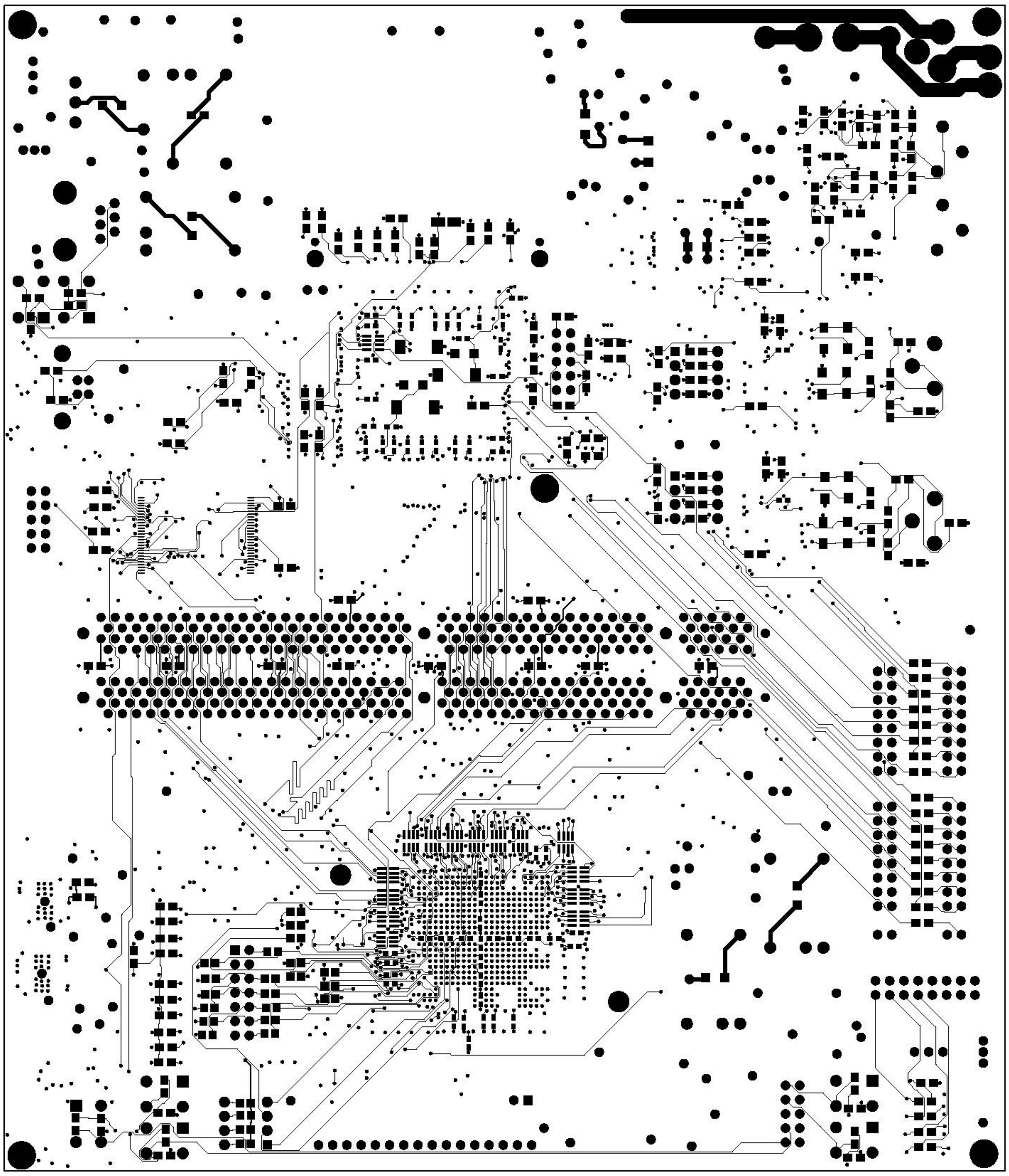


Figure B-8: Bottom routing layer (mirrored).

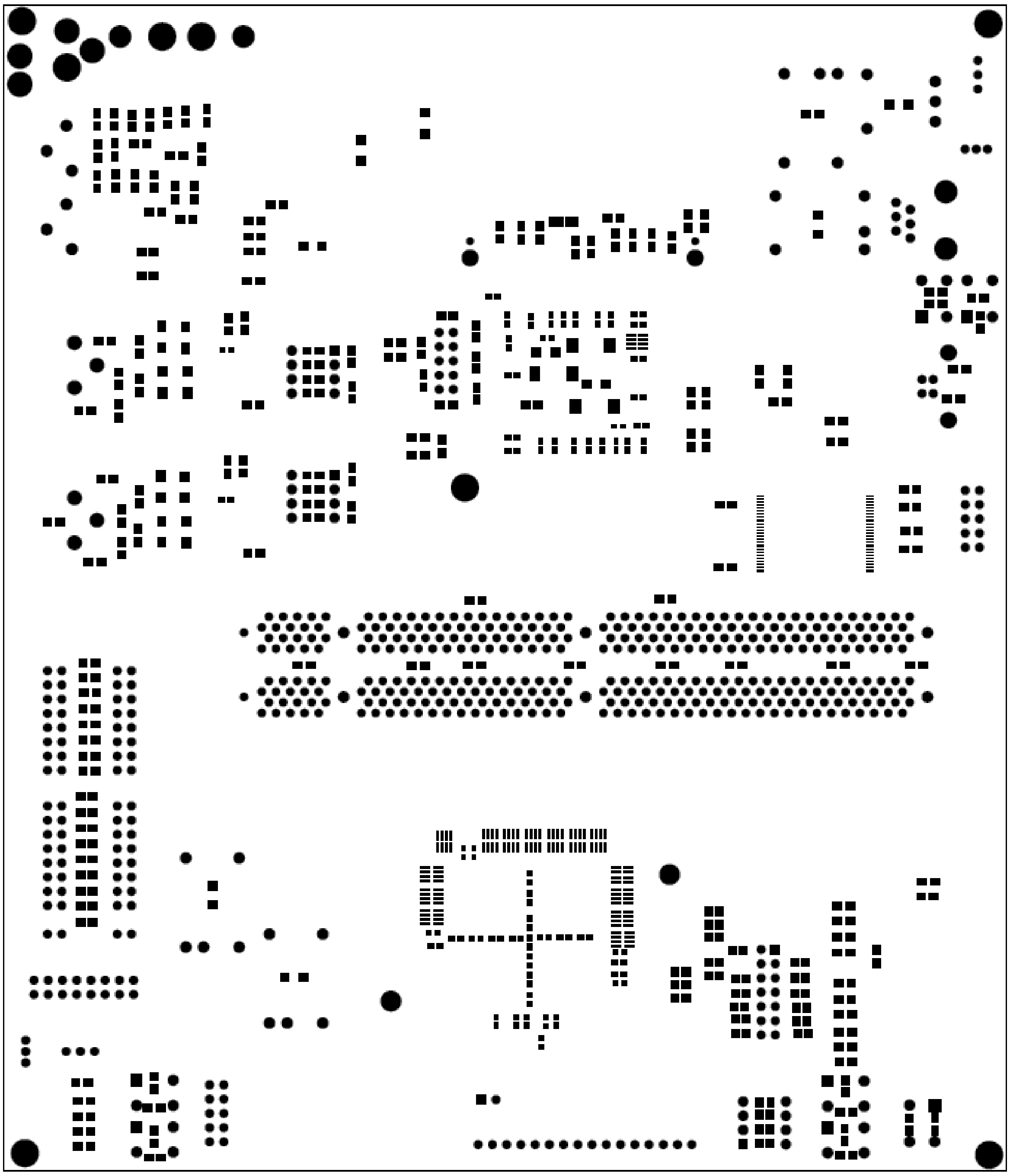


Figure B-9: Bottom solder mask (mirrored).

