

Allarme GSM batteria scarica e mancanza corrente

File del progetto — sketch Arduino (monitor_batteria_moto.ino)

officinasottocasa.it

```
/*
 * Allarme GSM per batteria scarica e mancanza corrente
 * Arduino Nano + SIM800L + display OLED SSD1306 (I2C)
 *
 * Misura in continuo la tensione della batteria tramite un partitore
 * resistivo; sotto la soglia impostata, sveglia il modulo GSM SIM800L
 * (tenuto in sleep-mode) e invia un SMS di allarme.
 *
 * Sketch coerente con l'articolo "Allarme GSM per batteria scarica e
 * mancanza corrente: Arduino Nano + SIM800L" - officinasottocasa.it
 */

#include <U8x8lib.h>          // libreria per gestione schermo OLED
#include "SoftwareSerial.h"    // libreria per seriale con modulo SIM800L

// --- Attribuzione dei pin ---
#define VoltTestPin A0        // pin analogico collegato al partitore per la misura del voltaggio
#define DTR_PIN 5             // pin DTR del SIM800L collegato al pin D5 di Arduino
#define SIM_TX_PIN 6          // pin TX del SIM800L collegato al pin D6 di Arduino
#define SIM_RX_PIN 7          // pin RX del SIM800L collegato al pin D7 di Arduino

// --- Inizializzazione degli oggetti ---
SoftwareSerial serialSIM800(SIM_TX_PIN, SIM_RX_PIN); // seriale software verso il SIM800L
U8X8_SSD1306_128X64_NONAME_HW_I2C u8x8(/* reset= */ U8X8_PIN_NONE); // oggetto per il display OLED

// --- Definizione delle variabili ---
int read_value = 0;           // valore intero letto sul pin A0 (0-1023)
float ref_voltage = 5.0;      // tensione di riferimento massima per il calcolo
float read_voltage = 0.0;     // tensione letta sul pin A0
float batt_voltage = 0.0;     // tensione reale della batteria, dopo la formula inversa del partitore
float R1 = 100000.0;          // valore di R1 del partitore
float R2 = 10000.0;           // valore di R2 del partitore
int v_dummy = 0;              // flag per inviare l'SMS una sola volta, fino al reset

void setup() {
    serialSIM800.begin(57600); // inizializza la seriale con il modulo GSM
    delay(50);
    digitalWrite(DTR_PIN, HIGH);
    delay(50);
    Screeninit();              // avvia lo schermo OLED
    delay(50);
    sim800sleepmode();         // mette il modulo GSM in sleep-mode
}

void loop() {
    Readvoltage();
    if (batt_voltage < 12.0 && v_dummy == 0) {
        sim800wakeup();
        sendsmsalarm();
        v_dummy = 1; // impedisce l'invio di ulteriori SMS fino al reset della scheda
    }
}

void Screeninit() {
    u8x8.begin();
    u8x8.setPowerSave(0);
    u8x8.setFont(u8x8_font_chroma48medium8_r);
}
```

```

    u8x8.clear();
    u8x8.drawString(0, 0, "Initializing...");
    delay(2000);
    u8x8.clear();
}

void Readvoltage() {
    read_value = analogRead(VoltTestPin);
    read_voltage = (read_value * ref_voltage) / 1024.0;    // converte la lettura in tensione
    batt_voltage = read_voltage / (R2 / (R1 + R2));        // formula inversa del partitore
}

void sim800sleepmode() {
    serialSIM800.println("AT+CSCLK=2");
}

void sim800wakeup() {
    digitalWrite(DTR_PIN, LOW);
    delay(200);
    digitalWrite(DTR_PIN, HIGH);
    serialSIM800.println("AT");
    serialSIM800.println("AT+CSCLK=0");
}

void sendsmsalarm() {
    serialSIM800.println("AT+CMGF=1");
    serialSIM800.println("AT+CMGS=\"+39xxxxxxxxxx\"");
    serialSIM800.print("testo del messaggio");
    serialSIM800.write(26); // Ctrl+Z, invia il messaggio
}

```