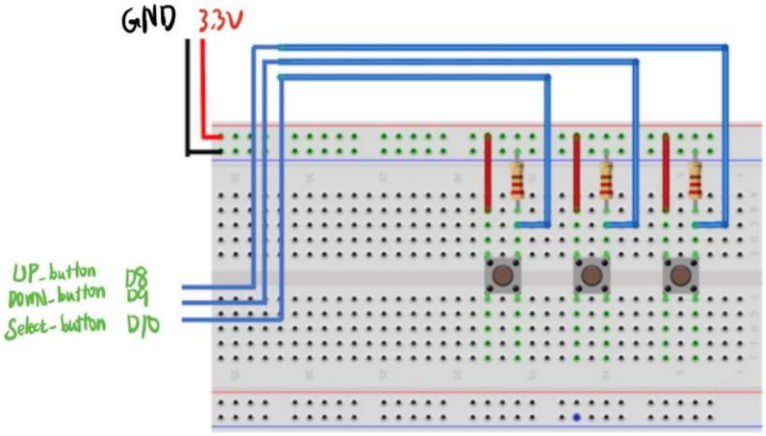
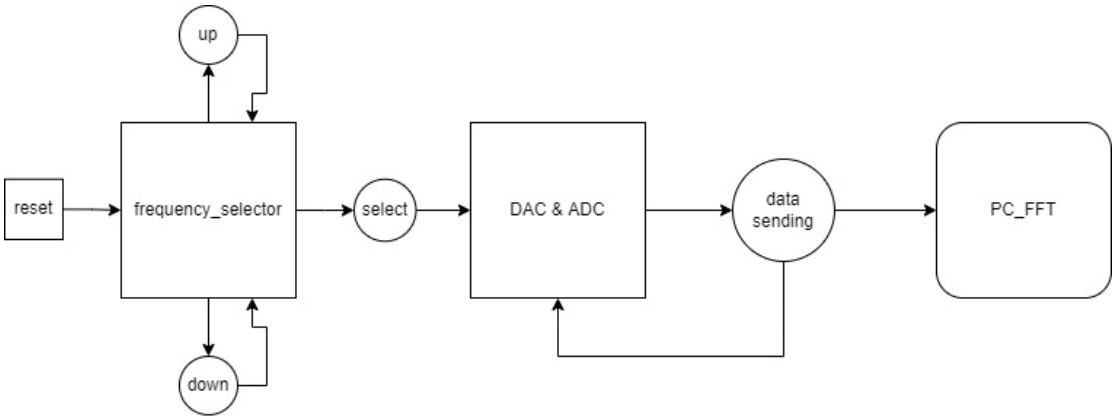
In homework 1 we use what we learned from lab1~4 to implement the spec.

(1) Implementation and Algorithm



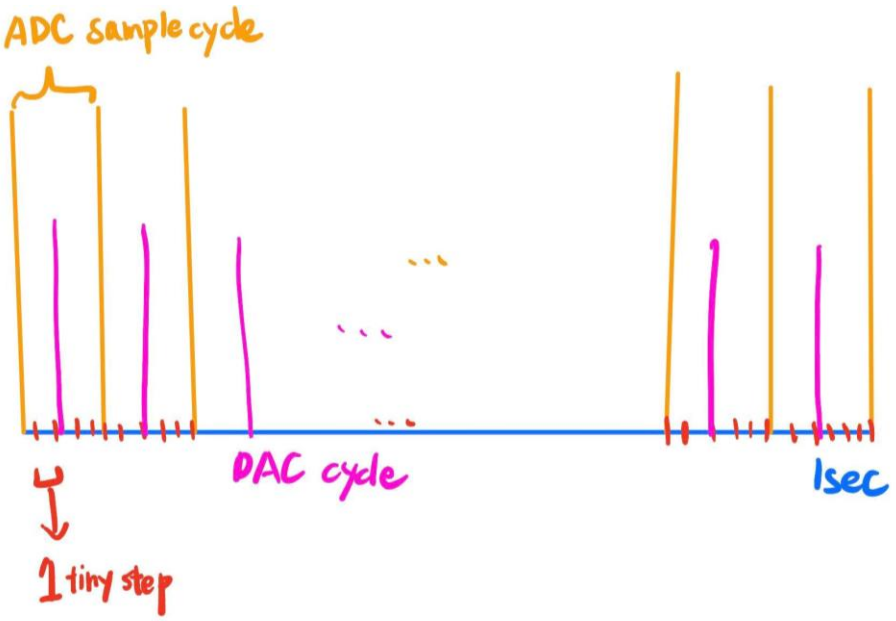
Function	Pinout on mbed
UP	D8
Down	D9
Select	D10
ADC input	A0

In this part we first set up the circuit with buttons, resistor, and wire.



Next we construct our flowchart, after resetting the system, we stay in frequency Selecting mode, use UP button and DOWN button to choose target frequency I Implement button debouncing method to avoid bouncing with buttons.
After pushing the SELECT button, we go in to DAC & ADC function, ADC will catch data and store them into an array while DAC continuously generating desire frequency with desire waveform. To output data in array, user must press the SELECT

button and data will transfer to FFT, analyzing data in frequency domain.



