GSoC 2021 Project Proposal

AXIOM Remote: Firmware Improvement And Extension

Aman Singh

Mentors: Sebastian and Priya

13th April, 2021

Table of Content

0. Curriculum Vitae of the Student

1. About AXIOM Remote

2. About Firmware improvement and Extension Task

3. Qualification Task

4. Progress Student wants to make During GSoC Period, in below tasks
   - Page Setup
   - Adding new UI classes
   - Event Handling
   - Text Input and Transition Animation
   - Histogram, Scopes and other Graph
   - Drawing Optimization in Firmware
   - Performance Optimization in Firmware

5. Timeline
   - Pre-GSoC Phase : Till May 17th, 2021
   - Community Bonding Period : May 17th, 2021 - June 7, 2021
   - Coding Phase I : June 7, 2021 - July 12, 2021
   - Coding Phase II : July 12, 2021 - August 16, 2021
   - Post GSoC Phase : Till Infinity And Beyond

6. Q&A
   - What interests you most about the apertus° AXIOM project?
   - As mentors and project coordinators, how can we get the best out of you?
   - Is there anything that you'll be studying or working on During GSoC?
   - Are there any techniques and tools which you use to keep yourself organized?

Curriculum Vitae of the Student

Name: Aman Singh

Undergrad Course: Computer Science

Email: theamanamanaman@gmail.com

Country of Residence: India

Primary Language: English

Time Zone: UTC+5:30

IRC Name: eppisai

Github Handle: eppisai

Link to the CV: Link

In this proposal, I am not trying to establish myself as someone who knows all the technology and is extremely well experienced with tech stack needed in AXIOM Remote.

I ***want to learn*** *and contribute to things that add meaning to other's lives. Hence, I am applying* ***as a student who wants to be part of Apertus in the long run, and GSoC will prove to be a great motivator for him to learn things faster, at the organization.***

About AXIOM Remote

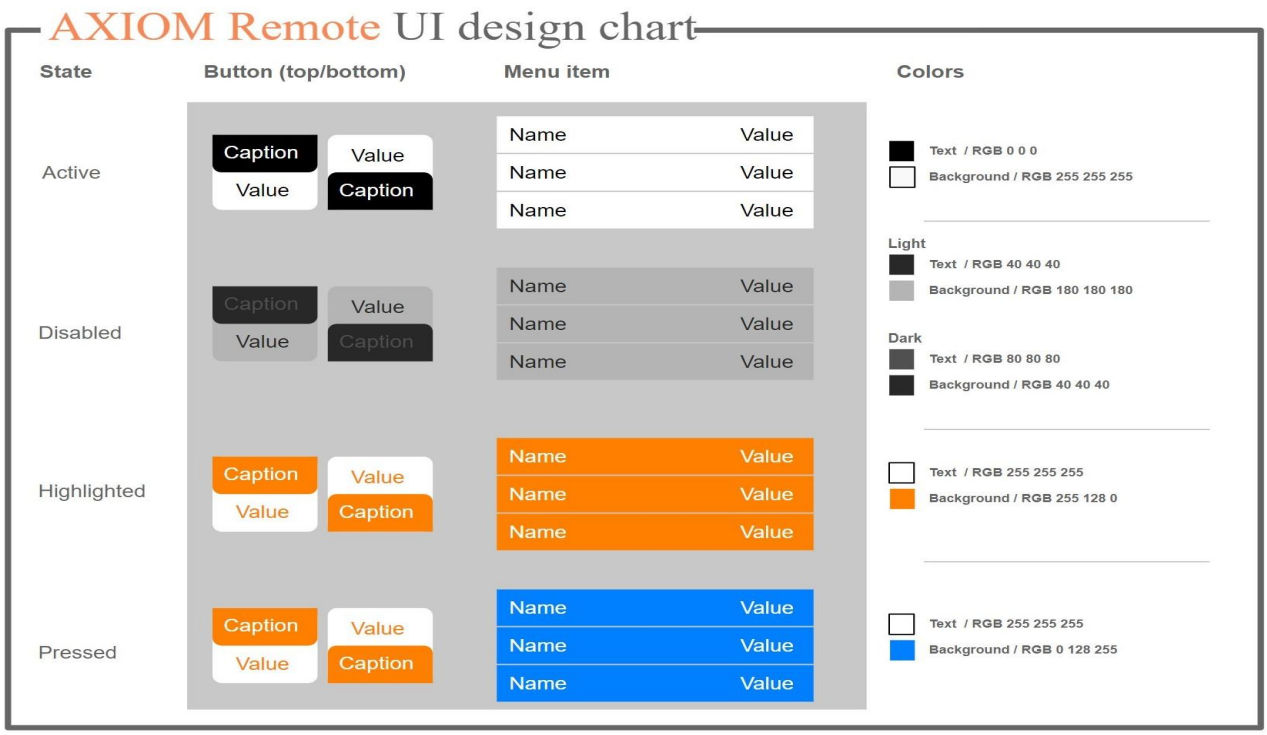
AXIOM Remote is the remote control for AXIOM beta. It's hardware includes PIC32MZ, two PIC16s and a LCD for the menus and options, with different buttons and rotor. The software runs "bare metal" and has no graphic acceleration, so each pixel is drawn manually in the software.

