Istituto d'istruzione superiore

Michelangelo Bartolo

Pachino (SR)

Tesina di Maturità

Gianfriddo Giuseppe

Classe: 5^B I.T.I.S

Indirizzo: Elettronica ed elettrotecnica

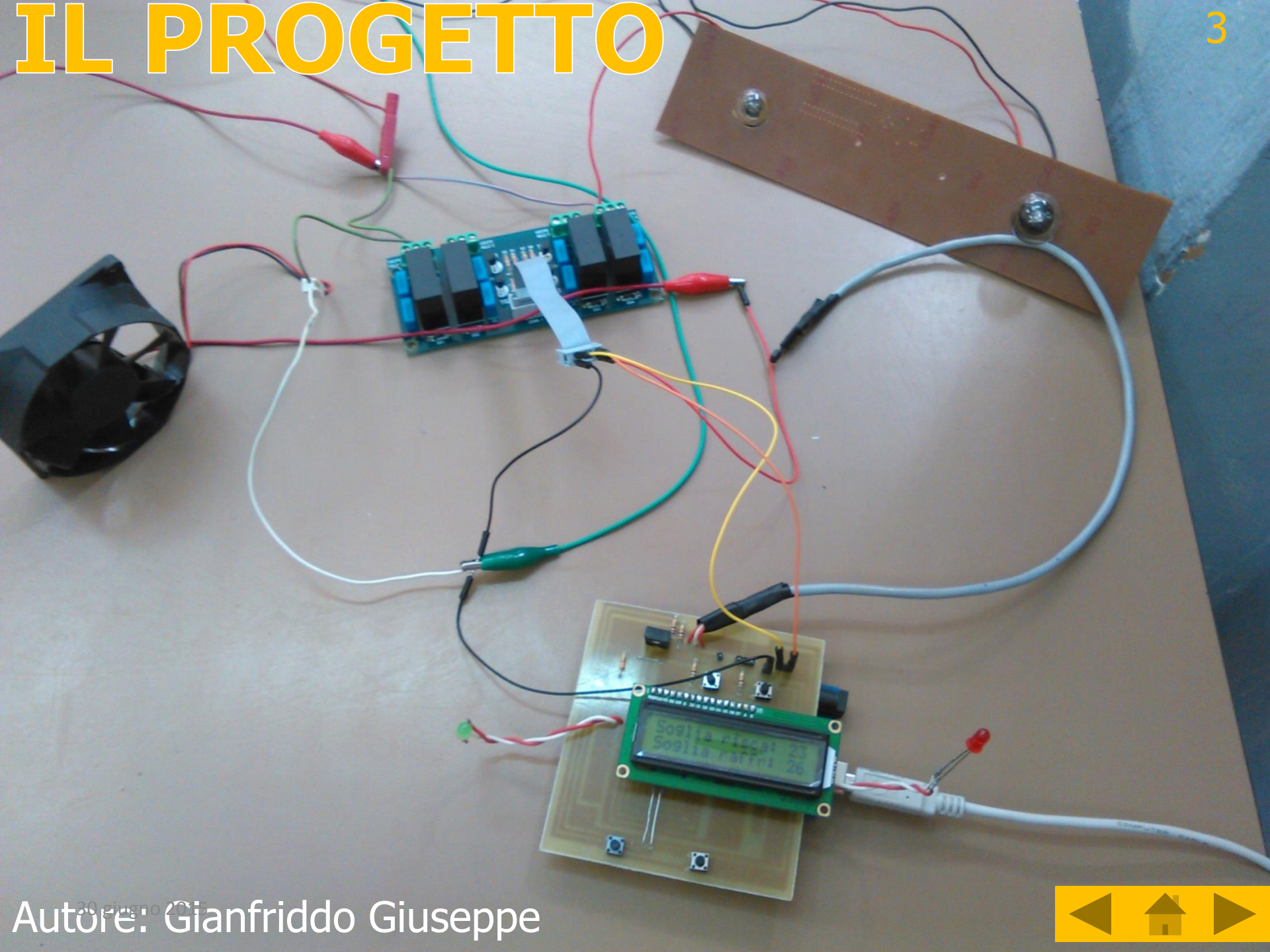
Anno: 2014/15

T.P.S.E.E

Acquisire la temperatura e variare due soglie per attivare o disattivare uno o più attuatori

Autore: Gianfriddo Giuseppe

IL PROGETTO



Autore: Gianfriddo Giuseppe

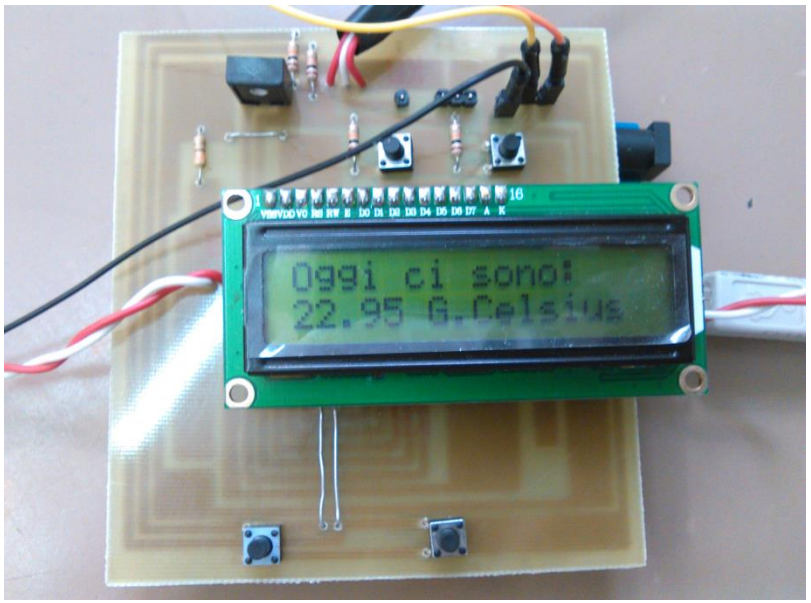
Premesse

L'obiettivo che mi sono prefissato, inizialmente era quello di progettare, stampare e assemblare, un circuito che fosse in grado di acquisire la temperatura, mostrare il valore della temperatura stessa su un display LCD, e variare una soglia prefissata; tutto ciò utilizzando il μC (microcontrollore) Arduino, a cui ho collegato come input, il trasduttore di temperatura (LM35), e come output una scheda relè, dove a sua volta ho collegato una ventola, che simula un condizionatore per raffreddare l'ambiente e una lampada che simula un impianto di riscaldamento.

Premesse

In aggiunta ho pensato di inserire anche due led, che nell'elettronica sono molto utilizzati come spie per indicare il funzionamento o meno di un circuito, in questo caso il led verde accesso segnerà che la temperatura è troppo bassa e il led rosso che la temperatura è troppo alta.
Una possibilità molto importante, è sicuramente la presenza dei *4 pulsanti* che servono all'utente per **regolare la soglia** di riscaldamento e quella di raffreddamento, ***senza intervenire via software***.