To implement ADC & DAC simultaneously, we take this algorithm. We know that FFT need samples for power of 2, which we can use 8, 16, 32… I take 256 samples in homework 1 since it can analyze up to 128hz (of course there may be some limitation in real world.) So we need

$$\frac{1\ \text{sec}}{256\ \ samples} = 3.90625\ \ ms$$

But the function

Thisthread::Sleep_for(int)

Only accept integer, so I take another function that is more precisely

Wait_us(int)

Which is more convenient to implement, though we have a smaller function to make time pass, we can't do anything when we call wait_us function, So I take the algorithm that I had learned from Logic design lab, I take counter to implement DAC & ADC more precisely. I divided 1 second into 25600 tiny pieces, so we need each time duration is

$$\frac{1\ \text{sec}}{25600\ \ step} = 39.0625\ \ ns$$

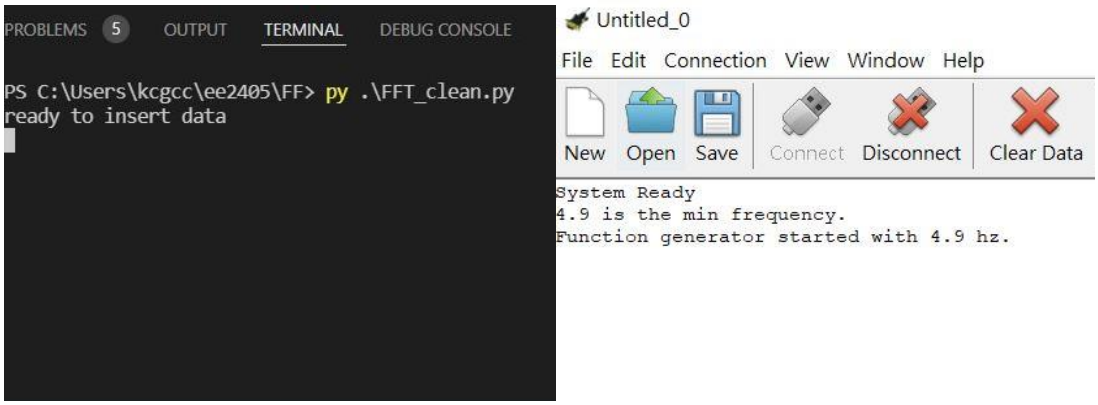
Every 100 steps we trigger ADC once, then in 1 second we will have 256 samples which fit our FFT desire requirement.

Then with different desire steps we can create waveform in different frequency, with high resolution and high speed.

Now we have most of the desire spec, we implement digital low-pass filter by following the tutorial, we take

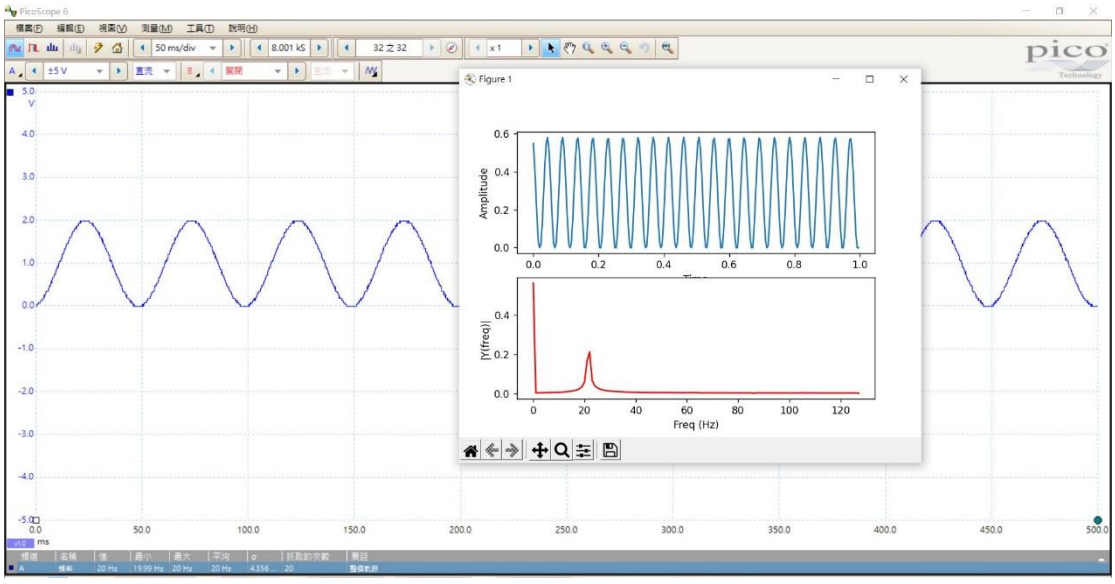
$$y_{lp[x]} = \frac{y[x] + y[x-1]}{2}$$

Also I implement FIR to check more in digital signal processing and add more user-friendly information to tell the system is ready to receive(Python will send a signal) or transmit the data(If FFT array is not ready, pushing button won't affect).