Electronic Components

- PIC32MZ as the core processor, 2 small PIC16s are used for the handling push buttons, rotor and the LED I/0 operations
- 2.8'' 320x240 TFT from Adafruit is used as the display
- USB-C Connector is used

About PIC32MZ

- M4K 32 bit core from MIPS Technologies - Harvard architecture based core
- The M4K core uses a 5 stage execution pipeline
- PIC32 uses the high-performance version of the Multiply and divide hardware module.
- Bus Masters - CPU, ICD, USB, DMA
- SYSCLK Peripherals - Prefetch Cache, USB, DMA, SRAM, Interrupts, I/O PORTs
- PBCLK Peripherals run for PBCLK -SPI, UART, ADC, RTCC, I2C
- Concurrent Accesses, that is while the first transaction started by one bus master is in progress, another Bus Master may initiate a second transaction to yet another target.



About Firmware Improvement and Extension Task

Firmware as the name suggests is a bridge between user and hardware/features of AXIOM Remote. Hence, there is a need for Firmware to follow good UI/UX principles,at the same time should not be heavy on the CPU. Axiom Remote uses ***PIC32MZ2048ECG100*** MCU, which has *512KB of data memory*, and *2MB of Program memory*.
AXIOM Remote general UI guidelines (Link to Guidelines) clearly mentions how things flow in AXIOM Remote firmware, and the correct structure and plan one should keep in mind. I tend to follow the structure and at the point of dilemma or turmoil would discuss with mentors.

Task mentioned in Firmware Improvement and Extension

1. ***Event handling (button interactions, turning the rotary knob, etc.)***
   Currently, there is no Behavioral Design Pattern like Command Pattern Design for handling events in Remote. A switch case is used for handling events on each screen.
2. ***Page setup***
   Generally, while adding a new page, few components are common or can be predefined in Interface, like some Buttons around the TFT, can be pre-set for navigation, so setting up the new page would be easier.
3. ***Add new UI classes***
   Currently, there are different UI classes like button, checkbox, text that are used in various screens, adding new UI classes like header, divide , scrollbar, could improve UI designing.
4. ***Text input GUI***
   Text Input is needed in AXIOM Remote, proper text input method, with keyboard, and text field, that is not heavy on CPU, and is easy for users to navigate.
5. ***Histogram, scopes and other graphs drawing***
   AXIOM Beta features a small program called cmv_hist3 that calculates raw histogram value. Defining a protocol (between AXIOM beta and Remote) for Graph variables and then a base graph widget, from which different graphs can be mapped.
6. ***Optimize drawing***
   Currently, the frame buffer is drawn again and again like a video game. But as the Display memory of ILI9341, stores the image persistently. Hence we can optimize it.
7. ***Add transition animations***
   Navigating between the screens in GUI can be made attractive, the old AXIOM remote version had beautiful transition animations involved, which can be improved and readded.
8. ***Unit tests to verify proper implementation***

Qualification Task(T1191)

I had started Qualifying Task, with converting the Image to 2 bit using online converters, and then displaying the image in firmware. For that my draw2bit Icon method was working fine. But the offline conversion process for 1 bit image was using xbm, a format which does not support 4 different colors. So, I tried looking for other formats and found xpm, but it was an ascii format.
Hence, upon discussion with mentors, it was decided to make a python script that can convert an svg image icon to an array needed for the task.

