UNIVERSIDAD
COMPLUTENSE
MADRID

ESP32 (low) power

IoT node architecture

Power consumption of ESP32

- According to Espressif

CPU Active	240 MHz	Dual-core	30mA - 68mA
		Single-core	NA
	160 MHz	Dual-core	27mA– 44mA
		Single-core	27mA – 34mA
	80 MHz	Dual-core	20mA – 31 mA
		Single-core	20 mA – 25 mA

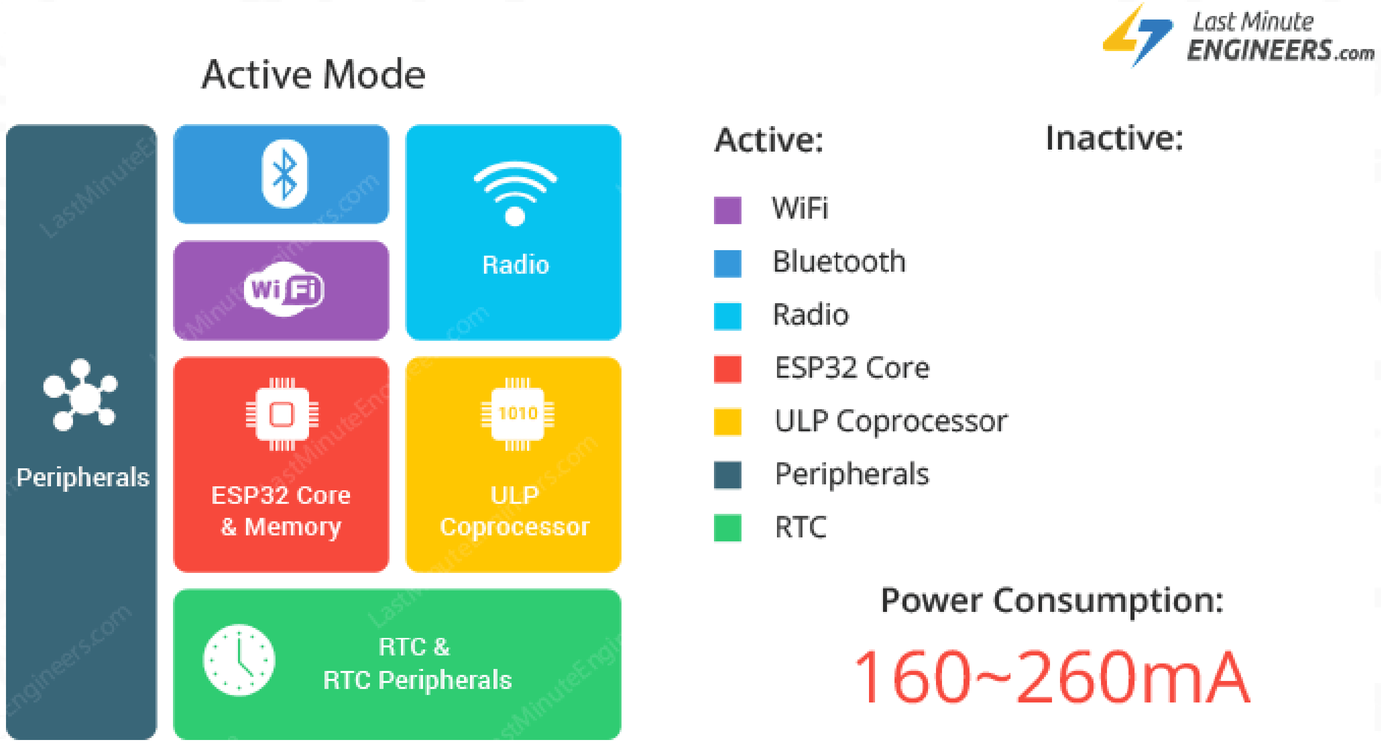
- Sometimes it can be advantageous from the energy point of view to use 2 cores and reduce frequency
  - Cubic relation between frequency (voltaje) and power
  - Performance is achieved by exploiting parallelism
- 5 power modes

https://lastminuteengineers.com/esp32-sleep-modes-power-consumption/

Active Mode

- Default mode. All functionality is available
  - Cores, WiFi, Bluetooth… Active at all times
  - Requires 240mA constantly. Power spikes up to 790mA when WiFi and Bluetooth are used together

