UTOPIA 3.0:

Power Cycling Controller Board

By

Allan Chan in collaboration with Western Digital

COMPUTER ENGINEERING DEPARTMENT

California Polytechnic State University

San Luis Obispo

Faculty Advisor: Dr. John Oliver

March 2010

Table of Contents

Table of Figures

ABSTRACT

The Universal Test & Observation Power Interface Adapter (UTOPIA) is a device used for hard drive testing by Western Digital. Utopia is designed to provide an interface to communicate with a hard drive and perform power cycling functions on a hard drive.

The purpose of this project is to develop a prototype that retains the features of the previous Utopia boards while including two new features to be used in the next design: voltage margining and improved Serial Input/Output (SIO) drive control.

This document describes the design of version 3.0 of UTOPIA starting with component selection, schematic design and, PCB layout and, manufacturing. Necessary firmware modifications to support the new voltage margining features are described. Finally, testing results of the Utopia 3.0 is shown.

I. INTRODUCTION

The purpose of the UTOPIA device is to provide an interface for power cycling and communicating with a hard drive in a testing environment. During hard drive development, it is common to test the hard drives' various functions such as: spinning up or down the hard disk platters, power cycling, and performing random reads or writes onto the disk. UTOPIA allows engineers to communicate with a hard drive using SIO, which allows direct control over the hard drive. Utopia is typically used for small scale testing on a lab bench, but also has uses in long term reliability testing. Using UTOPIA, hard drive tests can be automated and data can be gathered for failure analysis.

Problem Statement

The objective of this project is to upgrade the existing UTOPIA 2.5 design to UTOPIA 3.0 by developing a prototype that retains the features of the current device and includes two new features: voltage margining and improved Serial Input/Output (SIO) drive control.

II. PROJECT REQUIREMENTS

Chapter 2 covers the features and limitations of the previous UTOPIA 2.5 design. The proposed new features for UTOPIA 3.0 are then discussed, which culminates with a list of the requirements and constraints for the design of UTOPIA 3.0.

UTOPIA 2.5 Features

UTOPIA 2.5 is the previous design and is also the 3rd generation of the device (after UTOPIA 1.0 and 2.0). It features a PIC 18F4550 microcontroller that includes a built-in USB transceiver to communicate with a lab bench PC. Users are able to perform power control and data acquisition. A serial input/output (SIO) interface is provided by the microcontroller and its purpose is to communicate with the hard drive. It is a low level channel that can control and return information from the hard drive. The UTOPIA 2.5 board also has two power connections, one for power input and the other for power output. The power input contains two rails that are referred to as the 5V rail and 12V rail in which each rail is connected to an external power supply that supplies power to the outputs. Each rail on the power output is independently controlled and support switching and ramping functionality. Figure 1, below, shows a simplified hardware block diagram for UTOPIA 2.5.

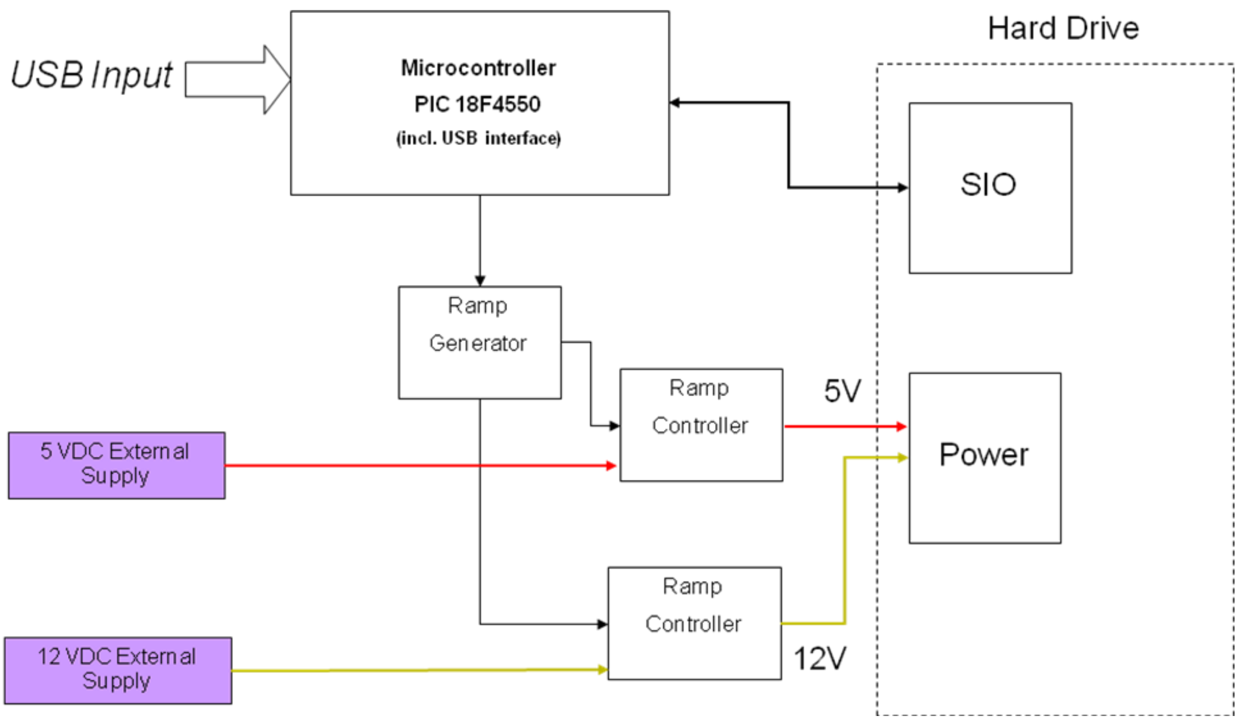


Figure 1: UTOPIA 2.5 Hardware Block Diagram

UTOPIA 2.5 Limitations

The hardware configuration in UTOPIA 2.5 prevents the use of WD hard drive test script software such as TREX, WINDEX, or SASDEX (WD proprietary testing scripts) while the hard drive is in SIO mode. This is a limitation due to the microcontroller. It requires a special command to enable SIO mode and it is not possible to adjust power cycling settings while in this mode. Also, large commands cannot be sent or received because the data buffers can only store up to approximately 1 Kbytes of data before overflowing. The on-chip USB transceiver in the microcontroller is only capable of USB 1.0 specifications which limit the maximum data transfer speeds up to 12 Mbits/second [1].

Although UTOPIA 2.5 supports independent control over the 5V and 12V power outputs, it lacks the ability to regulate the voltage supply. More specifically, it can toggle the power outputs on or off, but does not allow voltage margining. The existing design requires two external power supplies and power is manually set on the external equipment. For most applications, it is only a minor inconvenience. However, it becomes a problem when the device is used in the torture stand. The torture stand is a method for long-term reliability testing. In this application, multiple drives share a single power supply and are tested at the same voltages. This prevents the possibility of power cycling multiple hard drives at different voltages because only one test routine can be performed at any given time. The consequence is delays if more than one test routine is needed to be run simultaneously. There is also no over-voltage-protection (OVP) feature to prevent the risk of damaging the hard drives due to extremely high voltage settings. It is essential that the user has knowledge of the power limits of the hard drive and take precautions to prevent exceeding those limits.

Utopia 3.0 Proposed Features

In addition to the features and capabilities of UTOPIA v2.5, the new design adds three additional improvements. The following new features will be implemented in UTOPIA 3.0: independent interfaces for voltage control and SIO mode, high performance microcontroller, and voltage margining.

Independent Interfaces for Voltage Control and SIO Mode

In order to address the SIO mode issue, the hardware design in UTOPIA 2.5 will be modified to include a dual USB-to-UART IC chip. The proposed change will provide the lab bench PC with independent communication interfaces for voltage control and SIO mode. Thus, it will be possible to adjust power cycling settings and communicate with the hard drive via SIO simultaneously. By providing a direct channel between the PC and hard drive's SIO port, there is a high possibility that test script software such as SASDEX will be compatible. One of the goals in the new design is support of USB 2.0 specifications which will allow higher data transfer rates. With respect to the microcontroller, the proposed piece of hardware is expected to provide an improvement in data transfer rate because of the increase in available bandwidth and larger data buffers.

High Performance Microcontroller

While it is possible to reuse the current microcontroller, it will be advantageous to reevaluate the needs and expectations of the microcontroller. The microcontroller must retain the existing functionalities of UTOPIA 2.5, but there is opportunity for an upgrade by replacing it with a part that supports improved speeds and capabilities. The addition of the dual USB-UART IC renders the USB

function on the 18F4550 useless. Therefore, lower cost can be achieved by sourcing a part that excludes USB support. It is desired to acquire a microcontroller that includes high speed internal oscillator and faster CPU speeds.

Voltage Margining

The ability to provide onboard voltage margining will greatly improve the complexity of tests that could be performed. The UTOPIA board will allow voltages at the power outputs to be adjusted in software. Each rail will be adjustable independently and require only a single power supply rather than the two supplies required by UTOPIA 2.5. The range of power supply input is expected to range between 17 VDC and 24 VDC. The range of power supply output is expected to be able to supply up to 10 VDC for the 5V rail and 18 VDC for the 12V rail. To implement this feature, it will require completely new hardware and software to control it. Numerous benefits are achieved such as allowing multiple hard drives to be connected to a single power supply and power cycling them at different voltages, included voltage protection in software, and the ability to voltage margin with Western Digital test script software.

Design Requirements

Table 1 lists the overall project requirements and justification.

Table 1: Project Requirements

Engineering Requirements	Justification
1. The UTOPIA board components must be USB bus powered	Reducing the dependency on additional equipment will provide easy usability
2. The Utopia board must support USB 2.0 specifications of 480 Mbits/second	Higher bandwidth increases the rate of data transfers
3. The UTOPIA board components must be able to withstand ambient operating temperatures between -10°C and 75°C	In the torture stand, the devices may be subjected to temperatures ranging between -10°C and 75°C
4. Provide greater than 1Kbyte transmit and receive buffers on USB interface	Allows more data to be transferred at a time
5. Provide independent interfaces between PC and microcontroller, PC and hard drive SIO port	Allows simultaneous communication for SIO drive control and voltage control
6. Support UART baud rate of at least 57 Kbaud	Improves board response
7. Microcontroller must provide following: 1 UART, 1 SPI, 8 analog inputs, 2 PWM outputs	Retain UTOPIA 2.5 features
8. Power input must accept an allowable range between 17 VDC and 24 VDC	Provide compatibility with a wide range of power supply outputs
9. Power outputs must supply up to 10 VDC on the 5V rail and 18 VDC on the 12V rail	Allow wide range of voltage margining functionality
10. Allow independent control of voltage margining on the 5V and 12V rail	Each supply is separately adjustable for maximum flexibility

Constraints

The new UTOPIA 3.0 design must retain the features that are already in place from UTOPIA 2.5 which include data acquisition, voltage ramping, electrical switching, and SIO support. The UTOPIA board should be USB bus powered, as in the previous design, to reduce the necessary equipment required to operate the device. One application of the device is the torture stand. In this application, the devices are placed inside thermal chambers that may be subjected up to 75°C. All of the board components must be able to operate correctly at that temperature. Overall board size should be kept to a minimum and provide enough space to fit all necessary components. Overall board cost is not a huge concern. Since this is the initial prototype, the design is more likely regarded as a proof of concept. However, production runs should be approximately $5 per board in batches of 100 devices.

III. DESIGN

The overall system architecture and hardware block diagram is presented to provide a overview of how the device works. It is followed by parts selection for three components of the UTOPIA device: USB-UART IC, microcontroller, and voltage regulator. The chapter concludes with the test plan for the design.

System Architecture

The Utopia device provides an interface for the Lab Bench computer, external power supply, and the hard drive under test. Figure 2 is an example of a typical UTOPIA 3.0 configuration.

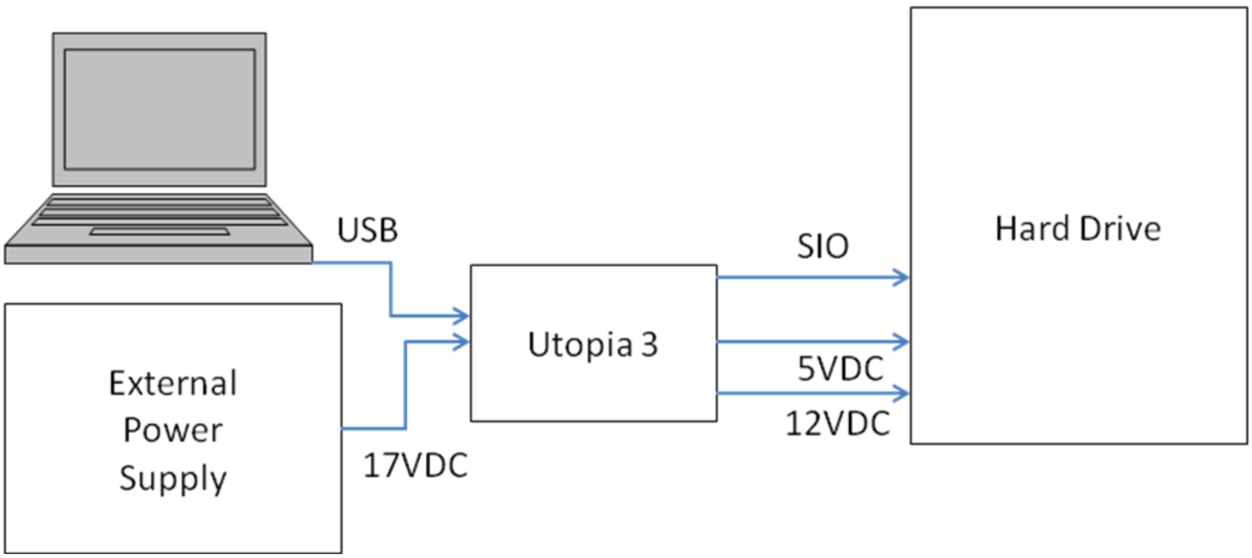


Figure 2: Top Level System Design

Lab Bench Personal Computer

The lab bench personal computer (PC) is used to operate and control the UTOPIA device. It is connected to the board via USB using a USB to mini-USB cable. The PC communicates to the board via an open COM port using any communication utility that supports serial protocols such as Hyperterminal or WD hard drive test script software such as SASDEX.

External Power Supply

The power supply is expected to provide an output of up to 17VDC during normal operations. The purpose of the power supply is to provide power to the hard drive. It is desirable to utilize a variable type power source, but not required. Any power source can be used as long as it outputs DC voltages on 17 V. The Utopia3 board will step down the input voltage to the desired voltages on each set of outputs. It is desired to supply the Utopia3 board with 17VDC to increase the range of voltage margining. The power supply is connected to the UTOPIA device's input header using a standard 4-pin Molex connector. The positive and negative terminals are located on pins 1 and 2 of the input header where pin 1 is identified as a square pad.

Hard Drive

The drive under test (DUT) is connected to the output header and the SIO connector of the Utopia board. The Utopia board powers the DUT from the output header and communicates with the hard drive via the SIO port.

Hardware Block Diagram

Figure 3 shows the hardware block diagram of UTOPIA 3.0. The components shaded in color are new features of the design.

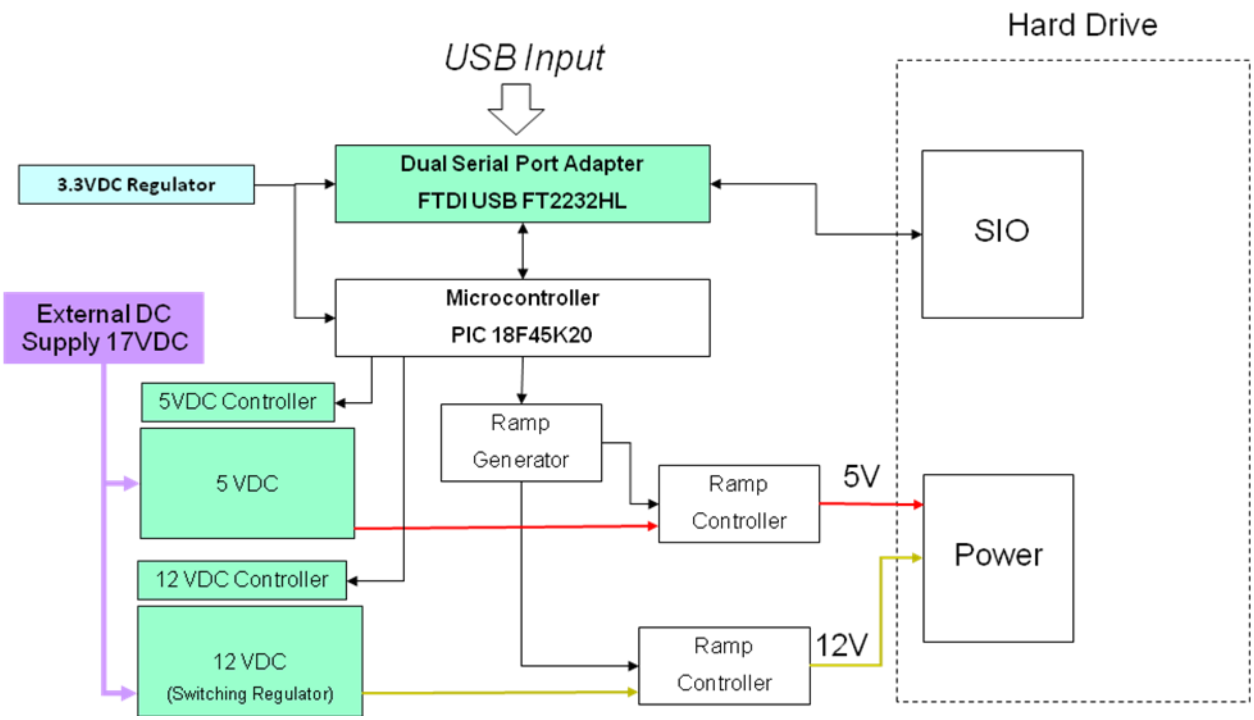


Figure 3: Utopia 3.0 Hardware Block Diagram

1. Dual USB-to-Serial Port Adapter

The dual USB-to-Serial Port Adapter allows the lab bench PC to communicate with the PIC microcontroller and hard drive via a USB-to-UART interface. Each interface is independent and allows simultaneous control of the microcontroller and hard drive. Data can be transmitted or received between the PC and microcontroller, likewise, between the PC and hard drive SIO port. The component is made by FTDI LTD and is known as the FT2232HL.

2. Microcontroller

The microcontroller is responsible for controlling the power outputs that interface the hard drive such as ramping up/down, switching on/off, and voltage margining. It can also take measurements of voltages and currents being output to the hard drive. The PIC microcontroller controls the regulator controllers and ramp controllers using the SPI communication protocol. It is the PIC 18F45K20, which is made by Microchip.

3. 5V and 12V Switch

The 5V and 12V switch consists of the ramp generator and ramp controllers. The ramp generator can set the ramping period between 1 microsecond and 1 second. Each controller can independently toggle the 5V or 12V output to be either on or off.

4. 5V and 12V Switching Regulator

Two switching regulators provide voltage margining for the 5V and 12V rail independently. The input is connected to the external power supply. Each output is interfaced with their respective switch and provide power to the power output header.

5. Onboard 3.3V LDO Regulator

The regulator is used to step down the 5V power supply provided by the USB connection to the 3.3V required by the board components such as the PIC microcontroller and FTDI USB-to-UART IC chip. The input is connected to the USB +5V pin and the output is connected to the board's inner power layer. The part used in this prototype is the MCP1700 which is made by Microchip.

6. USB Input

The mini-USB connector is used to interface the Utopia board to the PC. It provides a 5V source to power the board components and the USB data lines are connected to the FTDI USB-to-UART IC chip.

7. SIO Port

The 8-pin header is used to interface to the SIO port of the hard drive and provide communication between the PC and hard drive.

8. Voltage Input Header

The 4-pin header provides a connection to attach the external power supply. The header can accept the wires directly or a 4-pin Molex power connector to provide easy connectivity. The supplied voltage is sent to each switching regulator for the 5V rail and 12V rail on the board.

9. Voltage Output Header

The 4-pin header provides a connection for attaching a 4-pin Molex power connector to the DUT. The header can accept a 4-pin Molex power connector to provide easy connectivity.

USB-UART IC Component

This component provides independent ports to interface the microcontroller and the hard drive SIO port to the PC. It converts USB data to UART. One benefit is that Utopia commands can be executed simultaneously as data gets transferred between the hard drive and PC.

Requirements and Selection Criteria

The first requirement is that it must provide two interfaces to allow the PC to communicate with the microcontroller and SIO port independently. The goal is to provide a dedicated port to talk to hard drive via SIO and be able to send commands to the microcontroller at the same time. Another requirement is that the USB port must meet USB 2.0 specifications that support data transfer rates up to 480 Mbits/second. The previous implementation in UTOPIA 2.5 only supported USB 1.1 specifications which allows up to 12 Mbits/second. The higher bandwidth will allow more data to be transferred within the same time frame compared to the previous design.

The selection criteria are: support a baud rate of at 115Kbaud or greater, data buffers with at least 1Kbyte of storage, smallest package type, and be compatible with WD drive test script software. A baud rate of 115Kbaud is double that of the previous design and will increase the amount of data transfers. Larger data buffers will allow large amounts of data to be transferred from the hard drive SIO port and can collect large sets of data.

Options Considered

The FTDI FT2232HL is a dual USB-to-UART IC and contains two independent channels that can be configured to support interfacing USB to a wide variety of communication protocols such as UART, FIFO, SPI, I2C, JTAG, RS232, or parallel [2]. The entire USB protocol is handled on chip and each channel is configured as a USB-to-UART interface on default. The chip is compatible with USB 2.0 specifications. Both interfaces feature individual data buffers that can hold up to 4 Kbytes of data. The UART transfer rate is specified to handle up to 12 Mbaud. FTDI offers a royalty-free Virtual Com Port (VCP) and proprietary drivers to eliminate the need for USB driver development. The chip requires a 3.3V power supply to operate under normal conditions. It is rated to handle operating temperatures of -40°C to 80°C.

The only drawback is that overall board size of UTOPIA 3.0 must increase to fit the FT2232HL and its required components. The chip needs an external oscillator and two pairs of LEDs for transmit and receive status indicators.