Things/Subtasks I have done in Qualification Task

0. A Python Script to Convert Image to Array
1. DrawIcon2bit method in Painter to draw 2 Bit icons in firmware
2. Extended Icon.h, to better fit 2 bit icon data, and its colors
3. Added Transparency feature in Drawing 2 bit Icon
4. Unit Test to verify Proper Implementation

For me, the most time consuming and at the end, one of the easiest was **Alpha Compositing.** I first needed it for converting an RGBA image to its RGB equivalent, in python script and then, in firmware for applying transparency to icons.

So, What is Alpha Compositing?
It's just intermixing of background color and current color in ratio of **Alpha value**, that's it.

Below is the link to code base and complete explanation of the solution.

Link to Complete Task with codebase - Link

Progress Student wants to make during the GSoC

Optimize Drawing

There is no need to constantly update and display framebuffers, which reduces framerate and lowers performance. In this GSoC Period, Optimizations will be done in two part ways.

1. **Draw on Demand** - An efficient approach at the firmware, which will tell the remote, when to send new framebuffer data to the LCD. So, not sending data again and again constantly to the LCD.
2. **Dirty Rectangle** - Identifying which parts of the framebuffer are changing and sending only changed part of the framebuffer to the LCD.

I have read quite a lot about dirty rectangles algorithms, and have discussed little bit, about various things involved in dirty rectangles, or it is even necessary in AXIOM remote at the moment. Like a dirty rectangle overlap check, or where and how to use the DrawPixel() method to grab min and max coordinates of drawing, and involve some mechanism to grab min and max positions of Draw.

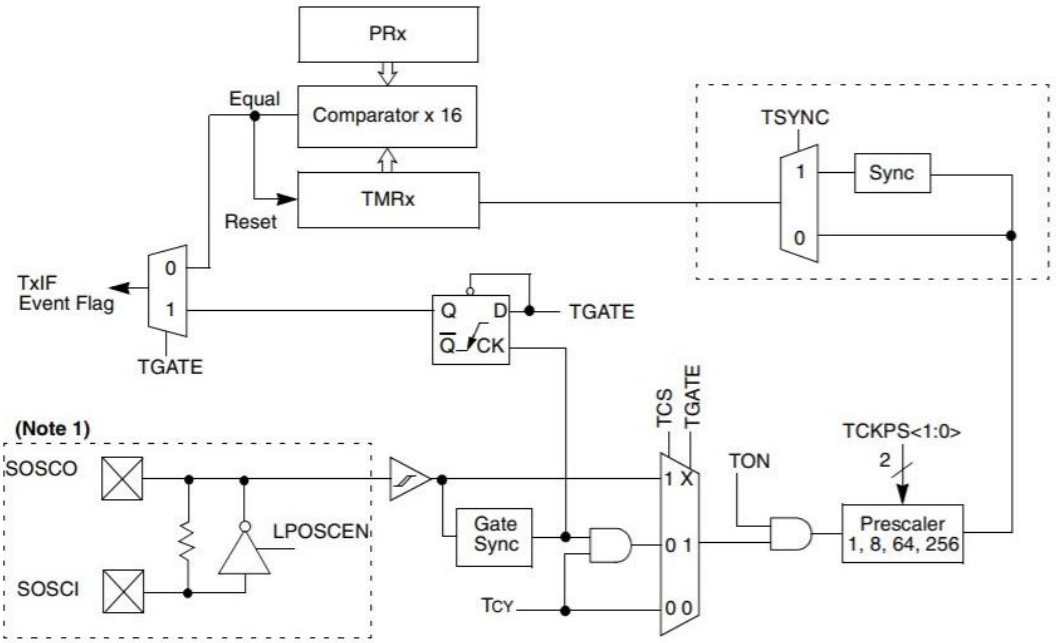
Optimize Performance

There are some sectors in firmware code that can be improved upon, like unnecessary calls and variables in Draw method in some screens. Some modification, in the existing structure of the objected oriented firmware code, like removing unnecessary functions,will pave way for better performance.

