Voltage Control of FPGA and CPLD Devices

Undergraduate Student: Corey Engelken

Faculty Advisor: Dr. Joanne Degroat

Department of Electrical and Computer Engineering

The Ohio State University

April 5th, 2013

Abstract:

Power consumption of components is of high importance to circuit designers. Lowering the power consumption of the components in the circuit can help the circuit be more energy efficient and lower the heat dissipated to the rest of the circuit.

FPGAs and CPLDs are devices found in circuits. Depending on the digital circuit programmed onto them, the recommended supply voltage may be too large. The devices may be able to use a lower supply voltage while maintaining a high degree of accuracy. If this is the case, then an opportunity exists to save power and become more energy efficient while simultaneously decreasing the heat dissipated to the rest of the circuit.

This paper aims to answer that question. The initial approach was to isolate the device and test it directly. This method had problems however, as we were not able to program the devices purchased with the materials at hand. Another approach was tried using an Altera DE2 Board which contains a Cyclone II FPGA. However, no change could be seen in the current consumed by the FPGA for various circuits because the DE2 consumed much more current than the FPGA and more precise measuring instruments would be needed.

To fully test this idea, the FPGA or CPLD would need to be isolated. Even though our attempts were unsuccessful, a large amount of design work was put into the circuit and is included in this paper.

Table of Contents:

List of Tables and Figures:

Introduction:

Power minimization is a high priority for circuit designers. Energy efficiency and heat dissipation are two large issues for them today. In both cases, minimizing the amount of power consumed can solve both of these issues. One way of lowering the power is by lowering the voltage being supplied to the components of the design.

This paper is focused on lowering the voltages being supplied to Field Programmable Gate Arrays (FPGAs) and Complex Programmable Logic Devices (CPLDs). FPGAs and CPLDs differ in design and implementation but behave similarly and will collectively be referred to as ‘devices’ in this paper. Both of these are programmable devices that implement digital logic circuits designed in software. Circuits designed in software that will be programmed onto the devices will be referred to as ‘digital circuits.’

Both of these devices are CMOS logic devices. As CMOS devices, they only consume power when switching from one state to another (i.e. from a logic high to a logic low). The amount of power consumed is proportional to the frequency of the logic switch times the voltage squared. In high frequency circuits, even a small decrease in the voltage can make a large impact on the power consumed.

As stated earlier, a lower power consumption leads to more energy efficient devices and less heat dissipation to the surrounding area. In today’s energy conscious world, more energy efficient devices are highly desired. Also, a large problem for hardware designers is dealing with the large amount of heat produced by running devices. Lower power consumption means less heat dissipation for the designers to consider.

Both of these are huge advantages for low power devices. As such, it is worth investigating the supply voltage given to modern-day devices. The hypothesis to be shown in this paper is if a lower voltage to supply to the devices can be found without introducing new errors then an opportunity exists for lower power consumption.

This paper will begin with the initial approach to test this hypothesis. Next, an alternative test using Altera’s DE2 Development Board will be discussed and the results given. Then, the initial approach from earlier will be discussed in great detail. Lastly, the paper will conclude and future ideas and directions will be given.

Initial Approach:

The most direct approach to test this hypothesis is to isolate the device and measure the current through it under normal operating conditions. Following that, a lower voltage can be found by incrementally stepping down the voltage to the device, checking for errors and measuring the current through it. At some point, errors will start occurring and the increment above this increment will be the new supply voltage. This approach would successfully find both measurements needed to prove the hypothesis; the lower voltage step and the current through the device.