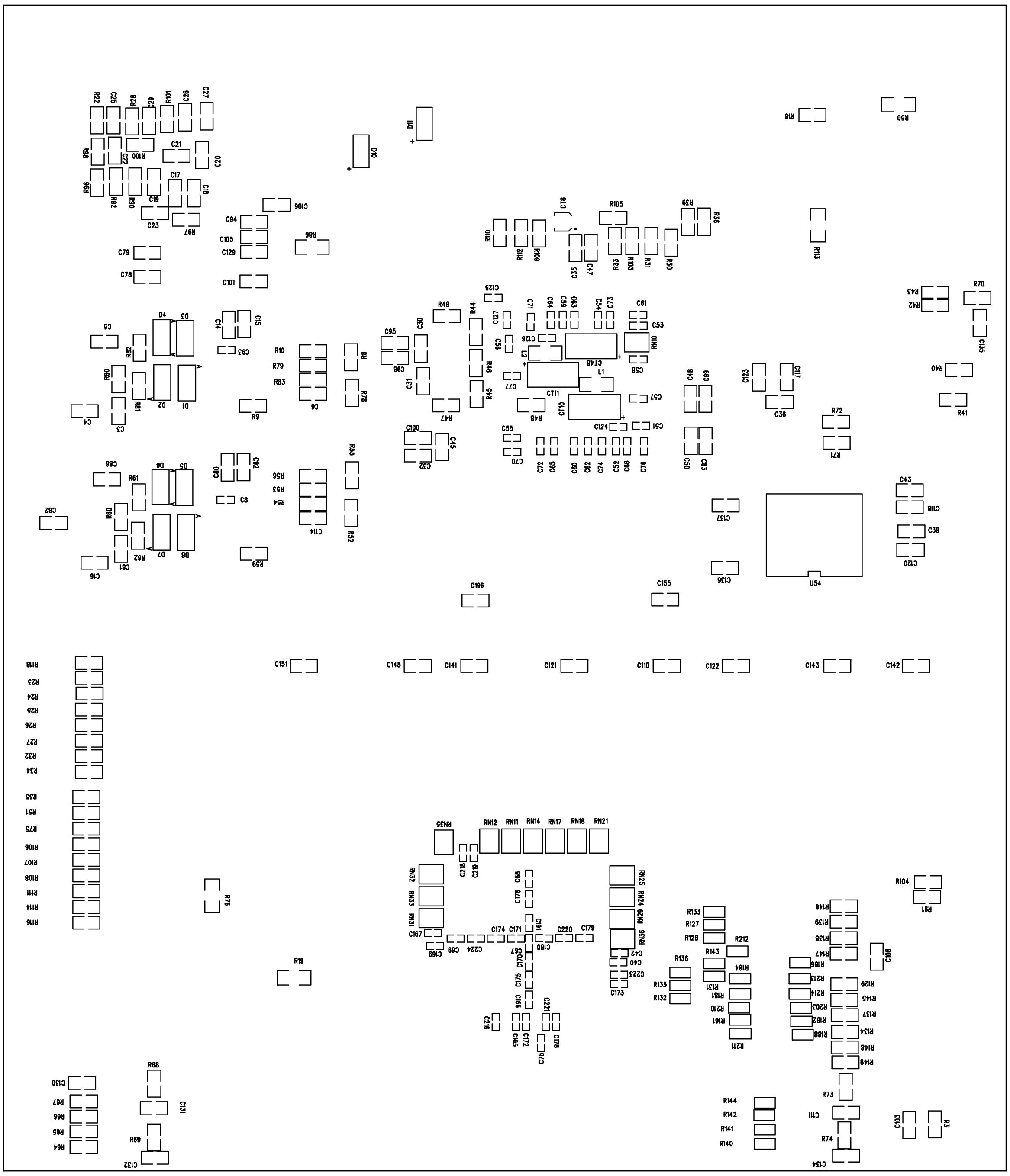


Figure B-10: Bottom silk screen (mirrored).

Appendix C

Bill of Materials

#	Qty	Reference Designator	Supplier	Device	Package	Value
1	84	C1,C6,C16,C21,C30,C31,C32,C33,C34,C35,C43,C45,C46,C47,C48,C49,C50,C83,C95,C96,C97,C98,C99,C100,C101,C102,C103,C104,C108,C110,C111,C112,C113,C114,C115,C116,C117,C121,C122,C123,C128,C130,C131,C132,C133,C134,C135,C136,C137,C141,C142,C143,C144,C145,C146,C147,C148,C149,C150,C151,C152,C153,C154,C155,C156,C157,C158,C159,C160,C161,C162,C163,C164,C195,C196,C197,C198,C199,C200,C201,C202,C203,C139,C140	Communica	Ceramic Capactors	805	100NF
2	1	C118	Communica	Ceramic Capactors	805	47NF
3	1	C138	Communica	Ceramic Capactors	805	6.9NF
4	2	C17,C28	Communica	Ceramic Capactors	805	330PF
5	2	C18,C27	Communica	Ceramic Capactors	805	680PF
6	6	C19,C23,C24,C29,C78,C79	Communica	Ceramic Capactors	805	100PF
7	49	C2,C8,C10,C11,C12,C13,C51,C52,C53,C54,C55,C56,C57,C58,C59,C60,C61,C62,C63,C64,C65,C70,C71,C72,C73,C74,C76,C77,C87,C88,C89,C90,C91,C93,C107,C127,C69,C75,C165,C166,C182,C184,C185,C187,C221,C222,C223,C224,C225	Communica	Ceramic Capactors	603	100NF
8	2	C20,C26	Communica	Ceramic Capactors	805	220PF
9	2	C204,C205	Communica	Ceramic Capactors	1206	4.7UF

10	2	C22,C25	Communica	Ceramic Capactors	805	2.2NF
11	14	C3,C4,C5,C14,C15,C80,C81 ,C82,C86,C92,C94,C105,C1 06,C129	Communica	Ceramic Capactors	805	1NF
12	1	C36	Communica	Ceramic Capactors	805	220NF
13	25	C38,C40,C42,C67,C168,C16 9,C170,C171,C172,C173,C1 74,C175,C176,C177,C178,C 179,C180,C186,C188,C189, C190,C191,C192,C193,C194	Communica	Ceramic Capactors	603	1000PF
14	3	C39,C119,C120	Communica	Ceramic Capactors	805	330NF
15	4	C66,C124,C125,C126	Communica	Ceramic Capactors	603	1NF
16	9	C68,C109,C167,C181,C183, C216,C218,C219,C220	Communica	Ceramic Capactors	603	10NF
17	7	C7,C9,C84,C85,C37,C41,C4 4	Communica	Ceramic Capactors	805	1UF
18	4	CT1,CT2,CT21,CT22	Communica	Electrolytic Capactors	CAPAE-E	47UF
19	6	CT12,CT14,CT35,CT43,CT4 4,CT53	Communica	Electrolytic Capactors	CAPAE-F	100UF
20	15	CT13,CT15,CT24,CT25,CT2 7,CT30,CT31,CT32,CT36,CT 37,CT42,CT54,CT56,CT61,C T62	AVX: TPSD107K01 6X0060	Tantulum Capactors	CAPTD	100UF
21	16	CT3,CT4,CT5,CT10,CT11,C T18,CT19,CT20,CT23,CT26, CT28,CT29,CT33,CT34,CT3 8,CT40	AVX: TPSD107C01 6X0060	Tantalum Capactors	CAPTD	10UF
22	6	CT39,CT49,CT50,CT57,CT5 9,CT60	AVX: TPME477M01 0X0023	Tantalum Capactors	CAPTD	470UF
23	3	CT47,CT48,CT55	AVX: TPME687J00 6 0023	Tantulum Capactors	CAPTD	680UF
24	2	CT6,CT7	Communica	Electrolytic Capactors	CAPAE-F	68UF
25	6	CT8,CT9,CT16,CT17,CT45, CT46	Communica	Electrolytic Capactors	CAPAE-A	10UF
26	8	D1,D2,D3,D4,D5,D6,D7,D8	RS:469-0758	MBRA120	DO214AC	
27	3	D9,D10,D11	RS:215-7237	US1B	DO214AC	
28	2	J1,J2	RS:542-2613	SDRAM DIMM	DIMM168	
29	8	L1,L2,L3,L4,L5,L6,L9,L7	RS:3275566		1206	600 OHM
31	1	LCD	PowerTip	PC2004LRU-AWA-B		
32	10	LED1,LED2,LED3,LED4,LED 5,LED6,LED7,LED8,LED9,L ED10	Communica	LEDS	1206	
33	1	P4	Mantech	Header	14x 2.54mm	