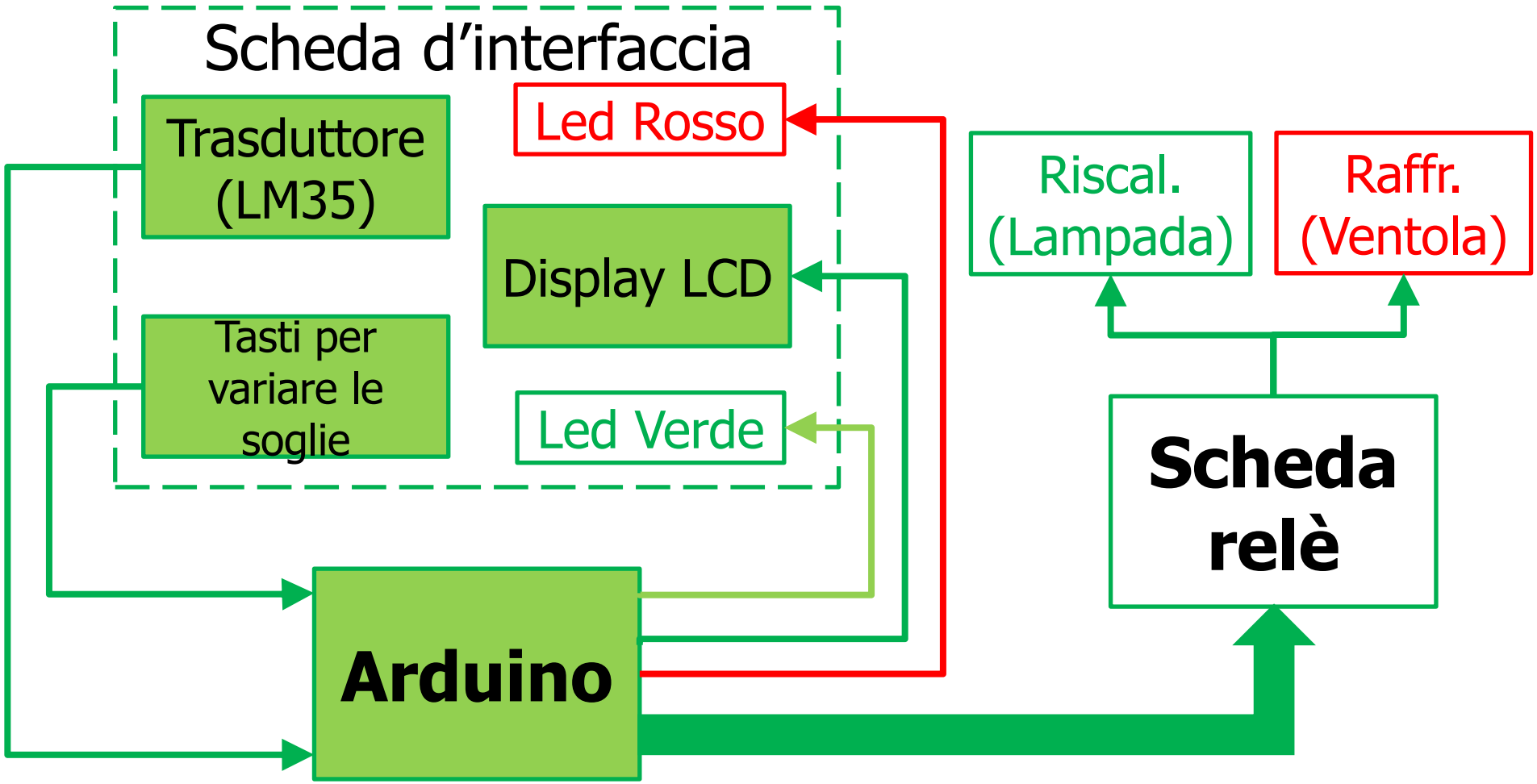
Dispositivi utilizzati



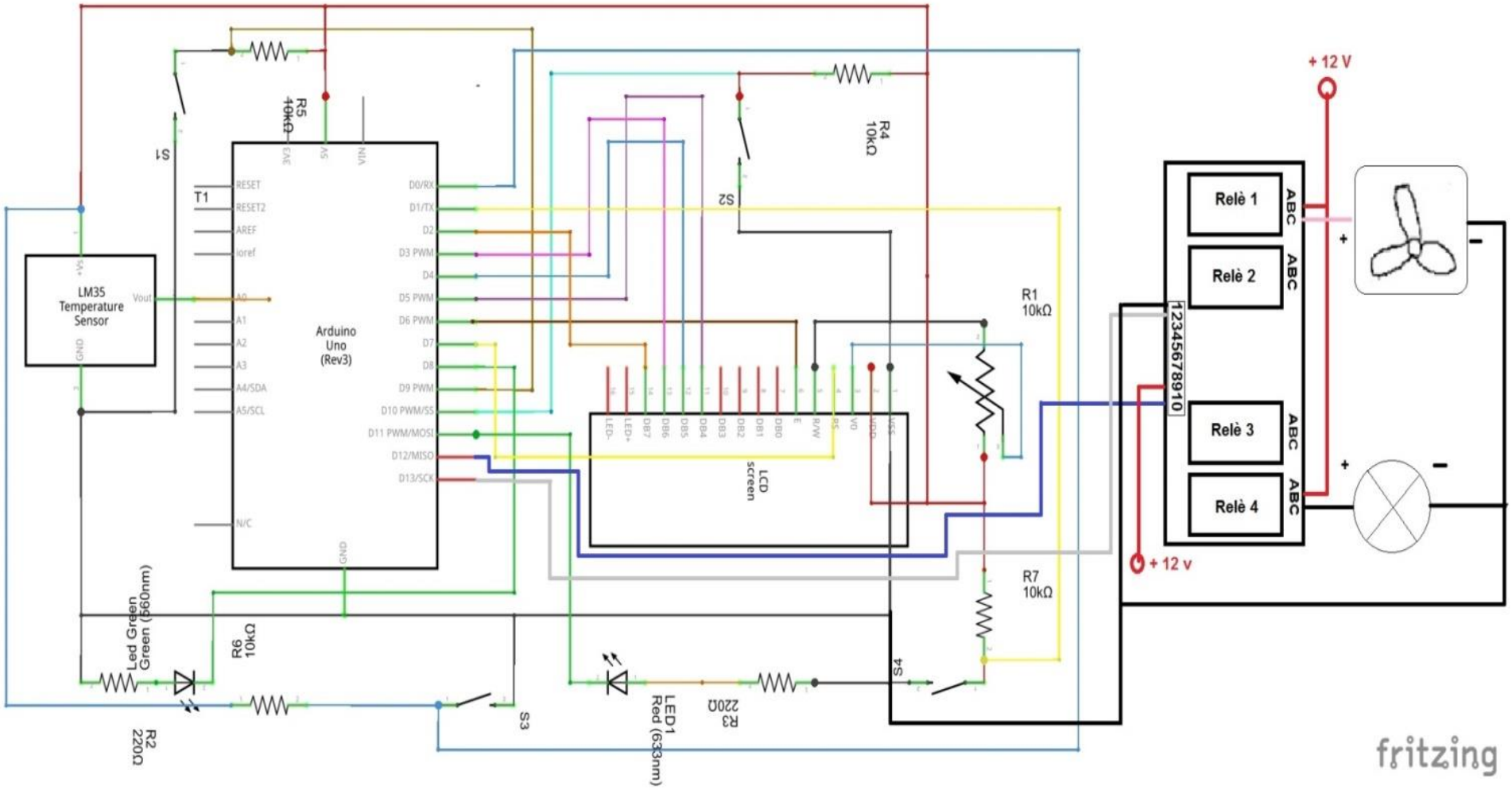
Per la creazione di questo circuito ho utilizzato:

- Arduino Uno
- Scheda relè
- Display LCD
- LM35
- 1 trimmer
- 2 Diodi led e 2 resistenze da 220Ω
- 4 pulsanti con 4 resistenze da 10KΩ

Schema a blocchi

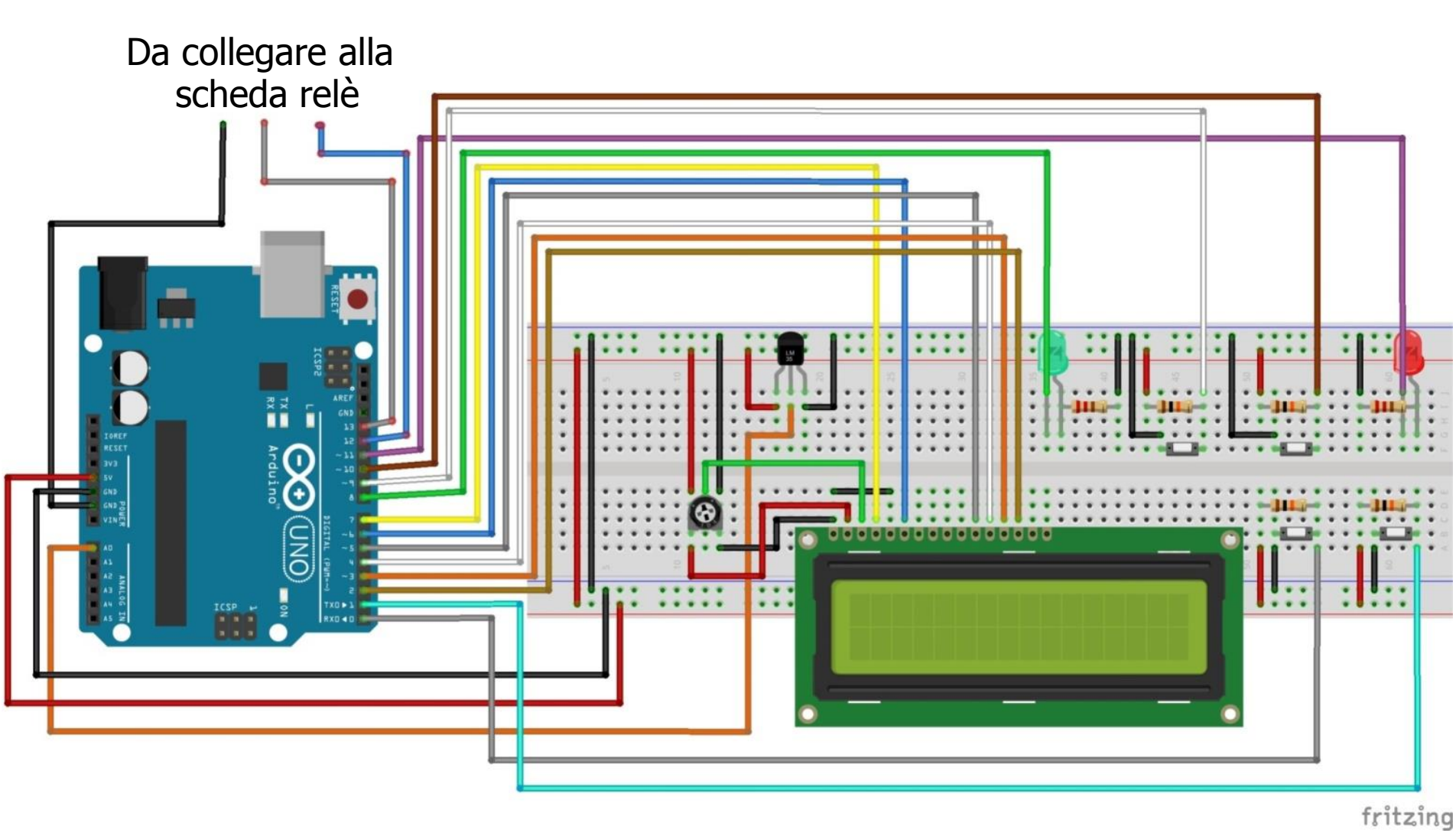


Schema elettrico

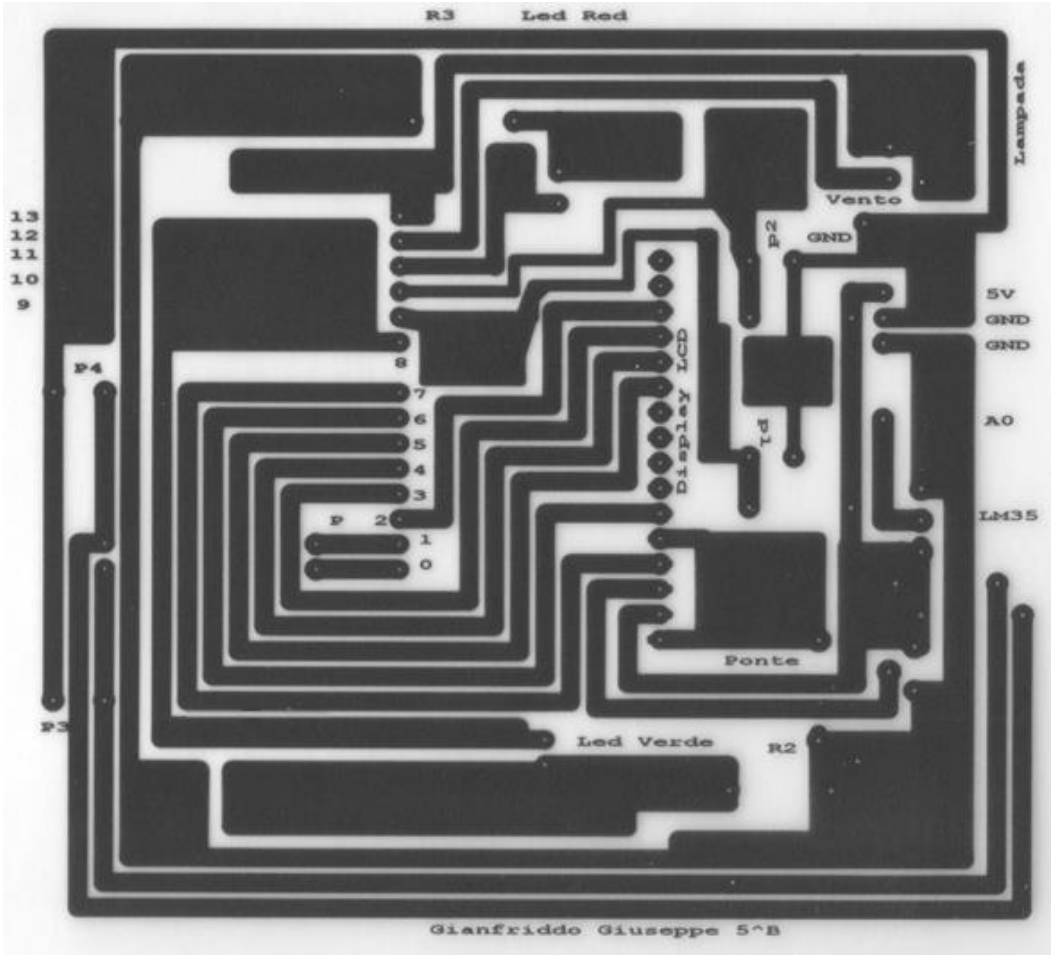


Autore: Gianfriddo Giuseppe

Schema di montaggio



Master scheda d'interfaccia

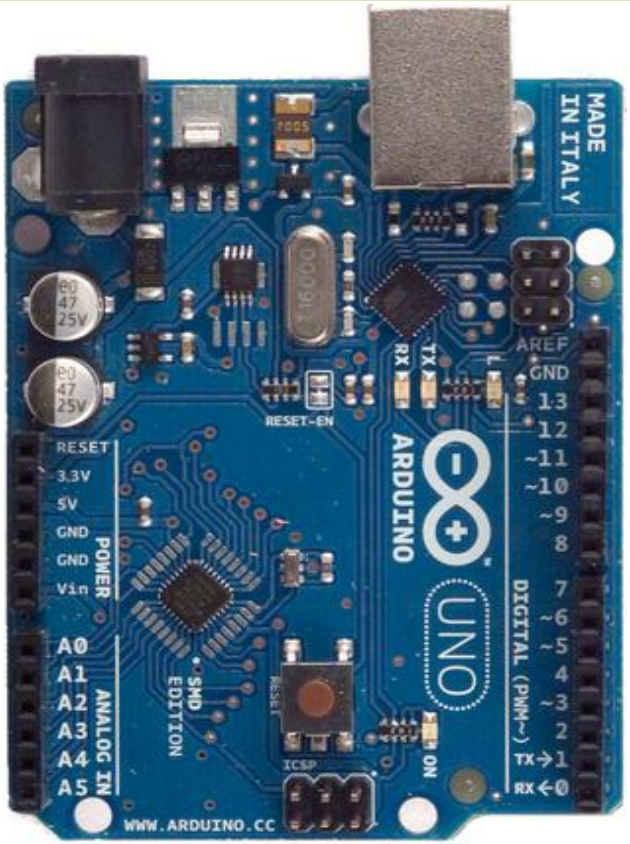


Il master in figura è stato progettato, disegnato e stampato da me. È progettato in modo tale che una volta saldati i componenti sulla basetta, quest'ultima può essere **inserita direttamente sulla scheda Arduino**, senza utilizzare nessun cavo.

I cavi utilizzati serviranno solamente per il collegamento con la scheda relè e per l'alimentazione di Arduino.

Arduino Uno

Clicca per approfondire ARDUINO



Arduino è una piattaforma open-source, è possibile trovare sul sito ufficiale **www.arduino.cc**, i circuiti, i componenti e addirittura le istruzioni per realizzare un progetto da soli. Ciò che dovrebbe interessare in realtà sono gli schemi circuitali: essendo Open, e quindi visionabili da tutti, possono essere continuamente migliorati dalla comunità e grazie ad essi sono state sviluppate un numero incredibile di librerie software che rendono davvero semplice l'interfaccia con periferiche di qualsiasi tipo.

Arduino Uno

Arduino Uno è una delle tante versioni di Arduino ed è una scheda basata sul microcontrollore Atmega328. Dispone di 14 digitali pin di ingresso/uscita (di cui 6 possono essere utilizzate come uscite PWM), 6 ingressi analogici, un 16 MHz risuonatore ceramico, una connessione USB, un jack di alimentazione, un header ICSP e un pulsante di reset. Esso si utilizza semplicemente collegandolo ad un computer con un cavo USB o alimentandolo con una batteria.