To do this, a circuit will be built around the device. Building such a circuit is challenging however since these devices are primarily surface mount devices and have no easy, nor cheap way of accessing each pin individually. That said, a few methods were tried. The first was to purchase a socket on a budget. Some sockets were bought and a device was put into them. All of the devices were to be programmed via a JTAG interface. This interface has designated pins that can communicate with some programming utilities through a USB port on a PC (for the Altera devices, these are a USB blaster programming hardware and Quartus software). These devices also include an ‘auto-detect’ feature that allows the software to figure out what device it is communicating with. However, when these devices were put in the sockets and the programming utilities connected, the devices could not be detected or programmed by the PC. A few different devices were tried (including MAX 7000S devices and FLEX 10K FPGAs), all via the JTAG interface. Multiple sockets were used as well but with the same results.

One aspect to note is that the only socket type tried were PLCC. Most other sockets would be too small to even attempt working with by hand and are much more expensive than PLCC sockets. Future work will include trying different socket types and finding different sockets

The other aspect to note about the sockets tried is that they do not have any mechanism to hold the devices into place. The device simply ‘sits’ inside the socket (no snapping into the socket nor anything else to hold it tightly and firmly into place). This is most likely the cause of the programming and detection issues. The device sitting in the socket, but not being held into place, most likely leads to faulty electrical connections and instability. However, better sockets are much more expensive and out of our current price range. With the outlook bleak for programming the devices without spending large amounts of money, a new idea for testing this hypothesis was realized using the Altera DE2 Development Board.

Second Approach: Using the Altera DE2 Development Board:

Since programming the devices was unsuccessful, a new approach to test the hypothesis would need to be engineered. This new approach was to use a DE2 Development Board (pictured below). The DE2 board contains an Altera Cyclone II FPGA chip. The idea is to measure the current through the DE2 board for various circuits and see if any conclusions can be drawn about the current consumed by the FPGA.

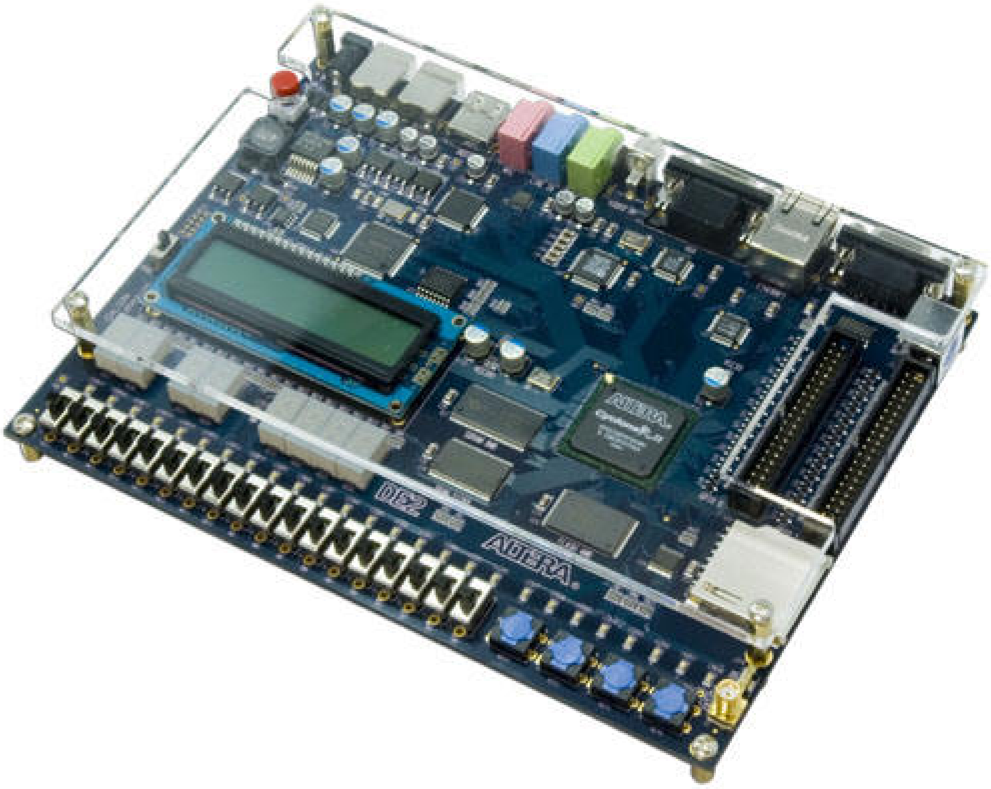

Figure 1: Altera DE2 Development Board
(http://www.altera.com/education/univ/materials/boards/de2/unv-de2-board.html)