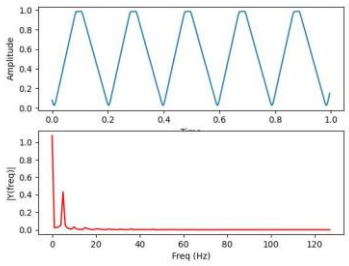
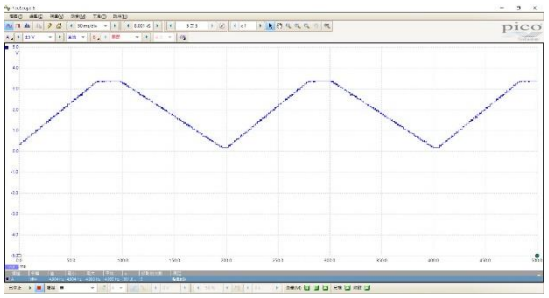
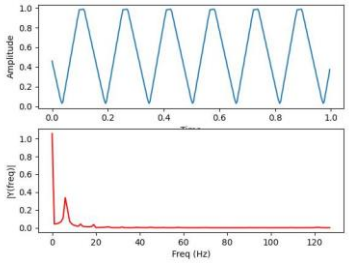
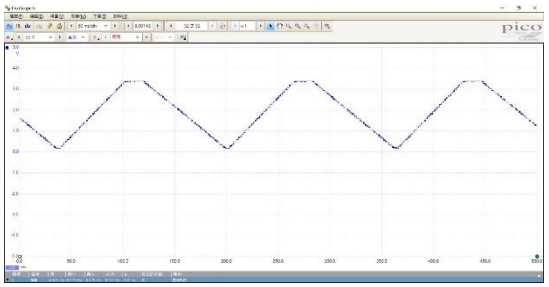
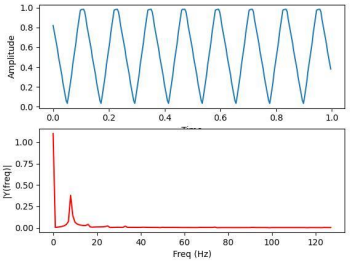
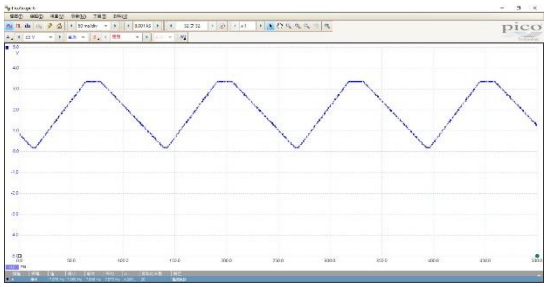
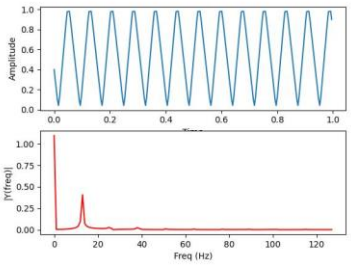
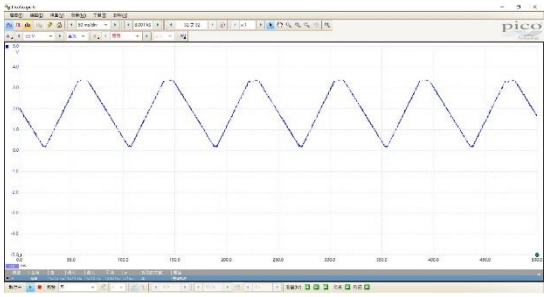
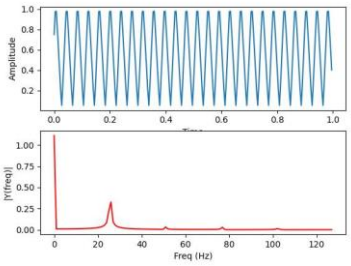
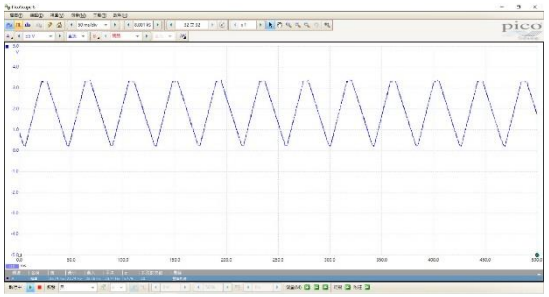


