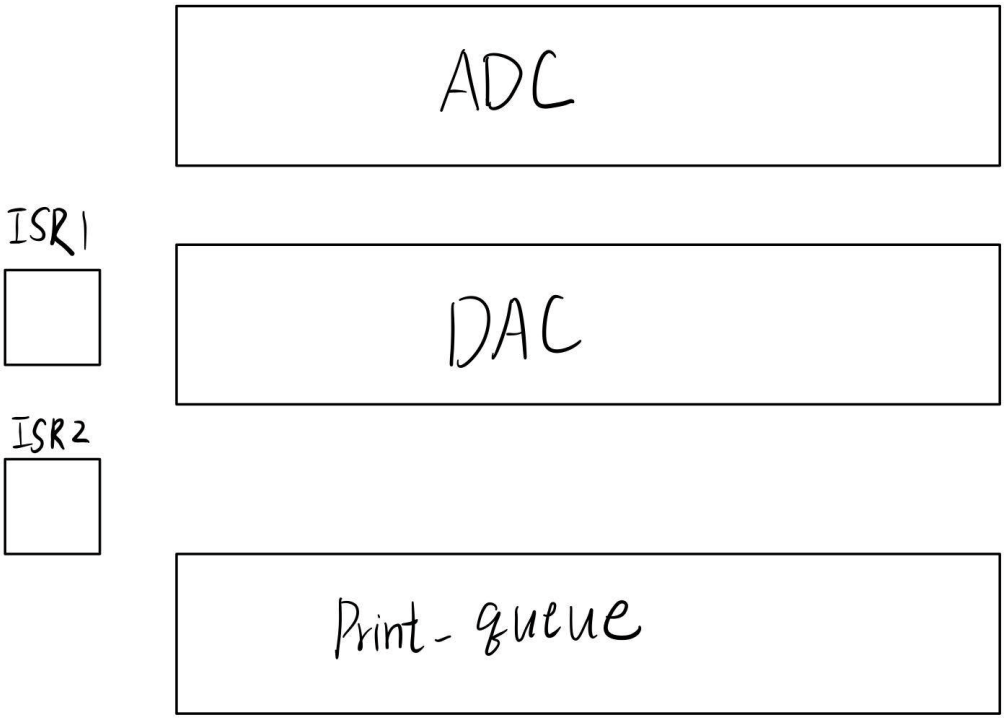
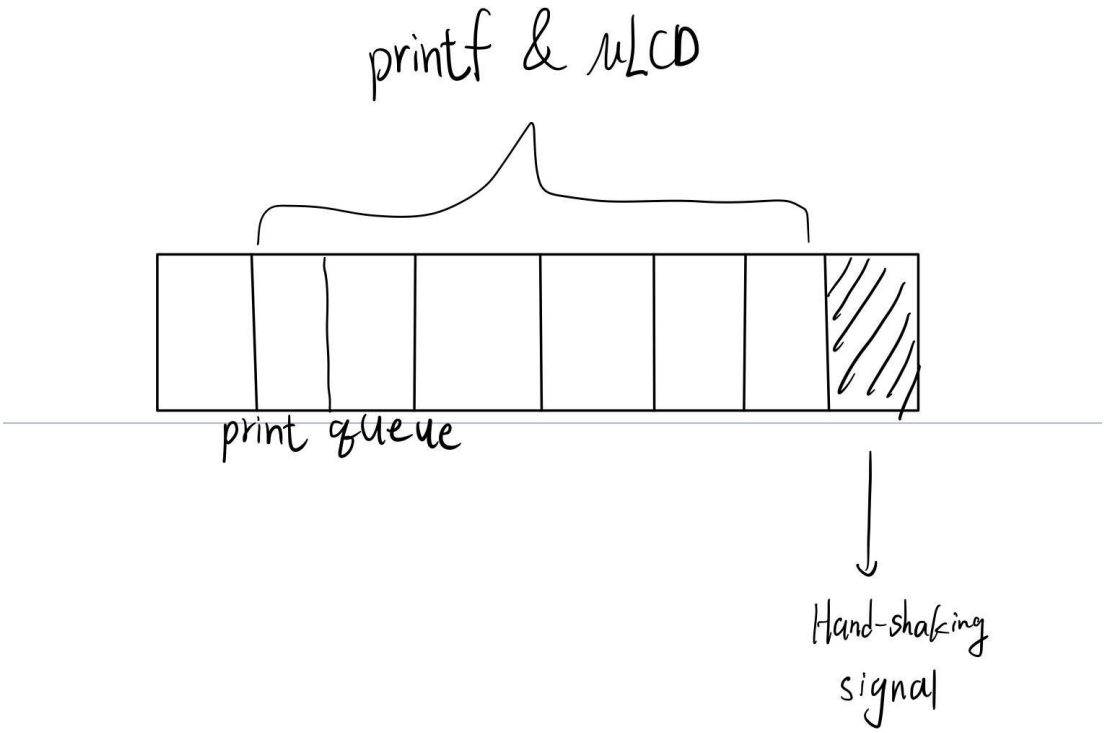
In homework 2 we use thread to make homework 1 easier to implement.