As can be seen in the picture, the DE2 board contains much, much more than just an FPGA chip. A means to differentiate between the current consumed by the device and the current consumed by the DE2 board;s peripherals is to program multiple, increasingly complex, high frequency digital circuits onto the FPGA and measure the current through the board each time. Two types of digital circuits would be used: ring oscillators and counters. The ring oscillators would become more and more complex with each circuit and would all output to just one LED on the DE2 board. If some change (should be an increase) of current through the board is found for each increasingly complex circuit, then it can be shown that the device does indeed consume a different amount of current as the circuit programmed onto the device changes even if no extra peripherals are used.

The counters would also become increasingly complex and would output to an increasingly larger amount of LEDs. The counters would show how much current is being used by the LEDs (peripherals) of the DE2 board. It is possible that the peripherals consume much more current than the FPGA and as such any change in the current through the DE2 board is due to a change in the peripherals used by the FPGA and not the FPGA itself.

This leads into another problem with this approach: the measurement tools may not be accurate enough to detect any change in the FPGA because the DE2 board consumes so much more current than the device. In this case, very accurate and sensitive measurement utilities would be required. If measurement tools that are not precise enough are used, no change in current would be seen regardless of whether or not a change actually occurred.

To test this approach, various ring oscillators and counters were programmed on the FPGA and the current was measured through the DE2 board using a Fluke 115 multimeter. This multimeter can measure up to three decimal places. Depending though, on the units measured from the DE2 board, this may not be accurate enough. Each digital circuit was programmed onto the FPGA ten times. The power to the DE2 board was reset after each trial. The measurements were taken after the current had settled for at least ten seconds. Some digital circuits that were programmed on the FPGA are shown below.