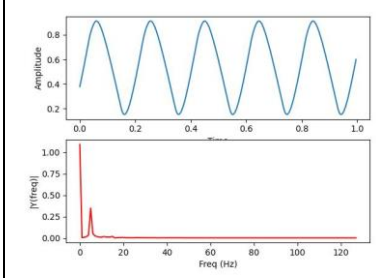
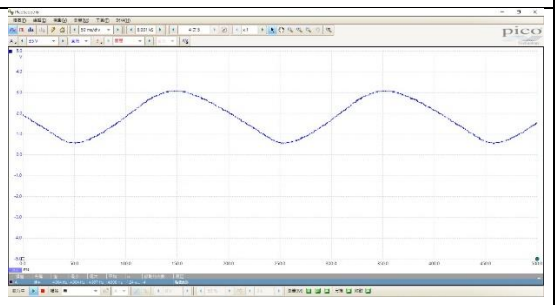
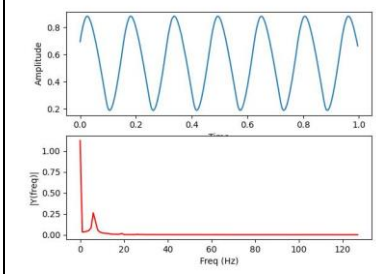
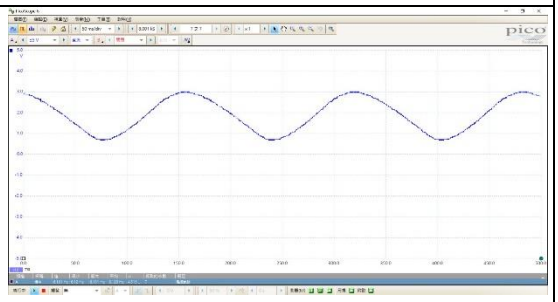
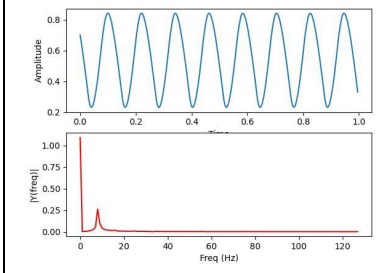
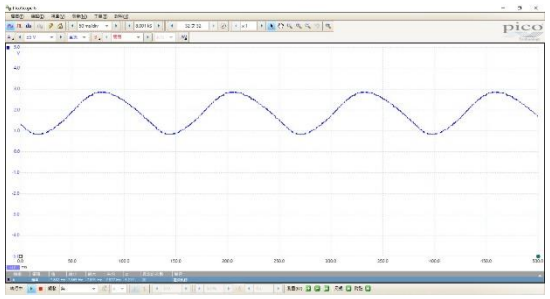
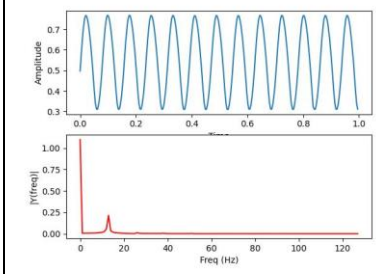
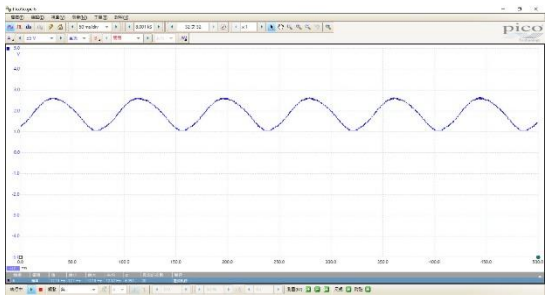
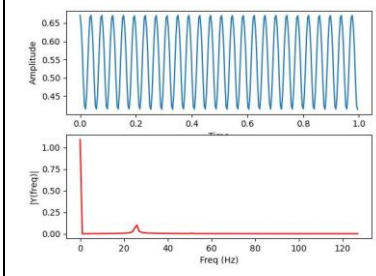
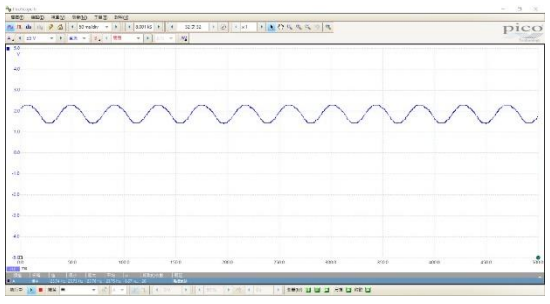
(2) Validation and Result

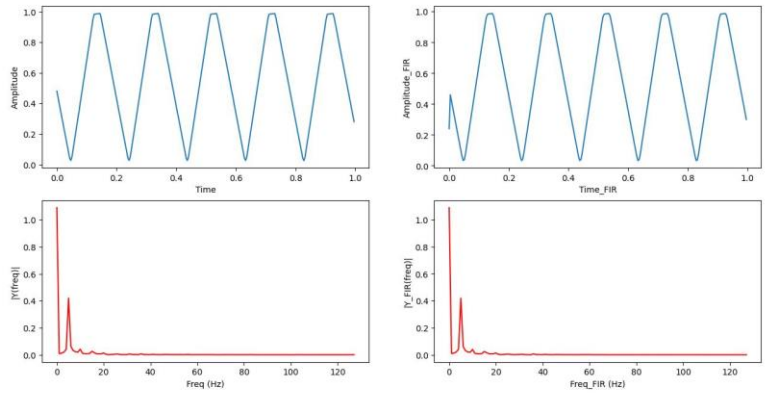
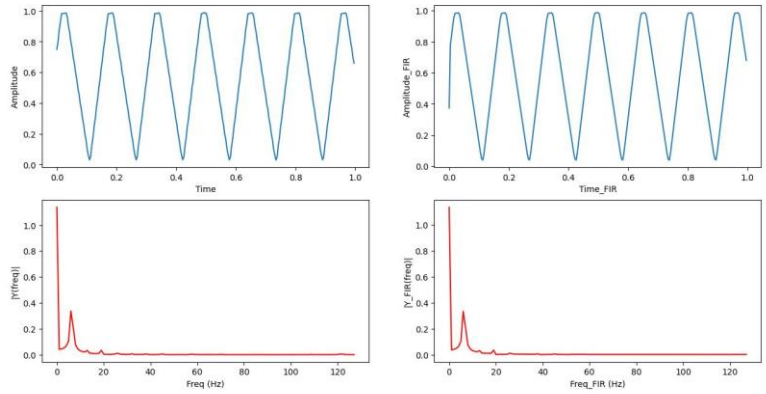
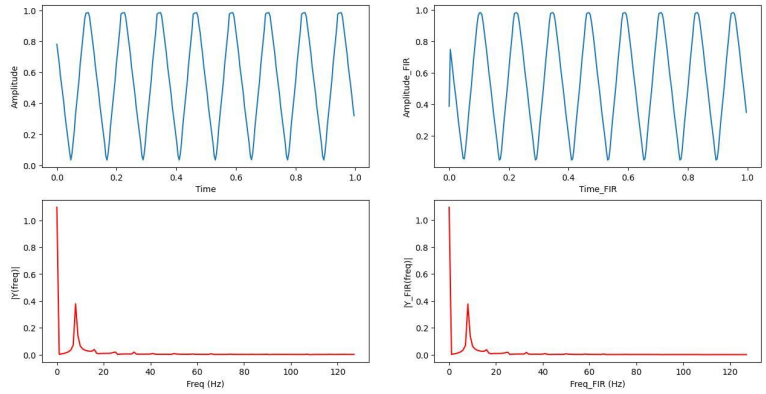
Most important is that we need to check FFT is working properly, so I use PicoScope to generated a 20hz sine waveform, read data from ADC on mbed while the ADC & DAC part is working simultaneously, luckily, the waveform is quite correct and the frequency analyzing is also precise as picture shown below.