					header	
34	48	R1,R2,R7,R14,R20,R21,R23,R24,R25,R26,R27,R32,R34,R35,R51,R75,R106,R107,R108,R111,R114,R116,R118,R122,R123,R124,R125,R126,R129,R130,R134,R137,R138,R139,R145,R146,R147,R148,R149,R150,R151,R160,R161,R181,R184,R210,R211,R212	Communica	Resistors	805	0
35	1	R104	Communica	Resistors	805	39K
36	15	R115,R117,R120,R127,R128,R131,R132,R133,R135,R136,R140,R141,R142,R143,R144	Communica	Resistors	805	510
37	1	R119	Communica	Resistors	805	3.3
38	2	R16,	Communica	Resistors	805	10K
39	1	R162	Communica	Resistors	805	2.7K
40	1	R163	Communica	Resistors	805	3K
41	1	R17	Communica	Resistors	805	2.74k
42	1	R19	Communica	Resistors	1206	12.1K
43	2	R22,R92	Communica	Resistors	805	49.9K
44	4	R28,R90,R94,R102	Communica	Resistors	805	5.49K
45	2	R29,R97	Communica	Resistors	805	2.74K
46	27	R3,R10,R15,R18,R33,R36,R39,R42,R44,R45,R46,R53,R54,R56,R68,R69,R70,R73,R74,R79,R83,R103,R105,R109,R110,R112,R164	Communica	Resistors	805	10K
47	2	R30,R31	Communica	Resistors	805	50K
48	2	R37,R121,R38,R88	Communica	Resistors	805	2.00K
49	10	R4,R77,R152,R153,R154,R155,R156,R157,R158,R159	Communica	Resistors	805	33
50	2	R40,R41	Communica	Resistors	805	100K
51	5	R43,R57,R58,R84,R85	Communica	Resistors	805	100
52	3	R47,R48,R49	Communica	Resistors	805	1K
53	12	R5,R6,R11,R12,R13,R93,R182,R186,R188,R203,R213,R214	Communica	Resistors	805	4.7K
54	1	R50	Communica	Resistors	1206	1.8K
55	4	R60,R80,R96,R98	Communica	Resistors	805	604
56	1	R63	Communica	Resistors	805	20
57	4	R64,R65,R66,R67	Communica	Resistors	805	270
58	6	R71,R72,R8,R52,R55,R78	Communica	Resistors	805	470
59	2	R76,R113	Communica	Resistors	1206	2K
60	1	R86	Communica	Resistors	1206	0
61	2	R87,R100	Communica	Resistors	805	3.32K
62	2	R89,R99	Communica	Resistors	805	1.65K
63	6	R9,R59,R61,R62,R81,R82	Communica	Resistors	805	10
64	1	R91	Communica	Resistors	805	110K

65	2	R95,R101	Communica	Resistors	805	11K
66	9	RN1,RN2,RN3,RN4,RN29,RN30,RN31,RN32,RN33	Digikey	Resistors Networks	CAY16	22
67	4	RN11,RN12,RN26,RN13	Digikey	Resistors Networks	CAY16	10
68	12	RN14,RN15,RN16,RN17,RN18,RN19,RN20,RN21,RN22,RN23,RN24,RN25	Digikey	Resistors Networks	CAY16	18
69	2	RN27,RN36	Digikey	Resistors Networks	CAY16	27
70	1	RN28	Digikey	Resistors Networks	CAY16	15
71	2	RN34,RN35	Digikey	Resistors Networks	CAY16	47
72	6	RN5,RN6,RN7,RN8,RN9,RN10	Digikey	Resistors Networks	CAY16	33
73	7	SW1,SW2,SW3,SW4,SW5,SW6,SW9	Mantech	Tact switch	6mm through hole	
74	2	U1,U24	RS:4785086	100MHz oscillator: SG8002CA	SG8002CA	
75	2	U10,U43	Texas Instruments	PGA2500	SOIC28	
76	1	U11	Analog Devices	ADSP-TS201S	BGA576	
77	1	U12	Digikey	Altera EPCS1	SOIC8	
78	1	U15	RS:348-6249	SHOTKEYDPAK	DPAK	
79	1	U16	RS:3015471	8MHz ceramic oscillator: CSTCC8	CSTCC8	
80	1	U17	Texas Instruments	MAX3232	SOIC DW	
81	1	U18	Cypress	CY7B9950	32 TPQF	
82	1	U19	Microchip	PIC18F4550	44 TQFP	
83	1	U20	Mantech	LCD_CONNECTOR	2.54mm HEADER16SINGLE	
84	1	U21	Texas Instruments	UA79M05CKVURG3	KTP	
85	1	U22	RS:4788956	12.288MHz oscillator SPXO020465	SPXO020465	
86	1	U23	RS:237-6084	Compact Flash connector	N7E50-7516	
87	3	U25,U53,U61	Texas Instruments	PTH12000WAH	EUS5	
88	1	U26	Texas Instruments	PTN78000A	EUS5	
89	3	U27,U28,U37	Communica	SWITCH_4	SWITCH_DIL4	
90	1	U29	Texas Instruments	PTH08080WAH	EUF	

91	2	U30,U31	Communica	IO_HEADER_17	IO_HEADER_17	
92	2	U32,U33	Texas Instruments	Opamp LMV722	SOIC-D	
93	2	U35,U36	RS: 344386	XLR MIC CONNECTOR	XLR	
94	1	U38	Texas Instruments	TPS3836	DBV	
95	3	U39,U50,U51	Maxim	MAX3002EUP	TSSOP	
96	1	U40	Analog Devices	AD1836A	MQFP52	
97	1	U41	Digikey	Altera EP2C5Q208C7	Q208C	
98	1	U44	Communica	RJ45	AMP-MJ6	
99	1	U45	Atmel	AT49BV040	PLCC32RS	
100	1	U46	Vantek	Iceberq FAN	FAN	
101	1	U48	Mantech	DC_POWER jack	DC_POWER	
102	1	U49	RS:5474894	ACM9070	ACM9070	
103	1	U5	RS529-8218	USB type b connector	USB type B	
104	1	U52	Maxim	MAX6898	UDFN6	
105	1	U54	AMD	S29GL128P	TSOP56	
106	1	U55	Texas Instruments	TPS77825	TPS778XX	
106	1	U60	Texas Instruments	TPS778G1	TPS778XX	
107	2	U56,U57	Mantech	FANHEADER	3x2.54mm	
108	2	U58,U59	Mantech	FANHEADER	3x2.0mm	
109	1	U6	Mantech	RCA	4PRCA	
110	1	U62	Mantech	HEADER16	2.54mm HEADER16	
111	1	U64	-	Power switch	panel mounted	
112	1	U7	mantech	FUSE 3.5A	25mm	
113	3	U8,U34,U47	Manteck	HEADER10	2.54mm HEADER10	
114	3	U9,U13,U14	Texas Instruments	TLV1117KVU	KVURG3	
Total	581					

Appendix D

Extracts from Datasheets

The following pages contain extracts of the following datasheets:

1. Pages 207 to 209: Analog Devices AD1836A Multi-channel 96kHz Codec (29)
2. Pages 210 to 225: Compact Flash Specification 4.0 (11)

SPI CONTROL REGISTERS

Note that all control registers default to zero at power-up.

Table 12. Serial SPI Word Format

Register Address	Read/Write	Reserved	Data Field
15:12	**11**	**10**	**9:0**
4 Bits	1 = Read 0 = Write	0	10 Bits

Table 13. Register Addresses and Functions

Register Address				RD/WR	Reserved	Function
Bit 15	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bits 9:0**
0	0	0	0	0	0	DAC Control 1
0	0	0	1	0	0	DAC Control 2
0	0	1	0	0	0	DAC1L Volume
0	0	1	1	0	0	DAC1R Volume
0	1	0	0	0	0	DAC2L Volume
0	1	0	1	0	0	DAC2R Volume
0	1	1	0	0	0	DAC3L Volume
0	1	1	1	0	0	DAC3R Volume
1	0	0	0	0	0	ADC1L—Peak Level (Read-Only)
1	0	0	1	0	0	ADC1R—Peak Level (Read-Only)
1	0	1	0	0	0	ADC2L—Peak Level (Read-Only)
1	0	1	1	0	0	ADC2R—Peak Level (Read-Only)
1	1	0	0	0	0	ADC Control 1
1	1	0	1	0	0	ADC Control 2
1	1	1	0	0	0	ADC Control 3
1	1	1	1	0	0	Reserved

Table 14. DAC Control Register 1

Packed Mode: Eight channels are "packed" in DSDATA1 serial input. Packed Mode 128: Refer to Figure 7. Packed Mode 256: Refer to Figure 8.