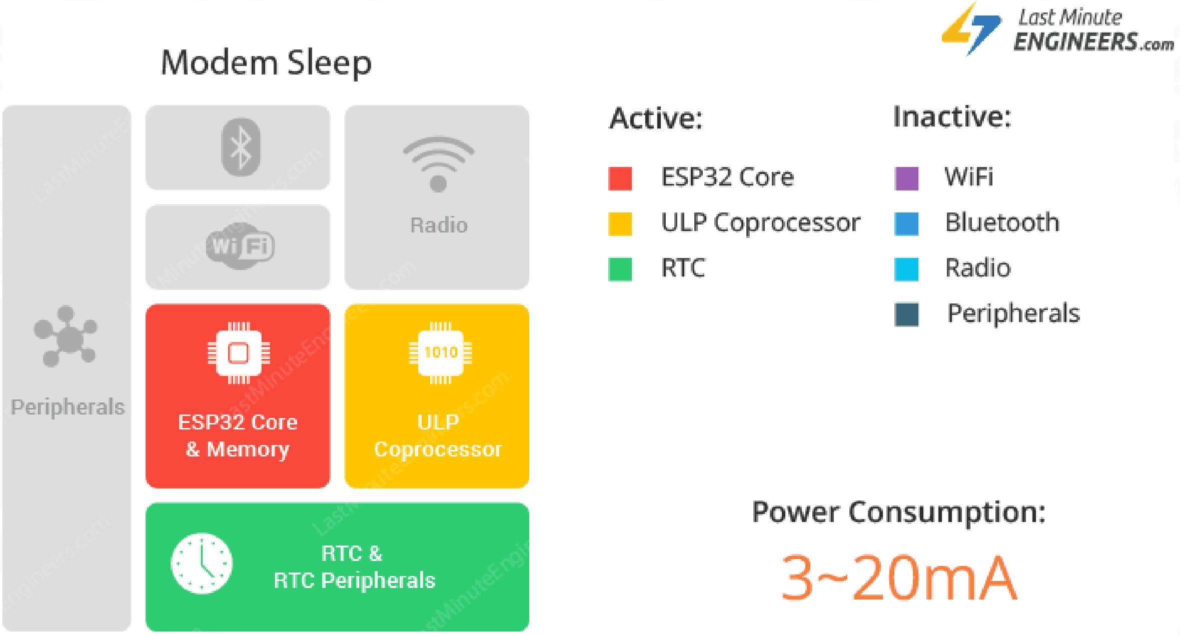
Mode	Consumption
Wi-Fi Tx packet 13dBm-21dBm	160mA – 260 mA
Wi-Fi/BT Tx packet 0dBm	120mA
Wi-FI/BT Rx and listening	80mA – 90mA



https://lastminuteengineers.com/esp32-sleep-modes-power-consumption/

Modem sleep

- Similar to *Active* except for WiFi and Bluetooth, which are disabled
  - CPUs are operational and the clock is configurable
  - Power between 3mA (low frequency) and 20mA (high frequency)
  - To keep connections alive, WiFI/BT modules are periodically woken up
    - Association Sleep Pattern