Option Selected and Justification

The FTDI FT2232HL did not have any other comparable competition. It is the best selection because it meets all of the component requirements. The dual UART interface meets Requirement 5 of Table 1. It provides a separate interface between the PC and two components on UTOPIA, the microcontroller and the SIO port. Also, the UART interface will support WD drive test script software such as TREX, WINDEX, and SASDEX. The 4Kbytes data buffers are four times greater compared to the previous UTOPIA design and exceed Requirement 4 of Table 1. The chip meets Requirement 2 in Table 1 because it is compatible with both USB 2.0 High Speed and Full Speed specifications which allows maximum flexibility for board design. The IC package is a 64-pin LQFP package. Its operating temperature range exceeds Requirement 3 of Table 1.

Microcontroller Component

This purpose of the microcontroller is to take commands from the PC for controlling the voltage regulators and the switch on the power outputs. It is also used to acquire measurements of the voltages and currents.

Requirements and Selection Criteria

The microcontroller component has four main requirements: UART data transfer rates greater than or equal to 57 Kbaud, support for at least 1 UART I/O port and 1 SPI I/O port, at least 8 analog inputs, and at least 2 PWM outputs. Higher data transfer rates improves UTOPIA board response. The I/O ports are needed to maintain the functionality of the previous designs.

The microcontroller will be selected based on the following criteria: include most, if not all, features of PIC 18F4550 excluding USB support, greater than or equal to 10 MIPS, at least 8 MHz internal oscillator, greater than or equal to 1024 bytes of RAM, lowest price, and availability in a 44-pin TQFP package. It should retain the same amount of I/O ports and capabilities of the PIC 18F4550 so that modifications to the previous UTOPIA 2.5 design is kept to a minimum. Any CPU speed greater than 10 MIPS will be an improvement over the previous design. The microcontroller in the previous design is a 44-pin TQFP package and using the same type of package is desirable.

Options Considered

Consideration was taken into account of reusing the PIC 18F4550 microcontroller from the previous design or replacing it with a different microcontroller. The contending microcontroller choices were narrowed down to two families, the 18F K-series and J-series. Table 2 compares the possible microcontroller choices.

Table 2: Comparison of Microcontrollers

	PIC 18F4550	PIC 18F45J11	PIC 18F45K20
CPU Speed	12 MIPS	12 MIPS	16 MIPS
RAM	2048 bytes	3800 bytes	1536 bytes
Flash Memory	32 Kbytes	32 Kbytes	32 Kbytes
Number of USART	1	2	1
Number of SPI	1	2	1
Internal Oscillator	8 MHz	8 MHz	16 MHz
Temperature Range:	-40 to 85 °C	-40 to 85 °C	-40 to 125 °C
USB	Yes	No	No
Package Type	44-pin TQFP	44-pin TQFP	44-pin TQFP
Price	$3.65	$2.09	$1.95

The PIC 18F45J11 and 18F45K20 were considered because of their similarity in capabilities and characteristics with respect to the PIC 18F4550 used in UTOPIA 2.5. Two advantages that the PIC 18F45K20 has over the other choices are the faster CPU and higher internal oscillator speed which will improve performance. The PIC 18F45J11 does not improve on performance with respect to the existing microcontroller, but has more I/O capabilities because it features 2 UART and 2 SPI ports and allow more design possibilities. The advantage of reusing the PIC 18F4550 is that only slight changes to the PCB layout are required. However, the disadvantage is that the overall cost per device will be increased due to the proposed additional components such as the voltage regulator and FTDI USB-UART IC.

Option Selected and Justification

After evaluating all of the choices, It was determined that the best choice will be to replace the PIC 18F4550 with the PIC 18F45K20. All of the microcontroller options meet requirement 6 and 7 listed in Table 1, support of I/O capabilities and UART baud rate of 57 Kbaud, respectively. However, the

selection criterion was used to ultimately decide upon the microcontroller choice. The PIC 18F45K20 exceed many items in the selection criteria including lower unit cost, same 44-pin TQFP package type as in UTOPIA 2.5, higher CPU speed, and higher internal oscillator speeds. These items are highlighted in the shaded boxes of Table 2. The operating voltage range will support 3.3V and allow compatibility with the FTDI chip.

The features included in the PIC 18F4550 did not justify the higher cost since the USB is not a required feature. The dual SPI ports in the PIC 18F45J11 was determined to be unnecessary with clever design. The SPI port specifies three required signals, clock, data, and chip select. Although multiple SPI ports are required to interface the voltage controllers and ramping controllers, it is possible to parallel the clock and data signals with all the controllers and use a single digital output pin for each controller. Thus, each controller only operates when their chip select line is selected.

Voltage Regulator Component

The onboard voltage regulators regulate the amount of power applied to the hard drive. It provides independent control over the 5V and 12V rails. The purpose of adding this component to the UTOPIA board is to provide voltage margining from a single external power supply and allow power control in software.

Requirements and Selection Criteria

The input voltage for the power supply requires approximately 20 VDC with an operating range between 17 VDC and 24 VDC. The output voltages must be capable of supplying 18 VDC. Both rails must be independently controlled to allow maximum flexibility.

The selection criteria are listed as follows: provide the largest range of operating voltage for input and output, highest current output, and smallest PCB package type. The larger range in voltages provides more testing capabilities. The lowest output voltage value is desired.

Options Considered

Below in Table 3, the technical specifications are compared between four DC/DC converters. Three out of the four choices are manufactured by Texas Instruments (TI) while the IR3802AM is made by International Rectifier (IR).

Table 3: Comparison of Switch Regulators

	Current [A]	Vin Max [V]	Vin Min [V]	Vout Max [V]	Vout Min[V]
IR3802AM	6	21	2.5	12	0.6
TI TPS5450	5	36	5.5	31	1.22
TI TPS54550	6	20	4.5	12	0.9
TI TPS54620	6	17	4.5	15	0.8

Legend
* Does not meet requirements   * Meets minimum requirements   * Exceeds requirements

These options were considered because of their ability to handle high currents. In the table, the minimum and maximum voltages for the input and output are specified. Different colors are used to distinguish each part's ability to meet design requirements. The option with the highest maximum voltage for both input and output is the TPS5450 with up to 36 VDC for input and up to 31 VDC for output [3]. The option with the lowest maximum voltages is the TPS54550 with 20 VDC for input and 12 VDC for output [4]. The minimum values for the input voltage are negligible for all of the choices because the operating range is not expected to be below 17 VDC.

Option Selected and Justification

Two of the choices, IR3802AM and TPS54550, were immediately taken out of consideration as a result of not meeting the minimum voltage requirements in requirement 9 of Table 1. Only two options remain and the selection of the TI TPS5450 was due to its much larger operating voltage ranges. The TPS5450 meets requirement 8 and 9 in Table 1. Each rail will include its own individual DC/DC converter to meet requirement 10 of Table 1 which requires that each rail have support independent voltage margining.

Test Plan

Software testing will ensure that user commands perform the intended operation. For example, UTOPIA 2.5 features the power on command that should turn on the power outputs of the UTOPIA board. This function will be tested by using Hyperterminal to send the PON command to turn on the UTOPIA device's power outputs. Then each rail on power output header will be measured by a digital multi-meter to determine whether the outputs are supplying power. Likewise, the POFF command should turn off the power outputs and measuring the power output header should read 0 VDC.

The data acquisition commands for measuring the voltages and currents will be tested to verify that the measurements are consistent with the actual value measured with a digital multi-meter. These tests involve sending the DATA command and a read out of the measurements are displayed in Hyperterminal. Each rail of the power input header and power output header will be measured and errors of 0.1 VDC are acceptable.

The voltage margining feature in UTOPIA 3.0 will be tested by setting the power output voltage with the VSET5 and VSET12 command for the 5V rail and 12V rail, respectively. The command requires a voltage value in milli-volts. Once the voltage has been set, the output voltage is measured by a digital multi-meter and compared with the voltage setting. The measured values should be approximate to the voltage setting and calibration will be needed if the values do not match.

The testing for the OVP feature is very straight forward and only requires that the user not be allowed to set a voltage on the power output that is higher than the OVP value for the specified rail. The OVP value is predetermined and hardcoded into the firmware. The power outputs are measured by a digital multi-meter and it is required that the measured voltage value not exceed the OVP value.

IV. UTOPIA 3.0 IMPLEMENTATION

This chapter discusses the implementation of the four main circuit components: microcontroller, USB-UART IC, 5V and 12V regulator, and 5V and 12V switch. Then, the PCB layout and board fabrication will be covered. Lastly, the embedded firmware design will be described in detail.

Overview of CAD Software

The UTOPIA 3.0 prototype was designed using ExpressPCB CAD software. It includes two applications, one called ExpressSCH for drawing schematics and one called ExpressPCB for circuit board layout. The software creators provide circuit board manufacturing services. This method of prototyping is relatively low cost because the software is free and board manufacturing time takes approximately a week.

Circuit Schematic Design

There are four main circuit components that exist in UTOPIA 3.0 in which two of them are new additions to this design. The microcontroller and switch circuit already exist, while the USB-UART IC and voltage regulator circuit are new. The circuit schematics can be found in Appendix A. Each section will describe its interfaces and hardware functionalities.

Microcontroller

This component consists of the PIC 18F45K20 microcontroller, U1. The schematic can be found in Appendix A-1. It is interfaced to the USB-UART IC via UART using two signals identified as PIC TX and PIC RX. An SPI interface is used to communicate with the voltage ramping controllers, U2 and U3, and voltage regulator controllers, U15 and U16. The SPI interface requires three signals for each interface, SCLK, SDATA, and an individual chip select for each controller: SYNC_12, SYNC_5, PROG_5V, PROG_12V. The 5V rail requires two signals for control of its switch which are "5 PWM" and 5Control. Similarly, the 12V rail requires "12 PWM" and 12Control for control of its switch. The "Drive Vin" signal is used to measure input voltages and two signals, "+5vdr sense" and "+12 vdr sense", measure voltage levels at the output. The signals that measure actual output voltages are 12VOLT and 5VOLT.

The ICSP header is used to flash firmware programs onto the microcontroller. An NTC1 thermistor is used to measure board temperatures. An external oscillator, X1, is used to provide it with a 20 MHz clock source. Future testing will determine whether the microcontroller's internal oscillator is accurate enough so that this part can be removed. Two LEDs, D4 and D5, are connected for debugging purposes. Power is supplied to the chip via the 3.3V low-dropout (LDO) regulator.

USB-UART IC

The data sheet for the FTDI FT2232HL provides a recommended circuit example for the UART configuration. The circuit schematic can be found in Appendix A-6 in which the IC is labeled as U10. It has two UART interfaces in which port A is interfaced to the hard drive SIO port and port B is interfaced to the PIC microcontroller. Each interface only requires two signals, TX and RX. The USB interface requires two signals for data and is connected to the USB port.

A pairs of LEDs are connected to each port as status indicators for the TX and RX lines. The LEDs will blink when data is being transmitted. Additional components are required such as a 3.3V LDO regulator , U11, to provide a 3.3V operating supply, external EEPROM, U12, to store configuration settings, and external 12 MHz oscillator identified as X2.

5V/12V Voltage Regulator

Each voltage regulator circuit is essentially a TI TPS5450 DC/DC converter attached to a resistor divider network. The circuit schematic can be found in Appendix A-5 for the 5V regulator and Appendix A-4 for the 12V regulator. Note that each regulator is essential the same circuit, which is slightly modified from the recommended configuration provided in the TPS5450 data sheet [3]. Each rail is regulated by a single converter; the 5V rail is regulated by U14 and the 12V rail is regulated by U13.

The 12V rail is powered directly by the external power supply through the power input header. The voltage output is located on the output of the 15 µH power inductor, L3. The output voltages of the DC/DC converter can be adjusted by increments of 1/16 VDC depending on the equivalent resistance at the VSENSE port on pin 4 of U13. The ADG738, identified as U15, is a serially controlled 8-input MUX switch [5] and is responsible for generating the equivalent resistances. The MUX switch accepts an 8-bit binary value and turns on the inputs corresponding to the bit positions that are set high. The inputs, ranging between S1 to S8, conduct to ground and are then paralleled together to produce a parallel resistance value. With all switches in the “off” state or open, the TPS5450 defaults to output 2 VDC. Any combination of closed switches will add to the 2 VDC and result in an increase in output voltage. The total voltage output can reach a maximum of approximately 18 VDC. Each MUX switch is connected to a specific resistance value that is designed to provide the output voltages shown in Table 4. For example, if the MUX switch is set with the binary value “1000000” or 127 in decimal, the input at S8 will be switched on and the rest will be switched off. Thus, the total output voltage on the rail will be 10 VDC because of the default 2 VDC added to the 8 VDC supplied by the MUX switch.

Table 4: MUX Switch Settings

Switch Input	S8	S7	S6	S5	S4	S3	S2	S1
Binary Bit Position	7	6	5	4	3	2	1	0
Output Voltage [V]	8	4	2	1	1/2	1/4	1/8	1/16

The 5V rail works in a similar fashion to the 12V rail. The only difference is that the resistors at the S7 and S8 inputs are not populated in the board fabrication. This limits the maximum total output voltage to approximately 6 VDC because the 8 VDC and 4 VDC outputs are not selectable.

5V/12V Switch

The switch circuit is referenced in Appendix A-2 and A-3. The 5V and 12V rails have nearly identical circuit configurations for the switch. The 12V switch will be described in detail and the characteristics can be applied to the 5V switch.

On the 12V switch, the 12Control signal turns on or off the power output in the 12V switch. The input voltages are passed into the source of a FET, U7, and forced through a feedback loop that is connected to the S1 input of the ADG738 MUX switch, U2. The output of the 12V switch is connected to pin 1 of the power output header, J5. Voltage ramping is controlled by a combination of PWM signals and a decade resistor network attached to the inputs of U2. The MUX switch provides a course method

of adjusting the ramping rate, while the "12 PWM" signal provides a fine tuning method. Depending on the MUX switch setting, the voltage ramps can be adjusted between 1 µs to 1s. The output of the MUX switch is connected to the positive input of a TL072 operation amplifier [6] identified as U6. The ADM4073F [7], identified as U4, is a voltage output, current-sense amplifier. It converts the current reading at the power output to an analog voltage value that can be used by the PIC microcontroller to measure the current of the 12V output in Amps. A LED, D2, is used to indicate the status of the switch output.

Similarly, the 5Control signal controls the output of the 5V switch. The input voltages are passed into the source of a FET, U9, and forced through a feedback loop that is connected to the S1 input of ADG738 MUX switch, U3. The output of the 5V switch is connected to pin 4 of the power output header, J5. Voltage ramping is controlled by a combination of PWM signals and a decade resistor network attached to the inputs of U3. The "12 PWM" is replaced with "5 PWM". The output of the MUX switch is connected to the positive input of a TL072 operation amplifier identified as U8. The ADM4073F, identified as U5, converts the current reading at the power output to an analog voltage value that can be used by the PIC microcontroller to measure the current of the 5V output in Amps. A LED, D3, is also used to indicate the status of the switch output.

PCB Layout Design

The board requires a relatively small footprint. The design utilizes a 4-layer PCB board which consists of a top, bottom, ground, and power layer. Components can be mounted on the top or bottom layer, while the power and ground layers are separately sandwiched in between. The board dimensions are approximately 2.5" x 2.5". It is almost double the size of UTOPIA 2.5 which was 2" x 1.5".

Overview of Design Procedures

The size of shape of each part is determined by its available package type in which some components only feature one specific one, while others have multiple types. Each part's data sheet specifies this information along with recommended PCB patterns. Most of the parts such as resistors, capacitors, and ICs were standard and there were no issues finding it in the parts library of ExpressPCB. However, certain components such as the mini-USB plug and power inductors, L3 and L4, had PCB patterns that the parts library did not support. As a result, these parts had to be custom made by grouping together pads according to the measurements provided by their data sheets. For example, the power inductor specified a 12.8 mm x 12.8 mm square in which two 2.9 mm x 5.4 mm pads were separated by 7 mm. ExpressPCB provides a method of making the pads and using the built-in ruler to design a PCB pattern that met those specifications. Similarly, the mini-USB port was created by grouping traces and pads according to the specifications provided by the manufacturer's data sheet.

Component placement was determined by the pin locations of each part. The strategy was to arrange each part so that traces lengths connecting two pins are minimized. All of the parts were first placed on the layout to determine the approximate board space required.

General PCB design rules were followed which includes running traces only horizontally, vertically, or at 45 degree angles [8]. The connections are routed with 7 mil traces that have a width of 0.007". The power signals are routed with thicker 10 mil traces that have a width of 0.010". A good

design practice is leaving at least a minimum gap of 7 mils surrounding traces. Filled planes are used to minimize current loss.

Top Layer

The PCB layout for the top layer is shown in Appendix B-2. It is subdivided into four parts in which the microcontroller is laid out on the top left section, the power output switches are laid out on the top right section, the voltage regulators are laid out on the lower right section, and the USB-UART IC is laid out on the lower left section.

Certain components had priority on the top layer. The large bulky components were placed on the top layer so that the board could lie flat on a lab bench surface. This included the aluminum capacitor, C16, which can be found in Appendix A-2 and C20 which can be found in Appendix A-3. The power inductors, L3 and L4, were also large components which are shown in Appendix A-4 and A-5, respectively. Also, the LEDs identified as D1, D2, D3, D4, D5, D8, D9, and D10 were placed on top so that they would be always visible.

Once all of the parts were laid out, it was a matter of orienting them such that placing the traces can be as short and direct as much as possible. For example, the FTDI USB-UART IC, U11, is oriented such that the traces for the TX and RX signals can be routed as short as possible to the PIC 18F45K20, U1. The output pins of the voltage regulators which are located on pin 2 of L3 and L4 are oriented to face the input of the power output switches. For L4, it is adjacent to C20 and is connected by a filled plane to minimize current loss. Likewise for L3, it is adjacent to C16 and provides the shortest path possible. Since there was no way to shorten the ground trace between the power input header and the power output header, a filled plane runs directly across the UTOPIA board to prevent current loss.

Bottom Layer

The bottom layer consists of the smaller or flat components such as resistors and the ADG738 MUX switches. Like the top layer, the board was subdivided into the sections mentioned in the previous top layer description above. The bottom layer is characterized by the long traces that run across the board. These are the clock, data, and chip select signals that connect between the microcontroller and the ADG738 MUX switches. In the lower right corner of the board, there is a filled plane that connects to the power input header and is spread out to reach the input terminals of the TI TPS5450 DC/DC converters on top. Due to the lack of board space, it was difficult to connect the inputs. The solution was to use vias that connect between the top and bottom layers to make the connections. Again, the filled plane is used to minimize the loss of currents.

Board Fabrication

The manufacturing of the boards were made by ExpressPCB. Once the circuit schematic and PCB layout were complete, a final verification was done to verify that all of the connections were implemented. The order is made online through the ExpressPCB software. The turnaround time for the boards took about a week from the date of the order. In order to reduce board costs, three copies of the layout were made onto a single PCB.

Upon visual inspection of the manufactured boards, two noteworthy errors were identified. A trace error caused by an unintentional short across capacitor C14 was found. This capacitor is part of the FTDI EEPROM circuit. It is an easy fix that only requires cutting the trace and should be corrected in the next board revision. The other error involved the PIC microcontroller and was caused by the routing an analog signal trace to a pin that was not assigned as an analog pin. Thus, it is not capable of reading the analog input. This will also be fixed in the next revision of the board, but for now, the incorrect pin and the correct pin will be shorted together to achieve correct functionality to develop the firmware.

While the boards were being manufactured, a parts order was placed at Digi-Key to obtain the remaining parts that were unavailable at Western Digital. In two weeks' time, two complete Utopia boards were produced and ready for testing and development. It is important to note that there are some differences between the prototype and the circuit schematic due to unavailability of parts. Some of the resistor values on the voltage regulator components are not the exact values specified. The external oscillator on the prototype is rated at a frequency of 25 MHz and not the 20 MHz as specified in the schematic.

The two figures below highlights the location of the main board components on UTOPIA 3.0. Figure 4 shows the top of the board and each main component is identified by a dotted line. The I/O interfaces such as the mini-USB plug, power headers, and SIO port are also labeled. Figure 5 shows the bottom of the board and identifies the components placed below.