Address	RD/WR	Reserved	Function					
			De-emphasis	Serial Mode	Data-Word Width	Power-Down	Interpolator Mode	Reserved
15, 14, 13, 12	**11**	**10**	**9, 8**	**7, 6, 5**	**4, 3**	**2**	**1**	**0**
0000	0	0	00 = None 01 = 44.1 kHz 10 = 32.0 kHz 11 = 48.0 kHz	000 = I^2S 001 = RJ 010 = DSP 011 = LJ 100 = Packed Mode 256 101 = Packed Mode 128 110 = Reserved 111 = Reserved	00 = 24 Bits 01 = 20 Bits 10 = 16 Bits 11 = Reserved	0 = Normal 1 = PWRDWN	0 = 8× (48 kHz) 1 = 4× (96 kHz)	0

Table 15. DAC Control Register 2

Address	RD/WR	Reserved	DAC Mute					
			DAC3R	DAC3L	DAC2R	DAC2L	DAC1R	DAC1L
15, 14, 13, 12	**11**	**10, 9, 8, 7, 6**	**5**	**4**	**3**	**2**	**1**	**0**
0001	0	00000	0 = On 1 = Mute	0 = On 1 = Mute	0 = On 1 = Mute	0 = On 1 = Mute	0 = On 1 = Mute	0 = On 1 = Mute

Table 16. DAC Volume Registers

Address	RD/WR	Reserved	Function
			Volume
15, 14, 13, 12	**11**	**10**	**9:0**
0010: DAC1L 0011: DAC1R 0100: DAC2L 0101: DAC2R 0110: DAC3L 0111: DAC3R	0	0	0 to 1023 in 1024 Linear Steps

Table 17. ADC Control Register 1

Address	RD/WR	Reserved	Function				
			Filter	Power-Down	Sample Rate	Left Gain	Right Gain
15, 14, 13, 12	**11**	**10, 9**	**8**	**7**	**6**	**5, 4, 3**	**2, 1, 0**
1100	0	00	0 = DC 1 = High Pass	0 = Normal 1 = PWRDWN	0 = 48 kHz 1 = 96 kHz	000 = 0 dB 001 = 3 dB 010 = 6 dB 011 = 9 dB 100 = 12 dB 101 = Reserved 110 = Reserved 111 = Reserved	000 = 0 dB 001 = 3 dB 010 = 6 dB 011 = 9 dB 100 = 12 dB 101 = Reserved 110 = Reserved 111 = Reserved

Table 18. ADC Control Register 2

Packed Mode: Eight channels are "packed" in ASDATA1 serial output. Packed Mode 128: Refer to Figure 5. Packed Mode 256: Refer to Figure 6. Packed Mode AUX: Refer to Figure 9 to Figure 11. Note that Packed AUX mode affects the entire chip, including the DAC serial mode.

Address	RD/WR	Reserved	Master/Slave AUX Mode	SOUT Mode	Word Width	ADC Mute			
						ADC2R	ADC2L	ADC1R	ADC1L
15, 14, 13, 12	**11**	**10**	**9**	**8, 7, 6**	**5, 4**	**3**	**2**	**1**	**0**
1101	0	0	0 = Slave 1 = Master	000 = I^2S 001 = RJ 010 = DSP 011 = LJ 100 = Packed Mode 256 101 = Packed Mode 128 110 = Packed Mode AUX	00 = 24 Bits 01 = 20 Bits 10 = 16 Bits 11 = Reserved	0 = On 1 = Mute	0 = On 1 = Mute	0 = On 1 = Mute	0 = On 1 = Mute

Table 19. ADC Control Register 3

When changing clock mode, other SPI bits that are written during the same SPI transaction may be lost. Therefore, it is recommended that these be set separately.

Address	RD/WR	Reserved	Clock Mode	Function					
				Left Differential I/P Select	Right Differential I/P Select	Left MUX/PGA Enable	Left MUX I/P Select	Right MUX/PGA Enable	Right MUX I/P Select
15, 14, 13, 12	**11**	**10, 9, 8**	**7, 6**	**5**	**4**	**3**	**2**	**1**	**0**
1110	0	000	00 = 256 × f_S 01 = 512 × f_S 10 = 768 × f_S	0 = Differential PGA Mode 1 = PGA/MUX Mode (Single-Ended Input)	0 = Differential PGA Mode 1 = PGA/MUX Mode (Single-Ended Input)	0 = Direct 1 = MUX/PGA	0 = I/P 0 1 = I/P 1	0 = Direct 1 = MUX/PGA	0 = I/P 0 1 = I/P 1

Table 20. ADC Peak Level Data Registers

Address	RD/WR	Reserved	Peak Level Data (10 Bits)	
			6 Data Bits	4 Fixed Bits
15, 14, 13, 12	**11**	**10**	**9:4**	**3:0**
1000 = ADC1L 1001 = ADC1R 1010 = ADC2L 1011 = ADC2R	1	0	000000 = 0.0 dBFS 000001 = −1.0 dBFS 000010 = −2.0 dBFS 000011 = −3.0 dBFS 111100 = −60 dBFS Min	0000 The 4 LSBs are always zero.

Table 30: CompactFlash Storage Card Configuration Registers Decoding

-CE2	-CE1	-REG	-OE	-WE	A10	A9	A8-A4	A3	A2	A1	A0	SELECTED REGISTER
X	0	0	0	1	0	1	00	0	0	0	0	Configuration Option Reg Read
X	0	0	1	0	0	1	00	0	0	0	0	Configuration Option Reg Write
X	0	0	0	1	0	1	00	0	0	1	0	Card Status Register Read
X	0	0	1	0	0	1	00	0	0	1	0	Card Status Register Write
X	0	0	0	1	0	1	00	0	1	0	0	Pin Replacement Register Read
X	0	0	1	0	0	1	00	0	1	0	0	Pin Replacement Register Write
X	0	0	0	1	0	1	00	0	1	1	0	Socket and Copy Register Read
X	0	0	1	0	0	1	00	0	1	1	0	Socket and Copy Register Write

Table 14: Attribute Memory Read Timing

Speed Version			300 ns	
Item	Symbol	IEEE Symbol	Min ns.	Max ns.
Read Cycle Time	tc(R)	tAVAV	300	
Address Access Time	ta(A)	tAVQV		300
Card Enable Access Time	ta(CE)	tELQV		300
Output Enable Access Time	ta(OE)	tGLQV		150
Output Disable Time from CE	tdis(CE)	tEHQZ		100
Output Disable Time from OE	tdis(OE)	tGHQZ		100
Address Setup Time	tsu (A)	tAVGL	30	
Output Enable Time from CE	ten(CE)	tELQNZ	5	
Output Enable Time from OE	ten(OE)	tGLQNZ	5	
Data Valid from Address Change	tv(A)	tAXQX	0	

Note: All times are in nanoseconds. Dout signifies data provided by the CompactFlash Storage Card or CF+ Card to the system. The -CE signal or both the -OE signal and the -WE signal shall be de-asserted between consecutive cycle operations.

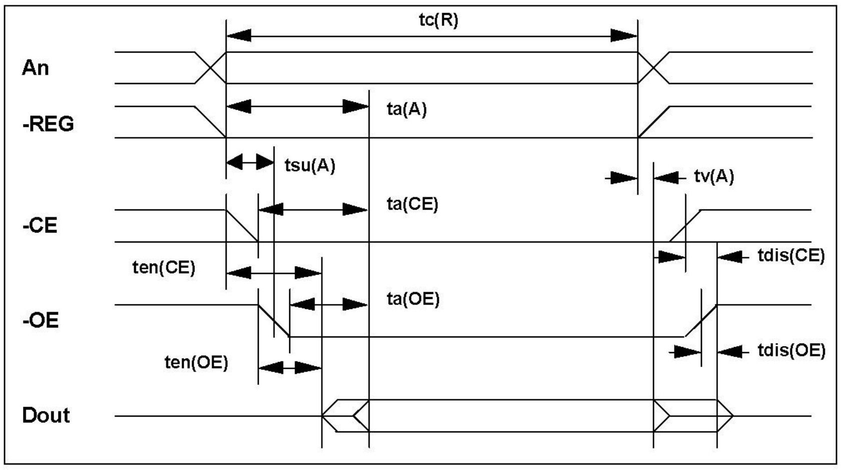


Figure 25: Attribute Memory Read Timing Diagram

4.3.11 Configuration Register (Attribute Memory) Write Timing Specification

The Card Configuration write access time is defined as 250 ns. Detailed timing specifications are shown in Table 15.

Table 15: Configuration Register (Attribute Memory) Write Timing

Speed Version			250 ns	
Item	**Symbol**	**IEEE Symbol**	**Min ns**	**Max ns**
Write Cycle Time	tc(W)	tAVAV	250	
Write Pulse Width	tw(WE)	tWLWH	150	
Address Setup Time	tsu(A)	tAVWL	30	
Write Recovery Time	trec(WE)	tWMAX	30	
Data Setup Time for WE	tsu(D-WEH)	tDVWH	80	
Data Hold Time	th(D)	tWMDX	30	

Note: All times are in nanoseconds. Din signifies data provided by the system to the CompactFlash Storage Card or CF+ Card.