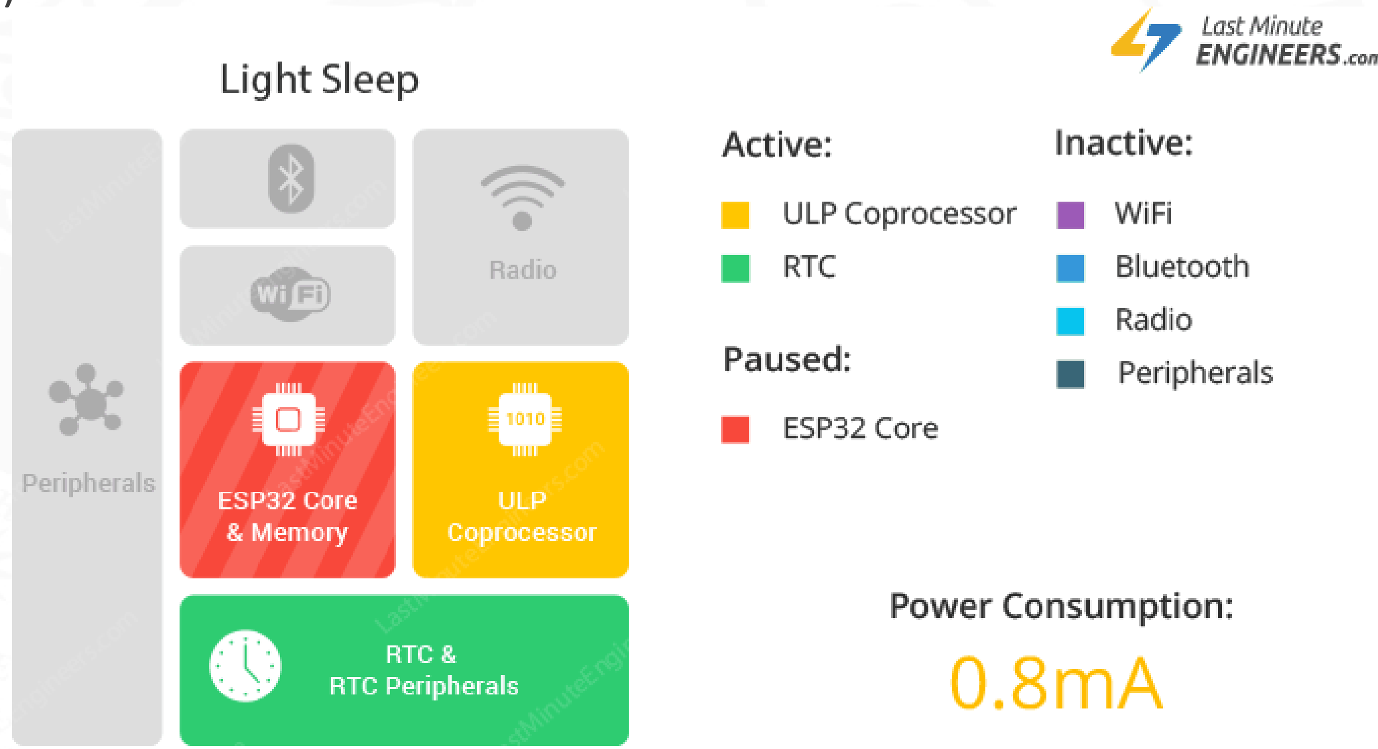


- ESP can use this mode only when it connects to the router/AP in *station mode*
  - ***DTIM beacon mechanism***
  - Modules are woken up when the *beacons arrive*
  - 100 ms – 1000 ms

Ligth Sleep

- ❑ Similar to *modem sleep*
  - It also follows *association sleep pattern*
  - *Clock-gating* for peripherals, CPUs and most of the RAM
    - CPU is *paused* (it has power but no clock pulses)
    - RTC and ULP (Ultra-Low Power processor) active

