GSoC 2021 Project Proposal

apertus° Association

Remote Test System for AXIOM Remote

Vinayak Sankar

Mentor: Herbert Poetzl (@Bertl)

About Me:

- Name: Vinayak Sankar
- IRC Nick: @vnksnkr
- Github: https://github.com/vnksnkr
- Email: vnksnkr@gmail.com
- Location : Kerala, India [UTC+05:30]
- University: Government Engineering College, Thrissur (2019-2023)
- You can find my CV here

- What interests you most about the apertus° AXIOM project?
  - Being an electronics and communication student who likes hacking into all kinds of devices, I was immediately drawn into the work being done by the Association when I came across the project.
  - The idea of having something like a camera whose technology is not actively shared by the industry,completely 0pen sourced, itself piqued my interest to learn more about the project.

- As mentors and project coordinators, how can we get the best out of you?
  - The idea of working on such a project itself motivates me to put in my best.
  - Regularly keeping track of my progress and asking for reports would help a lot to keep me in the zone when doing the tasks.
  - Until now I've got great responses from the mentor and other experienced contributors of the organization whenever I have questions, and that really encourages me to keep working.

- Is there anything that you'll be studying or working on whilst working alongside us?
    - Currently, my semester exams are slated to end within the first week of May which would give me free time during phase 1 as there will be only classes and not any exams.
    - If by any chance any other exams are scheduled during phase 2 I will adjust my timeline as needed to complete the tasks by the end of the coding period.

- Are there any techniques and tools which you use to keep yourself organized?
    - I do take notes on my phone when I need to plan out something, apart from that I'm familiar with Vim and Sublime editors although I'm getting acquainted with Vscode which was suggested by a few members from the organization's IRC channel.

Qualification Task:

https://github.com/vnksnkr/UART-SPI-BRIDGE-VHDL

Synopsis:

apertus° Association is an organization that aims at creating free and open source technologies in the field of film making and audiovisual media production.

Remote Test System for AXIOM Remote (T1233):

- The AXIOM Remote is a device used to control the AXIOM Beta range of cameras.

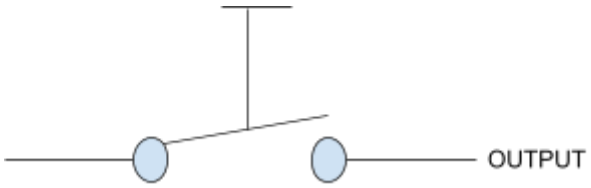
- The aim of this project is to develop an automated test system to test the various electronic components on the remote (buttons, switches and rotary encoders) from the AXIOM Beta.
- HDL needs to be implemented and tested for pressing various buttons and turning dials of the rotary encoders using the GPIOs of the routing fabric and a shield designed for interfacing with the remote.
- Testing actions should also cover corner cases like button bouncing and encoder skipping.
- A python framework is to be developed to automate the testing schemes.

The AXIOM Remote uses mainly push buttons, slide switches and rotary encoders as inputs. The idea is to simulate pressing buttons and rotating dials on the remote by sending signals to their terminals from the routing fabrics i.e the Lattice MachXO2 FPGAs.The signals needed to be generated depend upon the contact form of the component being actuated. In order to cover corner cases, button bounces and encoder skipping are also to be generated.

The signals are as follows :

Push Button :

The push button has an SPST contact form (Single Pole Single Throw) The signal is to be sent to the output pin of the push button from the FPGA.



A typical output signal(from push buttons) would start with a logic level high followed by a number of pulses with inconsistent widths followed by a logic level low until the button is released after which again a number of pulses are generated ending with a logic level high . To generate such a waveform accurately would not be possible as the time taken for the bounce to get stable and the width of each individual

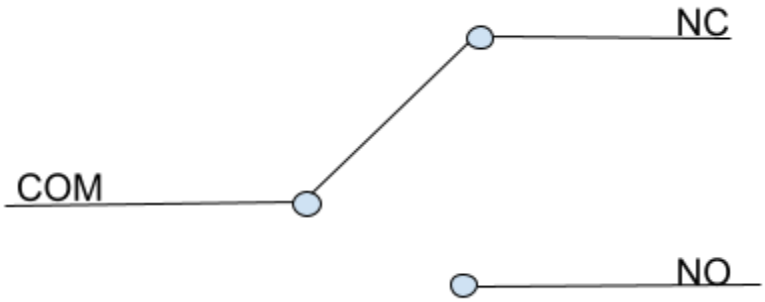
bounce is random.However empirical data from a few online resources (especially from A Guide to Debouncing by Jack G. Ganssle) suggests that a switch exhibits anywhere between 1.5ms to 6.5ms of bouncing.To be on the safe side the bounce duration can be taken as 10ms and the width of the individual bounce pulses varies between less than 1 µs and hundreds of µs.

So for pushing a button :

1. Random number of pulses of width between 1 µs and 100µs is to be generated within a total time of 10ms.
2. Followed by a low signal for a time greater than 10ms to simulate a button press.
3. Then again a random number of pulses are generated similar to that produced in 1.
4. And then a high signal to simulate the release of the button.
5. The number of pulses needs to be randomized using HDL.

Slide Switches:

The slide switches used in the remote have SPDT(Single Pole, Double Throw) contact form.