PIC32, has a set of timers, For **profiling the duration of functions/processes ,** we would need them. I am reading the manual of PIC32MZ , and at the time of writing this proposal, below is my understanding.

Basic Details of Timer Module in PIC32

- PIC32 device family supports 2 types of timers
  - Type A Timer (16-bit synchronous/asynchronous timer/counter with gate)
  - Type B Timer (16-bit or 32-bit synchronous timer/counter with gate and Special Event Trigger)



Timer A's Block Diagram

My understanding so far in Timer A,

1. It has two 16bit registers. **TMR1** (Timer 1) and **PR1** (Period 1)
    - **TMR1** counts each clock pulse applied to it.
    - The module has a comparator to set **T1IF** (Timer 1 interrupt flag) , when Timer 1 register (**TMR1**) matches the Period Register (**PR1**).
2. Timer A's control register decided various operating modes of timer A.
    - ON bit (**T1CON<15>**) is used to run or stop the timer 1.
3. The clock source can be a peripheral bus clock (**TPBCLK**) or external clock source from a secondary oscillator (**SOCSO/T1CK** , **SOSCI**).
    - Clock source is controlled by a TSC bit (**T1CON<1>**).
    - It can run in gated accumulation mode, it is enabled by **TGATE** bit
    - Timer A 's register starts incrementing, when the **T1CK** pin transitions to a high from a logic low level.
    - Incrementing from internal clock source, and stops when T1CK pin makes a transition to a low state, the T1 interrupt flag is set, and it can generate an interrupt.
    - When an external clock source is used, the external clock can be synced with the internal clock using, **TSYNC(T1CON<2>) bit** set to 1, in the timer A's control register.

I am still reading and trying to set timer is PIC32 - Link to my ongoing progress

I am using radare2 cutter and analysing elf files and finding worrying parts in code and trying out Valgrind's tool, callgrind for memory profiling (trying google's orbit in windows also). Link to my ongoing progress

Text Input

I have had a good amount of discussion, about text input, and how it's UI/UX should look like. Sebastian has helped on looks and how key input should be, so it will be best for the user to interact with it, given our input needs in remote.

Based on discussions with Sebastian and everyone, and help from Andrej in coding,I had coded a Text input screen.

Work to be done on Text Input Screen

1. Moving keyboard to widget.
2. Unit Test for keyboard

Screenshots of text input screen

[Link to code of above]

Transition Task

I had looked and understood the algorithm used in transition in Previous Axiom Remote firmware version, by Sebastian. Sebastian had explained to me the algorithm, and based on that, I had implemented the same in the current version of Axiom remote.

Previous version of AXIOM remote was written in C so was more functional Oriented (although Object oriented approaches were used in menu system and draw method, which was very fascinating and creative to see in C language), and the current Remote version is more object oriented. Andrej had helped me in thinking how and where the transition variables and functions could go.

Currently we use observer design patterns, one-to-many dependency between objects. I have so far extended the VirtualLCDDevice and its interface for handling framebuffer transitions and have made the attributes needed for transition in each screen. And have made different transitions, which can be called using enum, and transition speed can be set and changed.

I have added all transition animations in the current remote, to test it first, I have extended them in Visualizer. But framebuffer manipulations are currently happening in Visualizer and not in firmware.

Work to be done in Transition Task

1. Properly completing transition animation code structure, so they will happen in firmware, and not in Visualizer.
2. Adding the option to change the speed of transition in UI/UX.

