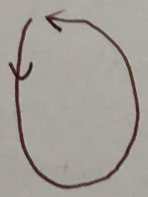
CSE 2037, Week 10

Interrupts

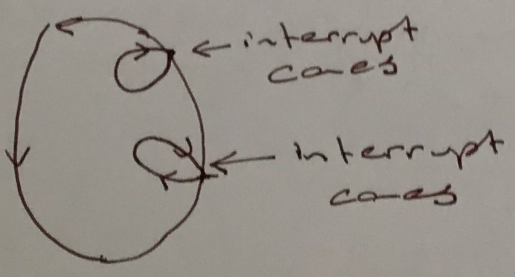
→ The Interrupt concept
  → Interrupts in general
  → Event driven programming
  → Interrupts in the microcontroller
→ Interrupt setup under Mbed (simulator)
→ Interrupt usage under Mbed

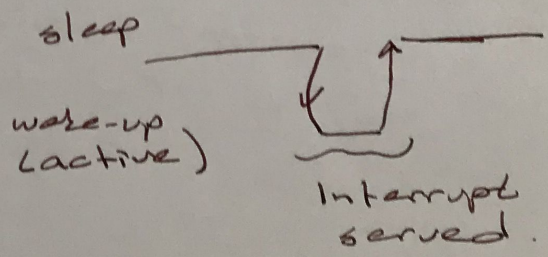
Interrupt Concept

1. No interrupt usage
   round robin approach, CPU active, we are waiting for an event to occur all the time.
   ⇒ while(1) { ⟷ } while(true) { } — infinite loop.



2. Interrupt usage
   2.1 CPU active
   2.2 CPU sleeps
   Low power mode
   Sleep mode
   Deep sleep mode



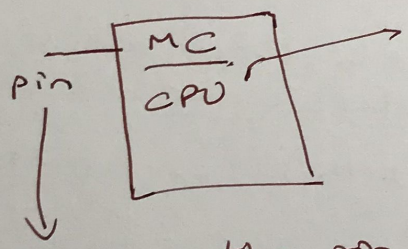


Interrupts in Microcontrollers

Interrupt source — interrupt generator (trigger)
- → hardware ★
- → software (syoten) Exclude

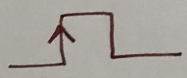
- → GPIO pin interrupts
- → Timer interrupts
- → ADC interrupts
- → Digital Comm. interrupts
- → - - -

Let's focus on GPIO interrupts



Microcontroller changes its execution steps,
- → CPU in thread mode
- → CPU in handler mode

prepare the pin to accept interrupts.



physical change occurs, it triggers the interrupt

Thread mode ≈ CPU performs its usual operation

handler mode ≈ CPU "responds" to the interrupt

Event Driven (Interrupt based) Programming
- → Interrupt source
- → When an interrupt occurs, we should execute a "specific" code as the response
- → Interrupt service routine, call back function.
- → The CPU will be in the handler mode.

Important Issues a Interrupts

w10, 3

1. Prioritize the interrupts
2. The callback function should be short.
3. Allow serving multiple interrupts.
4. Use low power modes with interrupt.

→ Interrupt setup
→ Interrupt Usage

} on Mbed (simulator)