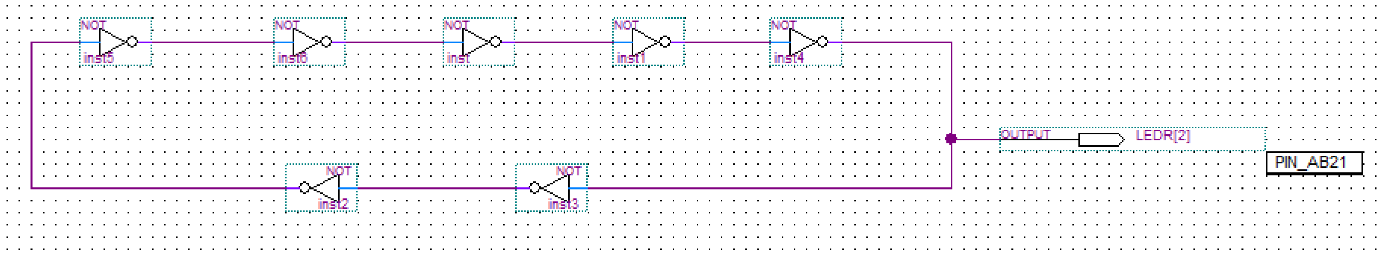

Figure 2: Seven-Ring Oscillator Digital Circuit

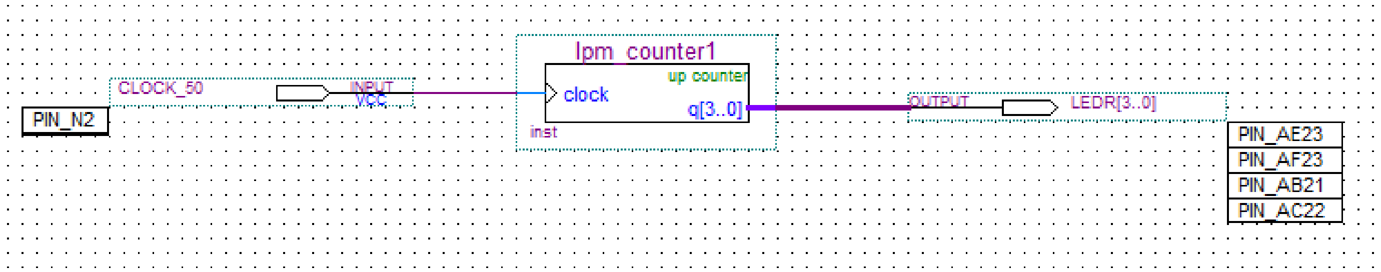

Figure 3: Four-Bit Counter Digital Circuit

The results for the various ring oscillators are shown below.

Trial	Circuit	Current (A)
1	3-Ring Oscillator	0.449
2	3-Ring Oscillator	0.420
3	3-Ring Oscillator	0.450
4	3-Ring Oscillator	0.420
5	3-Ring Oscillator	0.419
6	3-Ring Oscillator	0.422
7	3-Ring Oscillator	0.421

8	3-Ring Oscillator	0.420
9	3-Ring Oscillator	0.419
10	3-Ring Oscillator	0.419

Above: Table 1: Results for the 3-Ring Oscillator

Trial	Circuit	Current (A)
1	5-Ring Oscillator	0.418
2	5-Ring Oscillator	0.449
3	5-Ring Oscillator	0.418
4	5-Ring Oscillator	0.447
5	5-Ring Oscillator	0.447
6	5-Ring Oscillator	0.418
7	5-Ring Oscillator	0.418
8	5-Ring Oscillator	0.418
9	5-Ring Oscillator	0.418
10	5-Ring Oscillator	0.418

Above: Table 2: Results for the 5-Ring Oscillator

Trial	Circuit	Current (A)
1	7-Ring Oscillator	0.419
2	7-Ring Oscillator	0.449
3	7-Ring Oscillator	0.418
4	7-Ring Oscillator	0.418
5	7-Ring Oscillator	0.418
6	7-Ring Oscillator	0.446
7	7-Ring Oscillator	0.418
8	7-Ring Oscillator	0.419
9	7-Ring Oscillator	0.418
10	7-Ring Oscillator	0.419

Above: Table 3: Results for the 7-Ring Oscillator

Trial	Circuit	Current (A)
1	11-Ring Oscillator	0.420
2	11-Ring Oscillator	0.450
3	11-Ring Oscillator	0.419
4	11-Ring Oscillator	0.448
5	11-Ring Oscillator	0.419
6	11-Ring Oscillator	0.448
7	11-Ring Oscillator	0.449
8	11-Ring Oscillator	0.420
9	11-Ring Oscillator	0.420
10	11-Ring Oscillator	0.448

Above: Table 4: Results for the 11-Ring Oscillator

Trial	Circuit	Current (A)
1	15-Ring Oscillator	0.448
2	15-Ring Oscillator	0.419
3	15-Ring Oscillator	0.449
4	15-Ring Oscillator	0.419
5	15-Ring Oscillator	0.419
6	15-Ring Oscillator	0.449
7	15-Ring Oscillator	0.450
8	15-Ring Oscillator	0.420
9	15-Ring Oscillator	0.420
10	15-Ring Oscillator	0.419

Above: Table 5: Results for the 15-Ring Oscillator

The following are the results from the various counters.

Trial	Circuit	Current (A)
1	2-Bit Counter	0.421
2	2-Bit Counter	0.421

3	2-Bit Counter	0.450
4	2-Bit Counter	0.420
5	2-Bit Counter	0.450
6	2-Bit Counter	0.420
7	2-Bit Counter	0.451
8	2-Bit Counter	0.420
9	2-Bit Counter	0.449
10	2-Bit Counter	0.450

Above: Table 6: Results for the 2-Bit Counters.