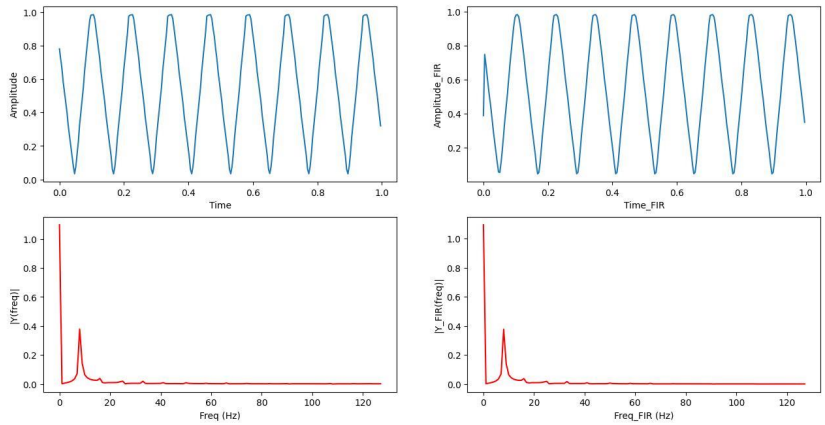
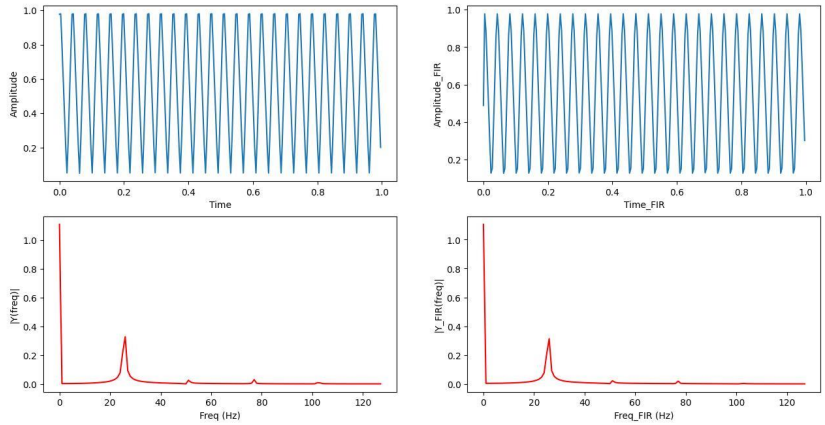


And all the results are shown below. (5 kinds of different frequency)