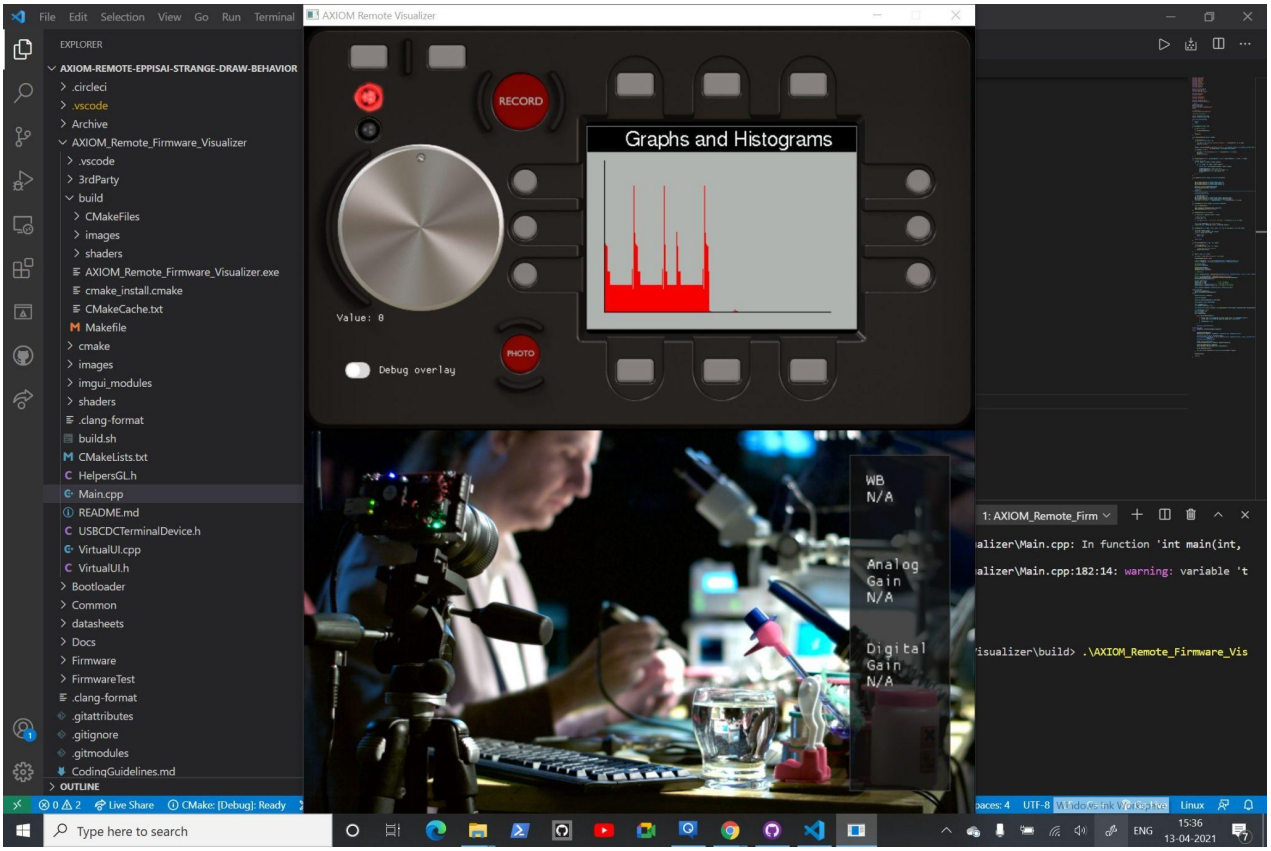
Link to the Video showing current remote visualizer with transitions

Histogram Scopes and Other Graph Drawing

Axiom Beta has a program, cmv_hist3, which outputs 4096 lines, and four columns for R,G,B,GB channels. These values are what is needed to draw a histogram, scope and other graph of an image. But these are not in correct format to display and map a graph on AXIOM Remote.

Defining a set of Protocols for the correct Data format is what is needed, between daemon and remote.

I had made a base graph widget, and had mapped a Histogram with it. I would need to improve on it alot, since I was inexperienced back then, and discussing and coming up with proper protocols about the graph data.



Screenshot of previous Progress

Adding New UI Classes

Current screens on Remote and new screens that will be added to the remote would be needing a few UI classes like header, divider, footer, slider, progress bar and scrollbar.
I will be adding new UI classes as per the common feature set.

Event Handling

Replacing Switch case Input Handler, with event handlers based on Behavioural Design Pattern.
I was trying out to come up with a Command design Pattern Event Handler in January, but the approach involved using Interface, and then defining a set of features that are associated with each button, I will complete the design and will implement it.

Page Setup

I have given time in setting up new screens (was adding a keyboard screen for Text Input Task , and histogram screen for showing Graph), and generally, while adding a new page, few components are common or can be predefined in Interface, like some Buttons around the TFT, can be pre-set for navigation, so setting up the new page would be easier.