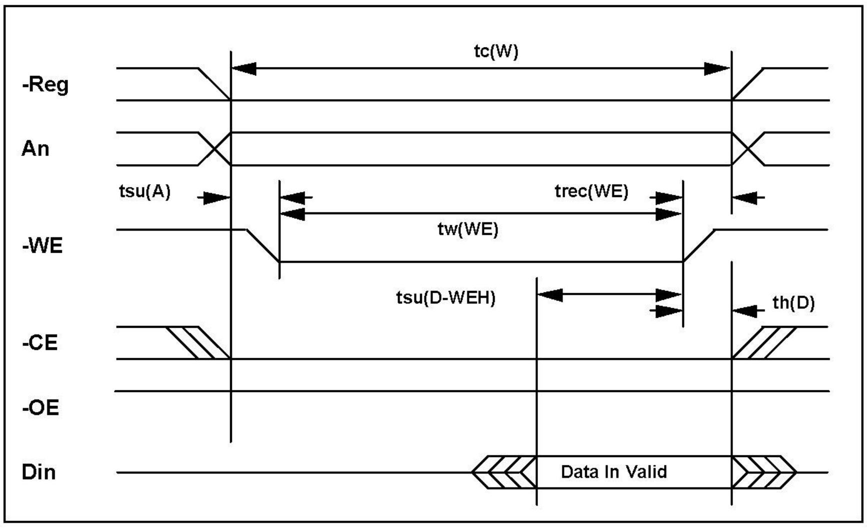


Figure 26: Configuration Register (Attribute Memory) Write Timing Diagram

4.3.12 Common Memory Read Timing Specification

Table 16: Common Memory Read Timing

		Cycle Time Mode:	250 ns		120 ns		100 ns		80 ns	
Item	**Symbol**	**IEEE Symbol**	**Min ns.**	**Max ns.**	**Min ns.**	**Max ns.**	**Min ns.**	**Max ns.**	**Min ns.**	**Max ns.**
Output Enable Access Time	ta(OE)	tGLQV		125		60		50		45
Output Disable Time from OE	tdis(OE)	tGHQZ		100		60		50		45
Address Setup Time	tsu(A)	tAVGL	30		15		10		10	
Address Hold Time	th(A)	tGHAX	20		15		15		10	
CE Setup before OE	tsu(CE)	tELGL	0		0		0		0	
CE Hold following OE	th(CE)	tGHEH	20		15		15		10	
Wait Delay Falling from OE	tv(WT-OE)	tGLWTV		35		35		35		na[1]
Data Setup for Wait Release	tv(WT)	tQVWTH		0		0		0		na[1]
Wait Width Time[2]	tw(WT)	tWTLWTH		350 (3000 for *CF+*)		350 (3000 for *CF+*)		350 (3000 for *CF+*)		na[1]

Notes: 1) –WAIT is not supported in this mode.

2) The maximum load on -WAIT is 1 LSTTL with 50 pF (40pF below 120nsec Cycle Time) total load. All times are in nanoseconds. Dout signifies data provided by the CompactFlash Storage Card or CF+ Card to the system. The -WAIT signal may be ignored if the -OE cycle to cycle time is greater than the Wait Width time. The Max Wait Width time can be determined from the Card Information Structure. The Wait Width time meets the PCMCIA PC Card specification of 12μs but is intentionally less in this specification.

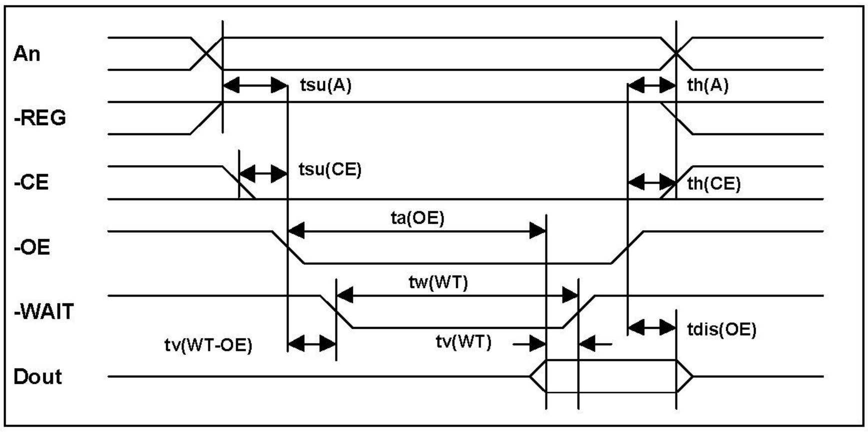


Figure 27: Common Memory Read Timing Diagram

4.3.13 Common Memory Write Timing Specification

Table 17: Common Memory Write Timing

		Cycle Time Mode:	250 ns		120 ns		100 ns		80 ns	
Item	**Symbol**	**IEEE Symbol**	**Min ns.**	**Max ns.**	**Min ns.**	**Max ns.**	**Min ns.**	**Max ns.**	**Min ns.**	**Max ns.**
Data Setup before WE	tsu (D-WEH)	tDVWH	80		50		40		30	
Data Hold following WE	th(D)	tWMDX	30		15		10		10	
WE Pulse Width	tw(WE)	tWLWH	150		70		60		55	
Address Setup Time	tsu(A)	tAVWL	30		15		10		10	
CE Setup before WE	tsu(CE)	tELWL	0		0		0		0	
Write Recovery Time	trec(WE)	tWMAX	30		15		15		15	
Address Hold Time	th(A)	tGHAX	20		15		15		15	
CE Hold following WE	th(CE)	tGHEH	20		15		15		10	
Wait Delay Falling from WE	tv (WT-WE)	tWLWTV		35		35		35		na[1]
WE High from Wait Release	tv(WT)	tWTHWH	0		0		0		na[1]	
Wait Width Time[2]	tw (WT)	tWTLWTH		350 (3000 for *CF*+)		350 (3000 for *CF*+)		350 (3000 for *CF*+)		na[1]

Notes: 1) –WAIT is not supported in this mode.

2) The maximum load on -WAIT is 1 LSTTL with 50 pF (40pF below 120nsec Cycle Time) total load. All times are in nanoseconds. Din signifies data provided by the system to the CompactFlash Storage Card. The -WAIT signal may be ignored if the -WE cycle to cycle time is greater than the Wait Width time. The Max Wait Width time can be determined from the Card Information Structure. The Wait Width time meets the PCMCIA PC Card specification of 12µs but is intentionally less in this specification.

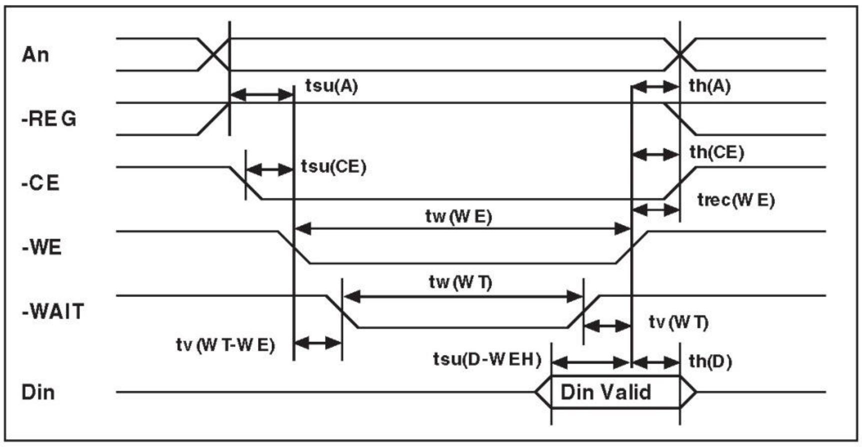


Figure 28: Common Memory Write Timing Diagram

6.1.3 Memory Mapped Addressing

When the CompactFlash Storage Card registers are accessed via memory references, the registers appear in the common memory space window: 0-2K bytes as follows:

Table 45: Memory Mapped Decoding

-REG	A10	A9-A4	A3	A2	A1	A0	Offset	-OE=0	-WE=0	Notes
1	0	X	0	0	0	0	0	Even RD Data	Even WR Data	1, 2
1	0	X	0	0	0	1	1	Error	Features	1, 2
1	0	X	0	0	1	0	2	Sector Count	Sector Count	
1	0	X	0	0	1	1	3	Sector No.	Sector No.	
1	0	X	0	1	0	0	4	Cylinder Low	Cylinder Low	
1	0	X	0	1	0	1	5	Cylinder High	Cylinder High	
1	0	X	0	1	1	0	6	Select Card /Head	Select Card/Head	
1	0	X	0	1	1	1	7	Status	Command	
1	0	X	1	0	0	0	8	Dup. Even RD Data	Dup. Even WR Data	2
1	0	X	1	0	0	1	9	Dup. Odd RD Data	Dup. Odd WR Data	2
1	0	X	1	1	0	1	D	Dup. Error	Dup. Features	2
1	0	X	1	1	1	0	E	Alt Status	Device Ctl	
1	0	X	1	1	1	1	F	Drive Address	Reserved	
1	1	X	X	X	X	0	8	Even RD Data	Even WR Data	3
1	1	X	X	X	X	1	9	Odd RD Data	Odd WR Data	3

Notes: 1) Register 0 is accessed with -CE1 low and -CE2 low as a word register on the combined Odd Data Bus and Even Data Bus (D15-D0). This register may also be accessed by a pair of byte accesses to the offset 0 with -CE1 low and -CE2 high. Note that the address space of this word register overlaps the address space of the Error and Feature byte-wide registers that lie at offset 1. When accessed twice as byte register with -CE1 low, the first byte to be accessed is the even byte of the word and the second byte accessed is the odd byte of the equivalent word access.