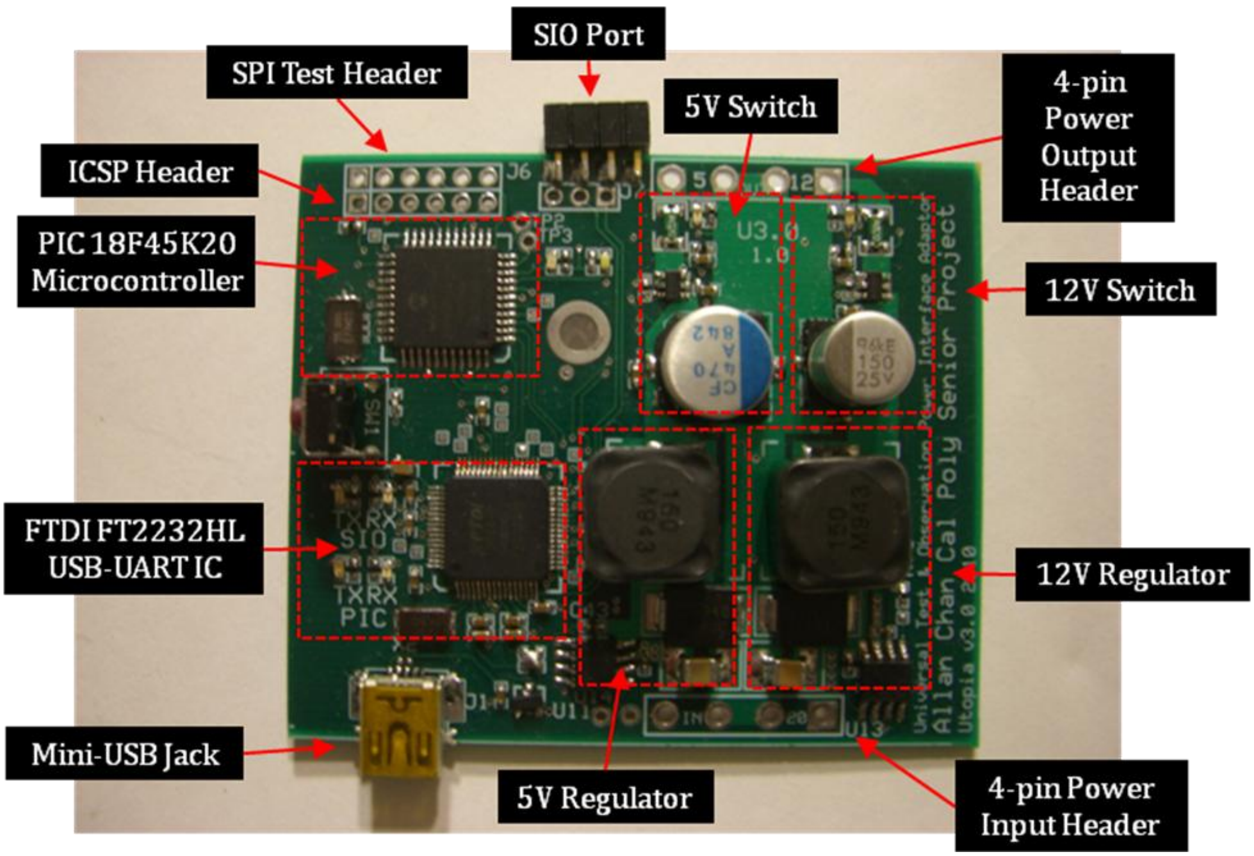


Figure 4: Photo of UTOPIA 3.0 (Top)

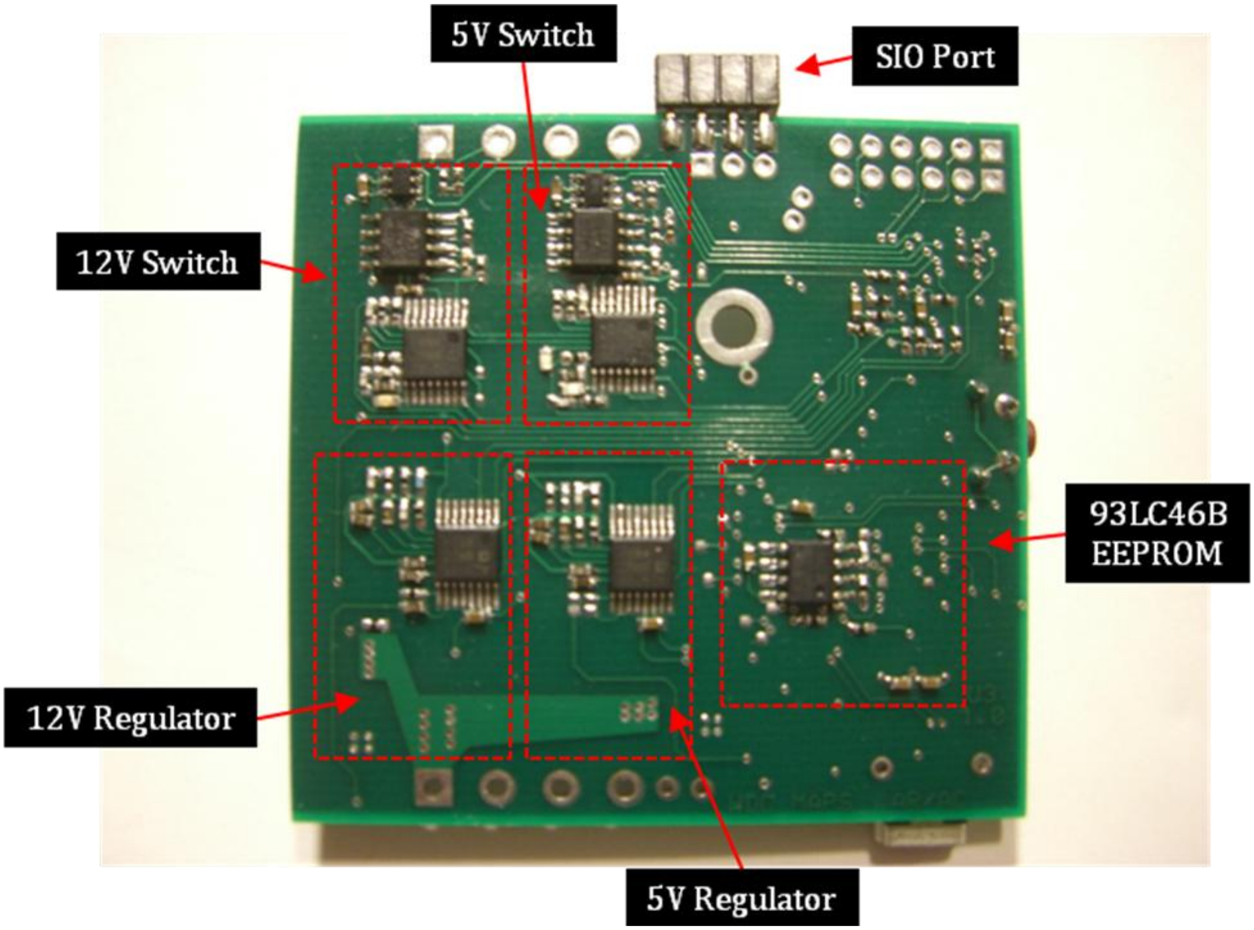


Figure 5: Photo of UTOPIA 3.0 (Bottom)

Software Architecture

This section describes the algorithm of the embedded system and the new commands that were implemented. The firmware for UTOPIA 3.0 is designed for the PIC microcontroller. The programming language is Basic. Code development was done using Proton Basic software [9].

Software Algorithm

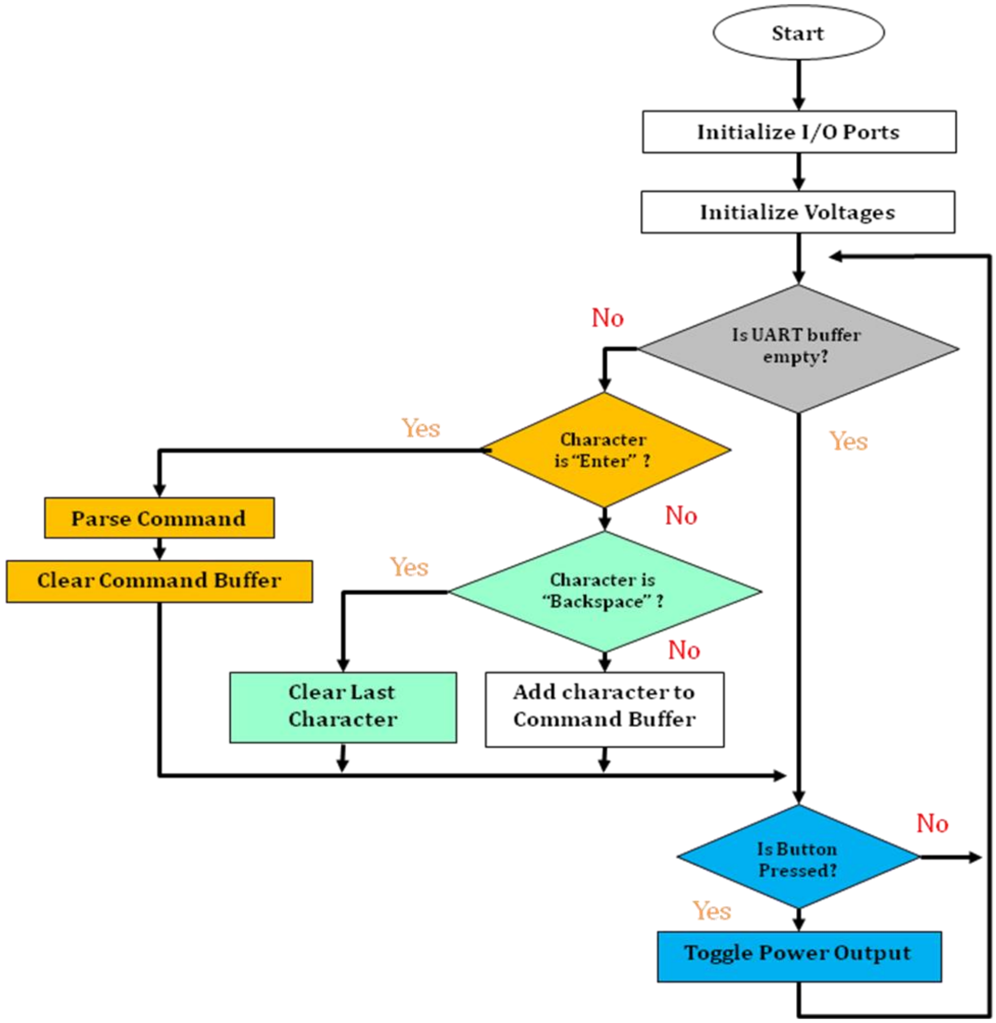


Figure 6: Software Flow Chart

The main loop algorithm is shown above in Figure 6. It is mainly used to parse commands from user input. It can be found in Appendix D on pages 49-50. Once the device has initialized its I/O ports and voltages, an infinite loop continuously checks the UART receive buffer for input from the user. If there is no data and the UART buffer is empty, then it checks whether the onboard button switch has been pressed. Pressing the button will toggle the power outputs and turn it on or off depending on the current state. If there is data in the UART buffer, each character is processed one at a time. Three conditions are checked: whether the input is an "Enter" key, a "backspace" key, or an alphanumerical key. The "enter" key indicates that the user is finished typing the command and the contents of the command buffer will be parsed to determine the action to perform. Once the command has been performed, the command buffer is cleared and ready to accept a new command. If the key press was a "backspace" key, then the previous character is removed from the command buffer. If the key is an alphanumerical character, then it is added to the command buffer.

Software Features

The software in UTOPIA 3.0 is responsible for three main tasks. It has to provide independent control of the power output switch, independent control of voltage margining, and perform data acquisition of voltages and currents. The power output switch functionality and data acquisition features have already been implemented in UTOPIA 2.5. New software for voltage margining needs to be developed to adjust the voltage settings and provide OVP on the power outputs.

The code for voltage margining is found in Appendix D between page 58-59. When the user calls the VSET5 or VSET12 command with a voltage value in millivolts, an if-statement checks whether the value is valid, either above the minimum voltage or below the OVP value. If it is valid, then it is converted to an 8-bit binary number for setting the MUX switch using the following formula:

$$SDATA = 256/16000 *(Vout - 2000)$$

Vout represents the value passed in from the VSET command. The value SDATA is a binary number ranging from 0-255 which is set on the MUX switch controlling the voltage rails.

The OVP feature was suggested during firmware development. The user can change the OVP for each rail independently or query for the OVP value. Its implementation is simple and only requires adding two registers to store the value in memory. The code can be found in Appendix D on page 53. On page 48, the registers are initialized with the values `MAXOVP5` and `MAXOVP12` which are the default OVP values on each rail. The user calls the command OVP5 or OVP12 with a voltage value in millivolts. An if-statement determines whether the value is valid and sets the new OVP value if it is. The valid voltage range for the 5V rail is 3 VDC to 6 VDC and for the 12V rail is 3 VDC to 18 VDC.

Existing Commands

The commands listed below in Table 5 are already implemented in UTOPIA 2.5. The PON and POFF commands are used to power on or power off the power outputs, respectively. Independent switching is also available for each power output using the PON5, POFF5, PON12, and POFF12 commands. The remaining commands in the list are used to take measurements of input voltages, output voltages, output currents, and board temperature.

Table 5: Existing UTOPIA 2.5 Commands

Output Switch

PON	Turns on 5V and 12V outputs
POFF	Turns off 5V and 12V outputs
PON5	Turns on 5V output
POFF5	Turns off 5V output
PON12	Turns on 12V output
POFF12	Turns off 12V output

Status

TEMP	Displays board temperature
DATA	Displays voltage and current measurements
CUR	Displays current measurements
VOLTS	Displays voltage measurements
U3ID	Displays the firmware version

New Commands

In addition to the existing commands, there are seven new commands for UTOPIA 3.0 listed in Table 6 that has been developed to control the voltage regulators. One pair of commands, VSET5 and VSET12, adjusts the voltage settings on the power outputs. Four commands are related to the over-voltage-protection (OVP) feature. The commands, OVP5 and OVP12, set the maximum voltage possible on the power output. The commands, GETOVP5 and GETOVP12, read back the OVP value on the 5V rail and 12V rail, respectively.

Table 6: New UTOPIA 3.0 Commands

Power Settings

VSET5 *volts*	Sets the voltage value on the 5V rail in mV
VSET12 *volts*	Sets the voltage value on the 12V rail in mV

OVP Settings

OVP5 *volts*	Sets the over-voltage-protection value on the 5V rail in mV
OVP12 *volts*	Sets the over-voltage-protection value on the 12V rail in mV

OVP Status

GETOVP5	Reads back the over-voltage-protection value on the 5V rail
GETOVP12	Reads back the over-voltage-protection value on the 12V rail

Miscellaneous

HELP	Provides descriptions of all user commands

V. INTEGRATION AND TEST

Chapter 5 describes the testing that was involved for the UTOPIA 3.0 device. Testing verified that the design is sound and meets requirements. The results are presented and recommendations for the next board design conclude the chapter.

Acceptance Testing

The final prototype is expected to meet the requirements listed in Table 1. Each component of the board must be tested to ensure successful operation upon integration. Specifically, the UTOPIA device was tested on accepting commands, supplying power to hard drive, and communicating with SIO port.

The UTOPIA 3.0 device's ability to accept commands was tested first. Only the commands related to setting the voltage, controlling the power outputs, and data acquisition were tested. For each tested voltage setting, the power outputs were measured and recorded. It was deemed a success as long as the voltage measurement was within 1 VDC difference because the hardware has not been calibrated. The power outputs were tested by issuing the commands and making sure that the power on command applied power to the output terminals and the power off command removes power from the output terminal. The voltage measurements in software were compared to measurements by electronic measuring tools. The current measurements and board temperature were not included in the final testing. It will be performed in the next stage of development where precision tools will be available.

The ability to supply power to the hard drive was tested next. The main task was to make sure voltage margining was functional and the power outputs could be switched on and off. Initially, the voltages were set by sending a value between 0 and 255 via SPI protocols to the ADG738 MUX switches which it uses to adjust the voltage setting. Different values switch on different combinations of the 8 input switches and, in turn, provide different voltage outputs.

Communication with the SIO port was tested at Western Digital and involves running SASDEX, test script software, to run an automated test script that repeatedly powers up and down the hard drive at two different pairs of voltage settings. The test results have been shown that it meets Western Digital specifications.

Test Report

The tests included in this report demonstrate basic operation of UTOPIA 3.0. The voltage ramping features was not tested at this time because it was beyond the scope of the project. There are five tests and they aim to verify basic functionality of UTOPIA 3.0.

Test:	Program PIC microntroller
Description:	This test flashes a blinking LED program onto the PIC microntroller. It demonstrates that the microcontroller can be flashed with a program. The two debugging LEDs are set to alternately blink continuously at a rate of 100 ms.
Equipment:	UTOPIA 3.0 device, PICkit 2 Programmer, mini-USB cable, PICkit 2 v2.61 programming software
Procedure:	1. Connect PICkit 2 to PC using mini-USB cable 2. Connect PICkit 2 to ICSP header 3. Import HEX file into PICkit 2 software and click "Write" button 4. PIC microcontroller will perform a power-on reset and begin running the program
Results:	The blinking LED program loaded successfully. After the reflash, the PIC microcontroller performs a reset and the LEDs start blinking in the middle of the board.

Test:	SPI Test Header
Description:	The SPI test header provides an external connection for communication via SPI 3-wire protocols. It supports input and output of data. This test demonstrates that the PIC microcontroller sends valid data to the ADG739 MUX switches for voltage margining and ramping control
Equipment:	UTOPIA 3.0 device, Oscilloscope, 3 scope probes, Hyperterminal, mini-USB cable
Procedure:	1. Connect UTOPIA 3.0 device to mini-USB cable 2. Attach 1 scope probe to pin 2 of SPI test header to measure SCLK 3. Attach 1 scope probe to pin 4 of SPI test header to measure SDATA 4.Attach 1 scope probe to pin 5 of SPI test header to measure SPI head CS (chip select) 5. Set Oscilloscope to trigger on SPI head CS on falling edge 6. Open Hyperterminal and type in "SPI5 10" 7. Read output from oscilloscope
Results:	The oscilloscope measures all three signals, SCLK, SDATA, SPI head CS. The port is activated by the SPI head CS signal when it selects low and data transmission stops when the chip select turns off and idles high. Data transmission occurs in 8-bit words and is transferred bit by bit on each clock cycle. Output is expected to show the chip select toggled low and a binary sequence of the value 10 that is timed to the clock signal. The binary equivalent of '10' (0b00001010) is shown on the scope in addition to a toggling clock source. The results meet expectation.

Test:	Software OVP
Description:	Provides a over-voltage-protection limit on the UTOPIA 3.0 power outputs. This test makes sure that voltages cannot be set higher than the default OVP settings, 5.3 VDC on the 5V rail and 13.3 VDC on the 12V rail.
Equipment:	UTOPIA 3.0 device, mini-USB cable, External 20 VDC power supply, digital multi-meter
Procedure:	1. Connect UTOPIA 3.0 device to PC using mini-USB cable 2. Connect digital multi-meter to 5V power output 3. Connect power supply to UTOPIA 3.0 power input header 4. Open Hyperterminal and connect to UTOPIA 3.0 device 5. Call "VSET5 6000" command to set 6 VDC on 5V rail 6. Call "VSET12 15000" to set 15 VDC on 12V rail 7. Measure voltages at power output header
Results:	The test sets the power outputs to settings that exceed the OVP value. The results meet expectations. In both situation, the power outputs are unchanged and an error message notifies the user that the voltage setting was too high.

Test:	Power Output Switch
Description:	This test is performed to ensure that the power on command switches on the power output and the power off command switches off the power output.
Equipment:	UTOPIA 3.0 device, mini-USB cable, External 20 VDC power supply, digital multi-meter
Procedure:	1. Connect UTOPIA 3.0 device to PC using mini-USB cable 2. Connect power supply to UTOPIA 3.0 power input header 3. Open Hyperterminal and connect to UTOPIA 3.0 device 4. Call "VSET5 6000" command to set 6 VDC on 5V rail 5. Connect digital multi-meter to 5V power output and measure voltages at power output header 6. Call "VSET12 15000" to set 15 VDC on 12V rail 7. Connect digital multi-meter to 12V power output and measure voltages at power output header
Results:	The test sets the power outputs to settings that exceed the OVP value. The results meet expectations. In both situation, the power outputs are unchanged and an error message notifies the user that the voltage setting was too high

Test: Voltage Margining

Description: Ensure that voltage margining is functional on the 5V rail and 12V rail. A range of voltage settings are selected and set on the UTOPIA 3.0 device. The power outputs are measured and compared to the voltage settings

Equipment: UTOPIA 3.0 device, mini-USB cable, External 20 VDC power supply, digital multi-meter

Procedure:
1. Connect UTOPIA 3.0 device to PC using mini-USB cable
2. Connect power supply to UTOPIA 3.0 power input header
3. Open Hyperterminal and connect to UTOPIA 3.0 device
4. Call "VSET5 5000" command to set 5 VDC on 5V rail
5. Connect digital multi-meter to 5V power output and measure voltages
6. Call "VSET12 12000" to set 12 VDC on 12V rail
7. Connect digital multi-meter to 12V power output and measure voltages
8. Repeat steps 4 through 7 for each voltage setting desired for testing

Results: The voltage settings and measured voltages are compared in the tables below. Table 7 lists the measurements for the 12V rail. The values are very closely matched to the voltage setting and have less than 1% error approximately. In Table 8, the 5V rail has slightly more percent error with up to almost 2% error. The amount of error for both outputs is acceptable because the certain resistance values for the decade resistor network in each voltage regulator don't all match exactly to the circuit schematic.

Table 7: 12V Rail Power Output Measurements

12V Rail Measurements		
Setting [V]	Measured Voltage [V]	Percent Error [%]
3.00	2.99	0.3
5.00	4.97	0.6
11.50	11.42	0.7
12.00	11.93	0.6
12.50	12.42	0.7
13.30	13.16	1.1

Table 8: 5V Rail Power Output Measurements

5V Rail Measurements		
Setting [V]	Measured Voltage [V]	Percent Error [%]
3.00	2.98	0.7
4.00	3.96	0.9
4.50	4.44	1.3
5.00	4.95	1.1
5.10	5.01	1.8

Results

Successes

The PCB board has a relatively low number of defects due to design. The problems were not severe and fixed with temporary solutions that did not hinder our ability to make progress on board development. There have been no known defects due to manufacturing, so it appears that all of the spacing and trace widths are acceptable. All of the parts have been tested for correct functionality and compatibility. The goal of the project has been reached with the PC able to communicate with the Utopia board and control the voltages. This board is now able to be used as a base for further development such as calibrating the voltage outputs to be more accurate, testing higher data transfer rates, and determining the potential of the powerful microcontroller.

Shortcomings

There have been no major failures so far that prevent the UTOPIA device from functioning normally. However, an issue with the Hyperterminal interface has not been resolved. The initial character of the very first typed command is always corrupted due to an apparent hardware issue. That is, the first character does not match the character typed on the keyboard on startup. The issue is resolved by deleting this character and it never occurs again until the next restart. This issue will require more debugging and testing to determine a solution.

Design Recommendations

1. One of the capacitors (C14) connected to the FTDI IC chip contain an electrical short across its pins. This can be fixed by removing the electrical short.
2. Resistors R3, R4, R13, and R14 should have resistance values of 1 Kohm rather than 249 ohm. The resistors placed on the board are actually 1 Kohm and the change only needs to be made to the circuit schematic.
3. There is an analog signal trace that is routed to a wrong pin on the PIC microcontroller. The analog pin for the 5CUR signal should be connected to pin 22 and not pin 23. This was due to an error in the circuit schematic that connected the analog signal to a pin that does not support analog measurements.

VI. FURTHER DEVELEPMENT/TESTING

There are three items that need to be addressed in future development before UTOPIA 3.0 is ready for mass production.

One of the items is to determine whether the EEPROM for the FTDI USB-UART IC should be included for all boards. The EEPROM is not required for proper device operation. However, it is not possible to adjust the brightness of the TX and RX status LEDs without it. The advantages and disadvantages will need to be taken into consideration before a decision can be made.

Secondly, the PIC 18F45K20's built-in internal oscillator which is capable of clock speeds up to 64 MHz will need to be thoroughly tested for accuracy at clock speeds higher than 20 MHz. It is unknown whether it is advantageous to run at a higher speed, so the UTOPIA 3.0 prototype will be evaluated at different speeds.

It is also important to calibrate the analog readings of the voltages and currents from the power input and output in order to provide accurate measurements in software. Precise testing equipment will be used to measure the actual values and the software will be tuned to match the analog readings to the actual measurements.