Trial	Circuit	Current (A)
1	4-Bit Counter	0.422
2	4-Bit Counter	0.422
3	4-Bit Counter	0.451
4	4-Bit Counter	0.451
5	4-Bit Counter	0.422
6	4-Bit Counter	0.422
7	4-Bit Counter	0.450
8	4-Bit Counter	0.450
9	4-Bit Counter	0.450
10	4-Bit Counter	0.422

Above: Table 7: Results for the 4-Bit Counters.

Trial	Circuit	Current (A)
1	8-Bit Counter	0.425
2	8-Bit Counter	0.425
3	8-Bit Counter	0.427
4	8-Bit Counter	0.425
5	8-Bit Counter	0.428
6	8-Bit Counter	0.454

7	8-Bit Counter	0.426
8	8-Bit Counter	0.426
9	8-Bit Counter	0.425
10	8-Bit Counter	0.454

Above: Table 8: Results for the 8-Bit Counters.

As can be seen from the table, no change was observed through the various ring oscillators. Some small changes were observed from the various counters. From the datasheet of the Cyclone II FPGA, it is known that the device consumes current in the tens of microAmps at most. However, the DE2 board is consuming current in the hundreds of milliAmps. As such, even if some change did occur in the current consumed through the FPGA, it would not have been seen. This does not mean that the hypothesis has been disproved. It means that isolating the device will be the best, and possibly only, way to test this hypothesis for sure. This leads back into the initial approach.

Return to the Initial Approach:

The conclusions drawn from testing the circuit on the DE2 development board show that the only way to truly test this hypothesis is to isolate the device itself. Even though our first attempt at testing the devices was not successful, a great amount of design work has been put into the circuit. The following will be an in-depth look into the initial approach including setup, recommended components and overall operation.

Below is a block diagram of the circuit. Each section will be looked at independently in the following sections.

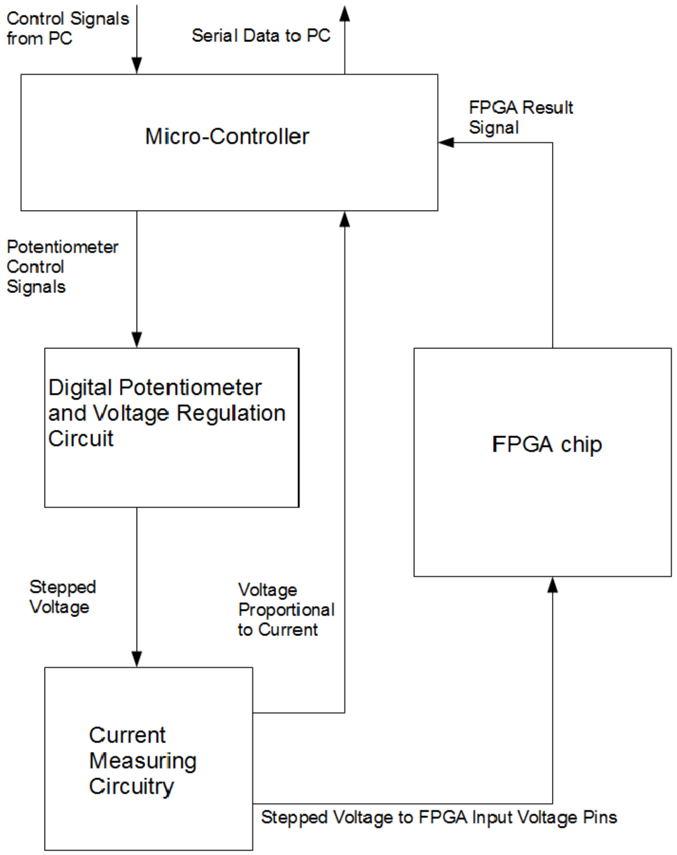
Control Signals from PC
Serial Data to PC
Micro-Controller
FPGA Result Signal
Potentiometer Control Signals
Digital Potentiometer and Voltage Regulation Circuit
FPGA chip
Stepped Voltage
Voltage Proportional to Current
Current Measuring Circuitry
Stepped Voltage to FPGA Input Voltage Pins