A byte access to address 0 with -CE1 high and -CE2 low accesses the error (read) or feature (write) register.

2) Registers at offset 8, 9 and D are non-overlapping duplicates of the registers at offset 0 and 1.

Register 8 is equivalent to register 0, while register 9 accesses the odd byte. Therefore, if the registers are byte accessed in the order 9 then 8 the data shall be transferred odd byte then even byte.

Repeated byte accesses to register 8 or 0 shall access consecutive (even then odd) bytes from the data buffer. Repeated word accesses to register 8, 9 or 0 shall access consecutive words from the data buffer. Repeated byte accesses to register 9 are not supported. However, repeated alternating byte accesses to registers 8 then 9 shall access consecutive (even then odd) bytes from the data buffer. Byte accesses to register 9 access only the odd byte of the data.

3) Accesses to even addresses between 400h and 7FFh access register 8. Accesses to odd addresses between 400h and 7FFh access register 9. This 1 Kbyte memory window to the data register is provided so that hosts can perform memory to memory block moves to the data register when the register lies in memory space.

6.1.5.1 Data Register (Address - 1F0h[170h];Offset 0,8,9)

The Data Register is a 16 bit register, and it is used to transfer data blocks between the CompactFlash Storage Card data buffer and the Host. This register overlaps the Error Register. Table 47: Data Register Access below describes the combinations of data register access and is provided to assist in understanding the overlapped Data Register and Error/Feature Register rather than to attempt to define general PCMCIA PC Card word and byte access modes and operations. See the PCMCIA PC Card Standard, for further definitions of the Card Accessing Modes for I/O and Memory cycles.

Note: Because of the overlapped registers, PC Card modes access to the 1F1h, 171h or offset 1 are not defined for word (-CE2 = 0 and -CE1 = 0) operations. These accesses are treated as accesses to the Word Data Register. The duplicated registers at offsets 8, 9 and Dh have no restrictions on the operations that can be performed by the socket.

Table 47: Data Register Access

Data Register Memory and I/O Modes	**-CE2**	**-CE1**	**A0**	**-REG**	**Offset**	**Data Bus**
Word Data Register	0	0	X	- [1]	0,8,9	D15-D0
Even Data Register	1	0	0	- [1]	0,8	D7-D0
Odd Data Register	1	0	1	- [1]	9	D7-D0
Odd Data Register	0	1	X	- [1]	8,9	D15-D8
Error / Feature Register	1	0	1	- [1]	1, Dh	D7-D0
Error / Feature Register	0	1	X	- [1]	1	D15-D8
Error / Feature Register	0	0	X	- [1]	Dh	D15-D8
Data Register True IDE Mode	**-CS1**	**-CS0**	**A0**	**-DMACK**	**Offset**	**Data Bus**
PIO Word Data Register	1	0	0	1	0	D15-D0
DMA Word Data Register	1	1	X	0	X	D15-D0
PIO Byte Data Register (Selected Using Set Features Command)	1	0	0	1	0	D7-D0

Notes: 1) -REG signal is mode dependent. Signal shall be 0 for I/O mode and 1 for Memory Mode.

Error Register (Address - 1F1h[171h]; Offset 1, 0Dh Read Only)

This register contains additional information about the source of an error when an error is indicated in bit 0 of the Status register. The bits are defined as follows:

D7	D6	D5	D4	D3	D2	D1	D0
BBK/ICRC	UNC	0	IDNF	0	ABRT	0	AMNF

Figure 50: Error Register

This register is also accessed in PC Card Modes on data bits D15-D8 during a read operation to offset 0 with -CE2 low and -CE1 high.

Bit 7 (BBK/ICRC): this bit is set when a Bad Block is detected. This bit is also set when an interface CRC error is detected in True IDE Ultra DMA modes of operation.

Bit 6 (UNC): this bit is set when an Uncorrectable Error is encountered.

Bit 5: this bit is 0.

Bit 4 (IDNF): the requested sector ID is in error or cannot be found.

Bit 3: this bit is 0.

Bit 2 (Abort) This bit is set if the command has been aborted because of a CompactFlash Storage Card status condition: (Not Ready, Write Fault, etc.) or when an invalid command has been issued.

Bit 1 This bit is 0.

Bit 0 (AMNF) This bit is set in case of a general error.

6.1.5.2 Feature Register (Address - 1F1h[171h]; Offset 1, 0Dh Write Only)

This register provides information regarding features of the CompactFlash Storage Card that the host can utilize. This register is also accessed in PC Card modes on data bits D15-D8 during a write operation to Offset 0 with -CE2 low and -CE1 high.

6.1.5.3 Sector Count Register (Address - 1F2h[172h]; Offset 2)

This register contains the numbers of sectors of data requested to be transferred on a read or write operation between the host and the CompactFlash Storage Card. If the value in this register is zero, a count of 256 sectors is specified. If the command was successful, this register is zero at command completion. If not successfully completed, the register contains the number of sectors that need to be transferred in order to complete the request.

6.1.5.4 Sector Number (LBA 7-0) Register (Address - 1F3h[173h]; Offset 3)

This register contains the starting sector number or bits 7-0 of the Logical Block Address (LBA) for any CompactFlash Storage Card data access for the subsequent command.

6.1.5.5 Cylinder Low (LBA 15-8) Register (Address - 1F4h[174h]; Offset 4)

This register contains the low order 8 bits of the starting cylinder address or bits 15-8 of the Logical Block Address.

6.1.5.6 Cylinder High (LBA 23-16) Register (Address - 1F5h[175h]; Offset 5)

This register contains the high order bits of the starting cylinder address or bits 23-16 of the Logical Block Address.

6.1.5.7 Drive/Head (LBA 27-24) Register (Address 1F6h[176h]; Offset 6)

The Drive/Head register is used to select the drive and head. It is also used to select LBA addressing instead of cylinder/head/sector addressing. The bits are defined as follows:

D7	D6	D5	D4	D3	D2	D1	D0
1	LBA	1	DRV	HS3	HS2	HS1	HS0

Figure 51: Drive/Head Register

Bit 7: this bit is specified as 1 for backward compatibility reasons. It is intended that this bit will become obsolete in a future revision of the specification. This bit is ignored by some controllers in some commands.

Bit 6: LBA is a flag to select either Cylinder/Head/Sector (CHS) or Logical Block Address Mode (LBA). When LBA=0, Cylinder/Head/Sector mode is selected. When LBA=1, Logical Block Address is selected. In Logical Block Mode, the Logical Block Address is interpreted as follows:

LBA7-LBA0: Sector Number Register D7-D0.

LBA15-LBA8: Cylinder Low Register D7-D0.

LBA23-LBA16: Cylinder High Register D7-D0.

LBA27-LBA24: Drive/Head Register bits HS3-HS0.

Bit 5: this bit is specified as 1 for backward compatibility reasons. It is intended that this bit will become obsolete in a future revisions of the specification. This bit is ignored by some controllers in some commands.

Bit 4 (DRV): DRV is the drive number. When DRV=0, drive (card) 0 is selected. When DRV=1, drive (card) 1 is selected. Setting this bit to 1 is obsolete in PCMCIA PC Card modes of operation. If the obsolete functionality is support by a CF Storage Card, the CompactFlash Storage Card is set to be Card 0 or 1 using the copy field (Drive #) of the PCMCIA PC Card Socket & Copy configuration register.