Uscire dall'approfondimento

Caratteristiche Arduino Uno

Microcontroller	ATmega328
Tensione di funzionamento	5V
Tensione in ingresso (consigliato)	7-12V
Tensione in ingresso (limiti)	6-20V
Pins Digital I / O	14 (di cui 6 forniscono PWM)
Pins di ingresso analogico	6
Corrente di CC per Pin O / I	40 mA
Corrente DC 3.3V per Pin	50 mA
Memoria Flash	32 KB (ATmega328) di cui 0,5 KB utilizzato da bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Frequenza di clock	16 MHz
Lunghezza	68,6 millimetri
Larghezza	53,4 millimetri
Peso	25 g

Uscire dall'approfondimento

Hardware – Parte 1

Successivamente sono sviluppati alcuni dei componenti dei quali, è composto Arduino, La scheda è composta da un piccolo e funzionale circuito stampato con dimensioni 7x5 cm circa.

La parte evidenziata è il microcontrollore AVR ATMEGA-328P (quello che eseguirà i nostri programmi).

AVR ATMEGA-328P

Uscire dall'approfondimento

Hardware – Parte 2

Uscire dall'approfondimento

1. Porta USB che ci permette di collegare il nostro Arduino al computer per poi caricare il nostro codice sulla memoria interna della scheda e fornirgli anche l'alimentazione che porta l'USB: 5 Volt.

2. Connettore se vogliamo che il nostro lavoro si alimenti senza collegamento al computer. Questo infatti permette di collegare alla vostra scheda un alimentatore esterno da 7 a 12V che sarà poi ridotta ai canonici 5V e a 3.3V da un regolatore interno.