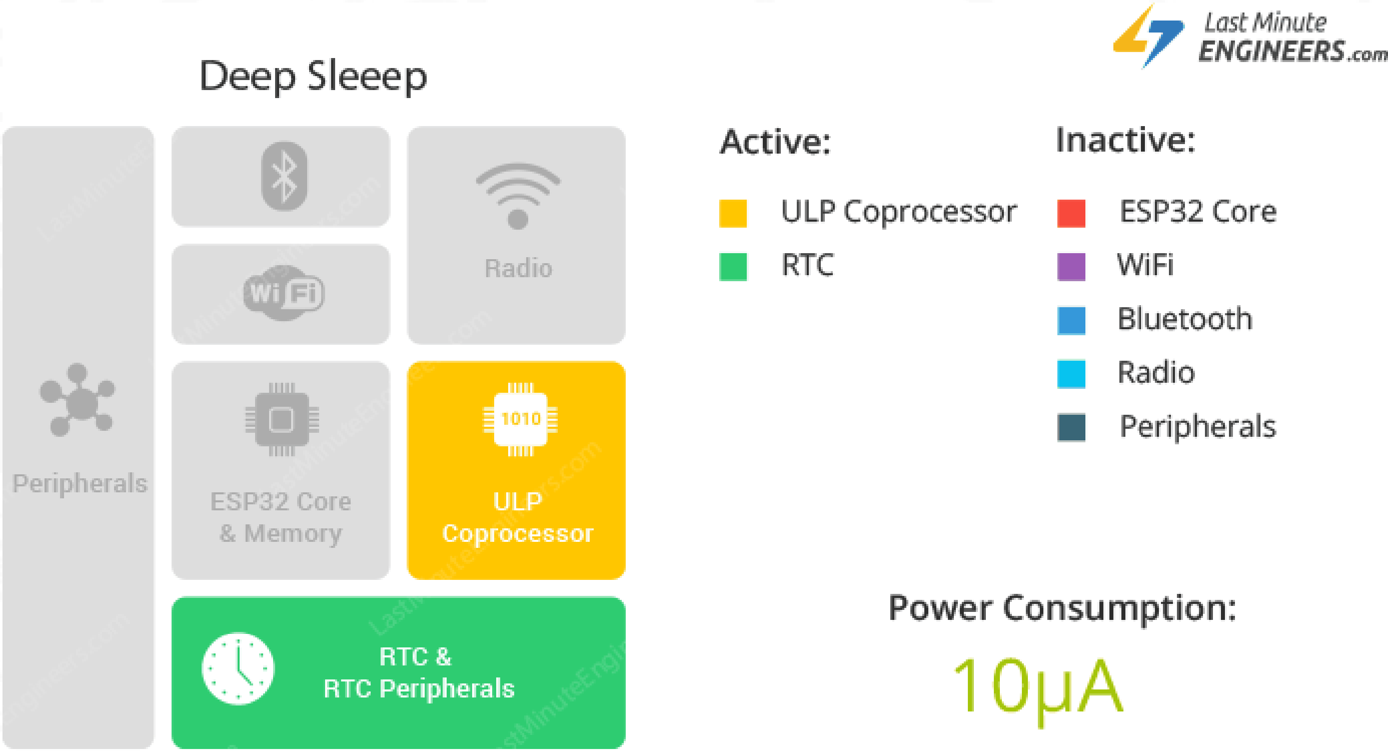
- Before entering this mode, ESP32 preserves its state
  - **FULL RAM retention**
- The wake-up mechanism needs to be configured before entering light sleep mode
  - CPU is NOT running