Bit 3 (HS3): when operating in the Cylinder, Head, Sector mode, this is bit 3 of the head number. It is Bit 27 in the Logical Block Address mode.

Bit 2 (HS2): when operating in the Cylinder, Head, Sector mode, this is bit 2 of the head number. It is Bit 26 in the Logical Block Address mode.

Bit 1 (HS1): when operating in the Cylinder, Head, Sector mode, this is bit 1 of the head number. It is Bit 25 in the Logical Block Address mode.

Bit 0 (HS0): when operating in the Cylinder, Head, Sector mode, this is bit 0 of the head number. It is Bit 24 in the Logical Block Address mode.

6.1.5.8 Status & Alternate Status Registers (Address 1F7h[177h]&3F6h[376h]; Offsets 7 & Eh)

These registers return the CompactFlash Storage Card status when read by the host. Reading the Status register does clear a pending interrupt while reading the Auxiliary Status register does not. The status bits are described as follows:

D7	D6	D5	D4	D3	D2	D1	D0
BUSY	RDY	DWF	DSC	DRQ	CORR	0	ERR

Figure 52: Status & Alternate Status Register

Bit 7 (BUSY): the busy bit is set when the CompactFlash Storage Card has access to the command buffer and registers and the host is locked out from accessing the command register and buffer. No other bits in this register are valid when this bit is set to a 1.

During the data transfer of DMA commands, the Card shall not assert DMARQ unless either the BUSY bit, the DRQ bit, or both are set to one.

Bit 6 (RDY): RDY indicates whether the device is capable of performing CompactFlash Storage Card operations. This bit is cleared at power up and remains cleared until the CompactFlash Storage Card is ready to accept a command.

Bit 5 (DWF): This bit, if set, indicates a write fault has occurred.

Bit 4 (DSC): This bit is set when the CompactFlash Storage Card is ready.

Bit 3 (DRQ): The Data Request is set when the CompactFlash Storage Card requires that information be transferred either to or from the host through the Data register.

During the data transfer of DMA commands, the Card shall not assert DMARQ unless either the BUSY bit, the DRQ bit, or both are set to one.

Bit 2 (CORR): This bit is set when a Correctable data error has been encountered and the data has been corrected. This condition does not terminate a multi-sector read operation.

Bit 1 (IDX): This bit is always set to 0.

Bit 0 (ERR): This bit is set when the previous command has ended in some type of error. The bits in the Error register contain additional information describing the error. It is recommended that media access commands (such as Read Sectors and Write Sectors) that end with an error condition should have the address of the first sector in error in the command block registers.

6.1.5.9 Device Control Register (Address - 3F6h[376h]; Offset Eh)

This register is used to control the CompactFlash Storage Card interrupt request and to issue an ATA soft reset to the card. This register can be written even if the device is BUSY. The bits are defined as follows:

D7	D6	D5	D4	D3	D2	D1	D0
X(0)	X(0)	X(0)	X(0)	X(0)	SW Rst	-IEn	0

Figure 53: Device Control Register

Bit 7: this bit is ignored by the CompactFlash Storage Card. The host software should set this bit to 0.

Bit 6: this bit is ignored by the CompactFlash Storage Card. The host software should set this bit to 0.

Bit 5: this bit is ignored by the CompactFlash Storage Card. The host software should set this bit to 0.

Bit 4: this bit is ignored by the CompactFlash Storage Card. The host software should set this bit to 0.

Bit 3: this bit is ignored by the CompactFlash Storage Card. The host software should set this bit to 0.

Bit 2 (SW Rst): this bit is set to 1 in order to force the CompactFlash Storage Card to perform an AT Disk controller Soft Reset operation. This does not change the PCMCIA PC Card Card Configuration Registers (see Section 4.4.4 to 4.4.9) as a hardware Reset does. The Card remains in Reset until this bit is reset to '0.'

Bit 1 (-IEn): the Interrupt Enable bit enables interrupts when the bit is 0. When the bit is 1, interrupts from the CompactFlash Storage Card are disabled. This bit also controls the Int bit in the Configuration and Status Register. This bit is set to 0 at power on and Reset.

Bit 0: this bit is ignored by the CompactFlash Storage Card.

6.1.5.10 Card (Drive) Address Register (Address 3F7h[377h]; Offset Fh)

This register is provided for compatibility with the AT disk drive interface. It is recommended that this register not be mapped into the host's I/O space because of potential conflicts on Bit 7. The bits are defined as follows:

D7	D6	D5	D4	D3	D2	D1	D0
X	-WTG	-HS3	-HS2	-HS1	-HS0	-nDS1	-nDS0

Figure 54: Card (Drive) Address Register

Bit 7: this bit is unknown.

Implementation Note:

Conflicts may occur on the host data bus when this bit is provided by a Floppy Disk Controller operating at the same addresses as the CompactFlash Storage Card. Following are some possible solutions to this problem for the PCMCIA PC Card implementation:

1) Locate the CompactFlash Storage Card at a non-conflicting address, i.e. Secondary address (377) or in an independently decoded Address Space when a Floppy Disk Controller is located at the Primary addresses.

2) Do not install a Floppy and a CompactFlash Storage Card in the system at the same time.

3) Implement a socket adapter that can be programmed to (conditionally) tri-state D7 of I/0 address 3F7h/377h when a CompactFlash Storage Card is installed and conversely to tri-state D6-D0 of I/O address 3F7h/377h when a floppy controller is installed.

4) Do not use the CompactFlash Storage Card's Drive Address register. This may be accomplished by either a) If possible, program the host adapter to enable only I/O addresses 1F0h-1F7h, 3F6h (or 170h-177h, 176h) to the CompactFlash Storage Card or b) if provided use an additional Primary / Secondary configuration in the CompactFlash Storage Card which does not respond to accesses to I/O locations 3F7h and 377h. With either of these implementations, the host software shall not attempt to use information in the Drive Address Register.

Bit 6 (-WTG): this bit is 0 when a write operation is in progress; otherwise, it is 1.

Bit 5 (-HS3): this bit is the negation of bit 3 in the Drive/Head register.

Bit 4 (-HS2): this bit is the negation of bit 2 in the Drive/Head register.

Bit 3 (-HS1): this bit is the negation of bit 1 in the Drive/Head register.

Bit 2 (-HS0): this bit is the negation of bit 0 in the Drive/Head register.

Bit 1 (-nDS1): this bit is 0 when drive 1 is active and selected.

Bit 0 (-nDS0): this bit is 0 when the drive 0 is active and selected.

6.2.1 CF-ATA Command Set

Table 48: CF-ATA Command Set summarizes the CF-ATA command set with the paragraphs that follow describing the individual commands and the task file for each.

Table 48: CF-ATA Command Set

Class	COMMAND	Code	FR	SC	SN	CY	DH	LBA
1	Check Power Mode	E5h or 98h	-	-	-	-	D	-
1	Execute Drive Diagnostic	90h	-	-	-	-	D	-
1	Erase Sector(s)	C0h	-	Y	Y	Y	Y	Y
1	Flush Cache	E7h	-	-	-	-	D	-
2	Format Track	50h	-	Y	-	Y	Y	Y
1	Identify Device	ECh	-	-	-	-	D	-
1	Idle	E3h or 97h	-	Y	-	-	D	-
1	Idle Immediate	E1h or 95h	-	-	-	-	D	-
1	Initialize Drive Parameters	91h	-	Y	-	-	Y	-
1	Key Management Structure Read	B9 Feature 0-127	C	C	C	C	D C	-
1	Key Management Read Keying Material	B9 Feature 80	C	C	C	C	D C	-
2	Key Management Change Key Management Value	B9 Feature 81	C	C	C	C	D C	-
1	NOP	00h	-	-	-	-	D	-
1	Read Buffer	E4h	-	-	-	-	D	-
1	Read DMA	C8h	-	Y	Y	Y	Y	Y
1	Read Long Sector	22h or 23h	-	-	Y	Y	Y	Y
1	Read Multiple	C4h	-	Y	Y	Y	Y	Y
1	Read Sector(s)	20h or 21h	-	Y	Y	Y	Y	Y
1	Read Verify Sector(s)	40h or 41h	-	Y	Y	Y	Y	Y
1	Recalibrate	1Xh	-	-	-	-	D	-
1	Request Sense	03h	-	-	-	-	D	-
1	Security Disable Password	F6h	-	-	-	-	D	-
1	Security Erase Prepare	F3h	-	-	-	-	D	-
1	Security Erase Unit	F4h	-	-	-	-	D	-
1	Security Freeze Lock	F5h	-	-	-	-	D	-
1	Security Set Password	F1h	-	-	-	-	D	-
1	Security Unlock	F2h	-	-	-	-	D	-
1	Seek	7Xh	-	-	Y	Y	Y	Y
1	Set Features	EFh	Y	-	-	-	D	-
1	Set Multiple Mode	C6h	-	Y	-	-	D	-