Hardware – Parte 3

- **5V**: pin da cui si possono prendere i 5V.
- **3.3V**: pin da cui possono prendere 3.3V.
- **GND**: Ground, terra.
- **IOREF**. Questo pin sulla scheda Arduino fornisce il riferimento di tensione con cui il microcontrollore opera. Uno scudo configurato correttamente può leggere la tensione pin IOREF e selezionare la sorgente di alimentazione adeguata o abilitare traduttori di tensione sulle uscite per lavorare con la 5V o 3.3V.

Uscire dall'approfondimento

Ingressi analogici

Permette di acquisire un segnale analogico solitamente fornito dai sensori. Per poter essere processato, prima di tutto il segnale analogico viene passato in un convertitore analogico digitale (A/D). A questa rilevazione gli viene attribuito un valore binario. L'intervallo di tensioni misurabile viene suddiviso in tanti livelli discreti quanti sono i valori esprimibili con i bit a disposizione. Il convertitore di Arduino è a 10 bit e quindi si può esprimere (2^{10}) 1024 valori distinti. Quindi in pratica con Arduino possiamo leggere valori analogici fra 0 e 5V con 1024 livelli discreti. La frequenza massima di lettura senza usare altre shield è di circa 9 kHz che permette quindi di campionare frequenze fino a 4,5 kHz.

Uscire dall'approfondimento

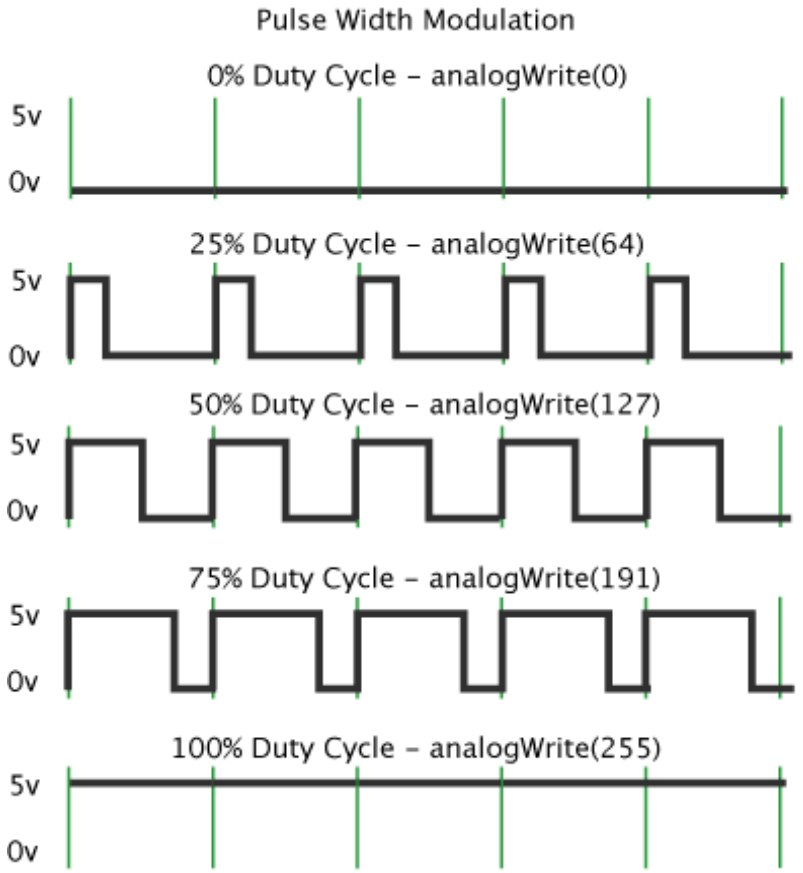
I/O digitali

I pin digitali sono stati progettati in modo che possano passare dalla modalità ingresso alla modalità uscita attraverso la scrittura di opportuni valori in appositi registri di controllo. Quando i pin sono settati come Output, i valori logici che possono assumere sono due: 1 e 0 che rappresentano rispettivamente i 5V e 0V.

Uscire dall'approfondimento

Uscita PWM

Le uscite analogiche-PWM sono contrassegnate dalle ondine presenti su alcuni pin digitali. Possiamo dire che la PWM è una tecnica per ottenere segnali analogici con uscite digitali. Variando la lunghezza dell'impulso posso generare dei valori analogici da 0 a Vcc (5V per Arduino). Il duty-cycle è il rapporto tra la durata dell'impulso al valore logico alto sul periodo in percentuale. La frequenza di lavoro del PWM di Arduino è circa 470Hz.



Uscire dall'approfondimento

Funzioni principali del programma

Le due funzioni principali sono:

- void setup
- void loop

Nella prima chiamata si inizializzano le variabili, e i vari pin, questa parte del programma si effettua una sola volta.
Void loop, invece, si ripete ogni ciclo, qui si inseriscono gli interventi tramite opportune istruzioni che deve effettuare Arduino.

Uscire dall'approfondimento

Variabili

boolean	→ true false
char	→ un carattere ASCII. Un byte in memoria
byte	→ memorizza un intero da 0 a 255
int	→ numero intero da -32768 a 32767.
unsigned int	→ numero da 0 a 65535
long	→ memorizza un numero tra -2147483648 e 2147483647
float	→ numero in virgola mobile, 7 cifre dopo il punto decimale.
Double	→ virgola mobile doppia precisione

Uscire dall'approfondimento

Alcune funzioni

```
pinMode(pin, mode);

digitalRead(pin);

digitalWrite(pin, value);

analogRead(pin);

delay(millisec);

Serial.begin(9600);
```

Uscire dall'approfondimento

Istruzioni di controllo

If-else

```
void loop() {
  if (temperatura<20){
    digitalWrite(Led_R,HIGH);
                          }
   else
    digitalWrite(Led_R,LOW);
    }
```

Break

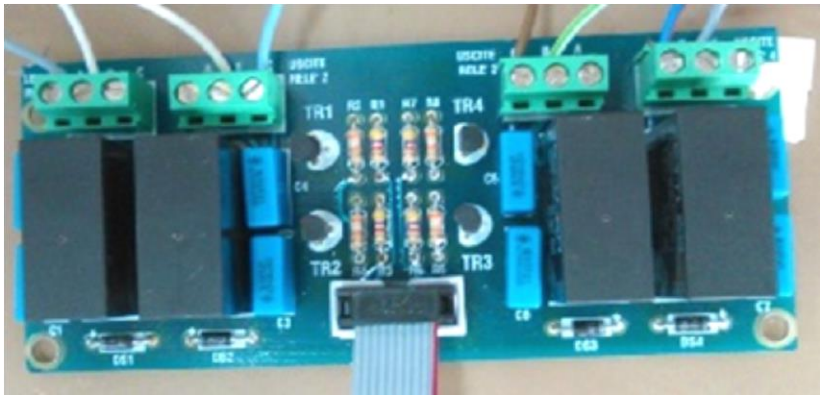
```
void loop() {
  do
  {
   delay(50);
   x = analogRead(A0);
   }
   while (true){
     if (analogRead(A0)==1023)
     break;
   }
```

