Individual Report(z5340468)

During this term, our aim is to achieve using kria board and i2s mic to catch audio, design and implement an app basic on hardware about dealing with audios. We separate those works to 3 parts, M2 for implement hardware and load it on board, M3 for software which can generate a wave file by the data we get from i2s mic, and M4 for app which our group achieved speech recognition. Reflection on the project, I think there are a lot of Insufficient, and it’s absolutely can do better.

Milestone 2:

In the part, We are committed to using Vivado to write VHDL code for hardware, which implemented to get the 1 bit data from i2s when bclk rising edge until we get 18 bits then we combine 14 ‘0’ bits to get a 32 bits data, then load it to fifo, in the meantime, we switch channel each 32 bclk circles failing edge which is actually when half circle before we start reading first bit.

Unfortunately, when we start writing the hardware code, we have no idea what we should do and how to implement it. I don’t quite understand the FSM and FIFO part, and in this process, our group cooperation is a great failure. So, I basically did all the code by myself. And it wasn't done until the deadline. Also, as I didn’t lean 3222 and 3211 before, so I don’t know how to use a test bench, so it is really hard for me to debug.

The good news was that in week 5, I finally got a sample256 output that looked correct, and I learned how to use test bench which really helps a lot. Nearly all VHDL code in i2s-master has been rewritten, because my groupmates’ codes are total wrong, and they always have no idea how to debug.

For M2, I think I should supervise my team members and urge them to complete their tasks on time. At the same time, I also realized that it was impractical to do everything by myself. I should find ways to involve my team members more, even though they may not be able to play a role.

Milestone 3:

In this part, we need to implement read fifo data in memory, then write it to a wav file,

Most of the code is already provided, we just need to write the frame's data to the wave file and deal with the noise. I deleted zero data from the frame first, and then reverse the rest, because the direction of adding data to the hardware and the direction of reading data in c is the opposite. We then write the processed data to a wave file, so we get a clear recording.

This part went fairly well, I built a 16-bit wave file generator early on, tested it with a simple sine wave, and in mian.c wrote the data from the frame to a buffer in order to generate the wave file. Then I asked my team to change the wave file from 16bit to 32bit. But they were never successful, and later even told me that the wave file could not be stored as 32 data for musical tone. At the same time, the wave file we generated only lasted for tenths of a second even though a large transfer run was used, so I once thought it was a problem of reading fifo in hardware and c program, so I invited our team members to find out the problem, but now it is obviously my fault. We should prioritize the generation of 32-bit wave files.

Therefore, before lab on Tuesday, I completed the 32bits wave file generator, and my team members also learned about the possible data inversion problem in office time. The best news is that in lab, we found a fatal problem in hardware during communication with other groups. We originally started reading 18-bit i2s-dout on channel, but it actually start half a cycle later.

As long as we know where the question is, then we have a direction to get closer to the answer. I asked Stanley and Asher dealing with date reversal, and I debug hardware code. Unfortunately, we did not achieve the goal on Tuesday, there were still bugs, and we had to verify whether the data after our inversion was correct, so Asher and I appeared in the lab again on Wednesday, and I finally realized the correct data was passed into the wave file generator, but the wave file we generated is still incorrect. Finally in my persistent efforts, I found that one was using the wave file generator to correct parameter errors, and finally at 4pm, I got a clear recording .

In m3, our group cooperation was still very advanced, but my team members participated in it and at least provided some useful things. I also found that Asher, the only one in our group with a windows laptop, couldn't connect kria to the computer yet, but he always asked me how to do it, so I taught him. However, in this process, I also made mistakes. I led the team members in the wrong direction and wasted our time. At the same time, our cooperation was obviously insufficient, and the content completed by the team members was too little, which needed more reasonable distribution.

Milestone 4:

In this part, we planned to implement two functions, audios mixing and speech recognition. Then we want to use speech recognition to let use voice control our audio mixer.

During the process, I implement multichannel first, Stanley finished a format for speech recognition by reading speech from a wave file. Then I force on using the code Stanley gave to me to implement speech recognition on kria board and add some functions, designing the wake words, Stanley and Asher force on audios mixer.

As we can get a really clear wave file from M3, so speech recognition will not be too difficult, I let my python code run 800 frames sample256 in subprocess then until the new wave file generate, convert the sound in the wave file into a string, and read whether the keyword of the function I designed appears in it. If a function is activated, this function may search for the variable it needs in the string and print the result into the terminal. Finally, I realized to get time, get date and get the weather of the city in city library of yesterday, today and tomorrow (I don’t have api for history weather, so the query result is not changed, but the data\_key will be verified by print to the terminal), a timer running in sub thread, and sleep and wake up our voice assistant. In the meantime, our mixer has some problems, after we mixed the wave file, it will be just noise, but I think we are able to finish it if we have enough time.

Overall, M4 is going very well, because we get a clear wave file in M3, and there are plenty of learning materials on the web for speech recognition using python. The biggest challenge we were facing is porting the speech recognition system that accesses the microphone on the computer to the kria board and changing the input to wave file. In terms of group cooperation, we have also improved, and everyone has participated in it.

Summary：

Reflecting on the project, it's evident that navigating the complexities of hardware and software integration presented a steep learning curve. The journey was fraught with challenges, from grappling with unfamiliar concepts like VHDL coding and FIFO mechanisms to handling the intricacies of audio processing through an I2S microphone. Notably, a lack of prior knowledge in foundational courses added to the hurdles.

The process underscored the vital importance of teamwork and the pitfalls of over-reliance on individual effort. There were significant moments of triumph, such as the successful generation of a clear recording, which marked a turning point and illuminated the path forward. This achievement, along with the realization of speech recognition functionalities, demonstrated the team's potential when focused and coordinated.

However, the insufficient group collaboration and misdirection at certain stages highlighted the need for better project management and more effective distribution of tasks. The experience has been a testament to the value of perseverance, continuous learning, and the iterative nature of problem-solving in engineering projects. Ultimately, despite the setbacks, the progress made lays a foundation for future improvements and the refinement of both technical skills and team dynamics.