Signal before Low-Pass filter.		
4.9hz		
6.12hz		
7.88hz		
12.12hz		
23.75hz		

Signal after Low-Pass filter.		
4.9hz		
6.12hz		
7.88hz		
12.12hz		
23.75hz		

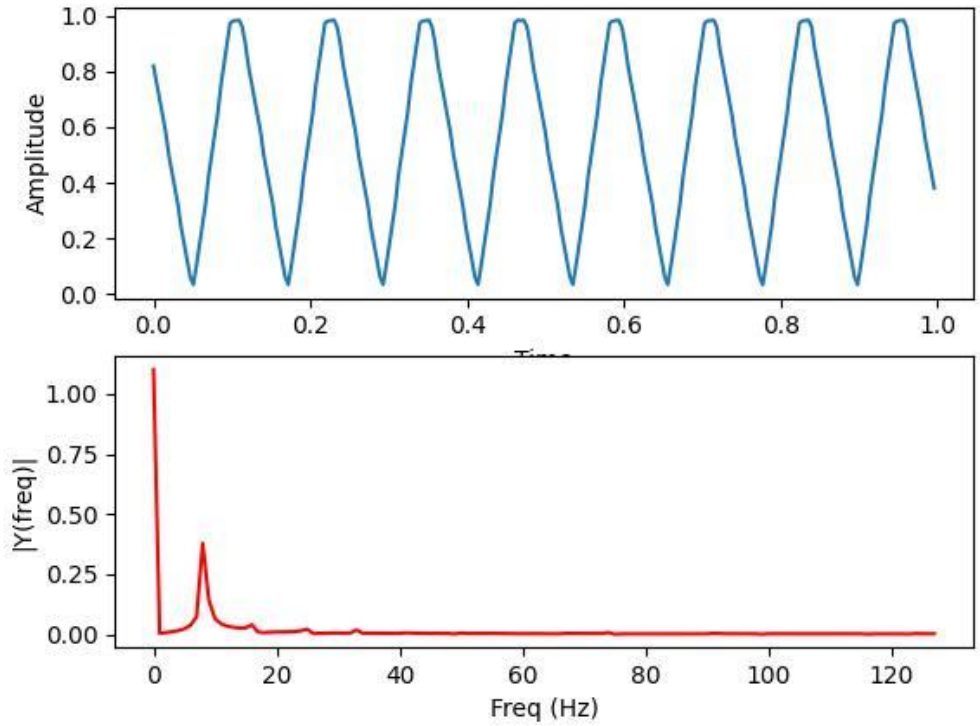
Signal after Digital Low-Pass filter.	
4.9 hz	
6.1 2hz	
7.8 8hz	

12. 12 hz	
23. 75 hz	

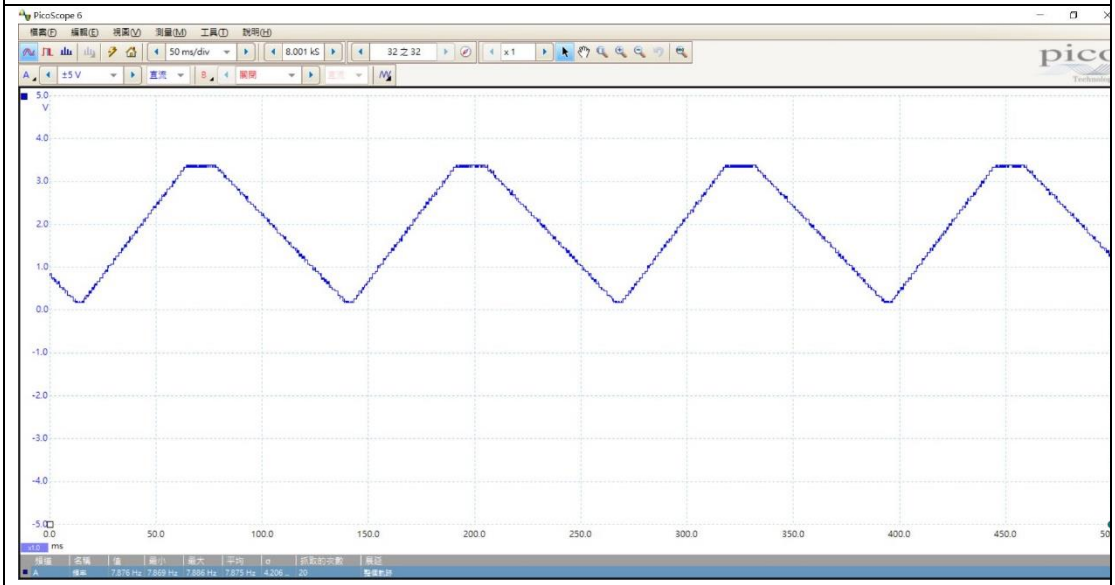
The results meet the specs, and I put frequency close to the cutoff-frequency

$$\frac{1}{2\pi RC} = \frac{1}{2\pi * 2000 * (10 * 10\text{^} - 6)} = 7.95 \ \ hz$$

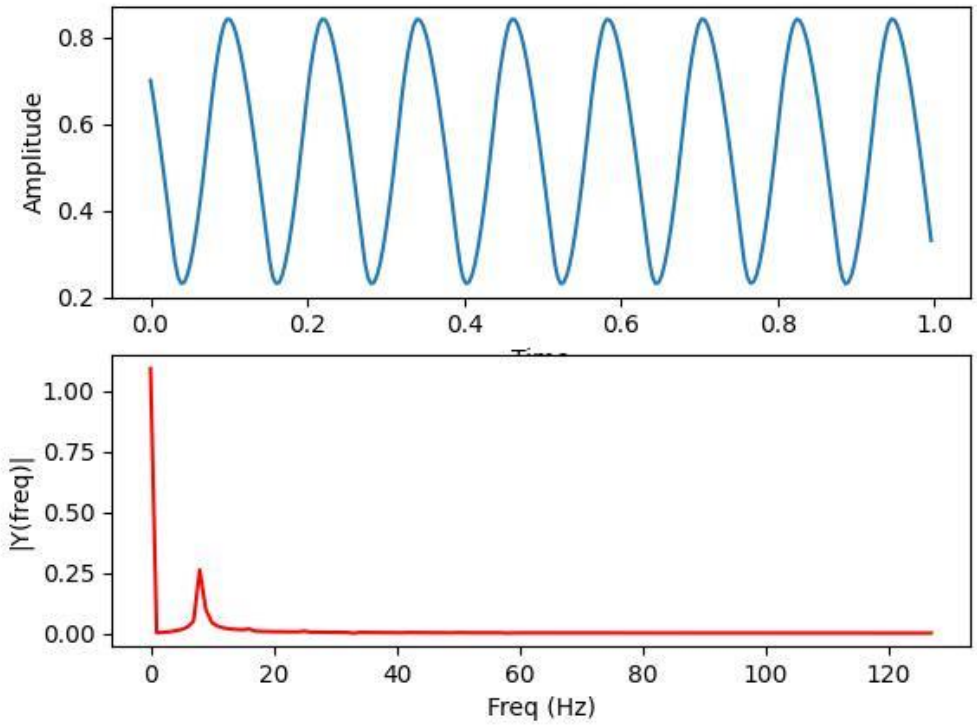
All of its result to show the detail.



