SOFT TIMERS

concept → ① to delay the execution of of fn.
② to execute a function periodically
③ Hardware timers: unique to the architecture
④ software timers: based on the running code/RTOS tick timer

Delaying execution of a function using FreeRTOS:

- vTaskDelay() : block currently running task for a set amount of time (in ticks)
- xTaskGetTickCount(): non-blocking delay w/ known timestamp

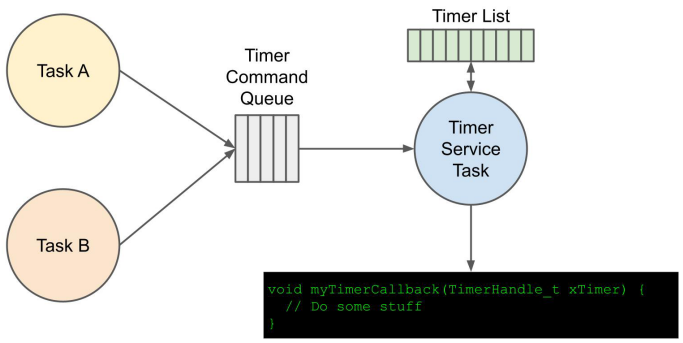
→ Microcontrollers & microprocessors include one or more hardware timers
↳ configured to count up or down & trigger an ISR when they expire.

→ software timers → exist in code
→ not hardware dependent
↳ exception: RTOS tick timer usually relies on a hardware timer

→ FreeRTOS → includes software timers that can delay calling of a function or call function periodically
→ there's an API that manages the timers
→ automatically runs a timer service task aka. "timer daemon" separately from all other tasks
↳ this task manages all timers we set & calls the various callback functions

→ the API function calls can communicate w/ service task through a queue to ensure all commands are received.

Software Timers in FreeRTOS



① when timer is created, a "callback function" is assigned whenever the timer expires

↳ however the timer resolution cannot be less than 1 tick as it is dependent on the tick timer

② can also set the software timers to be "one-shot"

↓

executes callback function once after the timer expires

or "auto-reload"

↳ executes callback function periodically every time after timer expires