(1) Implementation and Algorithm

We have tried bare metal in homework 1, this time we use multi thread to implement DAC and ADC simultaneously. The main system structure is shown below



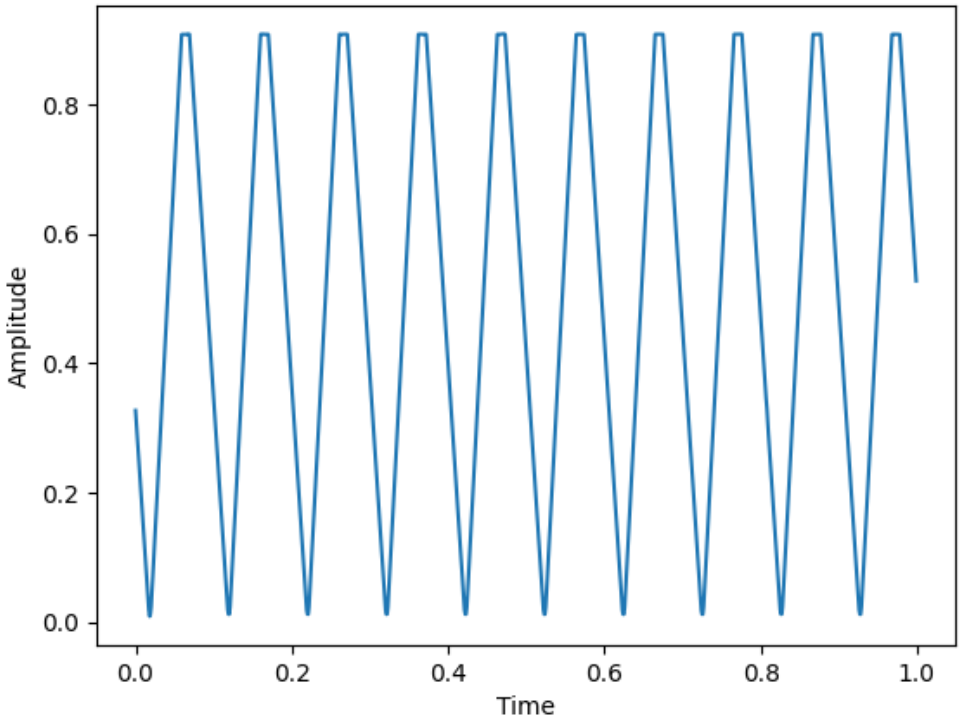
We have 3 thread and 2 ISR. For thread_ADC it produce waveform, thread_DAC for sampling and thread_print_queue to tackle the message of UART and uLCD. ISR will change a global variable "GenWave" , when it is true the waveform is produced and being sampled. When it is false, system will transmit data by UART interface once.