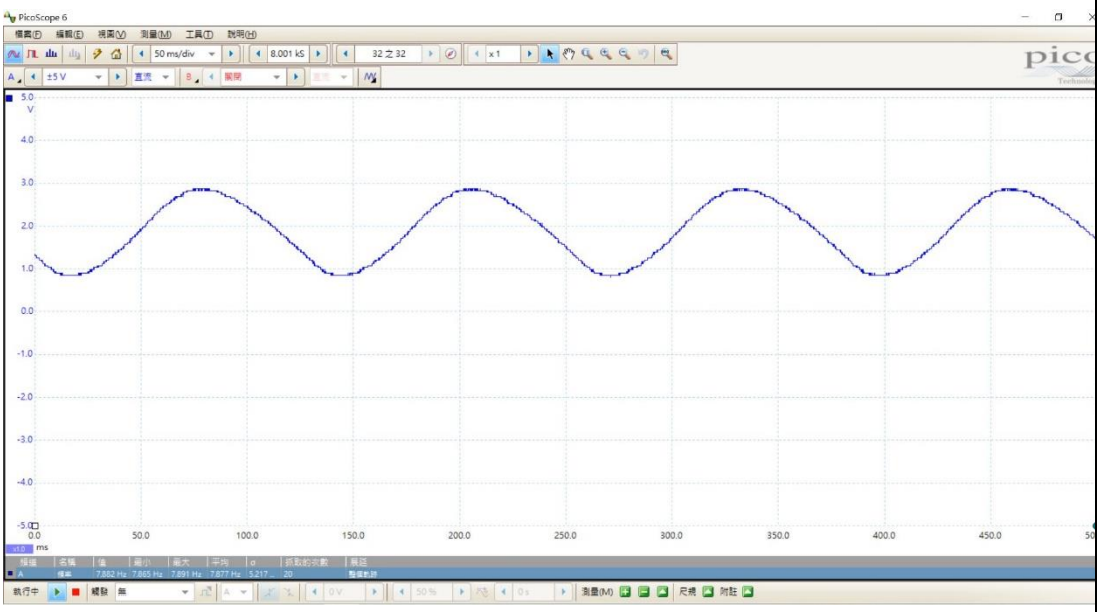
Before low-pass(FFT)



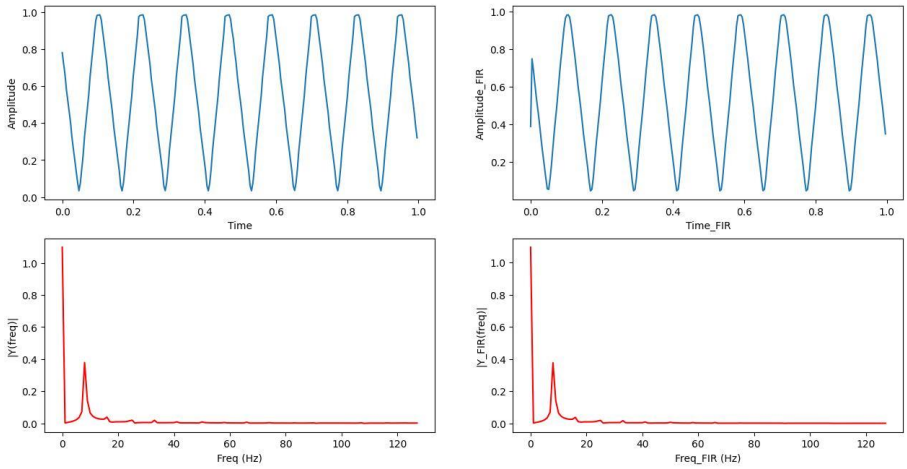
Before low-pass(Scope)



After RC-low-pass(FFT)



After RC-low-pass(Scope)