Figure 4: Block Diagram of the Circuit

Controlling the Voltage Supplied to the Device:

The voltage supplied to the device must be stepped down in increments of approximately 0.1 volts. This can be done using a microcontroller and two components: a LM317 Variable Voltage Regulator and a digital potentiometer.

The microcontroller we chose to use was an Arduino atmega. This microcontroller is very user friendly and fairly cheap. It also contains libraries to interact with several interfaces (SPI, Serial, etc) quickly and easily. It is programmed in C from a lightweight interface downloaded from the Arduino website. This microcontroller is generally used for hobby projects but is powerful enough and contains enough I/O ports for what is needed. These microcontrollers were also readily available.

The LM317 is a variable voltage regulator that allows its output voltage to be specified by two resistors, R1 and R2. The output voltage is given by an equation given on the datasheet: Vout = 1.25(R1/R2) * Iadj*R2 where Iadj is held to under 100uA. This allows the second term, Iadj*R2, to be negligible for small values of R2. This simpler equation was used when finding the preferred digital potentiometer. The voltage can be changed by choosing R1 to be constant at

240 Ohms (as stated in the datasheet) and replacing R2 with a digital potentiometer. This will allow the microcontroller to choose what voltage the LM317 should have by interfacing with the digital potentiometer. A circuit for using the LM317 is shown below.

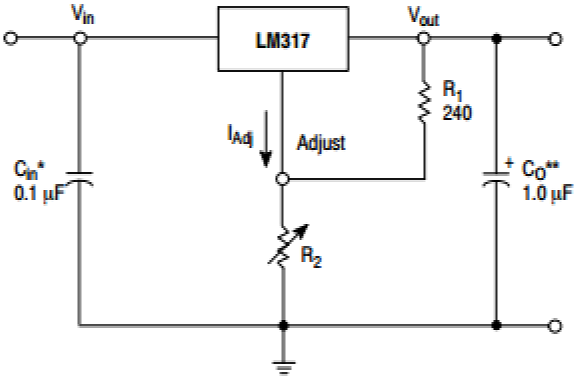

Figure 5: Circuit using the LM317 Voltage Regulator

The digital potentiometer must be able to have resistances that correspond to voltage steps of 0.1 Volts given that R1 is 240 Ohms. It is also desired that the potentiometer should be as small as possible so that Iadj will still be small enough to not have an impact on the output voltage. Some calculations were done that would find the ideal resistances for each voltage step. Then, a search began for a potentiometer whose resistances would match closely with the ideal resistances calculated. The closest potentiometer found was a 5 kOhm potentiometer with 257 steps. This is ~19 Ohms per step of the digital potentiometer.

The particular potentiometer selected was the MCP4151. This is a 5 kOhm, 257 step digital potentiometer that is communicated with via a SPI interface. Not by chance, the Arduino microcontroller can easily interface with this component through its SPI library. This library is included with the software. This allows the microcontroller to choose what step the potentiometer is at and control the voltage being supplied to the device.

This solution has two key problems however. When attempting to build this circuit, the digital potentiometer was found to be inaccurate at low resistances leading to incorrect voltages being outputted from the LM317. This is an inherent problem with potentiometers. One solution to this problem is to increase R1 and use larger values and larger steps of the potentiometer, which are more accurate. However, using larger values and larger steps increases R2. If R2 becomes too large than the Iadj * R2 factor is no longer negligible and the output of the LM317 will become very hard to predict. There is a dilemma in this problem with can solved in the method for measuring current.

