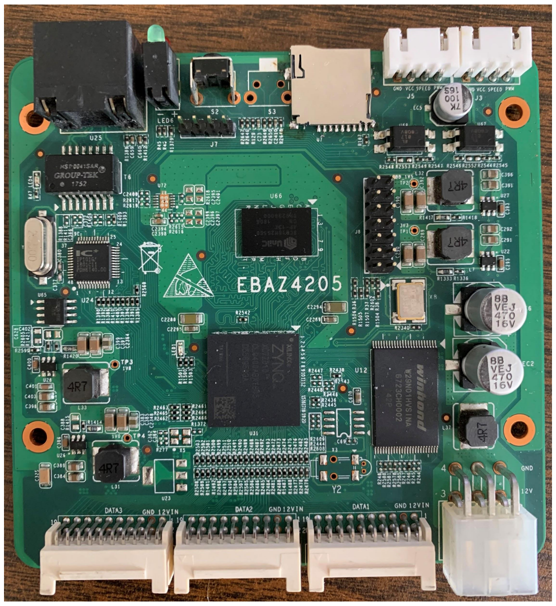
Final Project

Andrew Pond - u0666767

Overview

For my project, I set up linux on the Zynq7000 based board (EBAZ4205). I also generated a bitstream to program the FPGA inside of it.

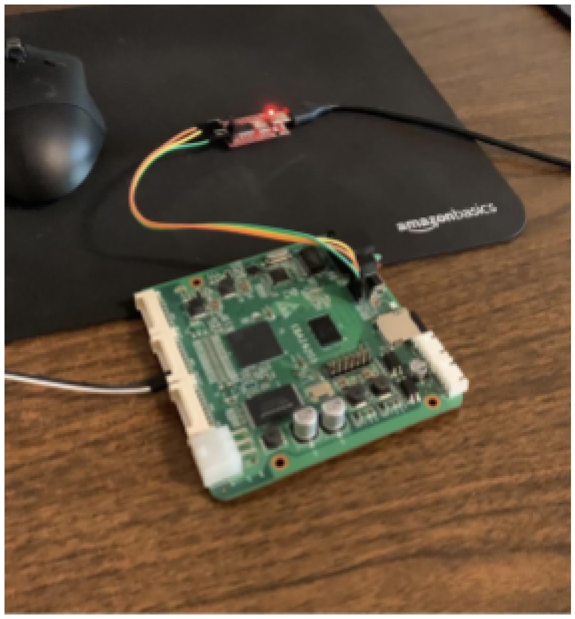
While this project does not cover 3 of the lab concepts directly (GPIO, interrupts, timers, UART, I2C, Analog), it involves those more basic components and still very well demonstrates embedded systems concepts. It was approved because of this.



Milestone 1

For the first milestone, I got the board booted and running. This board was a control board for old bitcoin miners, so it was preconfigured with a linux OS booting from the NAND flash.

I got the board to power on, then verified that the linux system booted correctly.


EBAZ4205 Connected with UART

Successful System Boot

Passoff Requirements

- ✓ Figure out how to power the board
- ✓ Connect to processor UART to verify that linux booted OK.
- ✓ Change board boot mode from NAND flash to SD card

Milestone 2

The objective of this milestone was to learn how to build a custom SD boot image that includes all of the peripherals in the linux system (ethernet, DDR, etc).

For the second milestone, I built a custom linux system using Yocto. I found a github repository that made the process much smoother. This was the route that worked best for me, as it had decent community support and it was almost just as easy as downloading it and running it.

I first tried to build the image using Yocto, and after 50 minutes of compiling, I got an error saying “name ‘ebaz4205’ is not defined (see github issue). After working with others on debugging, we found that in one of the build files, a string literal didn’t have the necessary double quotes, and therefore was causing the name error. It eventually ran without errors and I was able to flash the image that it generated onto an SD card.

Here is the video demonstration: https://youtu.be/7t_ZWefwDjE

Passoff Requirements

✓ Download buildroot utility
✓ Figure out what needs to be configured to boot correctly
✓ Flash image onto SD card
✓ Connect to linux after boot via UART

Milestone 3

For the third milestone, I would like to put a simple design on the FPGA. I used the JTAG interface to upload my bitstream onto the programmable logic of the Zynq 7000.

The objective of this milestone was to establish a process in which I can easily reprogram the FPGA inside of the Zynq 7000.

I ordered a USB to JTAG programmer, but it still has not come in the mail. I contacted P.E. and proposed an alternate route for this milestone. I had access to Digilent Zybo boards, which have the same Zynq7010 chips that my EBAZ4205 does, and a built in USB to JTAG programmer. He has allowed me to use the Zybo board for this milestone instead of the EBAZ4205 board.

The process was the same as before, but with the Zybo, I used the onboard switches and LEDs to demonstrate a working HDL design. In this design, the onboard switches control the inputs of an AND gate, and the output of the AND gate is connected to LED0.

Here is the video demonstration: https://youtu.be/ZggI3Qom9Iw

~~**Passoff Requirements**~~

- ~~❏ Download vivado, configure project for EABZ4205 board (Zynq7000)~~
- ~~❏ Create a simple bitstream that turns on one of the data lines~~
- ~~❏ Create device constraints file to map design to physical pins~~
- ~~❏ Upload the bitstream to the FPGA~~
- ~~❏ Multimeter should show logic 1 on the designated pin~~

New Passoff Requirements

- ✓ Download vivado, configure project for Zybo board (Zynq7000)
- ✓ Create a bitstream that contains an AND gate
- ✓ Create constraints file to map AND gate inputs to switches, AND gate output to LED0
- ✓ Upload the bitstream to the FPGA
- ✓ LED will light up when both switches (SW0 and SW1) are on, and turn off otherwise

Final Submission

For the final submission, I created a demonstration that would tell you which button on the board was pressed using the LEDs. For example, if BTN2 was pressed down, the LEDs would light up ‘0111’. If BTN3 was pressed, ‘1111’

This is to show how I can reprogram the FPGA quickly to adapt the design to whatever I need.

Here is the video demonstration: https://youtu.be/IgJiAmpQ4uQ

Passoff Requirements

- ✓ Create design that lights up LEDs according to which button is pressed

Conclusion

I successfully completed the project! I took the retired bitcoin control board and turned it into a personal development board. I will have learned how to configure both the PS and PL so that I can use the board to run my own personal projects.

Future Ideas

- Learn how to program NAND flash, get system booting from my own image on the NAND flash
- Get communication between PS and PL
  - Set up linux drivers
  - Learn how to communicate with the AXI bus