Controller for Distributed Power Processing in Battery Packs

John Miller

***Abstract*— A Dual Active Bridge is used to step up the voltage of battery cells. It is driven by an FPGA (Field Programmable Gate Array) with Pulse-Width Modulation (PWM) output which is controlled by a Proportional Integration Derivative (PID) controller. This project has three main development goals; 1.) PWM generator in VHDL, 2.) sensor input through ADC, and 3.) PID controller. The PWM generator is complete and functions properly. The ADC and sensor input is partially working. The PID controller still needs to be built. Future work will include the completion of the sensor input and the PID controller.**

I. INTRODUCTION

Power storage is necessary for modern power applications. Traditionally, batteries are connected in series to increase the voltage and connected in parallel to increase capacity.The main problem with this is the more batteries connected, the more potential for failure. Voltage is dependent on the worst cell, and the entire bank must be disconnected in order to replace a bad cell.

A Dual Active Bridge is a solution that would allow each cell to be monitored and replaced without having to disconnect the entire battery bank. The voltage of each cell could be stepped up. This would be a more reliable solution with potential economic and environmental returns.

Dual Active Bridges do have their own limitations. Currently, Dual Active Bridges are large, inefficient, and expensive. Engineers are experimenting with designs that minimize these drawbacks. Sandia National Laboratories partnered with New Mexico State University in order to explore the potential of this developing technology.

To that end, researchers at NMSU have already designed and built a prototype. The Dual Active Bridge does work, but it does not have a driver. The driver needs to adjust according to the input voltage, which is a battery cell, in order to maintain a near constant voltage output. A battery cell with a load will have a voltage that decreases over time.

This project is an attempt at driving a Dual Active Bridge using an FPGA that generates PWM signals and controls the PWM with a PID controller.

II. PROBLEM STATEMENT

A Dual Active Bridge (DAB) is driven by four Pulse-Width Modulation (PWM) signals. The signals are controlled by an FPGA with inputs from multiple sensors and processed with a PID controller. The design challenge is to build a self-contained control system that takes serial input from sensors and will change the duty-cycle and phase of several PWM signals.The original target device is the Altera De0-CV FPGA programmed in VHDL using Quartus II 15.0. Later designs targets the Altera DE10-Nano FPGA using Quartus II 17.0.

A. Dual Active Bridge

A Dual Active Bridge steps up the voltage of a DC power source to a higher voltage. This is achieved by converting the DC voltage to high frequency AC square wave, stepped up with a transformer, then rectified back into DC. There are several limitations to this process, however the most significant limitations are size and efficiency. The hope is that efficiency will increase over time and that economies of scale will bring down the cost and the size. [6]

B. Pulse-Width Modulation

Pulse-Width Modulation (PWM) is a high frequency square wave that is often used to effectively change the voltage of the load by varying the duty cycle and frequency of the square wave. For

instance, an LED could be dimmed by applying a PWM to it. For the application of the DAB, the PWM drives switches that convert a DC power source to AC.[5]

C. Serial Communication

Serial communication is used to communicate with sensors. For this project, a VHDL component could be built, however since Quartus has a built in soft processor and peripherals, those are used. The DE10-Nano has an on-board Analog to Digital Converter (ADC) that communicates using Microwire, which is a subset of SPI. Some FPGAs have on-die ADCs that do not rely on serial communication.

D. Analog to Digital Converter

The DE10-Nano has an on-board LTC2308 ADC. The LTC2308 is Low Noise, 500ksps, 8-Channel, 12-Bit ADC. It can be driven at 20 MHz, which is what this design does.

E. PID Control

A proportionalintegralderivative controller (PID controller) is a control loop feedback mechanism used to control a system, in this case the DAB.[2] The process variable (PV) is the sensed position of the output. The set point (SP) is desired output. The difference between the SP and PV is the error. A change to the PV could be proportional to the error to bring it close to zero. This is a quick reaction, but an integral will change the error to zero over time. However, an integral is slow to change. A combination of the two is used to have a fast acting and accurate controller. The derivative attempts to flatten the rate of change so that the correction does not overshoot the SP.[4]

III. Implementation

A. Generating the Pulse-Width Modulation

The Pulse-Width Modulation signal is generated by the FPGA and programmed in VHDL. Both the DE0-CV and the DE10-Nano have internal clocks of 50 MHz. A 200 kHz square-wave is needed to drive the Dual Active Bridge. The number of clock ticks is calculated as 50MHz/200kH = 250. One period of the PWM takes 250 ticks. The state machine counts from 0-250. At count 0, the output is set to low and at count 125 the output is set hight. The duty cycle is hard-coded to 125, 50% of 250. The duty cycle could easily be a variable input if needed.

Phase shifts are achieved by starting the cycle count at a different number. For instance, two waves in phase would have the same starting count. A third wave with a 5 s offset would have a count that starts 125.

The duty-cycle is likewise achieved by indicating the number of clock-cycles the output value is high. Again, 125 = 0.50*250, a 50 percent duty cycle would remain high for 125 cycles. A 10 percent duty cycle would be 25 cycles.

The PWM is made of several components in VHDL. This allows for the reuse of code. In testing the phase-shift, the PWM component is used to slow down the counter that changes the phase. The output is mapped to the GPIO pins of the FPGA and the values were verified on the oscilloscope. The output voltage is 5 Volts.