Measuring the Current:

The next step is to measure the current through the device. This is more challenging since there are no easily available nor affordable components to autonomously measure current in the micro-amps range. The solution to this problem is to use a shunt resistor. A shunt resistor is a very small, high wattage resistor that is placed in the path between the LM317's output voltage and the device. This resistor is a known resistance (e.g. 5 Ohms). The microcontroller will measure the voltage between the resistor and the device and use Ohm's Law to calculate the current through the resistor. Due to the errors in the preceding section, it is necessary for the microcontroller to measure the voltage on both sides of the shunt resistor since the output from the LM317 will be unreliable at low resistances.

There are a few problems with this approach however. This first is that the microcontroller accepts a maximum value of 5 volts on the its analog input pins. Attempting to measure voltages around or over 5 volts may damage the microcontroller. Additionally, the microcontroller draws current when measuring the voltage. This will cause inaccuracies when measuring the current. The solution to this problem is to use an op-amp circuit with large resistors (resistors in the hundreds of kOhms range) to half the voltage on both sides on the shunt resistor before being measured by the microcontroller. The microcontroller has a Vref pin that it will measure the analog voltages relative to this Vref voltage. This Vref voltage can be changed to 2.5 Volts to keep the same accuracy without risking damage to the microcontroller.

The op-amp circuit is shown below.

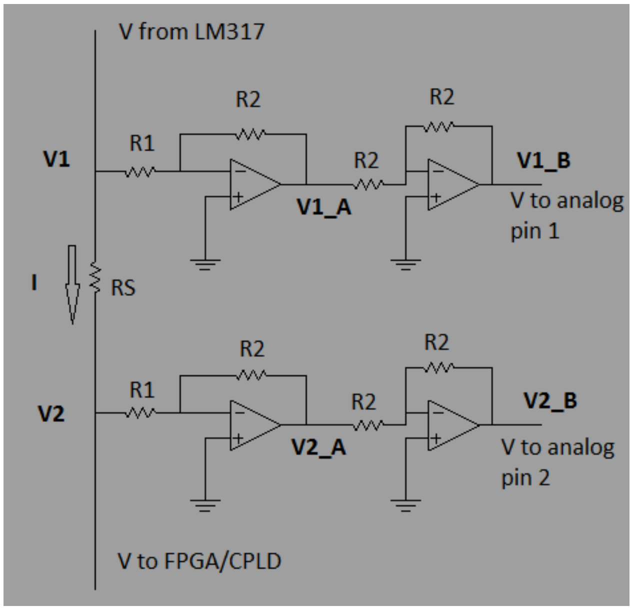

Figure 6: Op-Amp circuit to improve accuracy and protect the microcontroller

This circuit will half the voltages being measured by the microcontroller. Additionally, if large resistor values are used then the microcontroller will not draw as much current and will give more accurate readings. In the above circuit, R1 and R2 can be 200 k Ohms and 100 k Ohms respectively. This is just a proposed circuit. The values of R1 and R2 can be further tweaked for the best results.

Interfacing with the Device and the Circuit:

FPGAs and CPLDs have two voltages: a supply voltage and an I/O voltage. Both of these voltages will be from the same source. The voltage to the entire device will be stepped down.

Also, the device itself will contain digital circuitry to monitor and report errors. This digital circuitry (most likely VHDL or Verilog code) will watch the results being produced and report any errors through a digital pin on the device. The microcontroller will monitor this results pin from the device and react when errors occur.

The needed results gathered from this circuit are the voltage supplied to the device, the current through the device at this voltage step and the status of the results pin. The microcontroller has, or can find, all of this information and pass data back to the PC. The Arduino microcontroller has a library for a Serial interface through a USB port. Several programming languages have Serial libraries as well (e.g. Python, Matlab, C++). Whatever language is selected can open a Serial port on the same USB drive as the microcontroller and receive the data from it. From there, the data can be recorded to a text file, excel file, or whatever is needed.