While-Do

```
void loop() {
  while(pot<500){
    digitalWrite(Led_R,HIGH);
    pot=analogRead(A1);
    }
```

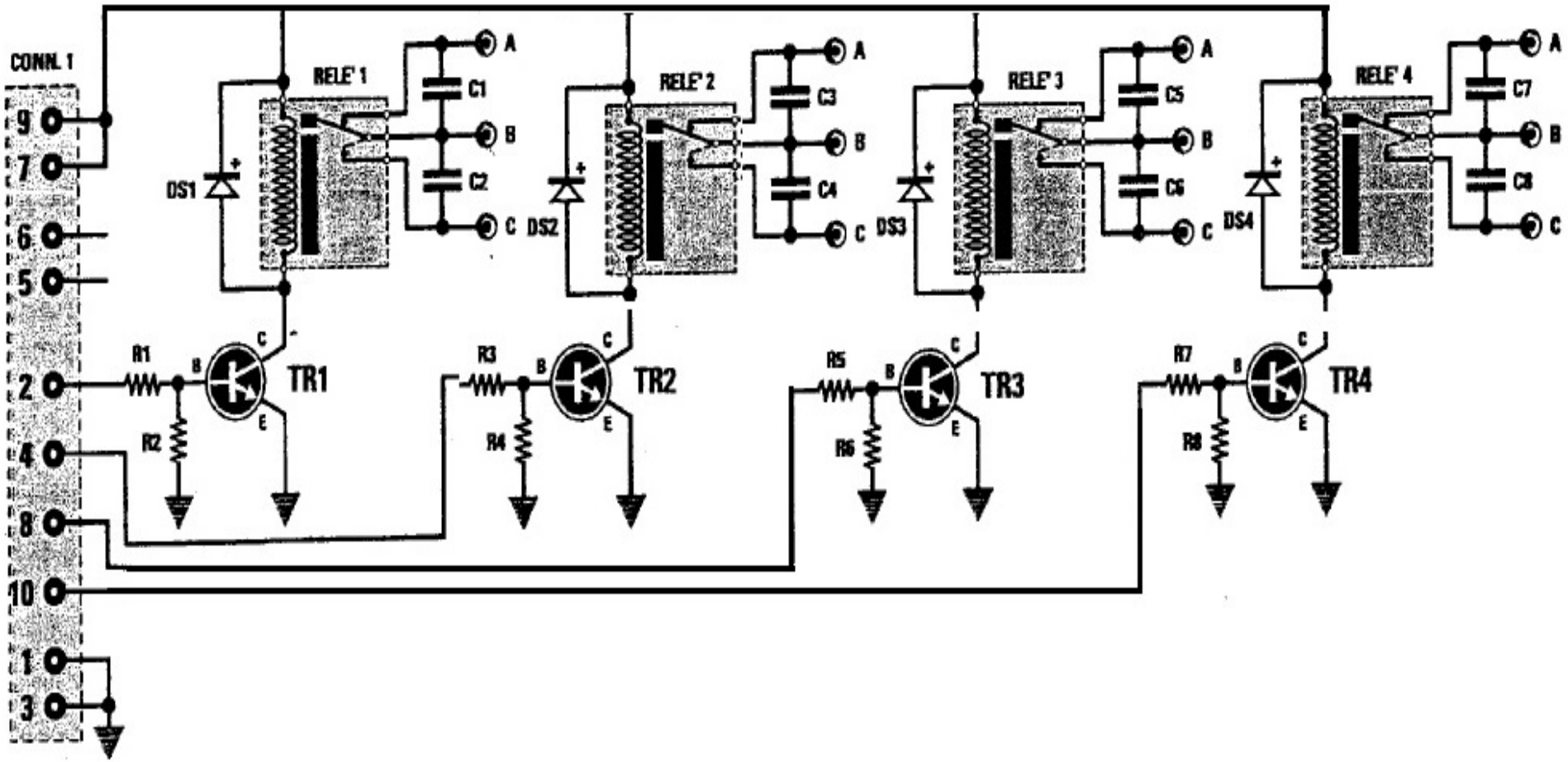
Uscire dall'approfondimento

Scheda relè



Una scheda relè è un dispositivo elettronico formato da 2 o più relè (in questo caso 4). E' comandato dal Arduino (µC) tramite il cavo flat a 10 pin.
Il relè è un dispositivo elettrico comandato dalle variazioni di tensione per influenzare le condizioni di un altro circuito.