VII. BIBLIOGRAPHY

[1] Microchip. "PIC 18F2455/2550/4455/4550 Data Sheet." 27 October 2009. Microchip.com. 14 November 2009 <http://ww1.microchip.com/downloads/en/DeviceDoc/39632e.pdf>.

[2] FTDI Ltd. "FT2232H Datasheet v2.06." 21 September 2009. ftdichip.com. 21 January 2010 <http://www.ftdichip.com/Documents/DataSheets/DS_FT2232H_V206.pdf>.

[3] Texas Instruments. "TPS5450EVM-254 5-A, Swift Regulator Evaluation Module." 9 March 2007. focus.ti.com. 10 December 2009 <http://www.ti.com/litv/pdf/slvu211>.

[4] Texas Instruments. "TPS54550EVM-158 Swift Regulator EVM." 29 March 2006. focus.ti.com. 13 November 2009 <http://www.ti.com/litv/pdf/slvu151>.

[5] Analog Devices. "ADG738: CMOS, Low Voltage, Single 8 to 1 Multiplexer, Serially Controlled Matrix Switch." May 2000. analog.com. 8 March 2010 <http://www.analog.com/static/imported-files/data_sheets/ADG738_739.pdf>.

[6] Texas Instruments. "TL072 Dual Low-Noise JFET-Input Gen. Purpose Operational Amplifier." 23 March 2005. focus.ti.com. 10 March 2010 <http://www.ti.com/lit/gpn/tl072>.

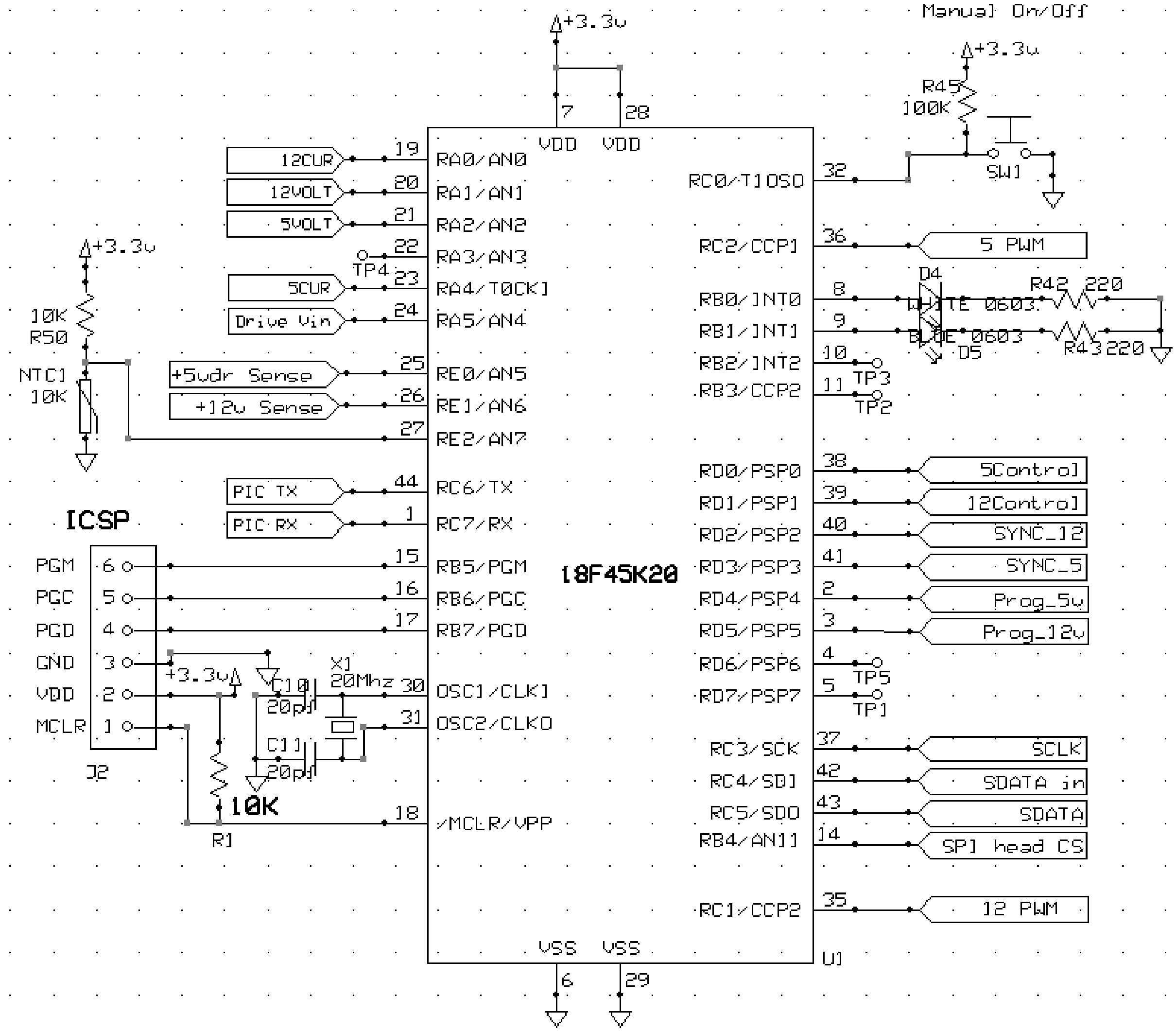
[7] Analog Devices. "ADM4073: LOW COST, VOLTAGE OUTPUT, HIGH-SIDE CURRENT SENSE AMPLIFIER." October 2006. analog.com. 16 March 2010 <http://www.analog.com/static/imported-files/data_sheets/ADM4073.pdf>.

[8] ExpressPCB. "Tips for Designing PCBs." expresspcb.com. 10 December 2009 <http://www.expresspcb.com/ExpressPCBHtm/Tips.htm>.

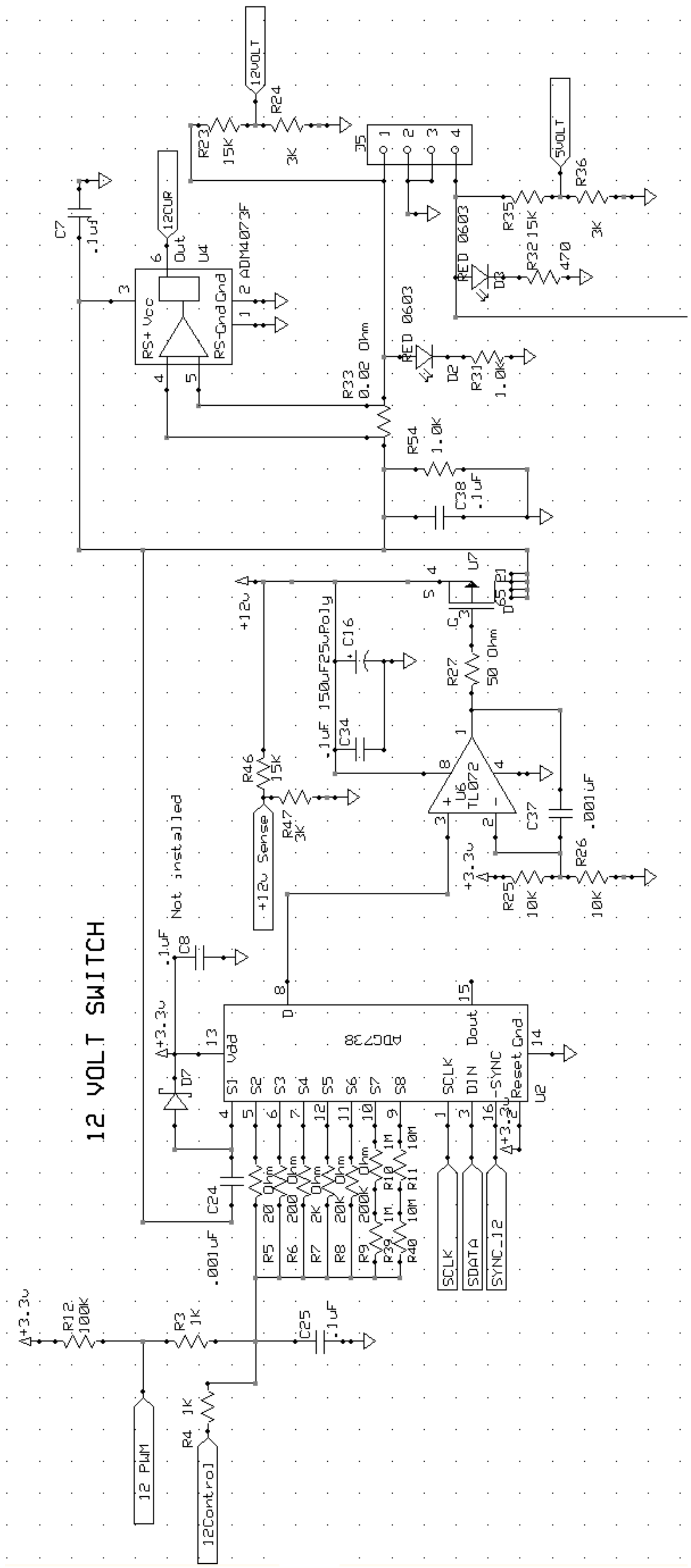
[9] Crownhill Associates. "Proton BASIC." picbasic.org. 26 January 2010 <http://www.picbasic.org/>.