Deep Sleep

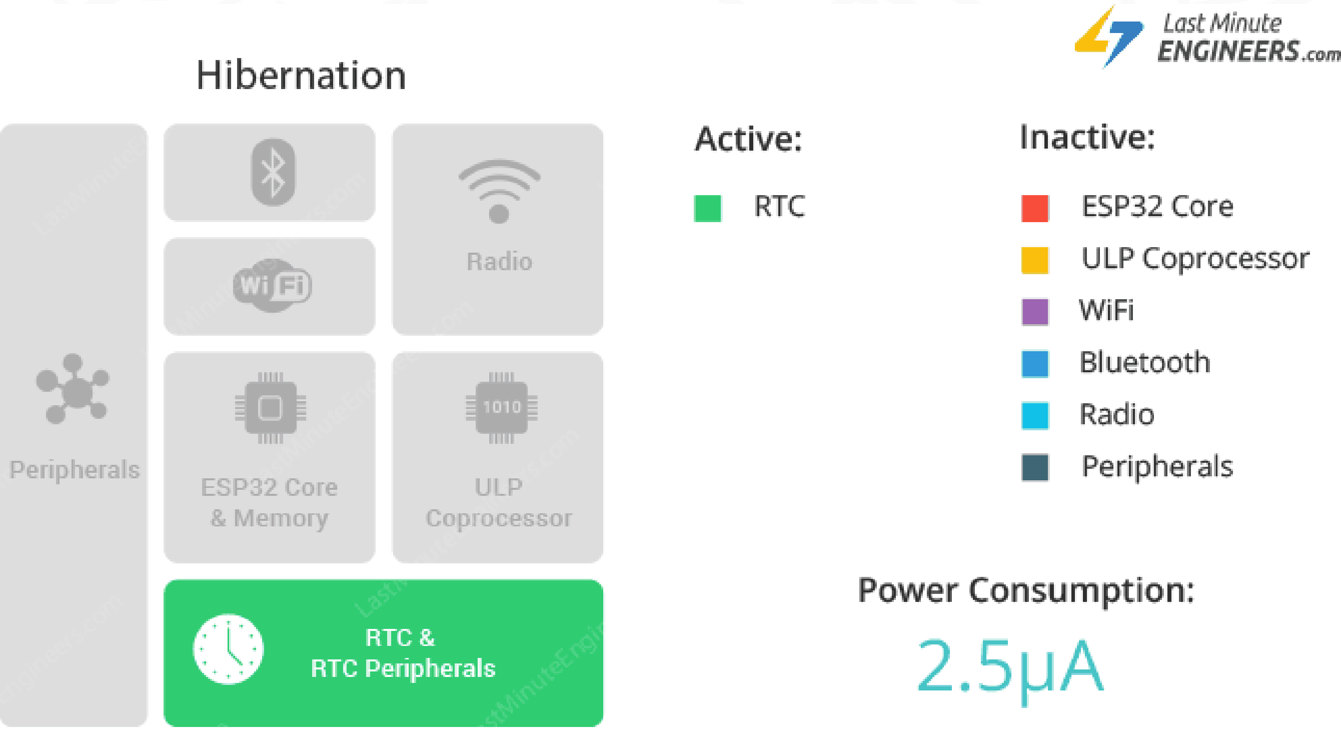
- CPU – Memory are powered-off
  - Only RTC controller/peripherals (including ULP coprocessor) and RTC memories (*slow* y *fast*) remain powered on
  - Consumption between 0.15mA (if ULP is powered on) and 7uA

- ULP allows reading sensors
  - It wakes up the main systems based on this measurements
  - ULP sensor-monitored pattern
- Main memory CANNOT be accessed
  - But RTC memory is powered.
- *Wake-up sources need to be configured*