B. Serial Communication

Initially, a UART component was made in VHDL in anticipation of an ADC.[3] The team working on the DAB still have not chosen the sensors, and an ADC will be needed regardless of sensor type. This is the reason for the change in the FPGA. It should be noted that the both the DE0-CV and DE10-Nano both have a The sensors are still not chosen,

Serial communication is needed in order for the FPGA to receive sensor data. The DE0-CV does not have an Analog to Digital Converter (ADC) or on board serial communication making the DE10-Nano a better choice. The DE10-Nano has an ADC which communicates with the FPGA via SPI. Fortunately, Quartus has a soft processor called NIOS II which handles the communication. A serial component in VHDL is no longer necessary, as was the case with the DE0-CV.[1]

C. Analog to Digital Converter

The DE10-Nano has an on-board ADC. The ADC is connected to the FPGA via SPI communication. In order to connect the ADC to the existing VHDL components, Qsys is used to connect the ADC to a soft processor and the on board memory. Qsys is software bundled in Quartus that allows for a level of abstraction in the soft circuit design.

Components can be connected graphically and a VHDL or Verilog file generated in order to integrate with other VHDL components.[1]

D. Controlling the PWM via SPI

Since the sensor has yet to be chosen, the design here is more hypothetical. The FPGA is connected to an ADC using Microwire, a subset of SPI. In the current iteration, NIOS II handles the communication.

E. Sensors

Sensors were not chosen and did not factor into the design. This posed a problem because it affected how the implementation of the design.

F. PWM and VHDL

The PWM signal was generated by writing a component in VHDL. This is a state machine that counts clock cycles and changes the output based on the number of 'ticks'. For instance, at count 'zero' the output is turned off and at count='duty cycle', the output is turned on. Another output is also created in the same component that is the inverse of the output.

A second PWM is also created similarly, but with a phase-shift. The phase-shift is an input into the FSM, which then offsets this PWM from the first. The input can vary as it is checked with each clock cycle. This PWM signal has an inverted signal output as well.

Both sets of output signals were demonstrated by compiling the VHDL code and uploading it to the FPGA. The sensor reading was simulated by an input into the PWM component that counts from zero to the number of 'ticks' in the frequency period. The result is one square wave appears to be stationary while the other is shifting right then left.

G. ADC and NIOS II

The on board ADC communicates with the FPGA over SPI. Quartus II has a soft processor called NIOS II that easily interfaces with components such as the ADC.[1]

IV. Problems

A. Passing Variables

An early challenge was passing an initial variable into a VHDL process. Every time a process begins, it initializes to zero, regardless of the value assigned. Many efforts were made to change this, however the solution was to create a state that compares the two waves. Passing a variable generated by a counter then worked.

B. ADC and Serial Communication

The documentation for the FPGA refers to the ADC as using SPI. The pin assignments are labeled as SPI. The DE10-Nano seems to be unique in that the pin assignments are Microwire, which does not match SPI.

C. JTAG Connection Lost

The JTAG connection gets lost in programming. The initial program seems to upload, however when communicating with the FPGA it consistently throws an error. This was not resolved.

D. Software

The documentation for the DE10-Nano refers to Quartus II 17.0. The lab computers have version 15.0 which works for most features, however some components are different across versions, notably the ADC. Further compounding the issue, the University does not have a license for version 17.0. Installing the trial version involved several days talking to IT.

It is also worth noting that configuration files seemed to be easily corrupted. Entire areas of the software would not function on other machines, including in virtual machines.

Another issue was compiling C code to operate on the soft processor. The Intel FPGA Monitor, a program that allows users to program in a higher level language, was inconsistent and often returned errors. It would compile the code, but then lose the JTAG connection or fail to implement the soft processor.

V. Future Research & Recommendations

A. Qsys Components

Qsys allows users to connect existing components together or build new components. Building

components appears to be necessary for using the ADC. This is an involved process that requires more time.

B. Full Administrative Privileges

The workstation used had several components of Quartus that were blocked by Windows Firewall. IT had to be called multiple times to allow code to compile properly.

C. Use a different manufacturer

Altera's documentation seemed to have missing information. Entire software components were missing or changed despite having the same version installed.

D. Investigate Advanced FPGA Design

There are several software packages that generate VHDL and Verilog files automatically which increase productivity. This allows the creation of more complicated designs while saving time.

E. Choose Sensors

This is an area of the design that should have been decided before starting. Know what is going to be used will allow for a much clearer design goal.

F. Investigate and Implement PID

The PID cannot be fully implemented without knowledge of the sensors, however building one would be useful. There may be designs already in Quartus, however it is more likely that a PID controller will need to be tailored for each application.

VI. Conclusion

This project had three main goals; generate PWM waves that could be controlled, receive and generate data from the ADC, and process that data and control the PWM with a PID controller. The first goal was completed. The second goal was nearly completed. The third goal requires more time to complete.

References

[1] Intel fpgas and programmable devices - intel fpga.

[2] Pid theory explained.

[3] Pong P. Chu. *FPGA prototyping by VHDL examples: Xilinx Spartan-3 version*. Wiley-Interscience, 2008.

[4] F. Radke and R. Isermann. A parameter-adaptive pid-controller with stepwise parameter optimization. *IFAC Proceedings Volumes*, 17(2):18851890, 1984.

[5] Xiaohong Wang, Zhifeng Pan, Thi Thu Giang Hoang, Lianfang Tian, and Yangquan Chen. New repetitive current controller for pwm rectifier. *IFAC-PapersOnLine*, 51(4):154159, 2018.

[6] Biao Zhao, Qiang Song, Wenhua Liu, and Yandong Sun. Overview of dual-active-bridge isolated bidirectional dcdc converter for high-frequency-link power-conversion system. *IEEE Transactions on Power Electronics*, 29(8):40914106, 2014.