APPENDIX A: Circuit Schematics

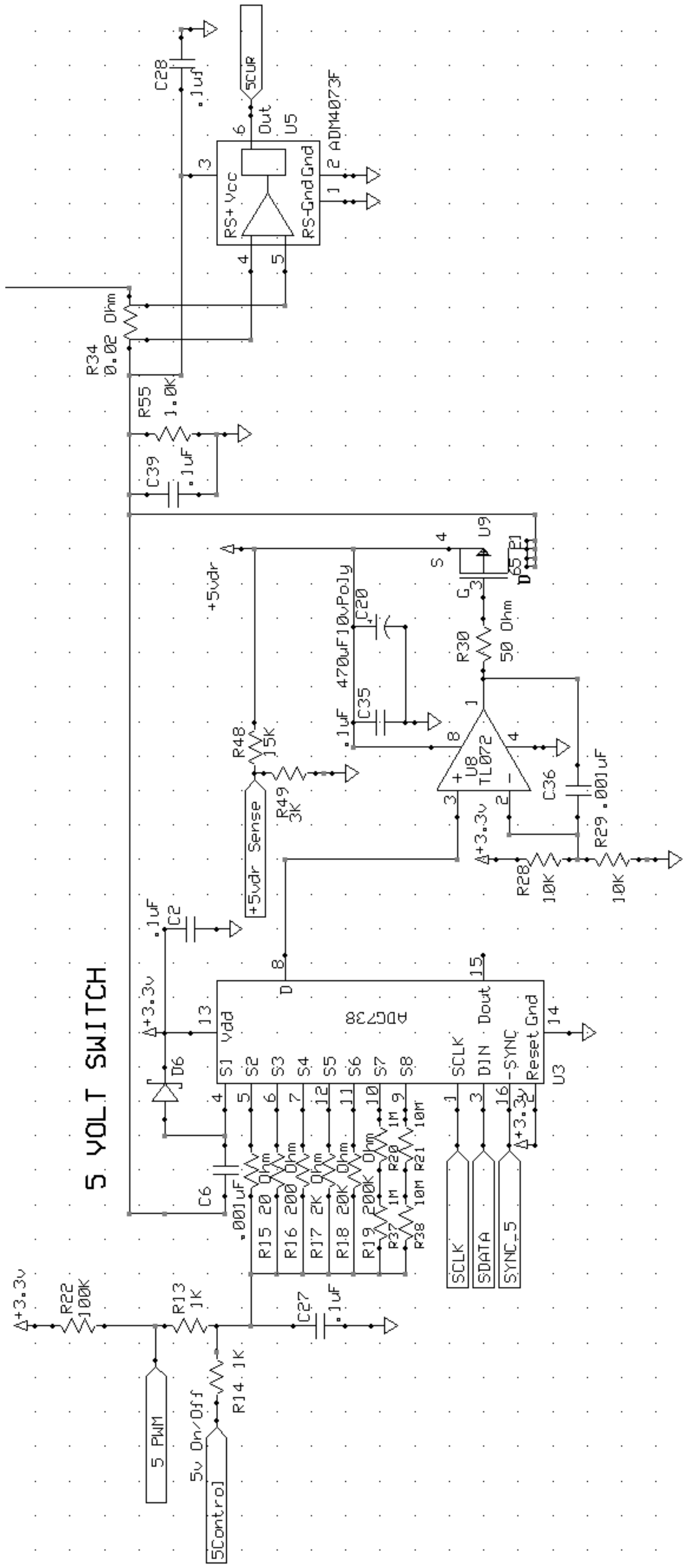
Appendix A-1: Microcontroller



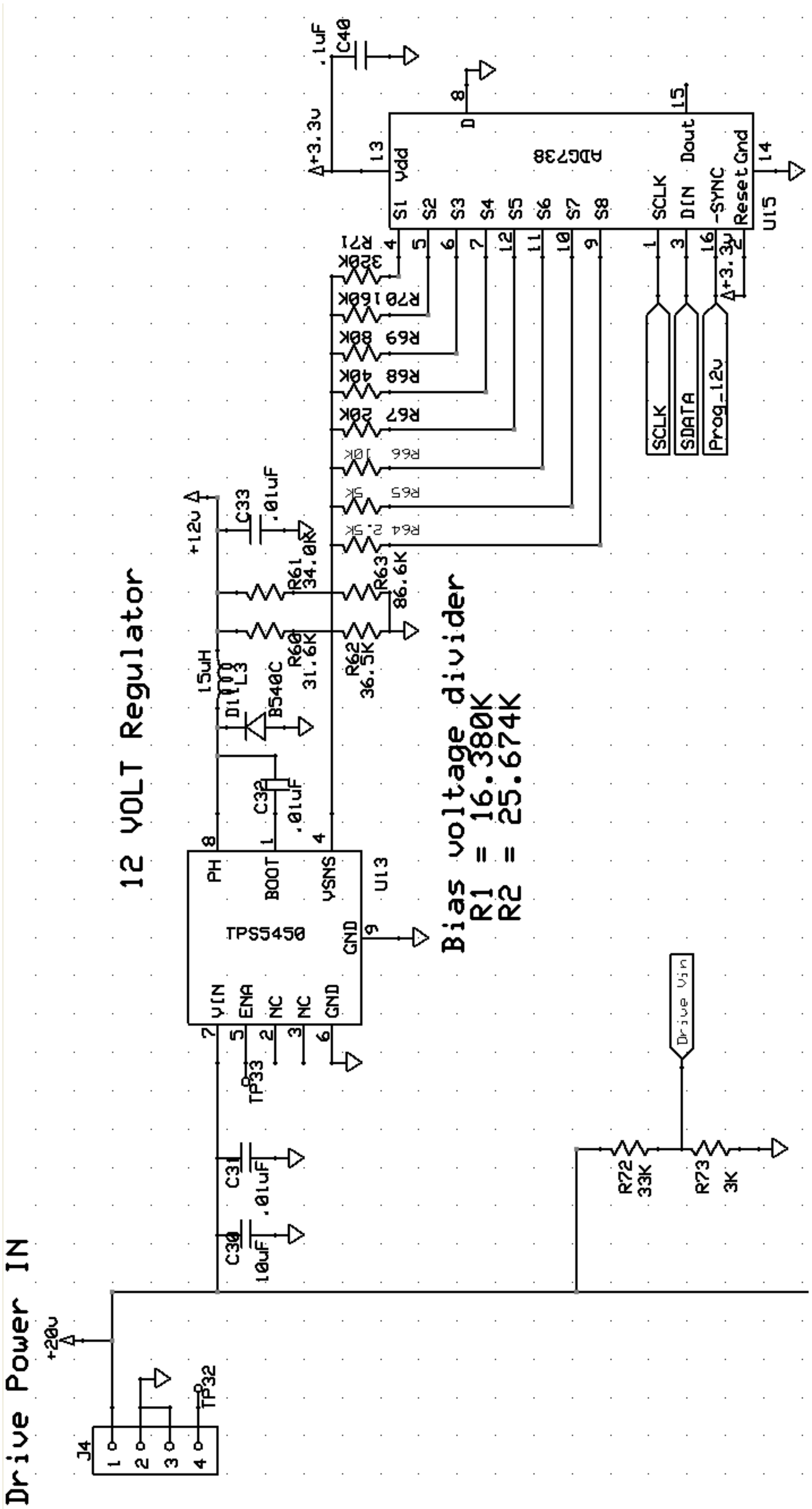
Appendix A-2: 12V Switch



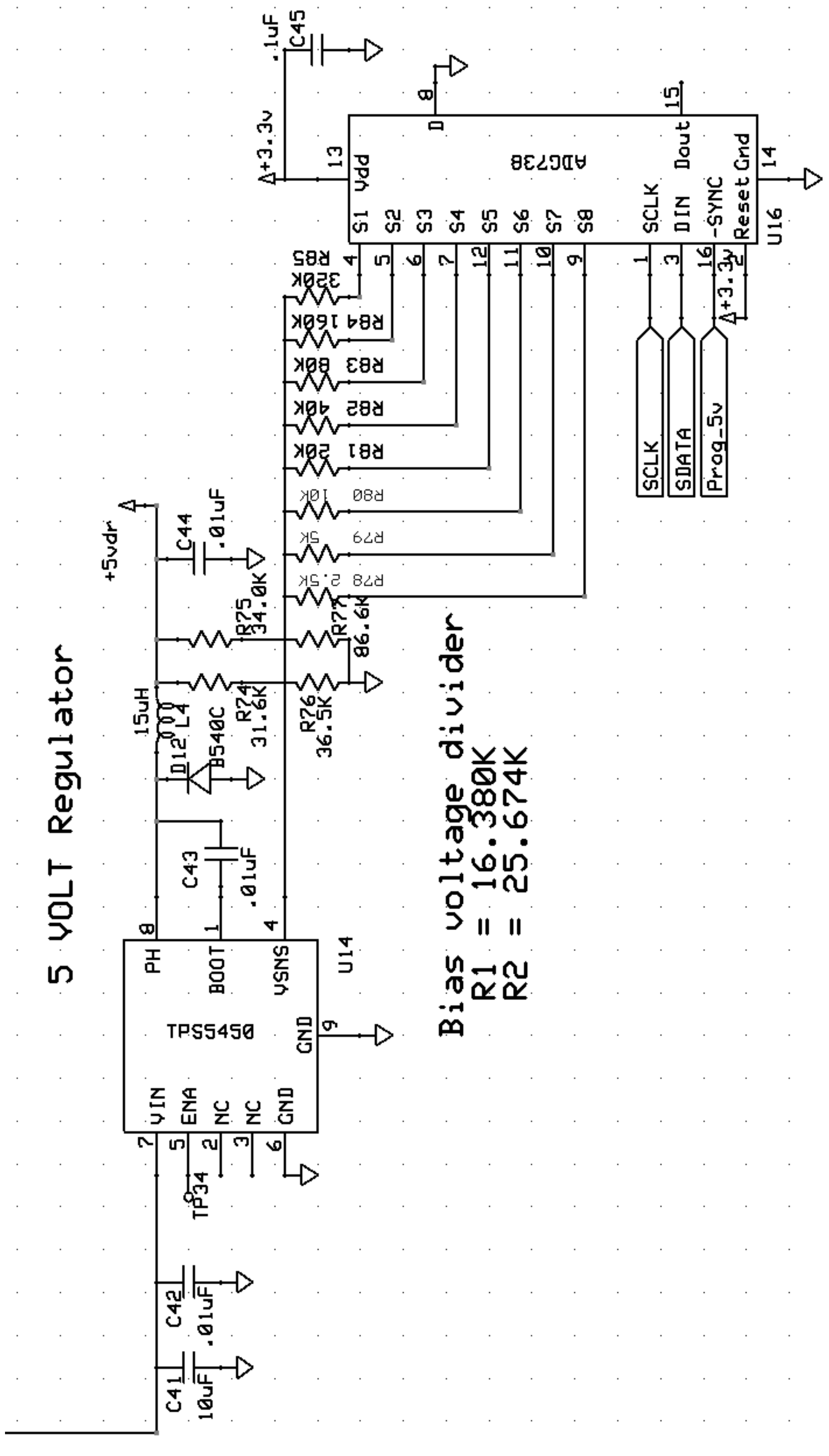
Appendix A-3: 5V Switch



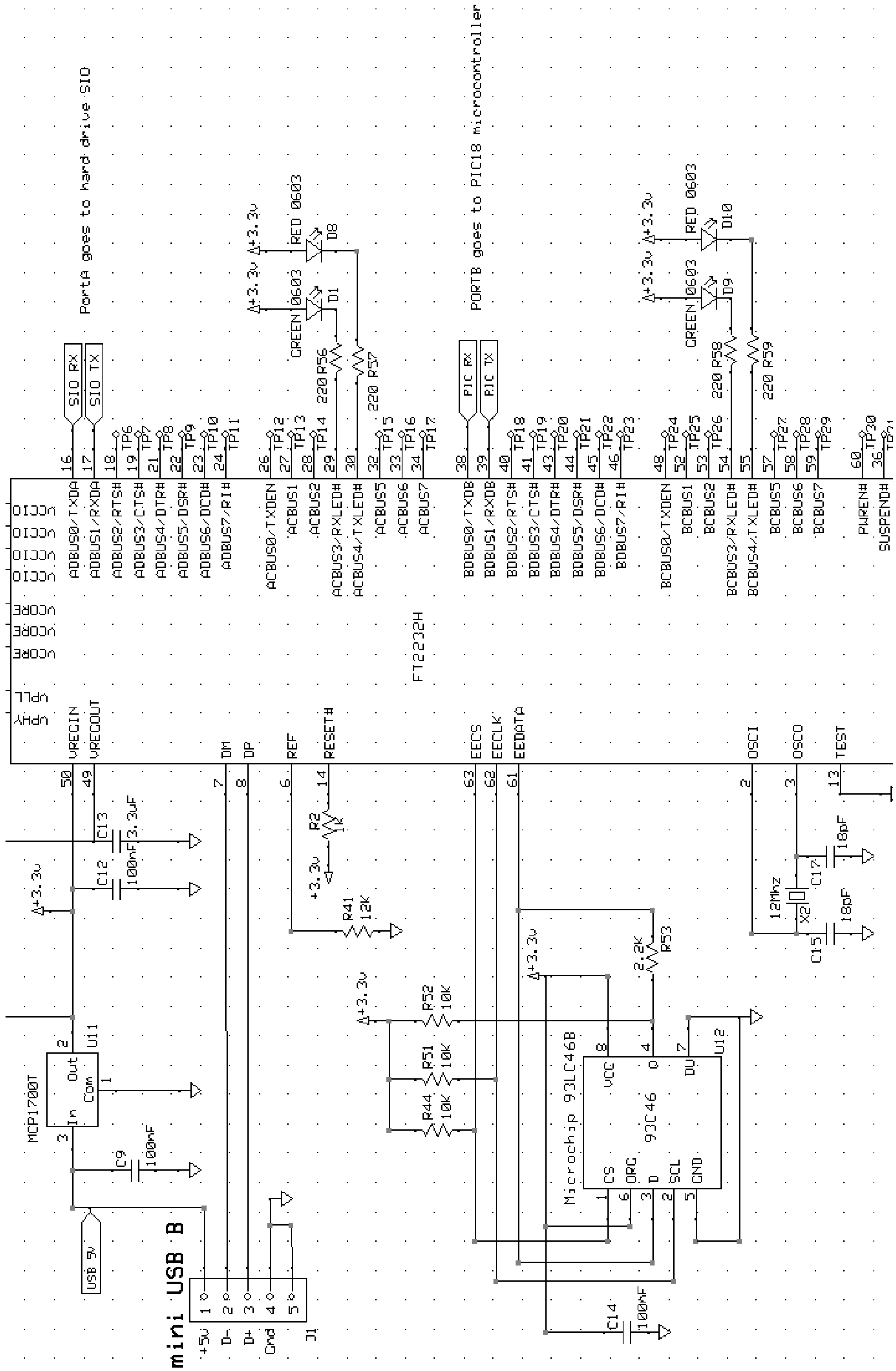
Appendix A-4: 12V Regulator



Appendix A-5: 5V Regulator

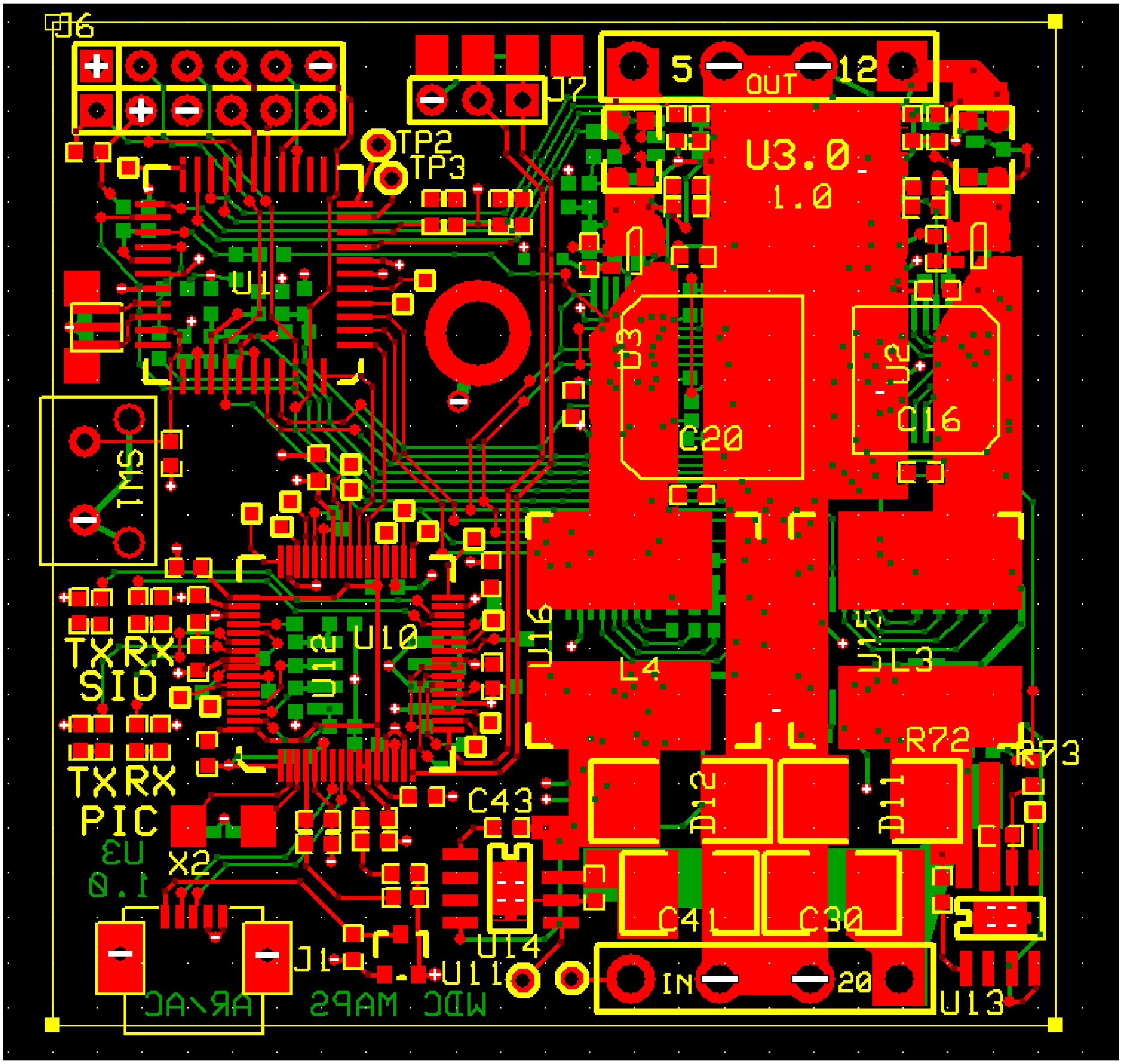


Appendix A-6: USB-UART IC

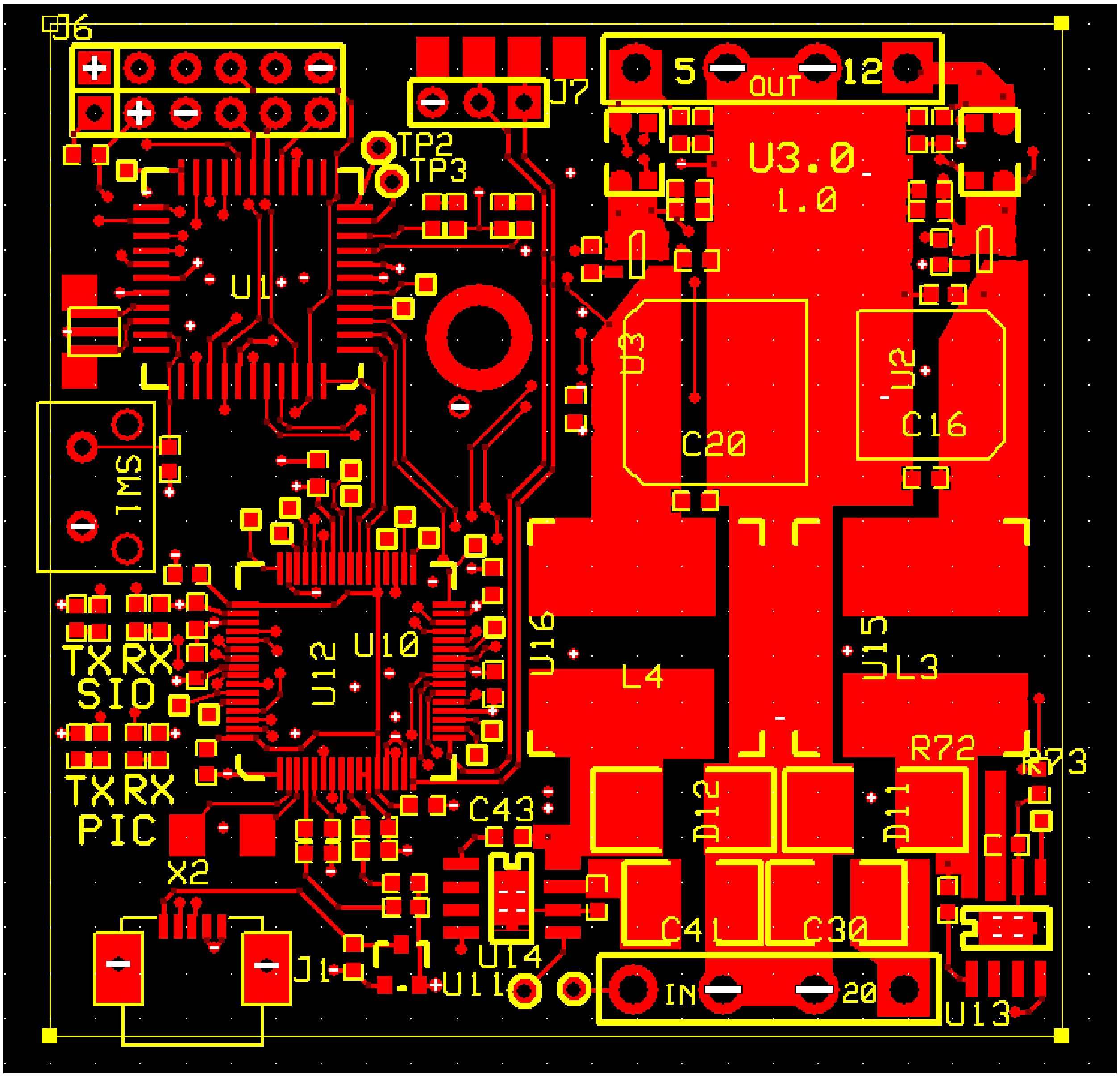


APPENDIX B: PCB Layout Design

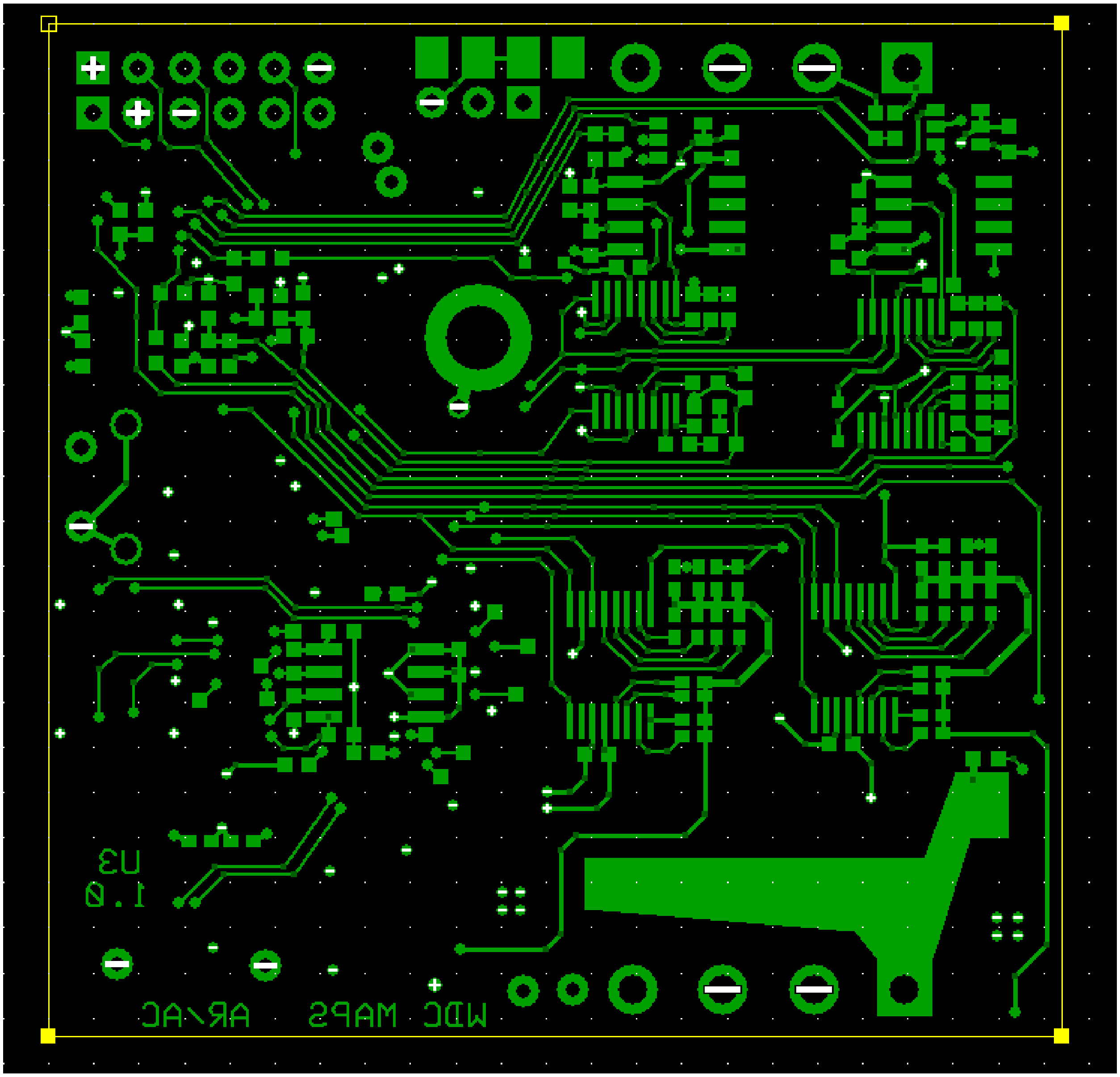
Appendix B-1: Top and Bottom Layers



Appendix B-2: Top Layer



Appendix B-3: Bottom Layer



APPENDIX C: Bill of Materials

PIC Microcontroller (18F45K20)								
Count	**Part ID**	**Value**	**Description**	**Size**	**MFR Part Number**	**MFR**	**Digi-Key Part Number**	**Unit Price**
2	C10, C11	20pf	Capacitor	0603		WD part		0.005
1	D4	Blue	Discrete LED	0603	APG1608QBC/E	Kingbright	754-1352-1-ND	0.72
1	D5	White	Discrete LED	0603		WD part		0.195
2	R1, R50	10K	Resistor	0603		WD part		0.005
1	R42, R43	220	Resistor	0603		WD part		0.005
1	R45	100K	Resistor	0603		WD part		0.005
1	NTC1	10K	Resistor	0603		WD part		0.005
1	X1	25 MHz	Crystal	5.0mm x 3.2mm	NX5032GA-25.000000MHZ	NDK	644-1041-1-ND	0.81
1	U1	18F45 K20	IC PIC MCU FLASH 16KX16 44-TQFP	44-TQFP	PIC18F45K20-E/PT	Microchip	PIC18F45K20-E/PT-ND	3.1
							Total	**4.86**

USB-UART IC (FTDI FT2232HL)								
Count	**Part ID**	**Value**	**Description**	**Size**	**MFR Part Number**	**MFR**	**Digi-Key Part Number**	**Unit Price**
2	C1, C3	4.7uF	Polarized Capacitor	0603		WD part		0.005
2	C15, C17	18pF	Capacitor	0603		WD part		0.005
12	C4, C5, C9, C12, C14, C18, C19, C21, C22, C23, C29, C26 C9, C12	.1uF	Capacitor	0603		WD part		0.005
0	L1, L2	--	Ferrite bead	0603		WD part		0.005

Count	Part ID	Value	Description	Size	MFR Part Number	MFR	Digi-Key Part Number	Unit Price
1	C13	3.3uF	Capacitor			WD part		0.005
1	R2	1K	Resistor	0603		WD part		0.005
1	R41	12K	Resistor	0603		WD part		0.005
3	R44, R51, R52	10K	Resistor	0603		WD part		0.005
1	R53	2.2K	Resistor	0603		WD part		0.005
4	R56, R57, R58, R59	220	Resistor	0603		WD part		0.005
1	X2	12Mhz	Crystal	5.0mm x 3.2mm	ABM3-12.000MHZ-B2-T	Abracon Corporation	535-9100-1-ND	1.88
2	D1, D9	Green	Discrete LED	0603		Kingbright Corp	754-1356-2-ND	0.195
2	D8, D10	Red	Discrete LED	0603		Avago Technologies	516-2005-1-ND	0.5
1	U10	FT2232H	FTDI USB-UART IC	64-LQFP	FT2232HL - REEL	FTDI, Future Technology Devices International Ltd	768-1024-1-ND	6.71
1	U11	MCP1700T	CMOS low dropout (LDO) Voltage Regulator, 3.3V	SOT-23 3-pin	MCP1700T-3302E/TT	Microchip	MCP1700T3302ETTCT-ND	0.45
1	U12	93LC46B	1Kbit low-voltage serial Electrically Erasable PROMs (EEPROM)	SOIC - 0.050 pitch - 8 pin	93LC46B-I/SN	Microchip	93LC46B-I/SN-ND	0.34
							Total	**10.91**

5V and 12V Regulator								
Count	**Part ID**	**Value**	**Description**	**Size**	**MFR Part Number**	**MFR**	**Digi-Key Part Number**	**Unit Price**
6	C31, C32, C33, C42, C43, C44	.01uF	Capacitor	0603		WD part		0.005
2	C40, C45	.1uF	Capacitor	0603		WD part		0.005
2	C30, C41	10uF	Capacitor	1812	C4532Y5V1H106Z	TDK	445-3483-1-ND	3.02
2	D11, D12	B540C	**Diode**	DO-214AB, SMC	B540C-13-F	Diodes Inc	**B540C-FDICT-ND**	1.08
2	R64, R78	2.5K	Resistor	0402		WD part		0.002
1	R73	3K	Resistor	0603		WD part		0.005
2	R65, R79	5K	Resistor	0402		WD part		0.002
2	R66, R88	10K	Resistor	0402		WD part		0.002

Count	Part ID	Value	Description	Size	MFR Part Number	MFR	Digi-Key Part Number	Unit Price
2	R67, R81	20K	Resistor	0402		WD part		0.002
2	R60, R74	31.6K	Resistor	0402		WD part		0.002
1	R72	33K	Resistor	0603		WD part		0.005
2	R61, R75	34.0K	Resistor	0402		WD part		0.002
2	R62, R76	36.5K	Resistor	0402		WD part		0.002
2	R68, R82	40K	Resistor	0402		WD part		0.002
2	R69, R83	80K	Resistor	0402		WD part		0.002
2	R63, R77	86.6K	Resistor	0402		WD part		0.002
2	R70, R84	160K	Resistor	0402		WD part		0.002
2	R71, R85	320K	Resistor	0402		WD part		0.002
2	L3, L4	15uH	Power Inductor	12.8mm x 12.8mm	CDRH127/LDNP-150MC	SUMIDA AMERICA COMPONENTS INC	**308-1333-1-ND**	
2	U15, U16	ADG738	IC MUX	16-TSSOP	ADG738BRUZ	Analog Devices	ADG738BRUZ-ND	3.85
2	U13, U14	TPS5450	**TPS5450 Step Down Conv**	8-SOIC Exposed Pad	TPS5450DDAR	Texas Instruments	296-21715-5-ND	6.25
							Total	**28.5**

5V and 12V Switch								
Count	**Part ID**	**Value**	**Description**	**Size**	**MFR Part Number**	**MFR**	**Digi-Key Part Number**	**Unit Price**
10	C2, C7, C8, C25, C27, C28, C29, C34, C35, C38	.1uf	Capacitor	0603		WD part		0.005
1	C16	150uF25vPoly	Aluminum Capacitor	0.327" L x 0.327" W	EMVE250ADA151MHA0G	United Chemi-Con	**565-2221-1-ND**	0.85
1	C20	470uF10vPoly	Aluminum Capacitor	0.406" L x 0.406" W	PCF1A471MCL1GS	Nichicon	**493-2996-1-ND**	1.99
4	C6, C24, C36, C37	.001uF				WD part		0.005
2	D2, D3	Red	Discrete LED	0603		Avago Technologies	516-2005-1-ND	0.5
2	D6, D7	Schottky				WD part		0.005
2	R33, R34	0.02 Ohm	Resistor	0603		WD part		0.005
2	R5, R15	20 Ohm	Resistor	0603		WD part		0.005
2	R27, R30	50 Ohm	Resistor	0603		WD part		0.005
2	R6, R16	200 Ohm	Resistor	0603		WD part		0.005
1	R32	470	Resistor	0603		WD part		0.005

Count	Part ID	Value	Description	Size	MFR Part Number	MFR	Digi-Key Part Number	Unit Price
3	R31, R54, R55	1.0K	Resistor	0603		WD part		0.005
4	R3, R4, R13, R14	1K	Resistor	0603		WD part		0.005
2	R7, R17	2K Ohm	Resistor	0603		WD part		0.005
4	R24, R36, R47, R49	3K	Resistor	0603		WD part		0.005
4	R25, R26, R28, R29	10K	Resistor	0603		WD part		0.005
4	R23, R35, R46, R48	15K	Resistor	0603		WD part		0.005
2	R8, R18	20K Ohm	Resistor	0603		WD part		0.005
2	R12, R22	100K	Resistor	0603		WD part		0.005
2	R9, R19	200K Ohm	Resistor	0603		WD part		0.005
4	R10, R20, R37, R39	1M	Resistor	0603		WD part		0.005
4	R11, R21, R38, R40	10M	Resistor	0603		WD part		0.005
2	U9, U7		FET	IC SWITCH ANALOG N-C	MMBFJ112	Fairchild	MMBFJ112CT-ND	0.54
2	U4, U5	ADM4073F	IC AMP CS 1.8MHZ	SOT-23-6	ADM4073FWRJZ-REEL7	Analog Devices	ADM4073FWRJZ-REEL7CT-ND	2.15
2	U6, U8	TL072	IC OPAMP JFET DUAL LN	8-SOIC	TL072BIDT	STMicroelectronics	497-7040-1-ND	1.33
2	U15, U16	ADG738	IC MUX	16-TSSOP	ADG738BRUZ	Analog Devices	ADG738BRUZ-ND	3.85
							Total	**19.88**

The board components cost a total of $64.15 from the 4 components above. The fabrication of the PCB by ExpressPCB cost $261. The grand total for all parts is $325.15.