Also, a hand-shaking method is added into the system to ensure its stability. To start producing waveform, the system will ensure that every task related to UART and uLCD had been done, which may cause unstable when waveform is generating or sampling. Also the system will deposit some data before the system is stable and keep reading in data to make the frequency stable.

(2) Validation and Result

After sampling our waveform, we send data through UART and show the waveform with python. Since we don't need to implement FFT this time (We asked Professor. Liou before). So I generate one data and sample one data this time. And the result is shown in the figure.



Also the message on uLCD and Terminal are shown below.


uLCD Loading...
94V-0
E481957
uLCD-144G2


94V-0
E481957
System Status:
Waveform is Genera
ting.
OLED-14402


94V-0
E481957
System Status:
Interupt Detect.
OLCD-1440

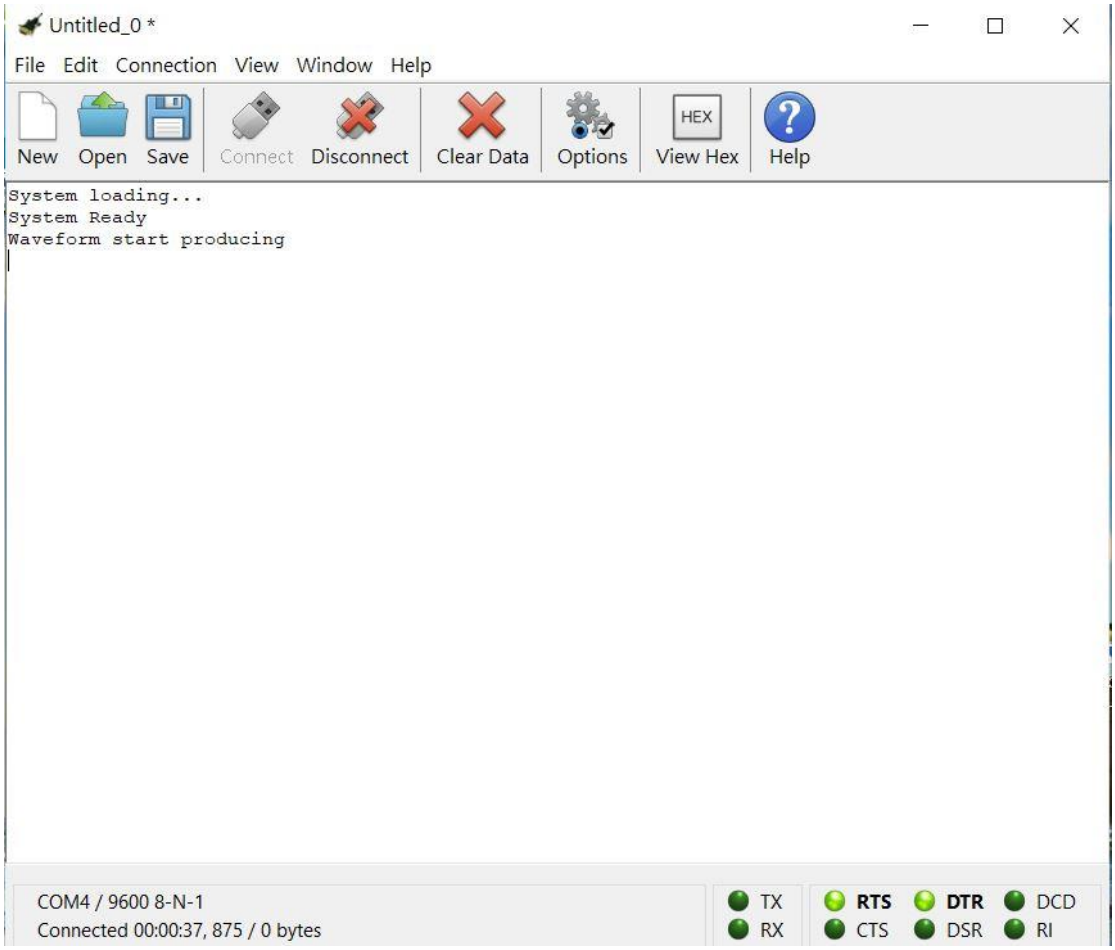

94V-0
E481957
System Status:
Data is transferin
g.
OLCD-144G2


94V-0
E481957
System Status:
Data transfering i
s finished.
OLCD-144G2

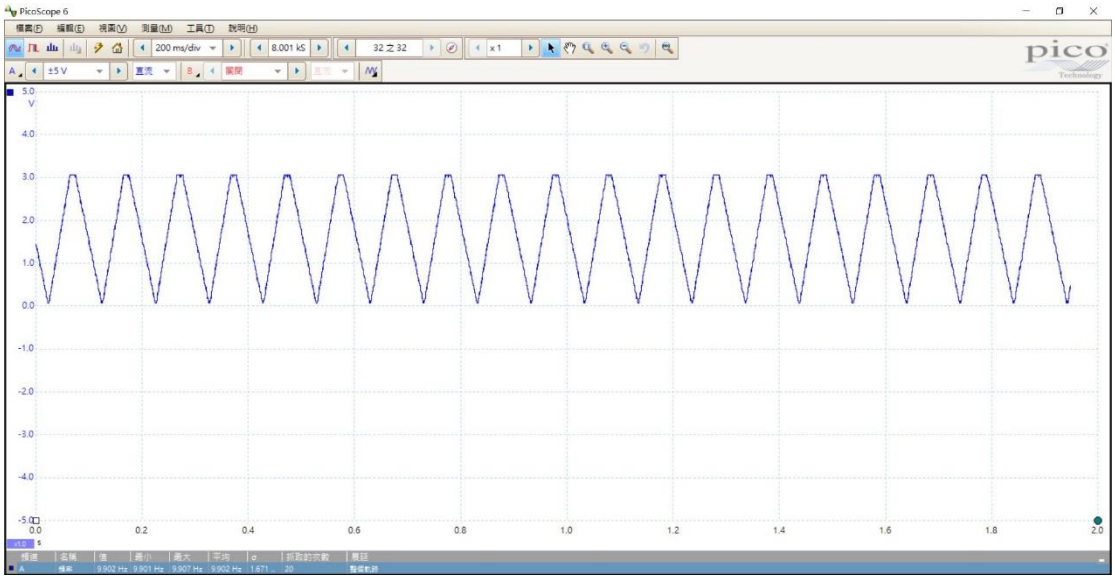

94V-0
E481957
System Status:
Interupt Detect.
OLCD-14402

The pictures shown above is a routine of system.

Initialing, waveform generating, Interrupt detected, Output data, Finished data transmitting, Interrupt detecting and back to waveform generating.



And the terminal will show the initializing message too.



And here's the waveform measured form the scope, which is what we expected to see.

(3) Encounter issue

The frequency is restricted to mbed system. Since it's minimal unit is ms in thread, which means the fastest frequency we can reached is 1000hz, however, the waveform we take need to cut into 100 pieces, which lead to a 10hz, besides that there are another thread of waveform sampling running simultaneously. It will also disturb waveform generating and make frequency go down.

Also, the hand-shaking is needed since we put our task into the queue, most of them take quite a lot of time. To avoid disturbing the best way is wait until the system tackle out the tasks and then resume waveform producing, which solve the problem.

(4) Discussion

The board we used is only single core. The benefit of thread is not quite significant. Restricted to our system, the performance is worse than bare metal, which can reach almost 166 hz in waveform generating. But the homework is still a great practice. The ISR make system more flexible and the thread make the task more clearly.

Also, though I use eventqueue to tackle the uLCD and UART. It still need a hand-shaking to ensure stability of the system, it worked same as the bare metal does. But if there is multi core I think the problem may be solved.