# 3 Timer experiment



## 3.1 the experimental goals

This lesson is mainly learning K210 timer function.

The present experiments the reference code path is: CanMV\03-Hardware\timer.py

## 3.2 experimental procedure

Module factory firmware has been integrated Timer Timer module, if you download the other firmware, please burn back to the factory firmware and then perform the experiment.

1. By machine import Timer.

```
from machine import Timer
import time
```

2. Create a Timer object, and the name of the timer.

```
timer = the machine. Timer(id, channel, mode=Timer. MODE_ONE_SHOT, period=1000,
unit=Timer. UNIT_MS, callback=None, arg=None, start=True, priority=1, div=0)
```

Parameters of content:

- `id`: Timer ID, [0~2] (the Timer. TIMER0~TIMER2)
- `channel`: Timer, channel [Timer. CHANNEL0~Timer. CHANNEL3]
- `mode`: Timer mode, `MODE_ONE_SHOT` Or `MODE_PERIODIC` Or `MODE_PWM`
- `period`: Timer cycle, after starting the timer `period` Time, the callback function will be called, (0,~)
- `unit`: Sets the cycle of the unit, the default bit milliseconds `ms`, and `Timer.UNIT_S` Or `Timer.UNIT_MS` Or `Timer.UNIT_US` Or `Timer.UNIT_NS`
- `callback`: Timer callback function, defined by two parameters, one is the timer object `Timer` And the second is in the definition of the object is the desired parameters to pass `arg` More please `arg` Parameter explanation

> Note: the callback function is the interrupt is called, so in the callback function, please do not take too long time and do dynamic allocation of Switch interrupt operation

- `arg`: Hope passed to the callback function parameter as a callback function as the second parameter
- `start`: Whether the object was successfully constructed immediately after the start of the timer, `True`: Immediate start; `False`:Do not immediately turn on, you need to call `start()` Function to start the timer
- `priority`: Hardware timer interrupt priority levels, with a particular CPU related, in the K210, take values in the range[1,7], the smaller the value the higher the priority
- `div`: Hardware timer frequency divider, the value range is[0,255], the default is 0, clk_timer (timer clock frequency = clk_pll0 (Phase-Locked Loop 0 frequency/2^(div+1)

3. The new timer callback function on_timer, inside print debugging information. And set the timer every 100 milliseconds called once on_timer。

Note: the callback function is the interrupt is called, print debug information will take longer CPU time, here just as in the demo, the actual development process do not interrupt the printing of debugging information.

```
def on_timer(timer):
    print("This is on_timer callback")

timer = Timer(the Timer. TIMER0, Timer. CHANNEL0,
    mode=Timer. MODE_PERIODIC, period=100,
    unit=the Timer. UNIT_MS, callback=on_timer, arg=None)
```

4. Create a new last_time variable for storing the current system time stamp, the units are milliseconds.

```
last_time = time. ticks_ms()
```

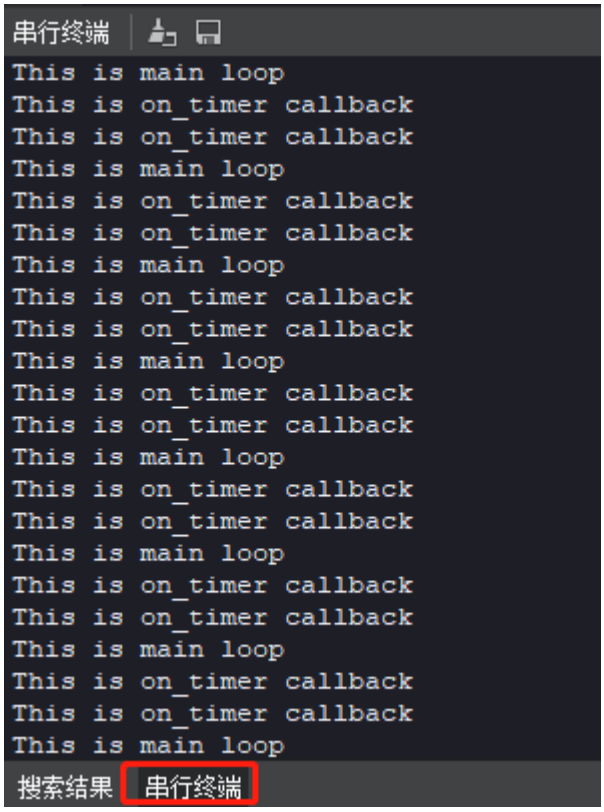
5. Create a while loop that lets the system every 200 milliseconds, print a debug message. And at the end of the program or exit the timer is shut off.

```
try:
     while True:
         if time. ticks_ms() - last_time >= 200:
         last_time = time. ticks_ms()
         print("This is main loop")
except:
     timer. deinit()
     del timer
```

## 3.3 experimental results

Connect the K210 module to the computer through the microUSB data cable, CanMV IDE click the connect button, after the connection is completed click the Run button to run the routine code. You can also download the code as main.py and run it in the K210 module.

Point the opening at the bottom of the `Serial Terminal`,You can see the print out debugging information. Since the timer is every 100 milliseconds of the print data once, whereas a while loop is for every 200 seconds printed data once, so the final look of the effect of each print once the loop information, it will print two times the timer information.




## 3.4 the experiments are summarized

Use CanMV IDE, with the factory firmware write a good MicroPython syntax, so that the control K210 timer very easy, just by the simple configuration of the timers, set the timer for the channel, period and the callback function, etc., you can call the normal timer, to avoid the timer callback function in the processing takes too long things should be as simple and fast process to finish things, and finally at the end of the program when you need to call the deinit function allows the timer to restore the default, otherwise the timer callback function does not stop automatically.