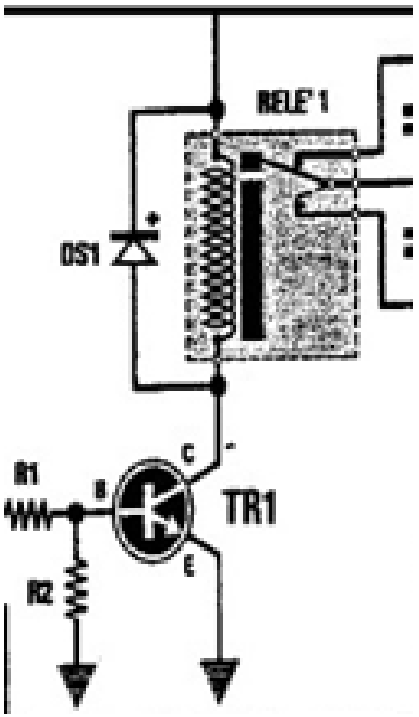
Schema elettrico scheda relè



Autore: Gianfriddo Giuseppe

Approfondimenti sullo schema elettrico della scheda relè

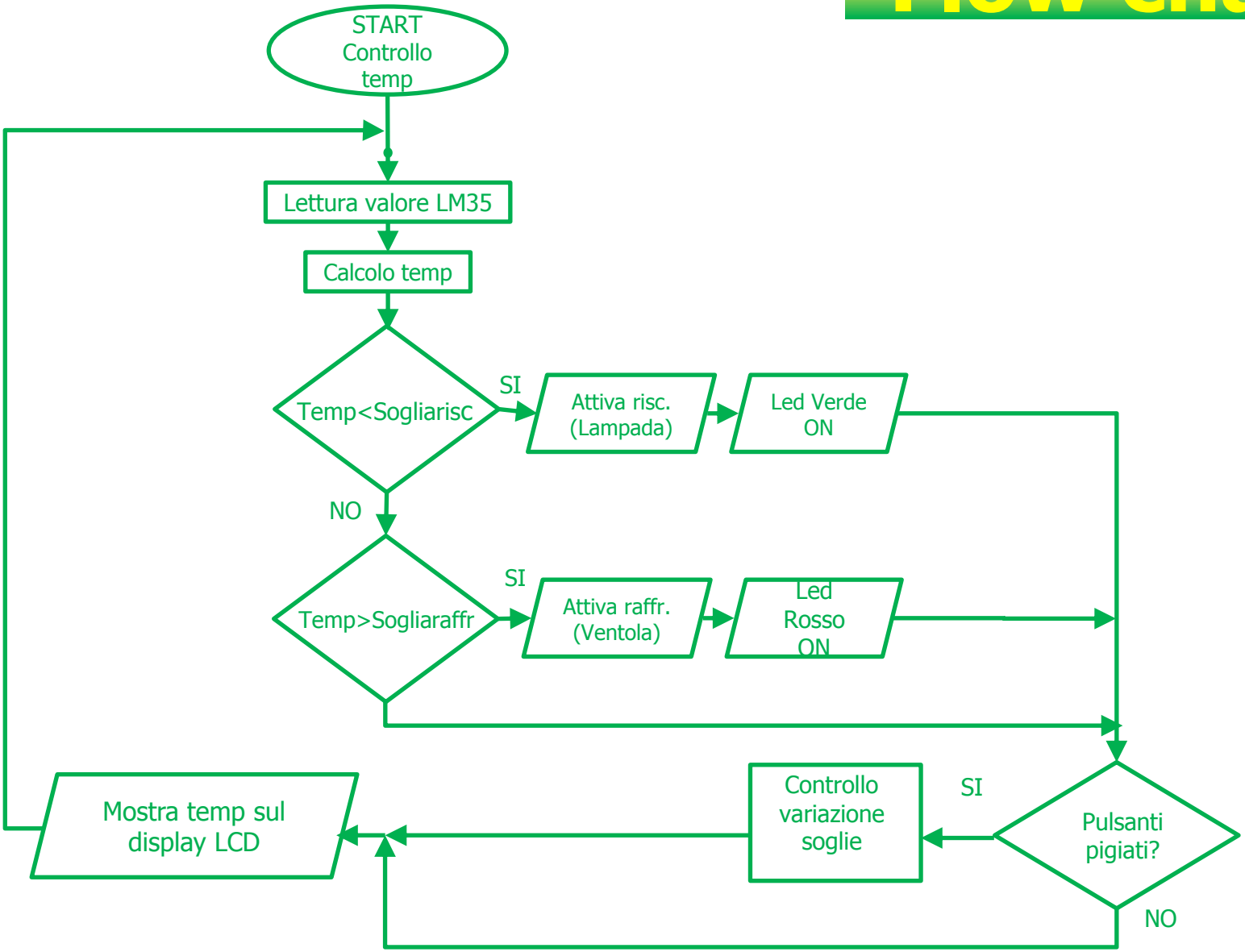
Il diodo in parallelo al relè serve a garantire un percorso conduttivo alla corrente di scarica dell'avvolgimento del relè (che è un'induttanza!), quando si apre il circuito. Si mette per evitare che questa corrente vada a bruciare il transistor di comando. Infatti se il circuito viene aperto bruscamente si sviluppano anche decine di volt ai capi dell'avvolgimento.



Display LCD 16x2

Il display LCD 16x2 HD44780 riportato in figura, ovviamente, serve, nel nostro caso, a mostrare il valore della temperatura corrente.
16 equivale al numero di colonne disponibili e 2 il numero di righe disponibili, per mostrare lettere o numeri o simboli.

Flow chart



Autore: Gianfriddo Giuseppe

Software – Parte 1

```
#include <LiquidCrystal.h>
float temperatura = 0;
float millivolts;
int sensor = 0;
int sogliaraffreddamento = 25;
int sogliariscaldamento = 22;
const int R_LED = 11;
const int G_LED = 8;
const int Ventola = 12;
const int Lampada = 13;
LiquidCrystal lcd(7,6,5,4,3,2);
```

In questa prima parte del programma ho inserito prima la libreria che si utilizza nel programma, *liquidCrystal.h* che permette ad Arduino di utilizzare il display LCD. Successivamente ho dichiarato le variabili, impostando anche le soglie, e poi ho inizializzato il display LCD.

Software – Parte 2

In questa seconda parte troviamo il void setup, dove impostare i pin da utilizzare, quindi utilizzando la funzione *PinMode* per impostare i piedini o come INPUT o OUTPUT.
Infine inizializzo il monitor seriale che ho deciso di utilizzare.

```
void setup ()
{ pinMode(A0,INPUT);
  pinMode(0,INPUT);
  pinMode(1,INPUT);
  pinMode(9,INPUT);
  pinMode(10,INPUT);
  pinMode(R_LED,OUTPUT);
  pinMode(G_LED,OUTPUT);
  pinMode(Ventola,OUTPUT);
  pinMode(Lampada,OUTPUT);
  Serial.begin(9600);       }
```

Software – Parte 3

```
void loop (){

sensor = analogRead(A0);
millivolts = (sensor/1024.0)*5000;
temperatura = millivolts/10;
```

Qui inizia il void loop, Arduino legge il valore ricevuto dal trasduttore, poi calcola il valore in millivolts, e successivamente la temperatura in °C.

Arduino è dotato di un convertitore A/D a 10bit (1024 livelli) e alla tensione di alimentazione di 5 volt corrisponde una risoluzione di 5/1023= 0.00488v (cioè circa 5mV); siccome l'LM35 fornisce in uscita 10mv per grado, se ne può dedurre che la **max precisione** è di circa **0.5 °C** (5/10). Attraverso una semplice proporzione dunque possiamo ottenere la temperatura (°C):
1024 : 5000 = X : millivolts → ***millivolts=(x/1024)*5000***

Considerando i 10 millivolt per grado i 5000 mV diventano → 5000/10 = **500**
Allora → 1024 : 500 = x : °C
Temp= x * 500 / 1024 dove x è il valore che riceviamo

Software – Parte 4

Questa parte di programma riguarda l'incremento delle variabili delle soglie, utilizzando 4 pulsanti:
2 pulsanti quando vengono premuti abbassano le soglie e gli altri 2 alzano le soglie.

```
if (digitalRead (10) == LOW)
{  sogliariscaldamento++; }

if (digitalRead (9) == LOW)
{  sogliariscaldamento--; }

if (digitalRead (1) == LOW)
{  sogliaraffreddamento++; }

if (digitalRead (0) == LOW)
{  sogliaraffreddamento--; }
```