This interface can also be used to give command signals to the circuit. A flowchart describing the behavior of the circuit is shown below.

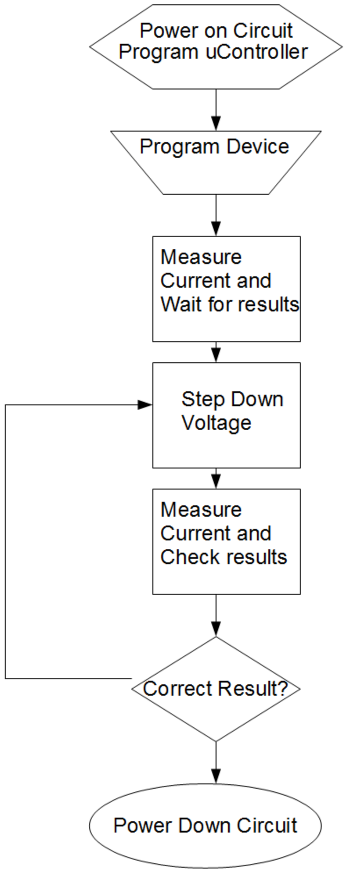

Figure 7: Flowchart of the circuit.

The microcontroller will need to know where it is in its execution. The microcontroller can wait for some code to be passed through the Serial interface before moving on. This is important because the device may not program properly if the voltage has already been stepped down too low.

Final Circuit Remarks:

By combining all of the blocks outlined previously, the final circuit can be built. The last step is to program and begin testing the devices. This step, however, is where problems occurred. Like most projects, with more time and more resources a circuit could be built. More resources would allow us to acquire better sockets, devices and programming utilities. With those, the circuit could be built and the device could be isolated and tested.

Conclusion and Future Work:

The hypothesis to be shown was that if FPGAs and CPLDs could be run at a lower voltage than specified with high accuracy then power can be saved. Due to problems using the devices, we were unable to isolate the device and test it directly. A new approach was tried which used an Altera DE2 Development Board but, unfortunately, this approach yielded inconclusive results. This does show that the only way to truly test this hypothesis will be to isolate and test the device directly. The groundwork for a circuit to perform this has already been done.

Future work will be to actually build this circuit and test the devices. The first step will be to find different sockets that have some way of firmly holding the device in place. After this, the devices can be tested in the circuit and the hypothesis further evaluated.

References:

LM317 Datasheet:
http://www.onsemi.com/pub_link/Collateral/LM317-D.PDF
USB Blaster Datasheet:
http://www.altera.com/literature/ug/ug_usb_blstr.pdf
MCP4151 Digital Potentiometer:
http://ww1.microchip.com/downloads/en/DeviceDoc/22060b.pdf
Flex 10K Datasheet:
http://www.altera.com/literature/ds/archives/dsf10k.pdf
MAX 7000S Datasheet:
http://www.altera.com/literature/ds/m7000.pdf
Cyclone II Device Handbook:
http://www.altera.com/literature/hb/cyc2/cyc2_cii5v1.pdf
Various Others Documents Available at:
http://www.altera.com/
Altera DE2 Development Board User Manual:
ftp.altera.com/up/pub/Webdocs/DE2_UserManual.pdf
Arduino ATMEGA Pin-Outs:
http://softsolder.com/2009/10/07/arduino-vs-atmega168-chip-pinouts/
Arduino Tutorials and Documentation:
http://arduino.cc/
Op-Amp Datasheet:
http://pdf1.alldatasheet.com/datasheet-pdf/view/194750/STMICROELECTRONICS/LM358N.html
Op-Amp Review:
http://webpages.ursinus.edu/lriley/ref/circuits/node5.html

Acknowledgements:

Thank you to Professor Joanne Degroat for allowing me the opportunity to work on this project and for supporting me through its duration.

Thank you to Professor Joanne Degroat and Professor Bradley Clymer for issuing my thesis defense and giving me feedback to include in this paper.