APPENDIX D: Source Code

```
'*************************************************************
'*  Name    : U3 - 0.1                                      *
'*  Author  : Allan Chan                                    *
'*  Notice  : Copyright (c) 2010 [select VIEW...EDITOR OPTIONS] *
'*          : All Rights Reserved                           *
'*  Date    : 3/19/2010                                     *
'*  Version : 0.1.7                                         *
'*  Notes   :  interupt
'*
'*          :  Version  1.7   3/19/10  based on K vesion.
'*          :    changed to "BUFFERED_HSERIN3.Inc"
'*          :    changed XTAL to 25
'*          :    New voltage setting commands: VSET5, VSET12
'*          :    New OVP commands: OVP5, OVP12, GETOVP5, GETOVP12
'*          :    added MAXOVP5 and MAXOVP12 symbols
'*          :    added MINVOLTS symbol
'*          :    changed Volt_adjust value
'*          :    added HELP command
'*          :    using HRSIn for UART
'*          :  Version  1.0m   10/26/09  based on K vesion.
'*          :    changed to "BUFFERED_HSERIN4.INC" which has interupts
                   turned off around the read section
'*          :    updated U2ID command to "Code Version 1.0m"
'*          :  Version  1.0k  7/15/09
'*          :    added main_loop_count (word) to help with house keeping
'*          :    added re calibrating the ADC every 2^16 main loops (about 600mS)
'*          :  Version  1.0j  7/14/09 built from 1.0e source to
'*          :    avoid a bug intorduced somewhere between e & i
'*          :    added a SIO counter to kick out if the drive runs away
'*
'*          :  Version  1.0e  12/05/08
'*          :  Added erase flash memory, write
'*          :  Version  1.0d  12/02/08
'*          :  Added delay to boot, config = eeprom 0,
'*          :  Version  1.0c  11/19/08                      *
'*          :  added ability for larger input strings.     *
'*************************************************************
    Device = 18F45K20         ' Use a device with full speed USB capabilities
    Xtal = 25                 ' Set the oscillator speed to 48MHz (using a 20MHz crystal)

   ' ON_HARDWARE_INTERRUPT GoTo Interrupt_section

    Dim Blue As PORTB.1   'Left LED
    Dim White As PORTB.0   'Right LED
    Dim butt As PORTC.0
    Dim temp As Byte
    Dim temp2 As Byte
    Dim Param1 As Word
    Dim param1_low As Param1.LowByte
    Dim param1_hi As Param1.HighByte
    Dim tempword As Word
    Dim tempword_low As tempword.LowByte
    Dim tempword_hi As tempword.HighByte
    Dim in_Buf_len  As Byte   ' the current legnth of IN_BUFFER

    Dim tempfloat As Float
    Dim IN_BUFFER As String * 64             ' Used to hold some input characters
    Dim CMD_BUFFER As String * 64
    Dim sio_buf[20] As Byte
    Dim Temp_String As String * 1
    Dim USB_out_buff As String * 65
    Dim TIMER0 As TMR0L.Word              ' Combine TMR0L, and TMR0H into WORD variable TIMER0
    Dim tempLong As Dword
    Dim Button_var As Word                ' used to debounce button
    Dim ADC_cal As Float
    Dim time0 As Word
    Dim inttemp As Byte                   ' temp byte for the interrupt
    Dim Ramp5up As Byte
    Dim Ramp12up As Byte
    Dim SIO_count As Word
    Dim main_loop_count As Word

    Dim OVP5  As Word       ' AC 2/4  Over Voltage Protection for 5VDC rail
    Dim OVP12 As Word       ' AC 2/4  Over Voltage Protection for 12VDC rail

    Dim PP2 As Byte System
```

```
    Symbol Timer0_reset = $0000
    Symbol CARRY_FLAG = STATUS.0      ' High if microcontroller doesn't have control over the buffer

    Symbol RBIF = INTCON.0    ' RB Port Interrupt Flag
    Symbol INT0IF = INTCON.1 ' INT0 External Interrupt Flag
    Symbol TMR0IF = INTCON.2 ' TMR0 Overflow Interrupt Flag
    Symbol RBIE = INTCON.3    ' RB Port Change Interrupt Enable
    Symbol INT0IE = INTCON.4 ' INT0 External Interrupt Enable
    Symbol TMR0IE = INTCON.5 ' TMR0 Overflow Interrupt Enable
    Symbol PEIE = INTCON.6    ' Peripheral Interrupt Enable     *********
    Symbol GIEL = INTCON.6    ' Peripheral Interrupt Enable
    Symbol GIE = INTCON.7     ' Global Interrupt Enable       **********
    Symbol GIEH = INTCON.7    ' Global Interrupt Enable
    Symbol TMR1IE = PIE1.0 ' TMR1 Overflow Interrupt Enable bit
    Symbol TMR2IE = PIE1.1 ' TMR2 to PR2 Match Interrupt Enable bit
    Symbol CCP1IE = PIE1.2 ' CCP1 Interrupt Enable bit
    Symbol SSPIE = PIE1.3  ' Master Synchronous Serial Port Interrupt Enable bit
    Symbol TXIE = PIE1.4   ' EUSART Transmit Interrupt Enable bit
    Symbol RCIE = PIE1.5   ' EUSART Receive Interrupt Enable bit  *********
    Symbol ADIE = PIE1.6   ' A/D Converter Interrupt Enable bit
    Symbol SPPIE = PIE1.7  ' Streaming Parallel Port Read/Write Interrupt Enable bit
    Symbol TMR1IF = PIR1.0 ' TMR1 Overflow Interrupt Flag bit
    Symbol TMR2IF = PIR1.1 ' TMR2 to PR2 Match Interrupt Flag bit
    Symbol CCP1IF = PIR1.2 ' CCP1 Interrupt Flag bit
    Symbol SSPIF = PIR1.3  ' Master Synchronous Serial Port Interrupt Flag bit
    Symbol TXIF = PIR1.4   ' EUSART Transmit Interrupt Flag bit
    Symbol RCIF = PIR1.5   ' EUSART Receive Interrupt Flag bit   ********
    Symbol ADIF = PIR1.6   ' A/D Converter Interrupt Flag bit
    Symbol SPPIF = PIR1.7  ' Streaming Parallel Port Read/Write Interrupt Flag bit

    Symbol control5 = LATD.0          'AC 2/8    was PORTD.0
    Symbol control12 = LATD.1         'AC 2/8    was PORTD.1
    Symbol Sync12 = LATD.2            'AC 2/8    was PORTD.2
    Symbol Sync5 = LATD.3             'AC 2/8    was PORTD.3
    Symbol Sdata = PORTC.5            'AC 2/1    was Symbol Sdata = PORTD.4
    Symbol Sclk = PORTC.3             'AC 2/1    was Symbol Sclk = PORTD.5
    Symbol M_Button = PORTC.0         'AC 2/1    was Symbol M_Button = PORTD.6
    Symbol TP4 = PORTA.3              'AC 2/1    was Symbol TP4 = PORTD.7
    Symbol PWM5 = PORTC.2
    Symbol PWM12 = PORTC.1

    Symbol Prog_5V = LATD.4             'AC 2/8    was PORTD.4  Chip select for 5 volt regulator
    Symbol Prog_12V = LATD.5            'AC 2/8    was PORTD.5  Chip select for 12 volt regulator
    Symbol SPIhead_CS = PORTB.4         'AC 2/4
    Symbol TP1 = PORTD.7                'input AC 2/10
    Symbol TP5 = PORTD.6                'input AC 2/10
    Symbol TP2 = PORTB.3                'input AC 2/10
    Symbol TP3 = PORTB.2                'input AC 2/10

    Symbol MAXOVP5 = 5300               'AC 2/18   the max ovp settings by default
    Symbol MAXOVP12 = 13300
    Symbol MINVOLTS = 3000              'AC 3/1    the minimum voltage setting

    Symbol Volt_adjust_input = 25.88    'AC 2/18  to obtain voltage from external power supply input
    Symbol Volt_adjust = 51.66667       'AC 2/18        to obtain voltages at power output

    'Symbol Volt_adjust =2370.421  'divide ADC float by this value to get volts
    Symbol Cur_adjust =  10000    'divide ADC float by this value to get Current
    Symbol QUANTA = 3300.0 / 1022.0 ' Quatasising constant for ADC reading to Voltage calculation
    Symbol T25 = 0.0033540 ' Thermistor's t25 value
    Symbol B = 1.0 / 3489.0 ' Thermistor's B value
    Dim THERMISTOR_VALUE As Word ' Holds the raw thermistor ADC reading
    Dim RESISTANCE As Float ' Holds the resistance of the thermistor for a given temperature
    Dim TEMPERATURE As Float ' Holds the floating point temperature in degrees Centigrade
    Dim TEMPF As Float ' Temporary variable for the thermistor linearisation
    Dim LN As Float ' Used to hold the LOG of the thermistor value
    ' Load the USART 1 interrupt handler and buffer read subroutines into memory
    Include "BUFFERED_HSERIN3.INC"
    'Change #1 for Special  used "BUFFERED_HSERIN4.INC"
    Declare Adin_Res 10

'--------------------------------------------------------------------------------
GoTo Start

Interrupt_section:

end_int:
      Context Restore


'----------------------------------------------------------------------------------
```

```
Start:
       TRISA  = %11111111     ' Tri-State register 1 = Input,  0 = Output
        PORTB  = %00010000    ' set portb.4 to output    AC 2/8
       TRISB  = %11101100     '  Bit.0 = White LED,   Bit.1 = Blue LED
       TRISC  = %11111111
        LATD  = %00111100      'AC 2/10  read-modify-write operation sets the control values for ADG
       TRISD  = %11000000
       TRISE  = %11111111

       TXSTA    = %10100111                'Setup RS232 port
       RCSTA    = %10110000               ' Setup RS232 port
       BAUDCON  = %00001010              ' Setup RS232 port
    SPBRGH   = 0                      'Setup RS232 port
    SPBRG    = 54                     'Setup RS232 port           AC 2/1

    Clear                            ' Clear all RAM before we start

    Low Blue        'Initialize Blue LED off
    Low White       'Initialize White LED off
    DelayMS 1000

   GoSub  init_PWM                  'set up the PWMs
                                     'including turning off the power
    temp = ERead 0                   'eeprom address 0 = config
    If temp = 255 Then
       EWrite 0,[0]                  'initlize value
       EWrite 1,[3]                 'set usb delay to 3 second default
       temp = 0
    End If
                                              'bit 0: power on at power up   0= off, 1 = on
     If temp.0 = 1 Then
        GoSub turn_on
     End If


     '***  Set up the timer interupt for the USB

     T0CON = %10000100   '


    '******************
    '*** Set up Interupts ***
    '************************


     INIT_USART_INTERRUPT     ' Initiate the USART 1 serial buffer interrupt
     CLEAR_SERIAL_BUFFER      ' Clear the serial buffer and reset its pointers

     GoSub  INIT_ADC          ' setup the ADC
     GoSub cal_adc            'Calibrate the ADC  The command "CALADC 1" will re-calibrate the
ADC

     OVP5 = MAXOVP5           ' Initialize values for 5VDC Over Voltage Protection     AC 2/4
     OVP12 = MAXOVP12         ' Initialize values for 12VDC Over Voltage Protection    AC 2/4

    Param1 = 5000             'AC 2/10    Initialize 5V and 12V rails
    GoSub vset5
    Param1 = 12000
    GoSub vset12

    Low White                 'AC 2/10    Light LED for 1 second
    Toggle White
    DelayMS 1000
    Toggle White

Main_loop:

   USB_out_buff = 13 + ">"    'AC 2/8
   GoSub Usb_out

    While 1 = 1

        If  HrsInBuffLen > 0 Then         ' Keep the USB interface alive

            ' Parse input
                 temp = HRSIn
                 Temp_String = temp

                 Select Temp_String
                    Case 0x0d           'check for a "Return" if so, Parse command
                        GoSub parse
```

```
                    Case 0x08          'backspace
                       If Len(CMD_BUFFER) > 1 Then           'more than 1 char in buffer
                          temp = -1 +Len(CMD_BUFFER)
                          CMD_BUFFER = Left$(CMD_BUFFER,temp)
                       Else
                           Clear CMD_BUFFER      '0-1 char in buffer
                       EndIf
                    Case Else
                       CMD_BUFFER = CMD_BUFFER +  Temp_String
                EndSelect

               If Temp_String[0]<> 13 Then           'check for and delete a "RETURN" hex
                       USB_out_buff =  Temp_String   'Send out keystroke
                       GoSub Usb_out

                       If Temp_String[0] = 0x08 Then     ' AC 2/8    backspace
                          USB_out_buff = " " + 0x08   ' Clear last keystroke and backspace again
                          GoSub Usb_out                 ' Make Hyperterminal backspace correctly
                       End If
               End If
       EndIf

       Clear temp2

       If temp2 = 1 Then
          Call Usb_out
       End If

       If Button_var <> 0 Then      ' If Button_var <> 0 we are in De-bounce
            If butt = 1  Then       '    butt = 1 - button is not pressed
               Inc Button_var       '    so we will Inc the Button_var to get out of de-bounce
            End If
       Else                         ' Else we are not in De-bounce
           If butt = 0 Then GoSub do_button  ' If butt = 0 the button has just been pressed
       End If

       If  main_loop_count = 0 Then
           GoSub  cal_adc 'Calibrate the ADC  The command "CALADC 1" will re-calibrate the ADC
       End If

       Inc main_loop_count   'count main loops (about 100,000 per second)
                             'so we can do house keeping

   Wend
'End of Main Loop


do_button:

   'reset
   Button_var = 60000   'Start the debounce counter: Overflow and stop at 65,536
   If control5 = 0 Or control12 = 0 Then    'if either of them is on - TURN OFF
        temp = 255-Ramp5up  'invert the ramp to get the right slope speed for down ramps
        CCP1CON.5=temp.1     'set up the 5 volt PWM
        CCP1CON.4=temp.0
        CCPR1L = temp /4
        temp = 255-Ramp12up     'invert the ramp to get the right slope speed for down ramps
        CCP2CON.5=temp.1        'set up the 12 volt PWM
        CCP2CON.4=temp.0
        CCPR2L = temp / 4
        control5 = 1
        control12 = 1
   Else               'TURN ON
        CCP1CON.5=Ramp5up.1     'set up the 5 volt PWM
        CCP1CON.4=Ramp5up.0
        CCPR1L = Ramp5up /4
        CCP2CON.5=Ramp12up.1     'set up the 12 volt PWM
        CCP2CON.4=Ramp12up.0
        CCPR2L = Ramp12up / 4
        control5 = 0
        control12 = 0

   End If
    Return

Usb_out:
       HRSOut USB_out_buff
       Return

parse:
      CMD_BUFFER =  ToUpper (CMD_BUFFER)
```

```
Clear Param1         'AC 2/18
GoSub Get_Param      'puts the numeric part of the command into the Param1 register

USB_out_buff = 13   'AC 2/17  prints a new line
Call Usb_out

Select    CMD_BUFFER
    Case "BON"          'AC 2/10  updated all LED commands to blue and white on/off
      High Blue
    Case "BOFF"
      Low Blue
    Case "WON"
      High White
    Case "WOFF"
      Low White

    Case "GON"     'AC 2/17 Added back Utopia2.5 commands for LEDs
      High PORTB.1
    Case "GOFF"
      Low PORTB.1
    Case "RON"
      High PORTB.0
    Case "ROFF"
      Low PORTB.0

    Case "TIM"
      USB_out_buff = "All Hail the Dark Lord!" + 13    'AC 2/2  added "13 +"
      Call Usb_out
    Case "STARS"
      GoSub Stars
    Case "HOMER"
      USB_out_buff = "Mmmmmm  Donuts!"  + 13
      Call Usb_out
      USB_out_buff = "D'oh!" + 13
      Call Usb_out
    Case "ROB"
      USB_out_buff = "Ring Master"  + 13
      Call Usb_out


    Case "ADC"
      If Param1 < 9 Then
          GoSub Get_ADC
          USB_out_buff = Str$ (Dec tempfloat) +13
      Else
          USB_out_buff = "Too big, choose 0-9" + 13
      EndIf

    Call Usb_out

    Case "DELAY"
      High White
      DelayMS Param1
      Low White
    Case "TEMP"
      GoSub Thermistor
      'USB_out_buff = Str$ (Dec3 TEMPERATURE) +"C" + 13
      USB_out_buff = "N/A"
      Call Usb_out

    Case "PWM1"
      CCP1CON.5=Param1.1
      CCP1CON.4=Param1.0
      Param1 = Param1 & 255        'trim to lower 8 bits
      CCPR1L = Param1 /4

      USB_out_buff = "PWM1 set to " + Str$ (Dec Param1)
      Call Usb_out

    Case "PWM2"
      CCP2CON.5=Param1.1
      CCP2CON.4=Param1.0
      Param1 = Param1 & 255        'trim to lower 8 bits
      CCPR2L = Param1 / 4

      USB_out_buff = "PWM2 set to " + Str$ (Dec Param1)
      Call Usb_out

    Case "SPI1"                               'For debugging SPI port on +5V ramp
      Sync5 = 0
```

```
          SHOut Sdata,Sclk, 1, [param1_low ]
          Sync5 = 1
          USB_out_buff = "SPI set to " + Str$ (Dec param1_low) + 13
          Call Usb_out
        Case "SPI2"                                 'For debugging SPI port on +12V ramp
          Sync12 = 0
          SHOut Sdata,Sclk, 1, [param1_low ]
          Sync12 = 1
          USB_out_buff = "SPI set to " + Str$ (Dec param1_low) + 13
          Call Usb_out

'*********************************************************************************************
'*********************************************************************************************
  'New code for Utopia 3 below       AC 2/8

        Case "SPI3"                           'For debugging SPI port on +5V regulator  AC 2/4
          Prog_5V = 0 'set chip select low for 5V control
          SHOut Sdata,Sclk, 1, [param1_low ]  'Bang out data
          Prog_5V = 1  'set chip select hi for 5V control
          USB_out_buff = "SPI set to " + Str$ (Dec param1_low) + 13
          Call Usb_out
        Case "SPI4"                          'For debugging SPI port on +12V regulator  AC 2/4
          Prog_12V = 0  'set chip select low for 12V control
          SHOut Sdata,Sclk, 1, [param1_low ]  'Bang out data
          Prog_12V = 1  'set chip select hi for 12V control
          USB_out_buff = "SPI set to " + Str$ (Dec param1_low) + 13
          Call Usb_out
        Case "SPI5"                       'For debugging SPI port on SPI header CS      AC 2/4
          SPIhead_CS = 0  'set chip select low for 12V control
          SHOut Sdata,Sclk, 1, [param1_low ]  'Bang out data
          SPIhead_CS = 1  'set chip select hi for 12V control
          USB_out_buff = "SPI set to " + Str$ (Dec param1_low) + 13
          Call Usb_out

        Case "GETVSET"                          'Voltage setting for debugging          AC 2/4
        ' input={2000-18000}: Convert Vout to SDATA using formula
                            ' Sdata = (256/16000) * (Vout - 2000)
        ' input=0: list all Sdata-Voltage pairs for each given voltage value
        ' input=1: list all Sdata-Voltage pairs for each given SDATA value

           If Param1 >= 2000 And Param1 <= 18000 Then
                  tempword = Param1
                  tempword = tempword - 2000      ' Vout - 2000
                  tempword = tempword * 2         ' Vout * (256/128)
                  tempword = tempword / 125       ' Vout / (16000/128)

                  If tempword > 255 Then tempword = 255
                  USB_out_buff = "REG will be set to " + Str$ (Dec tempword_low) + 13
                  Call Usb_out
           ElseIf Param1 = 0 Then
                  USB_out_buff = "   SDATA    Voltage"
                  Call Usb_out

                  For Param1 = 2000 To 18000 Step 125
                      tempword = Param1
                      tempword = tempword - 2000      ' Vout - 2000
                      tempword = tempword * 2         ' Vout * (256/128)
                      tempword = tempword / 125       ' Vout / (16000/128)

                      If tempword > 255 Then tempword = 255
                          USB_out_buff = 13 + "    " + Str$ (Dec tempword_low) + "          "
                                        + Str$ (Dec Param1)
                          Call Usb_out
                  Next

                  USB_out_buff = 13     'AC 2/17 print newline
                  Call Usb_out
           ElseIf Param1 = 1 Then
                  USB_out_buff = "   SDATA    Voltage"
                  Call Usb_out

                  For Param1 = 0 To 255 Step 1
                      tempword = Param1
                      tempword = tempword * 125      ' Vout - 2000
                      tempword = tempword / 2         ' Vout * (256/128)
                      tempword = tempword + 2000       ' Vout / (16000/128)

                      USB_out_buff = 13 + "    " + Str$ (Dec Param1) + "          "
                                        + Str$ (Dec tempword)
                      Call Usb_out
                  Next
```

```
                USB_out_buff = 13      'AC 2/17 print newline
                Call Usb_out
        Else
                USB_out_buff = "Please input a voltage(2000-18000)"
                Call Usb_out
                USB_out_buff = 13 + "Or input 0 to list possible SDATA values given voltage"
                Call Usb_out
                USB_out_buff = 13 + "Or input 1 to list possible voltage " +
                               "values given SDATA" + 13
                Call Usb_out
        EndIf

      Case "OVP5"
        If Param1 > MINVOLTS And Param1 < 6000 Then
            OVP5 = Param1
            USB_out_buff = "OVP set to " + Str$ (Dec OVP5) + 13
            Call Usb_out
        Else
            USB_out_buff = "Please specify valid OVP in mV [3000 - 6000]" + 13
            Call Usb_out
        EndIf
      Case "OVP12"                           ' AC 2/4
        If Param1 > MINVOLTS And Param1 < 18000 Then
            OVP12 = Param1
            USB_out_buff = "OVP set to " + Str$ (Dec OVP12) + 13
            Call Usb_out
        Else
            USB_out_buff = "Please specify valid OVP in mV [3000 - 18000]" + 13
            Call Usb_out
        EndIf
      Case "GETOVP5"       'AC 2/5 calls for current setting of OVP, specify 5 or 12
            USB_out_buff = "OVP setting for 5VDC: " + Str$ (Dec OVP5) + 13
            Call Usb_out
      Case "GETOVP12"
            USB_out_buff = "OVP setting for 12VDC: " + Str$ (Dec OVP12) + 13
            Call Usb_out
      Case "VSET5"        'Program 5 volt regulator
            GoSub vset5
      Case "VSET12"       'Program 12 volt regulator
            GoSub vset12
      Case "HELP"
            GoSub help
  'End new code for Utopia3
'***********************************************************************************************
'***********************************************************************************************

    Case "PON"
         DelayMS Param1        'delay if one is entered
         CCP1CON.5=Ramp5up.1     'set up the 5 volt PWM
         CCP1CON.4=Ramp5up.0
         CCPR1L = Ramp5up /4
         CCP2CON.5=Ramp12up.1     'set up the 12 volt PWM
         CCP2CON.4=Ramp12up.0
         CCPR2L = Ramp12up / 4
         control5 = 0
         control12 = 0
    Case "PON5"
         DelayMS Param1        'delay if one is entered
         CCP1CON.5=Ramp5up.1     'set up the 5 volt PWM
         CCP1CON.4=Ramp5up.0
         CCPR1L = Ramp5up /4
         control5 = 0
    Case "PON12"
         DelayMS Param1        'delay if one is entered
         CCP2CON.5=Ramp12up.1     'set up the 12 volt PWM
         CCP2CON.4=Ramp12up.0
         CCPR2L = Ramp12up / 4
         control12 = 0

    Case "POFF"
         'toggle green
         DelayMS Param1        'delay if one is entered
         'RCSTA.7 =0          'turn off uart
         temp = 255-Ramp5up   'invert the ramp to get the right slope speed for down ramps
         CCP1CON.5=temp.1     'set up the 5 volt PWM
         CCP1CON.4=temp.0
         CCPR1L = temp /4
         temp = 255-Ramp12up     'invert the ramp to get the right slope speed for down ramps
         CCP2CON.5=temp.1        'set up the 12 volt PWM
         CCP2CON.4=temp.0
         CCPR2L = temp / 4
```

```
     control5 = 1
     control12 = 1
Case "POFF5"
     DelayMS Param1       'delay if one is entered
     temp = 255-Ramp5up   'invert the ramp to get the right slope speed for down ramps
     CCP1CON.5=temp.1     'set up the 5 volt PWM
     CCP1CON.4=temp.0
     CCPR1L = temp /4
     control5 = 1
Case "POFF12"
     DelayMS Param1          'delay if one is entered
     temp = 255-Ramp12up     'invert the ramp to get the right slope speed for down ramps
     CCP2CON.5=temp.1        'set up the 12 volt PWM
     CCP2CON.4=temp.0
     CCPR2L = temp / 4
     control12 = 1

Case "CALADC"
     GoSub cal_adc
     USB_out_buff = Str$ (Dec8,ADC_cal)
     GoSub Usb_out
Case "VOLTS"
     GoSub volts
Case "CUR"
     GoSub cur

Case "SIO"
     USB_out_buff = "Command removed" + 13
     Call Usb_out

Case "RAMP"
     GoSub Ramp5
     GoSub Ramp12
Case "RAMP5"
     GoSub Ramp5
Case "RAMP12"
     GoSub Ramp12
Case "U3ID"
     USB_out_buff = "U3 Code Version 1.0m" + 13
     Call Usb_out
Case "U2ID"
     USB_out_buff = "U2 Code Version 1.0m" + 13
     Call Usb_out
Case "DATA"
     Call all_data
Case "CONFIG"
     EWrite 0, [Param1]
Case "USBDELAY"
     EWrite 1, [Param1]
Case "BOOT"
     GoTo (5000h)
Case "ADIN"                'AC 2/18
      Param1 = ADIn 0
      USB_out_buff = "12cur:        " + Str$ (Dec Param1)
      Call Usb_out

      Param1 = ADIn 3                       'AC 3/1
      USB_out_buff = 13 + "5cur:         " + Str$ (Dec Param1)
      Call Usb_out

      Param1 = ADIn 1
      USB_out_buff = 13 + "12v:          " + Str$ (Dec Param1)
      Call Usb_out

      Param1 = ADIn 2
      USB_out_buff = 13 + "5v:           " + Str$ (Dec Param1)
      Call Usb_out

      Param1 = ADIn 6
      USB_out_buff = 13 + "12vdr sense:  " + Str$ (Dec Param1)
      Call Usb_out

      Param1 = ADIn 5
      USB_out_buff = 13 + "5vdr sense:   " + Str$ (Dec Param1)
      Call Usb_out

      Param1 = ADIn 4
      USB_out_buff = 13 + "drive vin:    " + Str$ (Dec Param1) + 13
      Call Usb_out

Case ""
```

```
    'do nothing
  Case Else

    USB_out_buff = "   Unknown Command >" +  CMD_BUFFER + 13
    Call Usb_out

  EndSelect


   Clear CMD_BUFFER
   USB_out_buff = 13 + ">"
   GoSub Usb_out

   Return



Get_Param:
        temp = 0                        ' Start at position 0 within the string

        'look for a space or null' Create a loop
        While  CMD_BUFFER[temp] <> 0x20 And CMD_BUFFER[temp] <> 0x00
               Inc temp                  ' Move to the next position within the string
        Wend  ' Keep looping until the end of the string is found

        temp2 = Len(CMD_BUFFER)
        If temp2 <> temp Then                        'AC 2/28 A param value exist
              temp2 = temp2 - temp
              USB_out_buff = Right$(CMD_BUFFER,temp2)   ' Stores the param as a string
              Param1 = Val(USB_out_buff,Dec)            ' Convert param string to integer value
        Else
              Param1 = 0                        'AC 2/28 Set param to be "0" if no param exist
        EndIf

        CMD_BUFFER  = Left$(CMD_BUFFER,temp)

        Return

INIT_ADC:                            'Setup the ADC
        ADCON1 = %00000000      'ADCON1 = %00011000   'AC 2/18
        ADCON2 = %10001111      'ADCON2 = %10001100   'AC 2/18
                                    'ADCON2 = %10100111
        Return
cal_adc:
        ADC_cal = 1 'Param1        used CALADC 1 to calibrate
        Param1 = 3
        Call Get_ADC
        ADC_cal = 33000 /tempword                'AC 2/17
        Return
Get_ADC:    'line 335

        tempfloat = ADIn Param1
        DelayUS 2

        Return


'*******************************
'****    Calculate Ramps    ****
'*******************************
ramps:
       'Set up the resistor ladder and cap

       Sync5 = 0  'set chip select low for 5V control
       SHOut Sdata,Sclk, 1, [param1_low ]  'Bang out data
       Sync5 = 1  'set chip select hi for 5V control

       USB_out_buff = 13 + "SPI5 set to " + Str$ (Dec param1_low) +13 'report back what we did
       Call Usb_out

       Sync12 = 0 'set chip select low for 12V control
       SHOut Sdata,Sclk, 1, [param1_low ]   'Bang out data
       Sync12 = 1  'set chip select hi for 12V control

       USB_out_buff = "SPI12 set to " + Str$ (Dec param1_low) +13 'report back what we did
       Call Usb_out

       'Set up the PWMs

       '
       CCP1CON.5=param1_hi.1     'set up the 5 volt PWM
```

```
       CCP1CON.4=param1_hi.0
       'Param1 = Param1 & 255        'trim to lower 8 bits
       CCPR1L = param1_hi /4
       Ramp5up = param1_hi
       USB_out_buff = "PWM1 set to " + Str$ (Dec param1_hi) +13 'report back what we did
       Call Usb_out

       CCP2CON.5=param1_hi.1
       CCP2CON.4=param1_hi.0
           'Param1 = Param1 & 255        'trim to lower 8 bits
       CCPR2L = param1_hi / 4
       Ramp12up = param1_hi
       USB_out_buff = "PWM2 set to " + Str$ (Dec param1_hi) +13
       Call Usb_out
       Return

power_delay:        'delay if one is entered
       DelayMS Param1
       Return

volts:                                                'AC 2/18
        USB_out_buff = "5Vin  12Vin  5Vout 12Vout  Drive Vin" + 13  'Str$ (DEC8,ADC_cal)
        GoSub Usb_out

        Param1 = 5     '5Vin
        GoSub Get_ADC
        tempfloat = tempfloat / Volt_adjust
        USB_out_buff = Str$ (Dec3,tempfloat) + " "

        Param1 = 6     '12Vin
        GoSub Get_ADC
        tempfloat = tempfloat / Volt_adjust
        USB_out_buff = USB_out_buff + Str$ (Dec3,tempfloat) + " "

        Param1 = 2     '5Vout
        GoSub Get_ADC
        tempfloat = tempfloat / Volt_adjust
        USB_out_buff = USB_out_buff + Str$ (Dec3,tempfloat) + " "

        Param1 = 1     '12Vout
        GoSub Get_ADC
        tempfloat = tempfloat / Volt_adjust
        USB_out_buff = USB_out_buff + Str$ (Dec3,tempfloat) + " "

        Param1 = 4     'Drive Vin                'AC 2/18
        GoSub Get_ADC
        tempfloat = tempfloat / Volt_adjust_input
        USB_out_buff = USB_out_buff + Str$ (Dec3,tempfloat) + " "

        GoSub Usb_out

        Return

cur:
        USB_out_buff = 13 + "5VCur  12VCur" + 13  'Str$ (DEC8,ADC_cal)
        GoSub Usb_out

        Param1 = 3     '5V Current        'AC 3/1
        GoSub Get_ADC
        tempfloat = tempfloat / Cur_adjust
        USB_out_buff =   Str$ (Dec3,tempfloat) + " "

        Param1 = 0     '12V Current
        GoSub Get_ADC
        tempfloat = tempfloat / Cur_adjust
        USB_out_buff = USB_out_buff + Str$ (Dec3,tempfloat)
         GoSub Usb_out

        Return

