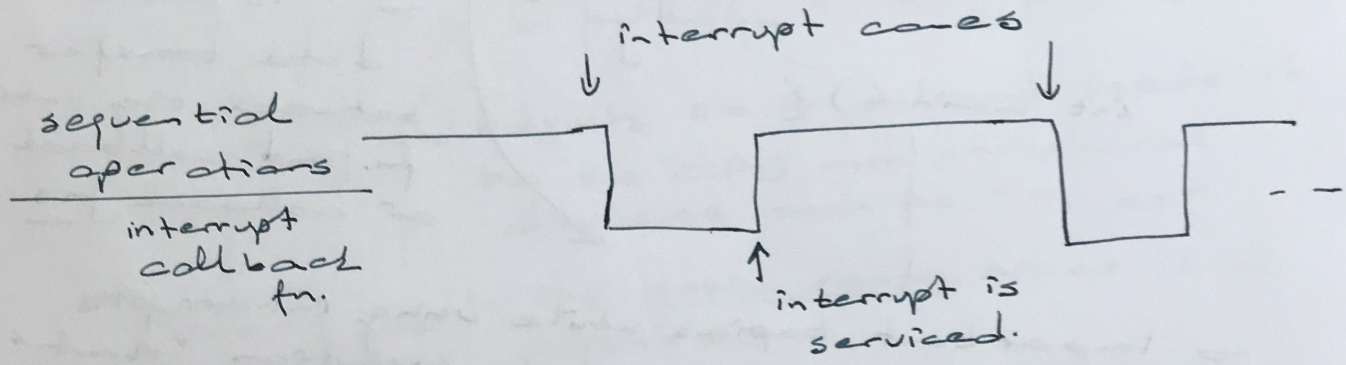
# CSE 2037, Week 11

- Interrupts, Revisited
  - Power management
  - Multiple interrupts
  - Data transfer between ISR, callback fn and the main code
  - Important issues in interrupt handling. Example on the actual board
  - Extra topics
- Timers, Introduction
  - Clock signals in the MC
    - Oscillators
  - Timers in Embedded systems
    - What is a timer
    - Timers in the STM32L0 MC, Mbed and its simulator.

## Interrupts

→ sleep mode
→ sequential op

→ Asynchronous event
↓
interrupt
↓
serve the interrupt.



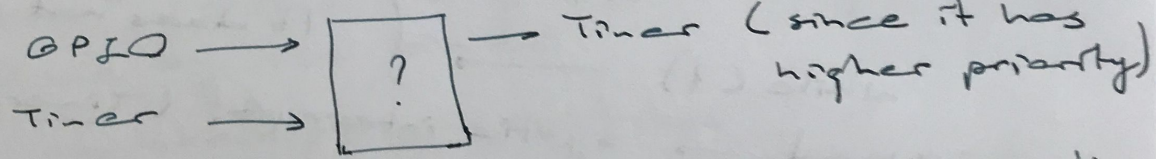

What happens
i-) more than one interrupt comes ?
ii-) successive interrupts come ?

→ Interrupt priority concept.

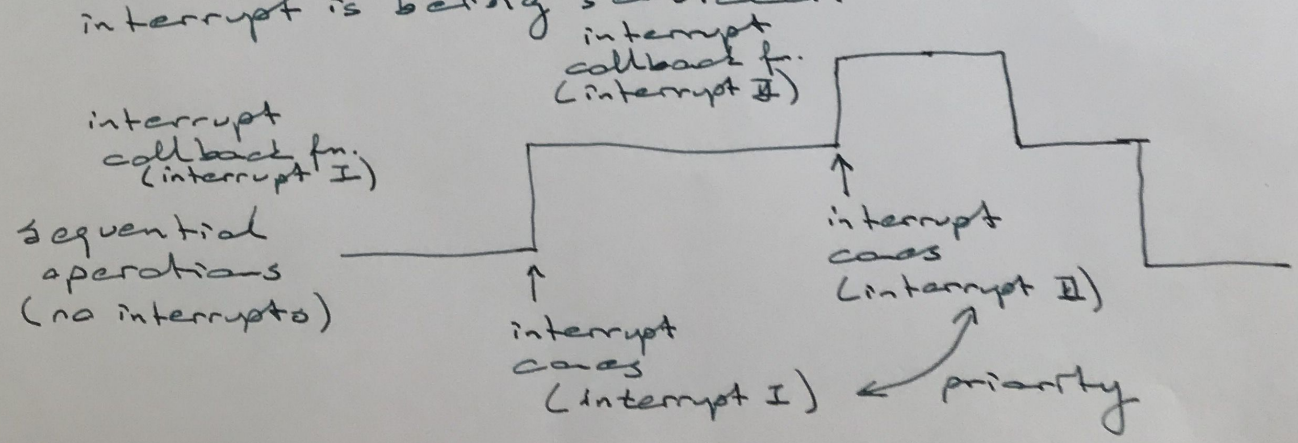
we have several interrupt sources
source1 (GPIO)
source2 (timer)
(ADC)

→ prioritized beforehand.
→ we can change priority by software.

i-) Two interrups come at the same time




ii-) A new interrupt comes while there is another interrupt is being serviced.



→ Data Transfer between the interrupt callback fn. and the main function.

→ Define a global variable

```
void callback_fn() {
        }
    }

int main() {
    |
    |
    |
```

→ This allows data transfer between the main fn. and callback fn. or callback fns.

→ Important topics while using interrupts

1. Keep the callback fn. execution "short"
2. Use global variables between the main and callback functions.
3. Prioritize the interrupts.
4. Use low power modes to get the best from the interrupts.

⇒ Low power mode?

wait_ms(osWaitForever) → To sleep mode. CPU → MC does not get the clock signal.

wait_ms(10) → ?
while(1) →

} The CPU waits afterwards. It does not sleep.

Mbed → systic interrupt. (Timers chapter)

# Introduction to Timers

→ Time based operations. Or when periodic operations are executed.

(inside the MC)

Oscillator ⟶ RC oscillator

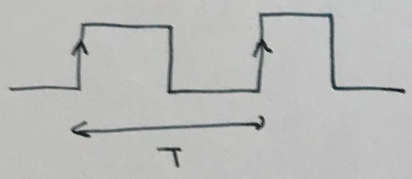
↓

Clock → Adjusts the signal and feeds it to the CPU and peripherals devices such as "Timer".

↓

Timer → In its basic sense, timer is a counter.

→ How do I perform time based operations in the MC using osc, clock, timer?




T → period
f → frequency
Example 1 kHz. frequency $10^{-3}$ sec. period

→ Every millisec, rising edge of the clock signal comes

→ If I want to toggle the LED every sec.
"count" 1000 rising clock edges.
⇒ Done by the timer.
↳ Interrupt generation.
→ callback fn. can be executed.