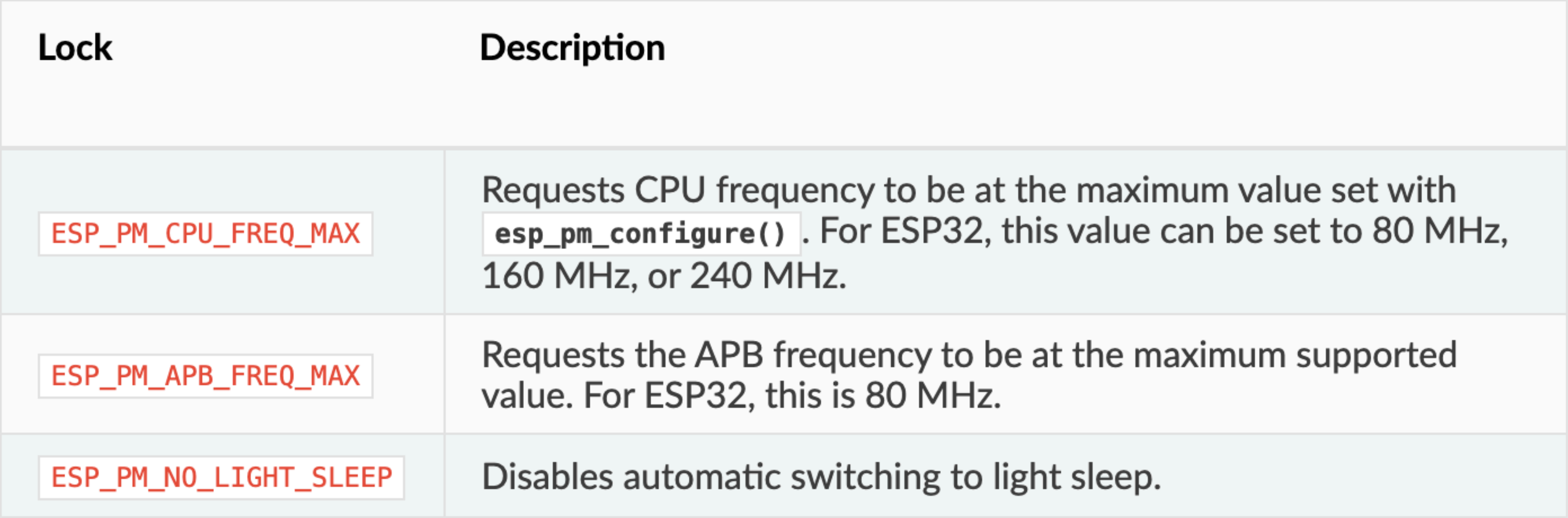
Hibernation

- Everything is disabled
  - Even the internal 8MHz oscillator and ULP, RTC memory
  - Nothing can be done or preserved
  - Only an *RTC timer and* some *RTC GPIOs are active*
    - To wake-up the chip



ESP-IDF (Power Management)