Autore: Gianfriddo Giuseppe

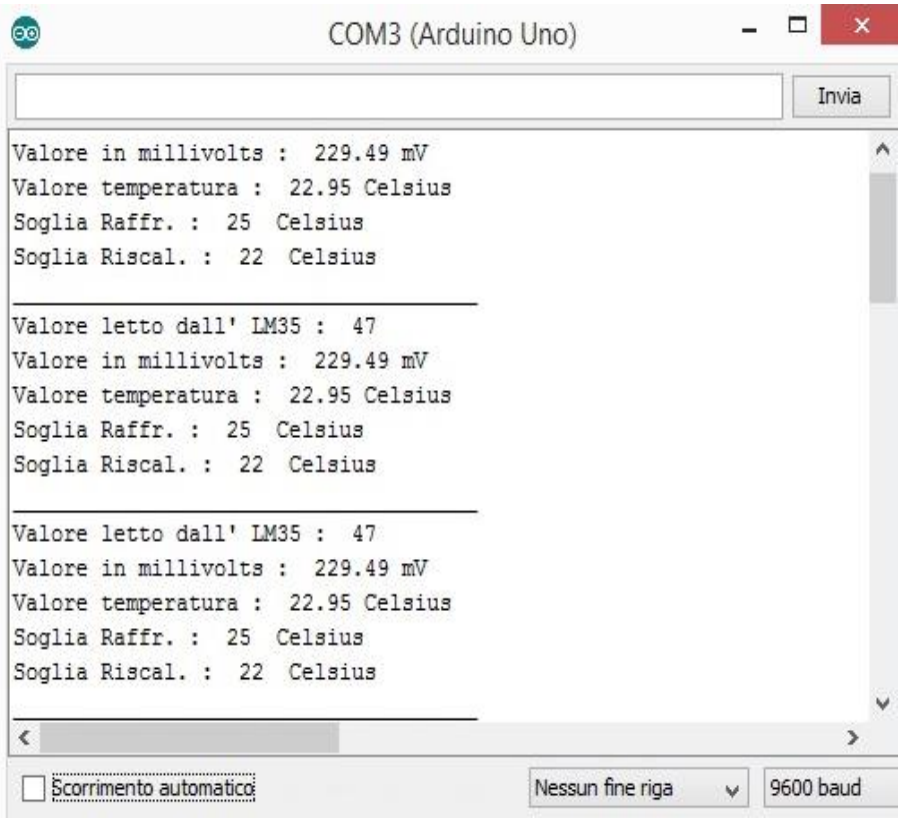
Software – Parte 5

```
if (temperatura<sogliaraffreddamento)
{   digitalWrite(Ventola,LOW);
    analogWrite(R_LED,0);       }
if (temperatura>sogliaraffreddamento)
{  digitalWrite(Ventola,HIGH);
   digitalWrite(Lampada,LOW);
   analogWrite(G_LED,0);
   analogWrite(R_LED,255);     }
    if (temperatura<sogliariscaldamento)
{   digitalWrite(Lampada,HIGH);
    analogWrite(G LED,255); }
if (temperatura>sogliariscaldamento)
{  digitalWrite(Lampada,LOW);
   analogWrite(G_LED,0); }
```

Per controllare gli attuatori, quindi accendere la ventola e il led rosso quando la temperatura supera la soglia di raffreddamento, invece, accende la lampada e il led verde quando la temperatura è minore della soglia di riscaldamento, se invece la temperatura si mantiene all'interno delle due soglie, il microcontrollore non deve attivare nessun attuatore.

Software – Parte 6

```
Serial.print("Valore letto dall' LM35 :  ");
Serial.println(sensor);
Serial.print("Valore in millivolts :  ");
Serial.print(millivolts);Serial.println(" mV");
Serial.print("Valore temperatura :  ");
Serial.print(temperatura);Serial.println(" Celsius");
Serial.print("Soglia Raffr. :  ");
Serial.print(sogliaraffreddamento);Serial.println("  Celsius");
Serial.print("Soglia Riscal. :  ");
Serial.print(sogliariscaldamento);Serial.println("  Celsius");
Serial.println("_____________________________________");
```



Questa parte di programma riguarda l'utilizzo del monitor seriale a disposizione su Arduino, qui vedremo il valore ricevuto dall'LM35, il valore in millivolts, poi quello della temperatura in °C. e infine il valore delle due soglie.

Software – Parte 7

```
lcd.begin(16, 2);
lcd.print("Oggi ci sono:  ");
lcd.setCursor(0,1);
lcd.print(temperatura);
lcd.println(" G.Celsius ");
delay(2000);
lcd.clear();
lcd.print("Soglia risca:");
lcd.print(sogliariscaldamento);lcd.println("C");
lcd.setCursor (0,1);
lcd.print("Soglia raffr:");
lcd.print(sogliaraffreddamento);lcd.println("C");
delay(2000);
}
```

L'ultima parte del programma consiste nel mostrare la temperatura e il valore delle due soglie, sul display LCD.

Funzionamento

Questo progetto, in pratica, serve a controllare la temperatura di un ambiente, impostando la soglia a cui si deve attivare il raffreddamento e la soglia a cui si deve attivare il riscaldamento.

Nel caso della foto:

Soglia risca: → 23°C

Soglia raffr: → 25°C

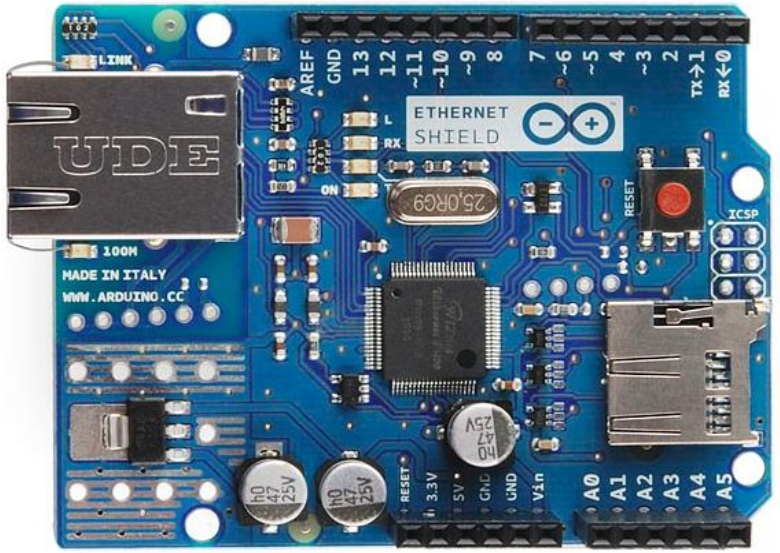
Come accennato precedentemente, le due soglie possono essere variate utilizzando 4 pulsanti.

Dove si può applicare questo progetto?

Un progetto simile potrebbe essere utilizzato nel campo dell'agricoltura (un settore molto sviluppato nelle nostre zone) per il controllo della temperatura all'interno delle serre.

Come si può migliorare

Per incrementare le funzioni di questo circuito si potrebbe aggiungere un controllo remoto, per esempio utilizzando la ***Ethernet Shield***, quindi, permettere all'utente di controllare comodamente da qualunque posto, la situazione del posto in cui è installato il circuito di controllo della temperatura.



FINE

Gianfriddo Giuseppe

Grazie della cortese attenzione!