'*************************************************************
'****  Return all of the important ADC data               ****
'*************************************************************
all_data:
        Param1 = 5     '5Vin
        GoSub Get_ADC
        tempfloat = tempfloat / Volt_adjust
        USB_out_buff = 13+ "5Vin= " + Str$ (Dec3,tempfloat) + " "
        GoSub Usb_out

        Param1 = 6     '12Vin
```

```
        GoSub Get_ADC
        tempfloat = tempfloat / Volt_adjust
        USB_out_buff = "12Vin= " + Str$ (Dec3,tempfloat) + " "
        GoSub Usb_out

        Param1 = 2     '5Vout
        GoSub Get_ADC
        tempfloat = tempfloat / Volt_adjust
        USB_out_buff = "5Vout= " + Str$ (Dec3,tempfloat) + " "
        GoSub Usb_out

        Param1 = 1     '12Vout
        GoSub Get_ADC
        tempfloat = tempfloat / Volt_adjust
        USB_out_buff = "12Vout= " + Str$ (Dec3,tempfloat) + " "
        GoSub Usb_out

        Param1 = 3     '5V Current           'AC 3/1
        GoSub Get_ADC
        tempfloat = tempfloat / Cur_adjust
        USB_out_buff = "5cur= " + Str$ (Dec3,tempfloat) + " "
        'USB_out_buff = "5cur= N/A "
        GoSub Usb_out

        Param1 = 0     '12V Current
        GoSub Get_ADC
        tempfloat = tempfloat / Cur_adjust
        USB_out_buff = "12cur= " + Str$ (Dec3,tempfloat) + " "
        GoSub Usb_out

        Param1 = 4     'Drive Vin                'AC 2/18
        GoSub Get_ADC
        tempfloat = tempfloat / Volt_adjust_input
        USB_out_buff = "Drive Vin= " + Str$ (Dec3,tempfloat) + " "
        GoSub Usb_out

        GoSub Thermistor
        'USB_out_buff = "Temp= " + Str$ (Dec1 TEMPERATURE) +"C"     'AC 3/1
        USB_out_buff = "Temp= N/A"
        Call Usb_out

        Return

init_PWM:
         PR2 = $3f
         CCPR1L = $08
         CCPR2L = $08
         T2CON = %00000100
         TRISC.1 = 0
         TRISC.2 = 0
         CCP1CON =  %00001100       ' turn on PWM mode
         CCP2CON =  %00001111       ' turn on PWM mode
         control5 = 1
         control12 = 1
         Sync12 = 1
         Sync5 = 1
         Param1 = 121      'default ramp (121) = 1ms ramp time
         GoSub Ramp5
         GoSub Ramp12
         temp = 255-Ramp5up  'invert the ramp to get the right slope speed for down ramps
           CCP1CON.5=temp.1     'set up the 5 volt PWM
           CCP1CON.4=temp.0
           CCPR1L = temp /4
           temp = 255-Ramp12up     'invert the ramp to get the right slope speed for down ramps
           CCP2CON.5=temp.1     'set up the 12 volt PWM
           CCP2CON.4=temp.0
           CCPR2L = temp / 4
           control5 = 1
           control12 = 1


         Return

Stars:
      USB_out_buff = "Stars Rule, Rats Drool!"
      Call Usb_out
      USB_out_buff = 13 + "Tim Ferris"
      Call Usb_out
      USB_out_buff = 13 + "Alan Rice"
      Call Usb_out
      USB_out_buff = 13 + "Ryan Roth"
```

```
      Call Usb_out
      USB_out_buff = 13 + "Andrew Kennedy"
      Call Usb_out
      USB_out_buff = 13 + "Sammy Guzman"
      Call Usb_out
      USB_out_buff = 13 + "Mingying" + 13
      Call Usb_out
      Return

Thermistor:

' Perform a Stein-Hart calculation to linearise the thermistor into degrees Centigrade
'
    THERMISTOR_VALUE = 0
    tempLong = 0
    For temp = 1 To 100
        'THERMISTOR_VALUE =   THERMISTOR_VALUE + ADIn 7
        tempLong = tempLong + ADIn 7
        DelayUS 5
    Next

    TEMPERATURE = tempLong/100 *  QUANTA   'THERMISTOR_VALUE * QUANTA ' \
    TEMPF = 4096 - TEMPERATURE ' Find the resistance across the thermisto
    RESISTANCE = (TEMPERATURE * 10000) / TEMPF ' /
    LN = Log (RESISTANCE / 10000) ' \
    TEMPERATURE = B * LN + T25 ' / Linearise the result using the stein-hart algorithm
    TEMPERATURE = (1 / TEMPERATURE) - 273.15 ' Convert from Kelvin to Centigrade

    Return ' Then return from the subroutine

help:
    USB_out_buff = "List of Commands"
    Call Usb_out
    USB_out_buff = 13+ "CUR       Displays the current measurements."
    Call Usb_out
    USB_out_buff = 13+ "DATA      Displays voltage and current measurements."
    Call Usb_out
    USB_out_buff = 13+ "GETOVP5   Displays the OVP setting for 5V rail in mV."
    Call Usb_out
    USB_out_buff = 13+ "GETOVP12  Displays the OVP setting for 12V rail in mV."
    Call Usb_out
    USB_out_buff = 13+ "HELP      Provides help information for commands."
    Call Usb_out
    USB_out_buff = 13+ "OVP5      Sets the OVP value for 5V rail in mV."
    Call Usb_out
    USB_out_buff = 13+ "OVP12     Sets the OVP value for 12V rail in mV."
    Call Usb_out
    USB_out_buff = 13+ "POFF      Turns off both outputs."
    Call Usb_out
    USB_out_buff = 13+ "POFF5     Turns off the output of the 5V rail."
    Call Usb_out
    USB_out_buff = 13+ "POFF12    Turns off the output of the 12V rail."
    Call Usb_out
    USB_out_buff = 13+ "PON       Turns on both outputs."
    Call Usb_out
    USB_out_buff = 13+ "PON5      Turns on the output of the 5V rail."
    Call Usb_out
    USB_out_buff = 13+ "PON12     Turns on the output of the 12V rail."
    Call Usb_out
    USB_out_buff = 13+ "TEMP      Displays board temperature."
    Call Usb_out
    USB_out_buff = 13+ "Volts     Displays the voltage measurements."
    Call Usb_out
    USB_out_buff = 13+ "U3ID      Displays the Utopia3 firmware version."
    Call Usb_out
    USB_out_buff = 13+ "VSET5     Sets the voltage on the 5V rail in mV."
    Call Usb_out
    USB_out_buff = 13+ "VSET12    Sets the voltage on the 12V rail in mV." + 13
    Call Usb_out

Return

vset5:                 'AC 2/4      Convert from voltage to SDATA setting
        ' Param1 = input voltage [mV]
        ' Checks if input is greater than over voltage protection else
        ' Checks if input is below minimum voltage
        ' Otherwise calculates SDATA value and sends out data (up to 6V)

        If Param1 > OVP5 Then
            USB_out_buff = "Voltage too high: 5V regulator not changed"
            Call Usb_out
```

```
            USB_out_buff = 13 + "Use OVP5 to raise OVP setting" + 13
            Call Usb_out
            Return
        ElseIf Param1 < MINVOLTS Then         'AC 3/1  MINVOLTS = 3000
            tempword = MINVOLTS
            USB_out_buff = "Voltage too small: 5V regulator set to 3VDC" + 13
            Call Usb_out
        Else
            tempword = Param1
        EndIf
                             ' Convert Vout to SDATA: Sdata = (256/16000) * (Vout - 2000)
        tempword = tempword - 2000      ' Vout - 2000
        tempword = tempword * 2         ' Vout * (256/128)
        tempword = tempword / 125       ' Vout / (16000/128)
        If tempword > 64 Then tempword = 64         'Vout >= 6000 so set to 6000

        Prog_5V = 0 'set chip select low for 5V control
        SHOut Sdata,Sclk, 1, [tempword_low ]  'Bang out data
        Prog_5V = 1  'set chip select hi for 5V control

Return

vset12:                       'AC 2/4     Convert from voltage to SDATA setting
        ' Param1 = input voltage [mV]
        ' Checks if input is greater than over voltage protection else
        ' Checks if input is below minimum voltage
        ' Otherwise calculates SDATA value and sends out data (up to 18V)

        If Param1 > OVP12 Then
            USB_out_buff = "Voltage too high: 12V regulator not changed"
            Call Usb_out
            USB_out_buff = 13 + "Use OVP12 To raise OVP setting" + 13
            Call Usb_out
            Return
        ElseIf Param1 < MINVOLTS Then           'AC 3/1  MINVOLTS = 3000
            tempword = MINVOLTS
            USB_out_buff = "Voltage too small: 12V regulator set to 3VDC" + 13
            Call Usb_out
        Else
            tempword = Param1
        EndIf
                                  ' Convert Vout to SDATA: Sdata = (256/16000) * (Vout - 2000)
        tempword = tempword - 2000      ' Vout - 2000
        tempword = tempword * 2         ' Vout * (256/128)
        tempword = tempword / 125       ' Vout / (16000/128)
        If tempword > 255 Then tempword = 255    ' Vout is >= 18000 so set to 255

        Prog_12V = 0  'set chip select low for 5V control
        SHOut Sdata,Sclk, 1, [tempword_low ]  'Bang out data
        Prog_12V = 1  'set chip select hi for 5V control

Return

   'End new code for Utopia3
'*********************************************************************************************
'*********************************************************************************************

Ramp5:


      tempword = CRead ADDRESS + (Param1 *4)   'Read and display memory location ADDRESS + LOOP
      'USB_out_buff = Str$ (Dec Param1) + "  " + Str$ (Dec tempword)
      'GoSub Usb_out
             'Set up the resistor ladder and cap
       'Param1 = tempword
       Sync5 = 0  'set chip select low for 5V control
       SHOut Sdata,Sclk, 1, [tempword_low ]  'Bang out data
       Sync5 = 1  'set chip select hi for 5V control

       'USB_out_buff =  "SPI5 set to " + Str$ (Dec tempword_low) +13 'report back what we did
       'Call Usb_out
       CCP1CON.5=tempword_hi.1     'set up the 5 volt PWM
       CCP1CON.4=tempword_hi.0
       'Param1 = Param1 & 255        'trim to lower 8 bits
       CCPR1L = tempword_hi /4
       Ramp5up = tempword_hi
       'USB_out_buff = "PWM1 set to " + Str$ (Dec tempword_hi) +13 'report back what we did
       'Call Usb_out
       'USB_out_buff = Str$ (Dec Param1) + "  " + Str$ (Dec tempword)
      ' GoSub Usb_out
```

```
Return

turn_on:
                     'TURN ON
             CCP1CON.5=Ramp5up.1     'set up the 5 volt PWM
             CCP1CON.4=Ramp5up.0
             CCPR1L = Ramp5up /4
             CCP2CON.5=Ramp12up.1     'set up the 12 volt PWM
             CCP2CON.4=Ramp12up.0
             CCPR2L = Ramp12up / 4
             control5 = 0
             control12 = 0
Return

turn_off:
             temp = 255-Ramp5up  'invert the ramp to get the right slope speed for down ramps
             CCP1CON.5=temp.1     'set up the 5 volt PWM
             CCP1CON.4=temp.0
             CCPR1L = temp /4
             temp = 255-Ramp12up     'invert the ramp to get the right slope speed for down ramps
             CCP2CON.5=temp.1     'set up the 12 volt PWM
             CCP2CON.4=temp.0
             CCPR2L = temp / 4
             control5 = 1
             control12 = 1
Return

Ramp12:
      tempword= CRead ADDRESS + (Param1 *4)+2   ' Read and display memory location ADDRESS + LOOP
      'USB_out_buff = Str$ (Dec Param1) + "  " + Str$ (Dec tempword)
      'GoSub Usb_out
      'Param1 = tempword

       Sync12 = 0 'set chip select low for 12V control
       SHOut Sdata,Sclk, 1, [tempword_low ]   'Bang out data
       Sync12 = 1  'set chip select hi for 12V control

       'USB_out_buff = "SPI12 set to " + Str$ (Dec tempword_low) +13 'report back what we did
       'Call Usb_out

       'Set up the PWMs
       CCP2CON.5=tempword_hi.1
       CCP2CON.4=tempword_hi.0
       CCPR2L = tempword_hi / 4
       Ramp12up = tempword_hi
       'USB_out_buff = "PWM2 set to " + Str$ (Dec tempword_hi) +13
      ' Call Usb_out
       'USB_out_buff = Str$ (Dec Param1) + "  " + Str$ (Dec tempword)
       'GoSub Usb_out
Return

Ramp_values:
         tempword= CRead ADDRESS + (Param1 *2)  ' Read and display memory location ADDRESS + LOOP
         USB_out_buff = Str$ (Dec Param1) + "  " + Str$ (Dec tempword)
         GoSub Usb_out

         Return
 Stop
 ' Reserve 10 spaces in FLASH memory
ADDRESS:
CData As Word  8194   ,      8194   '      Ramp   0      0.000001
CData As Word  12864  ,      8194   '      Ramp   1      0.000002
CData As Word  49472  ,      8194   '      Ramp   2      0.000003
CData As Word  56896  ,      8194   '      Ramp   3      0.000004
CData As Word  6784   ,      8194   '      Ramp   4      0.000005
CData As Word  19328  ,      8194   '      Ramp   5      0.000006
CData As Word  28800  ,      8194   '      Ramp   6      0.000007
CData As Word  34688  ,      8194   '      Ramp   7      0.000008
CData As Word  37760  ,      8194   '      Ramp   8      0.000009
CData As Word  4355   ,      4355   '      Ramp   9      0.000010
CData As Word  8195   ,      8195   '      Ramp   10     0.000011
CData As Word  12547  ,      12547  '      Ramp   11     0.000012
CData As Word  16387  ,      16387  '      Ramp   12     0.000013
CData As Word  20227  ,      20227  '      Ramp   13     0.000014
CData As Word  23299  ,      23299  '      Ramp   14     0.000015
CData As Word  25347  ,      25347  '      Ramp   15     0.000016
CData As Word  27907  ,      27907  '      Ramp   16     0.000017
CData As Word  30211  ,      30211  '      Ramp   17     0.000018
CData As Word  31747  ,      31747  '      Ramp   18     0.000019
CData As Word  33539  ,      33539  '      Ramp   19     0.000020
CData As Word  34819  ,      34819  '      Ramp   20     0.000021
```

```
CData As Word  36099   ,       36099   '       Ramp    21      0.000022
CData As Word  4357    ,       4357    '       Ramp    22      0.000023
CData As Word  7685    ,       7685    '       Ramp    23      0.000024
CData As Word  10757   ,       10757   '       Ramp    24      0.000025
CData As Word  13829   ,       13829   '       Ramp    25      0.000026
CData As Word  16645   ,       16645   '       Ramp    26      0.000027
CData As Word  18949   ,       18949   '       Ramp    27      0.000028
CData As Word  20997   ,       20997   '       Ramp    28      0.000029
CData As Word  23301   ,       23301   '       Ramp    29      0.000030
CData As Word  25349   ,       25349   '       Ramp    30      0.000031
CData As Word  27397   ,       27397   '       Ramp    31      0.000032
CData As Word  28677   ,       28677   '       Ramp    32      0.000033
CData As Word  29957   ,       29957   '       Ramp    33      0.000034
CData As Word  31237   ,       31237   '       Ramp    34      0.000035
CData As Word  32517   ,       32517   '       Ramp    35      0.000036
CData As Word  33797   ,       33797   '       Ramp    36      0.000037
CData As Word  35077   ,       35077   '       Ramp    37      0.000038
CData As Word  36101   ,       36101   '       Ramp    38      0.000039
CData As Word  37125   ,       37125   '       Ramp    39      0.000040
CData As Word  38149   ,       38149   '       Ramp    40      0.000041
CData As Word  38917   ,       38917   '       Ramp    41      0.000042
CData As Word  39685   ,       39685   '       Ramp    42      0.000043
CData As Word  40197   ,       40197   '       Ramp    43      0.000044
CData As Word  40965   ,       40965   '       Ramp    44      0.000045
CData As Word  41477   ,       41477   '       Ramp    45      0.000046
CData As Word  42245   ,       42245   '       Ramp    46      0.000047
CData As Word  42757   ,       42757   '       Ramp    47      0.000048
CData As Word  43525   ,       43525   '       Ramp    48      0.000049
CData As Word  44037   ,       44037   '       Ramp    49      0.000050
CData As Word  44549   ,       44549   '       Ramp    50      0.000051
CData As Word  45061   ,       45061   '       Ramp    51      0.000052
CData As Word  45573   ,       45573   '       Ramp    52      0.000053
CData As Word  46085   ,       46085   '       Ramp    53      0.000054
CData As Word  46597   ,       46597   '       Ramp    54      0.000055
CData As Word  46853   ,       46853   '       Ramp    55      0.000056
CData As Word  47365   ,       47365   '       Ramp    56      0.000057
CData As Word  47621   ,       47621   '       Ramp    57      0.000058
CData As Word  48133   ,       48133   '       Ramp    58      0.000059
CData As Word  48389   ,       48389   '       Ramp    59      0.000060
CData As Word  48901   ,       48901   '       Ramp    60      0.000061
CData As Word  49157   ,       49157   '       Ramp    61      0.000062
CData As Word  49669   ,       49669   '       Ramp    62      0.000063
CData As Word  49925   ,       49925   '       Ramp    63      0.000064
CData As Word  50437   ,       50437   '       Ramp    64      0.000065
CData As Word  50693   ,       50693   '       Ramp    65      0.000066
CData As Word  51205   ,       51205   '       Ramp    66      0.000067
CData As Word  51461   ,       51461   '       Ramp    67      0.000068
CData As Word  51973   ,       51973   '       Ramp    68      0.000069
CData As Word  52485   ,       52485   '       Ramp    69      0.000072
CData As Word  2569    ,       2569    '       Ramp    70      0.000076
CData As Word  6665    ,       6665    '       Ramp    71      0.000080
CData As Word  10761   ,       10761   '       Ramp    72      0.000084
CData As Word  14601   ,       14601   '       Ramp    73      0.000088
CData As Word  18441   ,       18441   '       Ramp    74      0.000093
CData As Word  22025   ,       22025   '       Ramp    75      0.000098
CData As Word  25097   ,       25097   '       Ramp    76      0.000103
CData As Word  27913   ,       27913   '       Ramp    77      0.000108
CData As Word  30473   ,       30473   '       Ramp    78      0.000114
CData As Word  32521   ,       32521   '       Ramp    79      0.000120
CData As Word  34313   ,       34313   '       Ramp    80      0.000126
CData As Word  36105   ,       36105   '       Ramp    81      0.000133
CData As Word  37897   ,       37897   '       Ramp    82      0.000140
CData As Word  39689   ,       39689   '       Ramp    83      0.000147
CData As Word  41481   ,       41481   '       Ramp    84      0.000155
CData As Word  43273   ,       43273   '       Ramp    85      0.000163
CData As Word  45065   ,       45065   '       Ramp    86      0.000172
CData As Word  46089   ,       46089   '       Ramp    87      0.000181
CData As Word  47113   ,       47113   '       Ramp    88      0.000190
CData As Word  47881   ,       47881   '       Ramp    89      0.000201
CData As Word  48905   ,       48905   '       Ramp    90      0.000211
CData As Word  49929   ,       49929   '       Ramp    91      0.000222
CData As Word  50697   ,       50697   '       Ramp    92      0.000234
CData As Word  51465   ,       51465   '       Ramp    93      0.000246
CData As Word  52233   ,       52233   '       Ramp    94      0.000259
CData As Word  53001   ,       53001   '       Ramp    95      0.000273
CData As Word  53769   ,       53769   '       Ramp    96      0.000287
CData As Word  54537   ,       54537   '       Ramp    97      0.000302
CData As Word  55305   ,       55305   '       Ramp    98      0.000318
CData As Word  55817   ,       55817   '       Ramp    99      0.000335
CData As Word  56329   ,       56329   '       Ramp    100     0.000352
CData As Word  56841   ,       56841   '       Ramp    101     0.000371
```

```
CData As Word  57353   ,       57353   '       Ramp    102     0.000391
CData As Word  57865   ,       57865   '       Ramp    103     0.000411
CData As Word  58377   ,       58377   '       Ramp    104     0.000433
CData As Word  58889   ,       58889   '       Ramp    105     0.000456
CData As Word  2577    ,       2577    '       Ramp    106     0.000480
CData As Word  6673    ,       6673    '       Ramp    107     0.000505
CData As Word  9745    ,       9745    '       Ramp    108     0.000531
CData As Word  12817   ,       12817   '       Ramp    109     0.000559
CData As Word  15633   ,       15633   '       Ramp    110     0.000589
CData As Word  18449   ,       18449   '       Ramp    111     0.000620
CData As Word  21009   ,       21009   '       Ramp    112     0.000652
CData As Word  23313   ,       23313   '       Ramp    113     0.000687
CData As Word  25617   ,       25617   '       Ramp    114     0.000723
CData As Word  27665   ,       27665   '       Ramp    115     0.000761
CData As Word  29713   ,       29713   '       Ramp    116     0.000801
CData As Word  31505   ,       31505   '       Ramp    117     0.000843
CData As Word  33297   ,       33297   '       Ramp    118     0.000887
CData As Word  35089   ,       35089   '       Ramp    119     0.000934
CData As Word  36625   ,       36625   '       Ramp    120     0.000983
CData As Word  38161   ,       38161   '       Ramp    121     0.001035
CData As Word  39697   ,       39697   '       Ramp    122     0.001090
CData As Word  40977   ,       40977   '       Ramp    123     0.001147
CData As Word  42257   ,       42257   '       Ramp    124     0.001207
CData As Word  43537   ,       43537   '       Ramp    125     0.001271
CData As Word  44561   ,       44561   '       Ramp    126     0.001338
CData As Word  45585   ,       45585   '       Ramp    127     0.001408
CData As Word  46865   ,       46865   '       Ramp    128     0.001482
CData As Word  47889   ,       47889   '       Ramp    129     0.001560
CData As Word  48657   ,       48657   '       Ramp    130     0.001642
CData As Word  49681   ,       49681   '       Ramp    131     0.001729
CData As Word  50449   ,       50449   '       Ramp    132     0.001820
CData As Word  51473   ,       51473   '       Ramp    133     0.001915
CData As Word  52241   ,       52241   '       Ramp    134     0.002016
CData As Word  53009   ,       53009   '       Ramp    135     0.002122
CData As Word  53521   ,       53521   '       Ramp    136     0.002234
CData As Word  54289   ,       54289   '       Ramp    137     0.002352
CData As Word  54801   ,       54801   '       Ramp    138     0.002475
CData As Word  55313   ,       55313   '       Ramp    139     0.002606
CData As Word  56081   ,       56081   '       Ramp    140     0.002743
CData As Word  56593   ,       56593   '       Ramp    141     0.002887
CData As Word  57105   ,       57105   '       Ramp    142     0.003039
CData As Word  57617   ,       57617   '       Ramp    143     0.003199
CData As Word  58129   ,       58129   '       Ramp    144     0.003368
CData As Word  58641   ,       58641   '       Ramp    145     0.003545
CData As Word  59153   ,       59153   '       Ramp    146     0.003731
CData As Word  59409   ,       59409   '       Ramp    147     0.003928
CData As Word  59921   ,       59921   '       Ramp    148     0.004135
CData As Word  60177   ,       60177   '       Ramp    149     0.004352
CData As Word  60433   ,       60433   '       Ramp    150     0.004581
CData As Word  2593    ,       2593    '       Ramp    151     0.004822
CData As Word  6689    ,       6689    '       Ramp    152     0.005076
CData As Word  9761    ,       9761    '       Ramp    153     0.005343
CData As Word  12833   ,       12833   '       Ramp    154     0.005625
CData As Word  15649   ,       15649   '       Ramp    155     0.005921
CData As Word  18465   ,       18465   '       Ramp    156     0.006232
CData As Word  21025   ,       21025   '       Ramp    157     0.006560
CData As Word  23329   ,       23329   '       Ramp    158     0.006905
CData As Word  25633   ,       25633   '       Ramp    159     0.007269
CData As Word  27681   ,       27681   '       Ramp    160     0.007651
CData As Word  29729   ,       29729   '       Ramp    161     0.008054
CData As Word  29729   ,       29729   '       Ramp    162     0.008478
CData As Word  29729   ,       29729   '       Ramp    163     0.008924
CData As Word  35105   ,       35105   '       Ramp    164     0.009394
CData As Word  35105   ,       35105   '       Ramp    165     0.009888
CData As Word  35105   ,       35105   '       Ramp    166     0.010409
CData As Word  39713   ,       39713   '       Ramp    167     0.010957
CData As Word  39713   ,       39713   '       Ramp    168     0.011533
CData As Word  39713   ,       39713   '       Ramp    169     0.012140
CData As Word  43553   ,       43553   '       Ramp    170     0.012779
CData As Word  44577   ,       44577   '       Ramp    171     0.013452
CData As Word  45601   ,       45601   '       Ramp    172     0.014160
CData As Word  46881   ,       46881   '       Ramp    173     0.014905
CData As Word  47905   ,       47905   '       Ramp    174     0.015690
CData As Word  48673   ,       48673   '       Ramp    175     0.016515
CData As Word  49697   ,       49697   '       Ramp    176     0.017385
CData As Word  50465   ,       50465   '       Ramp    177     0.018300
CData As Word  51489   ,       51489   '       Ramp    178     0.019263
CData As Word  52257   ,       52257   '       Ramp    179     0.020277
CData As Word  53025   ,       53025   '       Ramp    180     0.021344
CData As Word  53537   ,       53537   '       Ramp    181     0.022467
CData As Word  54305   ,       54305   '       Ramp    182     0.023650
```

```
CData As Word  54817   ,      54817  '      Ramp   183    0.024894
CData As Word  55329   ,      55329  '      Ramp   184    0.026205
CData As Word  56097   ,      56097  '      Ramp   185    0.027584
CData As Word  56609   ,      56609  '      Ramp   186    0.029035
CData As Word  57121   ,      57121  '      Ramp   187    0.030564
CData As Word  57633   ,      57633  '      Ramp   188    0.032172
CData As Word  58145   ,      58145  '      Ramp   189    0.033866
CData As Word  58657   ,      58657  '      Ramp   190    0.035648
CData As Word  59169   ,      59169  '      Ramp   191    0.037524
CData As Word  59425   ,      59425  '      Ramp   192    0.039499
CData As Word  59937   ,      59937  '      Ramp   193    0.041578
CData As Word  60193   ,      60193  '      Ramp   194    0.043766
CData As Word  60449   ,      60449  '      Ramp   195    0.046070
CData As Word  4417    ,      4417   '      Ramp   196    0.048495
CData As Word  7489    ,      7489   '      Ramp   197    0.051047
CData As Word  10561   ,      10561  '      Ramp   198    0.053734
CData As Word  13633   ,      13633  '      Ramp   199    0.056562
CData As Word  16449   ,      16449  '      Ramp   200    0.059539
CData As Word  19009   ,      19009  '      Ramp   201    0.062672
CData As Word  21569   ,      21569  '      Ramp   202    0.065971
CData As Word  24129   ,      24129  '      Ramp   203    0.069443
CData As Word  26433   ,      26433  '      Ramp   204    0.073098
CData As Word  28481   ,      28481  '      Ramp   205    0.076945
CData As Word  30529   ,      30529  '      Ramp   206    0.080995
CData As Word  32321   ,      32321  '      Ramp   207    0.085258
CData As Word  34113   ,      34113  '      Ramp   208    0.089745
CData As Word  35905   ,      35905  '      Ramp   209    0.094468
CData As Word  37441   ,      37441  '      Ramp   210    0.099440
CData As Word  38721   ,      38721  '      Ramp   211    0.104674
CData As Word  40257   ,      40257  '      Ramp   212    0.110183
CData As Word  41537   ,      41537  '      Ramp   213    0.115982
CData As Word  42817   ,      42817  '      Ramp   214    0.122087
CData As Word  44097   ,      44097  '      Ramp   215    0.128512
CData As Word  45377   ,      45377  '      Ramp   216    0.135276
CData As Word  46401   ,      46401  '      Ramp   217    0.142396
CData As Word  47425   ,      47425  '      Ramp   218    0.149890
CData As Word  48449   ,      48449  '      Ramp   219    0.157779
CData As Word  49473   ,      49473  '      Ramp   220    0.166083
CData As Word  50241   ,      50241  '      Ramp   221    0.174825
CData As Word  51009   ,      51009  '      Ramp   222    0.184026
CData As Word  51777   ,      51777  '      Ramp   223    0.193711
CData As Word  52545   ,      52545  '      Ramp   224    0.203907
CData As Word  53313   ,      53313  '      Ramp   225    0.214639
CData As Word  54081   ,      54081  '      Ramp   226    0.225936
CData As Word  54849   ,      54849  '      Ramp   227    0.237827
CData As Word  55361   ,      55361  '      Ramp   228    0.250344
CData As Word  55873   ,      55873  '      Ramp   229    0.263520
CData As Word  56385   ,      56385  '      Ramp   230    0.277390
CData As Word  56897   ,      56897  '      Ramp   231    0.291989
CData As Word  57409   ,      57409  '      Ramp   232    0.307357
CData As Word  57921   ,      57921  '      Ramp   233    0.323534
CData As Word  58433   ,      58433  '      Ramp   234    0.340562
CData As Word  58945   ,      58945  '      Ramp   235    0.358486
CData As Word  59457   ,      59457  '      Ramp   236    0.377354
CData As Word  59713   ,      59713  '      Ramp   237    0.397214
CData As Word  60225   ,      60225  '      Ramp   238    0.418120
CData As Word  60481   ,      60481  '      Ramp   239    0.440127
CData As Word  2689    ,      2689   '      Ramp   240    0.463291
CData As Word  5505    ,      5505   '      Ramp   241    0.487675
CData As Word  8833    ,      8833   '      Ramp   242    0.513342
CData As Word  11905   ,      11905  '      Ramp   243    0.540360
CData As Word  14721   ,      14721  '      Ramp   244    0.568800
CData As Word  17537   ,      17537  '      Ramp   245    0.598737
CData As Word  20353   ,      20353  '      Ramp   246    0.630249
CData As Word  22657   ,      22657  '      Ramp   247    0.663420
CData As Word  24961   ,      24961  '      Ramp   248    0.698337
CData As Word  27265   ,      27265  '      Ramp   249    0.735092
CData As Word  29313   ,      29313  '      Ramp   250    0.773781
CData As Word  31361   ,      31361  '      Ramp   251    0.814506
CData As Word  33409   ,      33409  '      Ramp   252    0.857375
CData As Word  34689   ,      34689  '      Ramp   253    0.902500
CData As Word  36225   ,      36225  '      Ramp   254    0.950000
CData As Word  37761   ,      37761  '      Ramp   255    1.000000
```