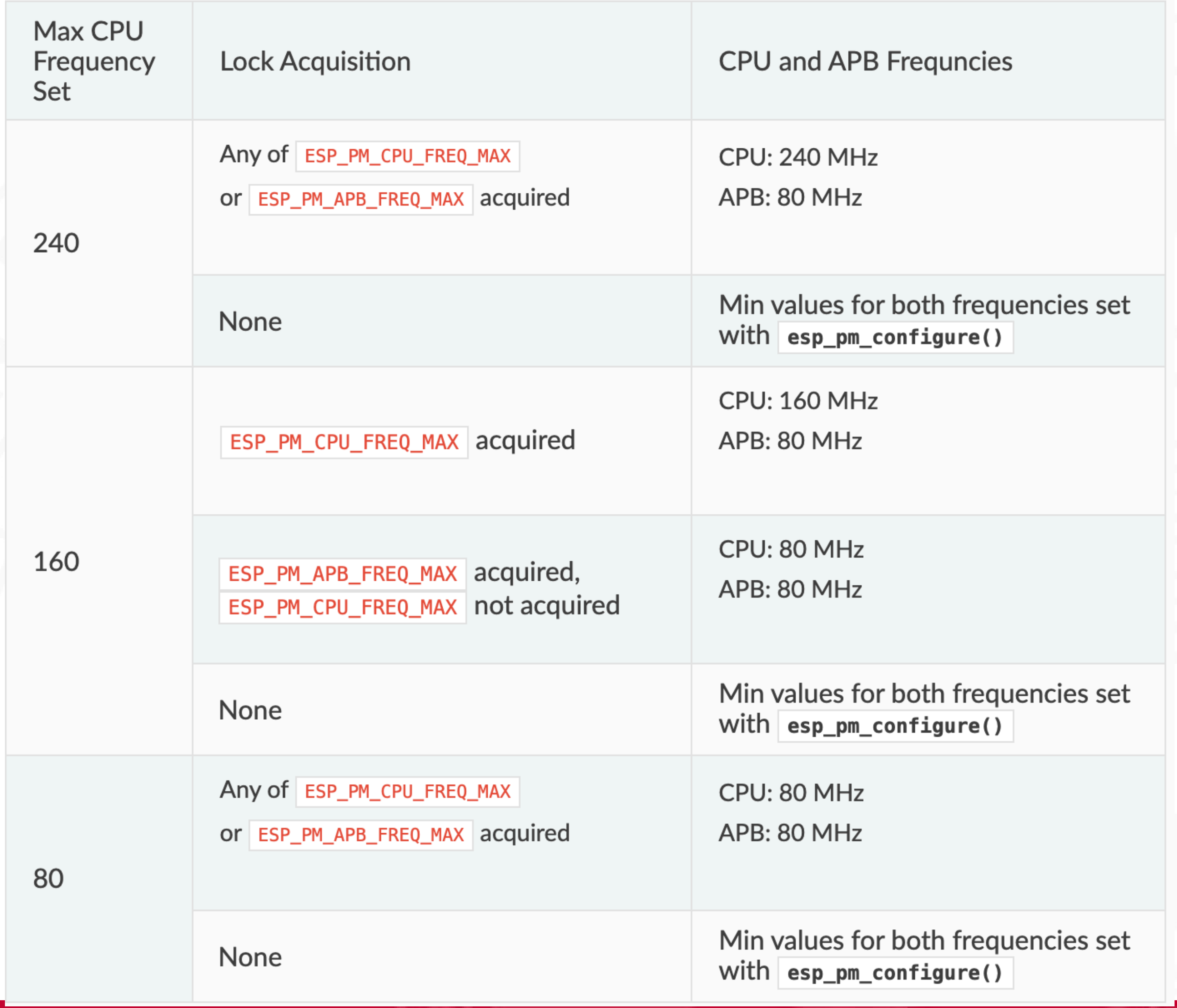
- Automatic management of power modes
  - https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/power_management.html
  - ESP-IDF defines *locks* that are used to express requirements of applications
- `esp_err_t esp_pm_configure(const void *config)`
  - max_freq_mhz (80MHz, 160MHz or 240MHz)
  - min_freq_mhz. XTAL$_{freq}$ usually 40MHz. Minimum 10Mhz.
  - Light_sleep_enable (boolean)



Lock	Description
`ESP_PM_CPU_FREQ_MAX`	Requests CPU frequency to be at the maximum value set with `esp_pm_configure()` . For ESP32, this value can be set to 80 MHz, 160 MHz, or 240 MHz.
`ESP_PM_APB_FREQ_MAX`	Requests the APB frequency to be at the maximum supported value. For ESP32, this is 80 MHz.
`ESP_PM_NO_LIGHT_SLEEP`	Disables automatic switching to light sleep.

- `esp_pm_lock_create()`
- `esp_pm_lock_acquire()`
- `esp_pm_lock_release()`

ESP-IDF (Power Management)



Max CPU Frequency Set	Lock Acquisition	CPU and APB Frequncies
240	Any of `ESP_PM_CPU_FREQ_MAX` or `ESP_PM_APB_FREQ_MAX` acquired	CPU: 240 MHz APB: 80 MHz
	None	Min values for both frequencies set with `esp_pm_configure()`
160	`ESP_PM_CPU_FREQ_MAX` acquired	CPU: 160 MHz APB: 80 MHz
	`ESP_PM_APB_FREQ_MAX` acquired, `ESP_PM_CPU_FREQ_MAX` not acquired	CPU: 80 MHz APB: 80 MHz
	None	Min values for both frequencies set with `esp_pm_configure()`
80	Any of `ESP_PM_CPU_FREQ_MAX` or `ESP_PM_APB_FREQ_MAX` acquired	CPU: 80 MHz APB: 80 MHz
	None	Min values for both frequencies set with `esp_pm_configure()`