Class	COMMAND	Code	FR	SC	SN	CY	DH	LBA
1	Set Sleep Mode	E6h or 99h	-	-	-	-	D	-
1	Standby	E2h or 96h	-	-	-	-	D	-
1	Standby Immediate	E0h or 94h	-	-	-	-	D	-
1	Translate Sector	87h	-	Y	Y	Y	Y	Y
1	Wear Level	F5h	-	-	-	-	Y	-
2	Write Buffer	E8h	-	-	-	-	D	-
2	Write DMA	CAh	-	Y	Y	Y	Y	Y
2	Write Long Sector	32h or 33h	-	-	Y	Y	Y	Y
3	Write Multiple	C5h	-	Y	Y	Y	Y	Y
3	Write Multiple w/o Erase	CDh	-	Y	Y	Y	Y	Y
2	Write Sector(s)	30h or 31h	-	Y	Y	Y	Y	Y
2	Write Sector(s) w/o Erase	38h	-	Y	Y	Y	Y	Y
3	Write Verify	3Ch	-	Y	Y	Y	Y	Y

Definitions:

- FR = Features Register
- SC = Sector Count Register
- SN = Sector Number Register
- CY = Cylinder Registers
- DH = Card/Drive/Head Register
- LBA = Logical Block Address Mode Supported (see command descriptions for use).
- Y - The register contains a valid parameter for this command. For the Drive/Head Register Y means both the CompactFlash Storage Card and head parameters are used; D - only the CompactFlash Storage Card parameter is valid and not the head parameter; C – The register contains command specific data (see command descriptions for use).

6.2.1.6 Identify Device – ECh

Bit ->	7	6	5	4	3	2	1	0
Command (7)	ECh							
C/D/H (6)	X	X	X	Drive	X			
Cyl High (5)	X							
Cyl Low (4)	X							
Sec Num (3)	X							
Sec Cnt (2)	X							
Feature (1)	X							

Figure 60: Identify Device

The Identify Device command enables the host to receive parameter information from the CompactFlash Storage Card. This command has the same protocol as the Read Sector(s) command. The parameter words in the buffer have the arrangement and meanings defined in Table 50. All reserved bits or words are zero. Hosts should not depend on Obsolete words in Identify Device containing 0. Table 50 specifies each field in the data returned by the Identify Device Command. In Table 50, X indicates a numeric nibble value specific to the card and aaaa indicates an ASCII string specific to the particular drive.

Table 50: Identify Device Information

Word Address	Default Value	Total Bytes	Data Field Type Information
0	848Ah	2	General configuration - signature for the CompactFlash Storage Card
	0XXX	2	General configuration – Bit Significant with ATA-4 definitions.
1	XXXXh	2	Default number of cylinders
2	0000h	2	Reserved
3	00XXh	2	Default number of heads
4	0000h	2	Obsolete
5	0000h	2	Obsolete
6	XXXXh	2	Default number of sectors per track
7-8	XXXXh	4	Number of sectors per card (Word 7 = MSW, Word 8 = LSW)
9	XXXXh	2	Obsolete
10-19	aaaa	20	Serial number in ASCII (Right Justified)
20	0000h	2	Obsolete
21	0000h	2	Obsolete
22	0004h	2	Number of ECC bytes passed on Read/Write Long Commands
23-26	aaaa	8	Firmware revision in ASCII. Big Endian Byte Order in Word
27-46	aaaa	40	Model number in ASCII (Left Justified) Big Endian Byte Order in Word
47	XXXXh	2	Maximum number of sectors on Read/Write Multiple command
48	0000h	2	Reserved

Word Address	Default Value	Total Bytes	Data Field Type Information
49	XX00h	2	Capabilities
50	0000h	2	Reserved
51	0X00h	2	PIO data transfer cycle timing mode
52	0000h	2	Obsolete
53	000Xh	2	Field Validity
54	XXXXh	2	Current numbers of cylinders
55	XXXXh	2	Current numbers of heads
56	XXXXh	2	Current sectors per track
57-58	XXXXh	4	Current capacity in sectors (LBAs)(Word 57 = LSW, Word 58 = MSW)
59	01XXh	2	Multiple sector setting
60-61	XXXXh	4	Total number of sectors addressable in LBA Mode
62	0000h	2	Reserved
63	0X0Xh	2	Multiword DMA transfer. In PC Card modes this value shall be 0h
64	00XXh	2	Advanced PIO modes supported
65	XXXXh	2	Minimum Multiword DMA transfer cycle time per word. In PC Card modes this value shall be 0h
66	XXXXh	2	Recommended Multiword DMA transfer cycle time. In PC Card modes this value shall be 0h
67	XXXXh	2	Minimum PIO transfer cycle time without flow control
68	XXXXh	2	Minimum PIO transfer cycle time with IORDY flow control
69-79	0000h	20	Reserved
80-81	0000h	4	Reserved – CF cards do not return an ATA version
82-84	XXXXh	6	Features/command sets supported
85-87	XXXXh	6	Features/command sets enabled
88	XXXXh	2	Ultra DMA Mode Supported and Selected
89	XXXXh	2	Time required for Security erase unit completion
90	XXXXh	2	Time required for Enhanced security erase unit completion
91	XXXXh	2	Current Advanced power management value
92-127	0000h	72	Reserved
128	XXXXh	2	Security status
129-159	0000h	64	Vendor unique bytes
160	XXXXh	2	Power requirement description
161	0000h	2	Reserved for assignment by the CFA
162	0000h	2	Key management schemes supported
163	XXXXh	2	CF Advanced True IDE Timing Mode Capability and Setting
164	XXXXh	2	CF Advanced PC Card I/O and Memory Timing Mode Capability
165-175	0000h	22	Reserved for assignment by the CFA

6.2.1.18 Read Sector(s) - 20h or 21h

Bit ->	7	6	5	4	3	2	1	0
Command (7)	20h or 21h							
C/D/H (6)	1	LBA	1	Drive	Head (LBA 27-24)			
Cyl High (5)	Cylinder High (LBA 23-16)							
Cyl Low (4)	Cylinder Low (LBA 15-8)							
Sec Num (3)	Sector Number (LBA 7-0)							
Sec Cnt (2)	Sector Count							
Feature (1)	X							

Figure 72: Read Sector(s)

This command reads from 1 to 256 sectors as specified in the Sector Count register. A sector count of 0 requests 256 sectors. The transfer begins at the sector specified in the Sector Number Register. When this command is issued and after each sector of data (except the last one) has been read by the host, the CompactFlash Storage Card sets BSY, puts the sector of data in the buffer, sets DRQ, clears BSY, and generates an interrupt. The host then reads the 512 bytes of data from the buffer.

At command completion, the Command Block Registers contain the cylinder, head and sector number of the last sector read. If an error occurs, the read terminates at the sector where the error occurred. The Command Block Registers contain the cylinder, head, and sector number of the sector where the error occurred. The flawed data is pending in the sector buffer.

6.2.1.41 Write Sector(s) - 30h or 31h

Bit ->	7	6	5	4	3	2	1	0
Command (7)	30h or 31h							
C/D/H (6)	1	LBA	1	Drive	Head (LBA 27-24)			
Cyl High (5)	Cylinder High (LBA 23-16)							
Cyl Low (4)	Cylinder Low (LBA 15-8)							
Sec Num (3)	Sector Number (LBA 7-0)							
Sec Cnt (2)	Sector Count							
Feature (1)	X							

Figure 95: Write Sector(s)

This command writes from 1 to 256 sectors as specified in the Sector Count Register. A sector count of zero requests 256 sectors. The transfer begins at the sector specified in the Sector Number Register. When this command is accepted, the CompactFlash Storage Card sets BSY, then sets DRQ and clears BSY, then waits for the host to fill the sector buffer with the data to be written. No interrupt is generated to start the first host transfer operation. No data should be transferred by the host until BSY has been cleared by the host.

For multiple sectors, after the first sector of data is in the buffer, BSY shall be set and DRQ shall be cleared. After the next buffer is ready for data, BSY is cleared, DRQ is set and an interrupt is generated. When the final sector of data is transferred, BSY is set and DRQ is cleared. It shall remain in this state until the command is completed at which time BSY is cleared and an interrupt is generated.

If an error occurs during a write of more than one sector, writing terminates at the sector where the error occurs. The Command Block Registers contain the cylinder, head and sector number of