APPENDIX E: Analysis of Senior Project

Project Title: UTOPIA 3.0: Power Cycling Controller Board
Quarter / Year Submitted: Winter Quarter/2010
Student: Allan Chan
Advisor: John Oliver

Summary of Functional Requirements

The UTOPIA 3.0 device is designed to provide an interface to communicate with a hard drive and perform power cycling functions on a hard drive. It is used in hard drive development at Western Digital. The board provides a low level communication protocol called Serial Input/Output (SIO) that can control and access information on the hard drive. Also, the device has the ability to regulate the power supply to the hard drive. Specifically, it is possible to switch on and off the power, control the voltage ramping or rate at which the power is applied, and provide voltage margining. The device is interfaced to a lab bench PC via USB. An external power supply is required to provide power to the hard drive under test.

Primary Constraints

Since the UTOPIA 3.0 device is an upgrade to an existing device, it was required that the features of the previous generation be retained, namely the ramping and switching function of the power outputs. This also included retaining the external interfaces of device which are USB port, 4-pin power input header, 4-pin power output header, ICSP header, and SIO port.

The size of the board and components was also a concern because these devices have application in thermal chambers in which space is a premium. The design of the PCB had to be as small as possible so it was intended to make the PCB sized just big enough to fit the components.

Economics

The original cost of the components were estimated to be somewhere around $500. The most expensive part is fabricating the board because it is custom made. Board components are relatively cheap and cost somewhere within the range of almost $0 to $5.00 per component. Note that this product is intended to be mass produced in large quantities and the cost for the components drops considerably when they are purchased in bulk.

The final cost of the component parts were approximately $300. The bill of materials specify the details of how the cost is broken down and can be found in Appendix C.

Additional equipment cost

The development of the firmware for the UTOPIA 3.0 device required software called Proton Development Suite for writing the code. The cost of the software was $199. A PICkit 2 programmer was also needed for flashing the microcontroller with new programs. It costs approximately $30. Other items were borrowed from Western Digital such as an oscilloscope and a DC power supply which did not have any monetary costs.

Original estimated development time

Development time was expected to require approximately five months. This amount of time takes into consideration the time that is needed to design the hardware, design the PCB, fabricate the PCB, and test functionality of the prototype.

Actual development time

The actual development time took about 3 ½ months total. It was broken down into 6 weeks for hardware design, 4 weeks for PCB design, and 4 weeks for firmware development and testing. That is nearly two months ahead of schedule which was largely due to the expertise of the engineer in charge of the project. When I had problems or questions, it was usually resolved fairly quickly because of his knowledge and familiarity with the UTOPIA 3.0 components. However, the total development time spanned across five months due to some downtime waiting for the boards to be made.

If manufactured on a commercial basis:

Estimated number of devices sold per year: 500-1500

Estimated manufacturing cost for each device: $5.00 in lots of 100 for PCB

Estimated purchase price for each device: N/A

Estimated profit per year: N/A

Estimated cost for user to operate the device, per unit time: 1 day to learn commands

Environmental

During the manufacturing process, there are environmental impacts that are indirectly associated with building a circuit board. Specifically, a large amount of electrical energy is required to operate the machinery that fabricates the board. In addition, the delivery of the parts also uses lots of fuel due to the use of trains, airplanes, or trucks that get the parts from the manufacturers to the consumer. With such a low number of samples created for this prototype, the cost quickly overcomes the worth of making the board. After the boards are assembled, each board is cleaned thoroughly using spray cleaners which give off unwanted toxic chemicals.

The device itself uses relatively low power. It is powered by a USB port from a computed with a rated voltage of 5 VDC and 1 A current draw. Therefore, power consumption is kept low and prevents waste of electricity.

Manufacturability

All of the components on the board are surface mount parts which are very small. I did not have any experience soldering parts that are so small especially without access to a stereo microscope. Without the proper equipment, it would have taken me a long time to get the hang of it and completely populate the entire board which has nearly 200 components.

Sustainability

The hope is that the components on the UTOPIA 3.0 device last a very long time and avoids having to replace entire boards. As long as the device provides its function, there will be no need to upgrade or implement a new design. In the long run, the benefit is that the device does not become electronic waste or e-waste. Once the components of the device have been assembled, it becomes a very time consuming and expensive task to reuse the parts because all the components are surface mounted. Thus, the device is most likely trashed or tossed away. It can be prevented by properly designing the product such as making sure that all components perform within their rated specifications.

When parts are used inappropriately, they tend to have shorter life due to the stress and strains that they weren't designed for.

Health and Safety

There are no safety concerns that are known at this time. Since the device is an electrical part, there is always the risk of components possibly burning or catching fire through improper use. One example is setting voltages higher than a part is rated to handle. All of the parts on UTOPIA 3.0 have been verified to meet the maximum voltage requirements specified in the Requirements section.

Development

During this project, I had to learn many new things. The development software that was used such ExpressPCB for designing the PCB and PROTON Basic for firmware development were new software that I learned how to use during the course of the project. They were not difficult to use, but just took some time to get used to the GUI and features like any other piece of new software. Prior to this project, I had never worked with PIC microcontrollers. Now that I have skills working with PICs, I realize that there are endless possibilities with the use of microcontrollers. They can pretty much be programmed to do anything. It is great having skills in PCB design because now I can custom fabricate a PCB within weeks. Tools such these can become very useful in the future.