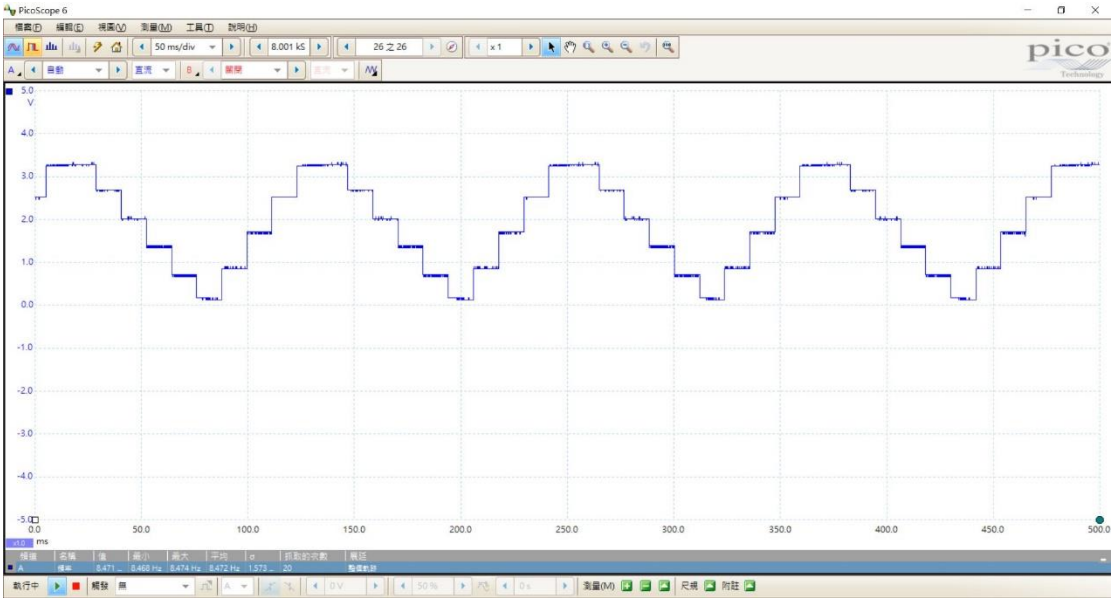
After Digital-low-pass(FFT)

(3) Encounter issue

In this homework, I encounter tons of issue, I came up three kinds of different algorithm to implement.

The first one is ADC and DAC working in parallel, but I can't figure out how to implement this now.

The second one is similar to the method I take in this homework, it also take the concept of counter, but the frequency is constrain to FFT sample rate which is 256hz . The highest frequency DAC can create is 25.6hz with waveform divided into 10 parts as shown below.



If it take higher resolution (divided into 100 parts), the result is 2.56hz, which is too slow to see the result. Besides that the frequencies are 25.6hz, 25.6/2 hz,

25.6/3 hz …. which also be a restriction.

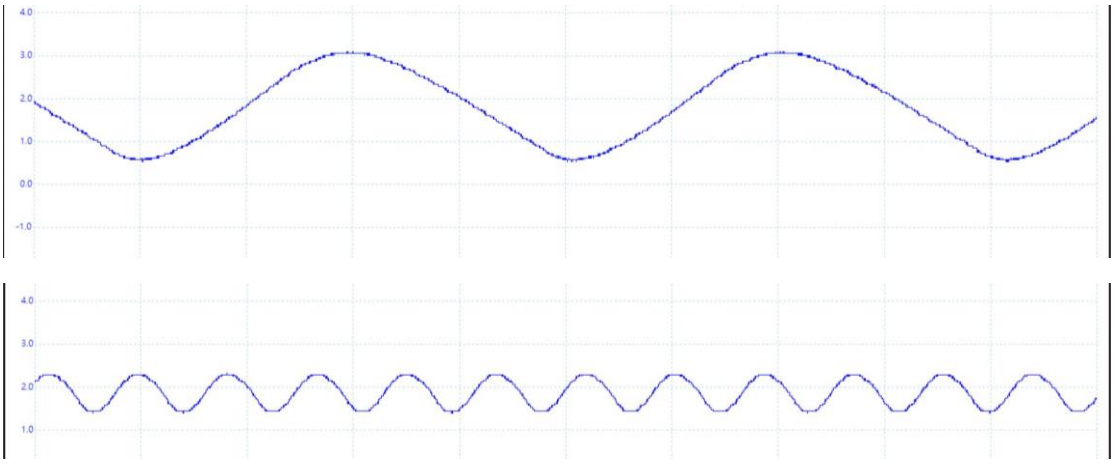
So I came up the third method which I finally take it to implement homework 1. Also I tried to change FFT spec, for example, change its time duration but I think it's beyond my ability, maybe I should take some advanced course in signal processing.

(4) Discussion

(a) I think that homework is harder than lab1~4, (maybe lab1 is also hard ☺) Button pressing part is quite simple, but the part of ADC & DAC is not. It took me a lot of time to try different kinds of method. Also FFT part is also a big problem to us, I discuss with my classmate the method to do homework 1, they said they change samples to 100 in FFT. 100 is not power of 2(I think it could go wrong, not quite sure about that.) Besides that, if take 256 samples, max frequency to reach is 128hz in ideal situation and the time we delay is not quite precise. Fortunately, I use FFT in logic design lab (though I didn't know how it work inside), a little bit too hard for us to do FFT analysis, but I think it's quite interesting.

(b) I also see the tradeoff between resolution and frequency from method 2 I tried, which is mention in DAC and ADC lecture on Wednesday, the concept is quite important to future learning.

(c) Also I found that there is a voltage drop with high frequency after passing through the RC filter, I think it's because of the impedance we learned from electric circuit , microelectronic and signal & system, the concept meets bode plot, distortion, frequency response and more.



(d) I find that FIR is quite interesting, maybe I will take some time reading it and take advance lecture with that to learn how it work, also the digital low-pass filter didn't significantly effect the signal around the slope, but I notice that the corner is smoother than input signal, this is the concept of interpolation mentioned in signal & system, this homework let us see many different

phenomenon told in class, I had learned a lot.