Aung Myin Kyaw (Max) weekly Research Progress Report

Date : 14/10/2018 to 20/10/2018

- **Scope of the work:**
  - ✓ **Troubleshooting of the issue with collection of data using CCS100.getScanData() function**
  - ✓ **Scanning of the data and collection in the excel format.**
  - ✓ **Understanding of the integration and implementing a a variable integration time for the device**
  - ✓ **Understanding of Thread.sleep() function**

➢ Research Progress in last week

- Integration time is the time that the CCD is allowed to collect photons before passing the processed data over to the processor. The range of integration time that can be set to CCS100 is 10 µs - 60 s (Thorlabs CCS-200 Spectrometer User Manual).
- There is no set of most suitable integration time for different device as integration time varies base on the situation. For very bright light sources, low integration times are required while for the weak sources, longer integration is suitable. Though higher integration times results in higher peaks in the measurement data. In addition, saturation can be occurred when the integration time is too large. Integration time can be use to varies the Signal to Noise Ration which is ≤2000:1 (Thorlabs CCS-200 Spectrometer User Manual).
- Integration time is different from data transfer speed.

➢ Research Progress in last week

- Manage to troubleshoot the issues that I was facing for the past two weeks. This was due to my lack on understanding how the CCS spectrometer works. Especially the understanding of the integration plays a big part in solving the issues.
- Initially, I set the integration to 0.1 second which is the minimum integration time that can be for the spectrometer. (Thorlabs CCS-200 Spectrometer User Manual)
- Though I was able to get the data, it was harder to understand as the collection of the photon time is too small.
- Thus, I have increased the integration time to same as Thorlabs OSA program which is 2.67 sec.
- This allows me to collect the data similar to what I would collect using Thorlabs OSA. However, this also requires me to give the device close to 3 sec to process the data before using the .getScanData() to read the data collect.

➢ Research Progress in last week

- I took several steps to solve the issue. For the first step I have hardcoded a timer to sleep the thread or program for the define set of time. For this I used Thread.Sleep(delayTime);
- Through my research, I have found out that this particular method is not advisable as this method will pause my program for the set of time that I have set. This will become an issue in future when I convert my program to UI base as it will lag the whole program.
- There are other method which is using Task.Delay().
- Task.Delay acts in a very different way than Thread.Sleep. Basically, Task.Delay will create a task which will complete after a time delay. Task.Delay is not blocking the calling thread so the UI will remain responsive. Behind the scenes there is a timer ticking until the specified time. Since the timer controls the delay, we can cancel the delay at any time simply by stopping the timer.
- However, at this moment I will continue to use Thread.Sleep since I am testing the workability of the program.

➢ Research Progress in last week

- Inserting the pause for 3 second > integration time allows me to scan, process and collect the data.
- However, as I have mention earlier there is no set of perfect integration time. Thus, it is necessary for the program to have a ability to let the user choose different integration time based on the environment.
- To test such function, before setting up the spectrometer, program will ask the user to key in the desire integration time in double variable format. Using the input provided by user, program will set the integration and read the integration time again for confirmation.

➢ Research Progress in last week

- To be able to use Thread.Sleep(); function

  - It can only process integer values and integration time that user provided is in double. Thus, I casted to convert the double to integer value.
  - In addition, I have round up the integration time as it is important for the Thread.Sleep() to pause longer than the integration. The reason behind rounding up is that if dealyTime is shorter than integration time it will be collecting that data which has not fully processed.
  - The outcome is times *1000 to convert from second to millisecond and assign to dalayTime integer variable.

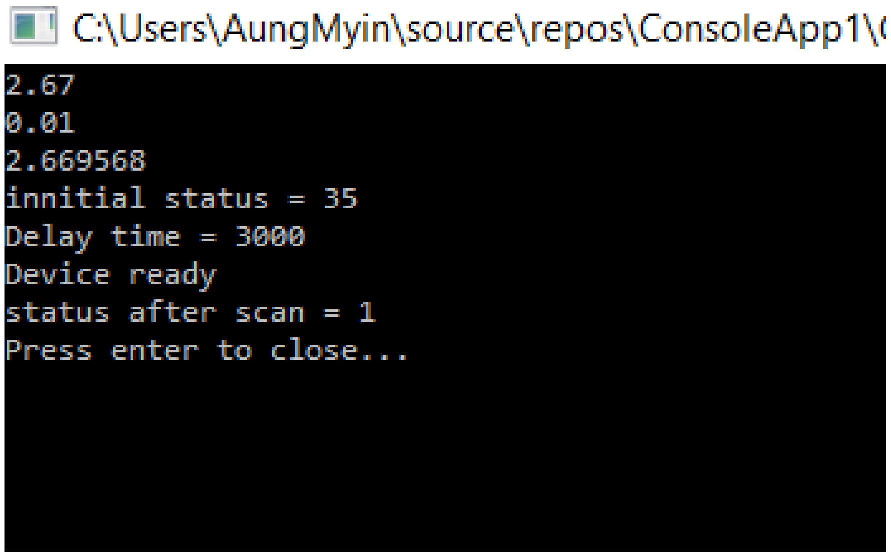
- delayTime = (int)Math.Ceiling(integration_Time) * 1000; ;

Casting

Round up

➢ Research Progress in last week

- Below is the screenshot of the program after execution



C:\Users\AungMyin\source\repos\ConsoleApp1\

```
2.67
0.01
2.669568
innitial status = 35
Delay time = 3000
Device ready
status after scan = 1
Press enter to close...
```

- In addition, I have also attached two excel files which are the data collected using my program.

➢ Research Progress in next week

- After the successful debugging, moving forward will be using the value collected to printing of graph.
- In addition, testing out the continuous scanning function.