- Depending on the configuration and on the *locks* adquired, the power management algorithm will choose the clock frequency
- If no *locks are adquired and light sleep is enabled*, the system will go into *ligth sleep mode. It will end*...
  - If a task is unblocked
  - If a Timer (High Resolution) ends
  - If there is a *wakeup source*

ESP-IDF. *Wakeup sources*

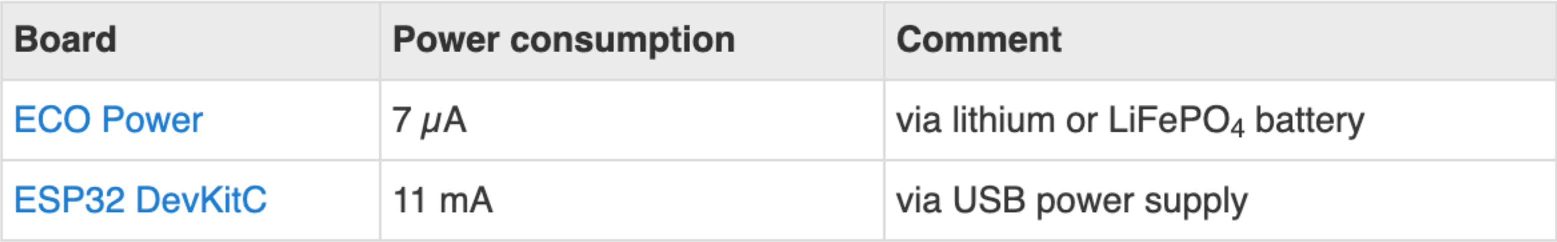
- Timer
  - Timer in the RTC controller
  - The systems goes into Active mode after *n* microseconds
  - esp_sleep_enable_timer_wakeup()
- TouchPad
  - The system goes into Active mode when a *touch sensor interrupt occurs*
  - Interrupts need to be configured before entering Deep sleep mode
  - esp_sleep_enable_touchpad_wakeup()
- External wakeup (ext0)
  - The system goes into Active mode when an RTC GPIO is set to a predefined logic level
  - esp_sleep_enable_ext0_wakeup()
- External wakeup (ext1)
  - The system goes into Active mode using multiple RTC GPIOs
    - Wake up if any of the selected pins is 1
    - Wake up if all the selected pins are 0
  - esp_sleep_enable_ext1_wakeup()
- Other options
  - https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/sleep_modes.html

ESP-IDF. Manual power management

- When power management is active, entering *light sleep mode* occurs automatically
- But, it is also posible do it to *manually* by calling
  - `esp_light_sleep_start()`
  - `esp_deep_sleep_start()`
- WiFi and Bluetooth in low power modes
  - Before entering *light/deep sleep* manually, they have to be disabled
  - `esp_bluedroid_disable(), esp_bt_controller_disable(), esp_wifi_stop()`
  - WiFi/BT connections will NOT remain when manually entering *ligth/deep sleep modes*
    - If we need them, we have to use automatic management and enable *automatic light sleep*

BEWARE: SoC vs board

- ESP32 is the SoC, but there are other elements that also consume in the board
  - Sometimes they cannot be powered-off



Board	Power consumption	Comment
ECO Power	7 μA	via lithium or $LiFePO_4$ battery
ESP32 DevKitC	11 mA	via USB power supply

https://www.radioshuttle.de/en/media-en/tech-infos-en/battery-powered-esp32/