In the older AXIOM Remote design GUI, by Sebastian had few features that I loved,

- Previous and next page can be accessed by pressing the left/right buttons
- The Page number button is used to change the page
- “Help” button: to know more about any label/button.

Finally, unit testing for each component added to the Firmware.

Timeline

Whenever I plan something to do, I try to break the task into smaller subtasks, and then work on smaller tasks. Similar approach, I plan to Follow Here.
I am gonna spend time understanding all the tasks, and completing my previous incomplete tasks Till 17th May, and then if I am selected, I'll focus on dividing each task into subtask in Community Bonding Period after discussion with mentors, and will be keeping a daily log of my progress, in a publicly viewable manner (google docs, github) so that necessary people in the organization will be aware about by my progress, and if necessary, can help me if they find something wrong.

Pre-GSoC Phase - 13th April Till 17th May

(I will be trying to prove that I'll be able to complete my respected task that might get assigned to me during GSoC period, and I have not been wasting my organization's time so far)

Task 0 - Start profile the firmware code
Task 1 - Complete Transition Task
Task 2 - Complete Keyboard Task
Task 3 - Discuss Approaches with mentors about Firmware Tasks

Community Bonding Period - 17th May to 7th June

Task 0 - Discuss and finalize the subtasks for each task.
Task 1 - Adding new UI classes
Task 2 - Page Setup
Task 3 - Event handling

Week 0 (17th May to 24th May)

- Discuss and divide each task into its smaller subtasks.
- Discuss with mentors your thought and approach on what UI classes can be added
- Start adding the UI Classes, and ask mentors for review
- Start a daily logging Progress, which is publicly viewable

Week 0.1 (24th May to 31st May)

- Adhere to the reviews by mentors and complete the UI classes Task
- Discuss with mentors your thought and approach on Page Setup can be added
- Complete the Page setup task in firmware,and get it reviewed by mentor

Week 0.2(31st May to 7th June)

- Discuss and Design various approaches in Event Handler Task
- What design Pattern you want to use, how you want to proceed, have solid understanding of these after discussion
- Complete the Event Handler Task, and get it reviewed by mentors

Coding Phase 1 (7th June to 12th July)

Task 0 - Set-up Timer in PIC32
Task 1 - Finalize the Optimizations in Firmware.
Task 2 - Histogram, and other graph screen
Task 3 - Defining proper protocol, for Graphs

Week 1(June 8 - June 14)

- Finish the Analysis of the firmware code
- Complete the Histogram screens, and other graph drawing.
- Discuss with mentors, how you wanna edit the code of firmware.
- Analysis of the time taken for buffer transfer and fps of remote variables and methods that are called unnecessarily in painters.

Week 2(June 15 - June 21)

- Document changes you want to make in firmware, and present it to mentors, before proceeding
- Start Optimizing the Firmware code.
- Analyse again the firmware code, and have a proper functioning of how else you can improve it.
- Present the data and optimization changes to mentors

Week 3(June 22 - June 28)

- Start various approaches and things you can do in Drawing Optimization
- Start with “Draw on Demand”
- Start analysing and designing the mechanism to grab min and max positions of Draw.

Week 4(June 29 - July 5)

- Complete things that are left to be done in all the task, till date

<u>Coding Phase 2</u> (16th July to 16th August)

Task 0 - Finalize the Optimizations in Drawing
Task 1 - Complete the changes and get them reviewed
Task 2 - Analyse all the changes
Task 3 - Complete previous tasks, if left

Week 6(July 13- July 19)

- Think of mechanism you can make in firmware to implement Draw on Demand
- Design the pseudo code and algorithm
- Try to implement

Week 7(July 20 - July 26)

- Present the Draw on Demand Implementation to mentors
- Adhere to their reviews and think and modify approach accordingly
- Start implementing Draw on Demand

Week 8(July 27 - August 2)

- Analyse the code again, for optimization in draw on demand
- Improve the approach
- Present the changes to mentors
- Again Improve the code further

Week 9(August 3 - August 9)

- Improve the code, and buffer this week to complete the incomplete stuff and things that needed improvement
- Make the Base for Dirty Rectangles Approach, in firmware

Week 10(August 10 - August 16)

- Reviewing the code written.
- Fixing the issues related to all the work done

Post GSoC Phase

- Continue working on Axiom Remote, and complete the GSoC tasks (Dirty Rectangles), and keep on improving Remote.
- Help others who are starting with Apertus and if allowed, mentor other beginner's in next year's GSoC.
- Ask mentors if you can also be involved in other projects that you are interested in, like Opencine and web viewer.

