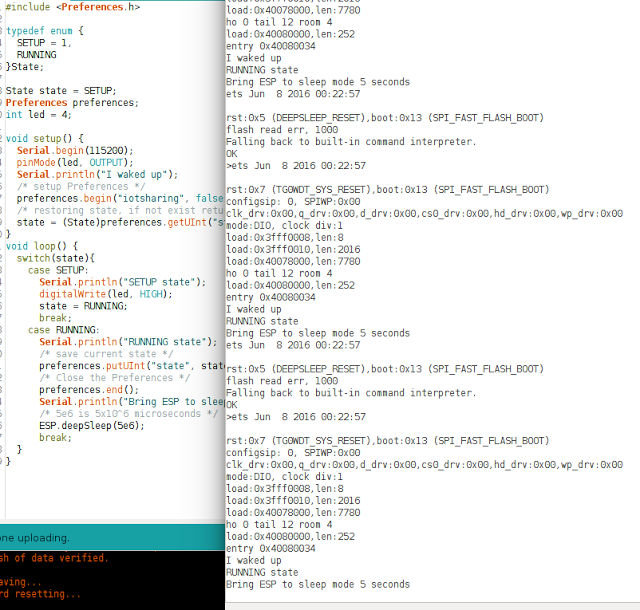
# **[Demo 24: How to bring ESP32 to low power-sleep mode to extend battery life](http://www.iotsharing.com/2017/06/bring-esp32-to-low-power-sleep-save-battery.html)**

1. **Introduction**  
   - When deploying IoT applications in area where power supply is not well-prepared, we often use battery for IoT node. The power of battery is limited so we have to save it. In order to save power, we will bring ESP to sleep mode in which ESP will not consume much power.  
   - Sleep mode is not applicable when ESP is in [Access Point](http://www.iotsharing.com/2017/05/how-to-turn-esp32-into-access-point.html" \t "http://www.iotsharing.com/2017/06/_blank) mode.  
   - When ESP is in sleep mode it need a trigger to wake it up. There are some types of trigger:  
   + by external signal using RTC\_IO.  
   + by external signal using RTC\_CNTL.  
   + by timer.  
   + by touchpad.  
   + by ULP program.  
   - I will make 2 demos that using timer and RTC\_IO for waking up.  
   **2. Demo 1 using timer to wake up**  
   - In order to bring ESP to sleep mode and wake up by timer, just use the function "ESP.deepSleep(time\_in\_micro\_seconds)" which**time\_in\_micro\_seconds** is the time ESP will spend in sleep mode.  
   **Note:**trigger wake ESP up by resetting it. You can see this by watching the Terminal or the setup() function will be invoked again. So you need to backup data before going to sleep mode using [Preferences](http://www.iotsharing.com/2017/06/how-to-use-preferences-to-backup-data-in-main-flash-when-power-off.html" \t "http://www.iotsharing.com/2017/06/_blank) or [sdacrd](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-store-data-to-sdcard.html" \t "http://www.iotsharing.com/2017/06/_blank).  
   - I will make a demo that has 2 states: SETUP and RUNNING. In SETUP state just print "SETUP state" and then move to RUNNING state, backup this state to Preferences and going to low power-sleep mode for 5 seconds. After waking-up ESP will read the state from Preferences and move to that state without running SETUP state again.

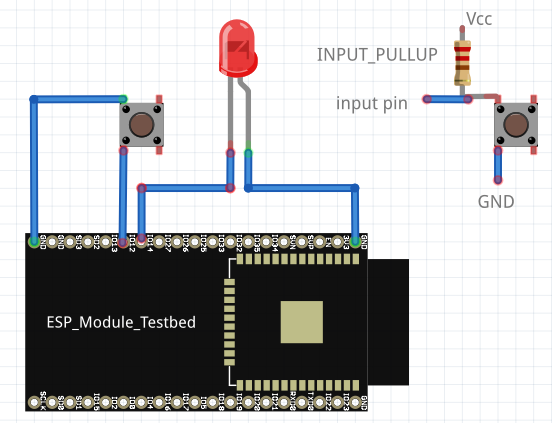
|  |
| --- |
| #include <Preferences.h>  /\* possible states \*/  typedef enum {  SETUP = 1,  RUNNING  }State;  State state = SETUP;  Preferences preferences;  void setup() {  Serial.begin(115200);  Serial.println("I waked up");  /\* setup Preferences \*/  preferences.begin("iotsharing", false);  /\* restoring state, if not exist return default SETUP sate \*/  state = (State)preferences.getUInt("state", SETUP);  }  void loop() {  switch(state){  case SETUP:  Serial.println("SETUP state");  state = RUNNING;  break;  case RUNNING:  Serial.println("RUNNING state");  /\* save current state \*/  preferences.putUInt("state", state);  /\* Close the Preferences \*/  preferences.end();  Serial.println("Bring ESP to sleep mode 5 seconds");  /\* 5e6 is 5x10^6 microseconds \*/  ESP.deepSleep(5e6);  break;  }  } |

[](https://3.bp.blogspot.com/-WAI4hPO7W9Q/WU98U02Z_yI/AAAAAAAAEF8/XlSEt4RQ5ro4iSRq1qpPZT0ISvSAWbbdQCLcBGAs/s1600/esp32sleep1.png)

**Figure: using timer to wake up**

**3. Demo 2 using external signal RTC\_IO to wake up**

In this demo, we will use a button to make an external signal LOW level to wake ESP up. We re-use the circuit in [Demo 21](http://www.iotsharing.com/2017/06/how-to-use-interrupt-in-arduino-esp32.html" \t "http://www.iotsharing.com/2017/06/_blank) but ignoring the LED.

[](https://4.bp.blogspot.com/-SAIoFz75OyQ/WUKx-FTo5eI/AAAAAAAAEEI/S8o79cqiGWkTBSeRCyJ46kVd-NmFXIqAQCPcBGAYYCw/s1600/esp32interrupt2.png)

Connections:

[ESP32 GIO12 - BUTTON - GND]

|  |
| --- |
| #include "esp\_deep\_sleep.h"  void setup() {  Serial.begin(115200);  Serial.println("I waked up");  /\* set up RTC\_IO for GPIO12 \*/  esp\_deep\_sleep\_pd\_config(ESP\_PD\_DOMAIN\_RTC\_PERIPH, ESP\_PD\_OPTION\_AUTO);  /\* enable pullup resister \*/  gpio\_pullup\_en(GPIO\_NUM\_12);  /\* disable pullup resister \*/  gpio\_pulldown\_dis(GPIO\_NUM\_12); // not use pulldown on GPIO  /\* anable deep sleep with external wakeup  and input level which will trigger wakeup is LOW on GPIO12 \*/  esp\_deep\_sleep\_enable\_ext0\_wakeup(GPIO\_NUM\_12, LOW);  Serial.println("Bring ESP to sleep mode until get external signal");  esp\_deep\_sleep\_start();  }  void loop() {  } |