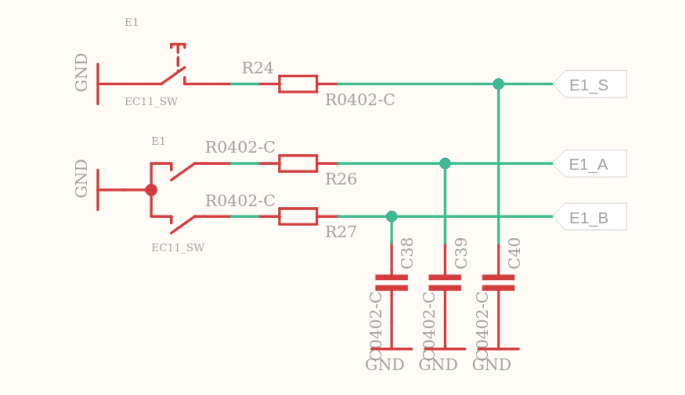
The same empirical data applies for the SPDT switches when COM(in the diagram) makes with either NC(normally closed) or NO(normally open).

For actuating the switch:

1. Generate pulses(bounces) similar to that for the push button at the terminal that breaks first i.e NO for 10ms.
2. Followed by sending a high signal until the switch is opened.

3. There would be an inter bounce delay before contact with NC is made.
4. An optimum value for this delay is 2.25 ms which is also an empirical value.
5. After staying in the current state for 2.25 ms the contact made with NC is simulated by generating pulses followed by a low signal until the switch is opened.
6. On opening the switch NO and NC come back to their original state after going through a few bounces again.

Rotary Encoders:



The rotary encoder used in the remote has three terminals, two terminals (E1_A, E1_B) that read the rotation of the encoder and one (E1_S) that reads whether the encoder has been pressed. The signal to be generated at the button(E1_S) is similar to that to be generated at the pushbuttons. For E1_A and E1_B, we need to generate signals that are apart by a phase of 90°. The terminal that will lead by 90° will depend upon the direction in which the dial is turned.
Here also we need to generate bounces when a transition takes place to cover corner cases of encoder skipping.

Once the HDL for the button presses and turning dials are implemented, the next goal is to develop a Python framework to automate the testing. This can be done by developing code that can send triggers to the

MachXO2 via the PIC16s below them using I2C. An ideal end result would be a framework that can run scripts that are both pre-written and something that can be generated from the command line.

Timeline and Deliverables :

Week	Deliverables
Community Bonding Period May 17 - June 7 2021	• Discuss the implementation of bounce generation with the mentor. • Get familiar with the Axiom Beta, AXIOM remote and other components (shield, breakout board) from remote access • Familiarize the codebase to plan out how to send triggers to the MachXO2 from the Axiom Beta.
Week 1 June 8 - June 14	• Learn more about the MachXO2 FPGA and the GPIOs • Set up the development environment • Develop code for generating signals to be sent to the terminals
Week 2 June 15 - June 21	• Continue writing code. Cover corner cases by implementing bounces. • Develop a Random Noise Generator for sending bounce signals.

Week 3 June 22 - June 28	● Finish working on the Random Noise Generator. ● Start testing for every component on the remote. ● Optimize code, fix errors if any.
Week 4 June 29 - July 5	● Begin interfacing with the shield and test the code on the Axiom Beta provided remotely. ● Test specifically corner cases and improve code if needed.
Week 5 July 6 - July 12	● Complete working on the HDL part. ● Document the current status. ● Start working on the Python Framework. ● Learn more about accessing the FPGA from the Zynq PS side over I2C,
Week 6 July 13 - July 19	● For the framework initially start working on writing a number of scripts that will test the components in a sequential manner. ● Work on sending commands via the PIC16 to the FPGAs. ● Test the scripts on the Axiom Beta provided remotely.
Week 7 July 20 - July 26	● Fix and improve the scripts if any errors. ● Complete work on the first stage of the Python Framework ● Start working on the second stage i.e feature to write your own script as an operator.

Week 8 July 27 - August 2	• Figure out and develop code to read inputs from the command line to control the remote.
Week 9 August 3 - August 9	• Testing and fixing bugs of the second stage of the framework • Format code and provide readme for writing a script from the command line.
Week 10 August 10 - August 16	• Finish documentations, verify all goals and finish working on project report

Other Relevant Background Info:

I chose this project because I believe the tasks needed to be achieved within this project and the prerequisites for the project fall under my area of expertise.

FPGA was something that really interested me ever since I laid my hands on the Nexys 4 DDR in my university lab and I felt working on this project would be a great experience for me. Although I was more comfortable working with Verilog, working on the VHDL Challenge helped me improve my VHDL skills.

Apart from that, I also have relevant experience working with Python and C. However, I felt Python would be a better choice for the project since it would be easier for the operator to write automated scripts.

I also have a little experience contributing to open source projects which started off during Hacktoberfest 2020. However, contributing to an organization like apertus° would give me a lot of experience in a field that really interests me.

References

https://www.eejournal.com/article/ultimate-guide-to-switch-debounce-part-1/

http://vserver.13thfloor.at/Stuff/AXIOM/BETA/axiom_beta_shield_remote_v0.2_r1.1.pdf