Q 1. ***What interests you most about the apertus° AXIOM project?***

Short Answer

The People Behind it, who made this Project and the organization

Long Proper Answer

I had joined Apertus back in December, with intermediate knowledge of C++ and object oriented Programming (had explained my reasoning of approaching Apertus in a mail to the organization).

I started slow and small, I was a beginner, **Metaldent** helped me with finding beginner friendly tasks, and after that helped me make my first pull request. Which embarked my first contribution to Apertus. I didn't expect such a treatment from a proper and well established open source organization. That made me stick with apertus and trying out other tasks

I started with Histogram, I mapped the coordinates, and had done coordinate transformation logic in firmware. I asked **BAndiT1983 ,** for help in the task. He explained

where i am going wrong in code, why it's not a good idea to do such calculations in Firmware. And the complete following week, explained me the concepts and all about the AXIOM Remote, even the basic things. So, It was obvious that histogram task might be tough for me at that time.

In the upcoming week, I was searching and trying out things in AXIOM Remote and wasn't able to make any significant progress. **Sebastian** had then texted me, and helped me find the task. My favorite task so far, "*The Text Input Task*" . I started with the task, and was guided by **Sebastian** at the start of the code approach, I used to discuss with him the approaches, and things I can do. I used to code them out, and **BAndiT1983** always used to help in coding, sharing various links, teaching new things.

My three month long experience, can take lot of words, but I guess the point I am trying to make is,
Everyone at Apertus has helped me grow into my skin as a developer, whether I am making silly mistakes or doing wrong approaches. They have always helped me at every stage. Once, when I was unable to understand transition algo, **Sebastian** spent 30-40 minutes explaining the concept. He is a very kind and considerate person. **BAndiT1983** has helped me alot and spend lot of time teaching me proper coding and discussing approaches**.**
These things meant and mean a lot to me, and as a student, I could not have imagined a better environment to grow.

I had sometimes wondered looking at current firmware code how someone can think so well defined an approach. I had recently seen the Opencine repository, the Qt framework and its uses, the cross platform scripting in cmake. Just WOW! I want to be Part of such work.

As a developer I want to be as good and as diverse as I can be. And here, I think I can be.

So,among other things, People behind the apertus interest me the most.

Q 2. ***As mentors and project coordinators, how can we get the best out of you?***

I try things, I can get stuck, if I can get a little hint/help, I move heaven and earth and can produce the required output.
Recently, I tried porting Visualizer to windows, whenever i used to get stuck, **BAndiT1983** had helped me at every stage with references and hints to proceed, with his guidance, I was able to Port Visualizer to windows.

So,the kind of mentoring provided by BAndiT1983 and Sebastian, is adequate and more than enough.

Q 3. ***Is there anything that you'll be studying or working on During GSoC?***

My college can give 4-5 weeks off during the GSoC duration. But I will have my final exam in the month of June or July (uncertain due to covid), which can occupy 10-12 days of my time.

I will inform my mentors as soon as my exam date sheet arrives.

I have **not** been able to work on Axiom Remote full-time due to college and other commitments. But during the GSoC period, I will have no other commitments, besides college, and my college is happening online now, and college can also grant me interim leave for 5 weeks(for GSoC) , and will be closed for 2 weeks,in the June.

So, out of 10 weeks period in GSoC. I'll get to give **40+ hours every week for 7-8 weeks**, and I'm really flexible, if my tasks or mentors require me to invest in some extra time.

Q 4. ***Are there any techniques and tools which you use to keep yourself organized?***

All of my organizational tools are old-fashioned....I use a tangible paper calendar, not an electronic one - this way if the power goes out, (we have rolling blackouts often) - I am not without my calendar! I use tablets of paper and "Post It!" notes because if I write it with my very own hand instead of a keyboard, it becomes etched in my mind's eye.

It's also very gratifying to cross something off a list... and because I enjoy this, it is incentive and motivation to be on top of time management and organization, to actually cross everything off in completion. It's a win-win.