## Istituto Tecnico Industriale Statale “C. Zuccante” – Mestre (Ve)

a.s. 2018-2019

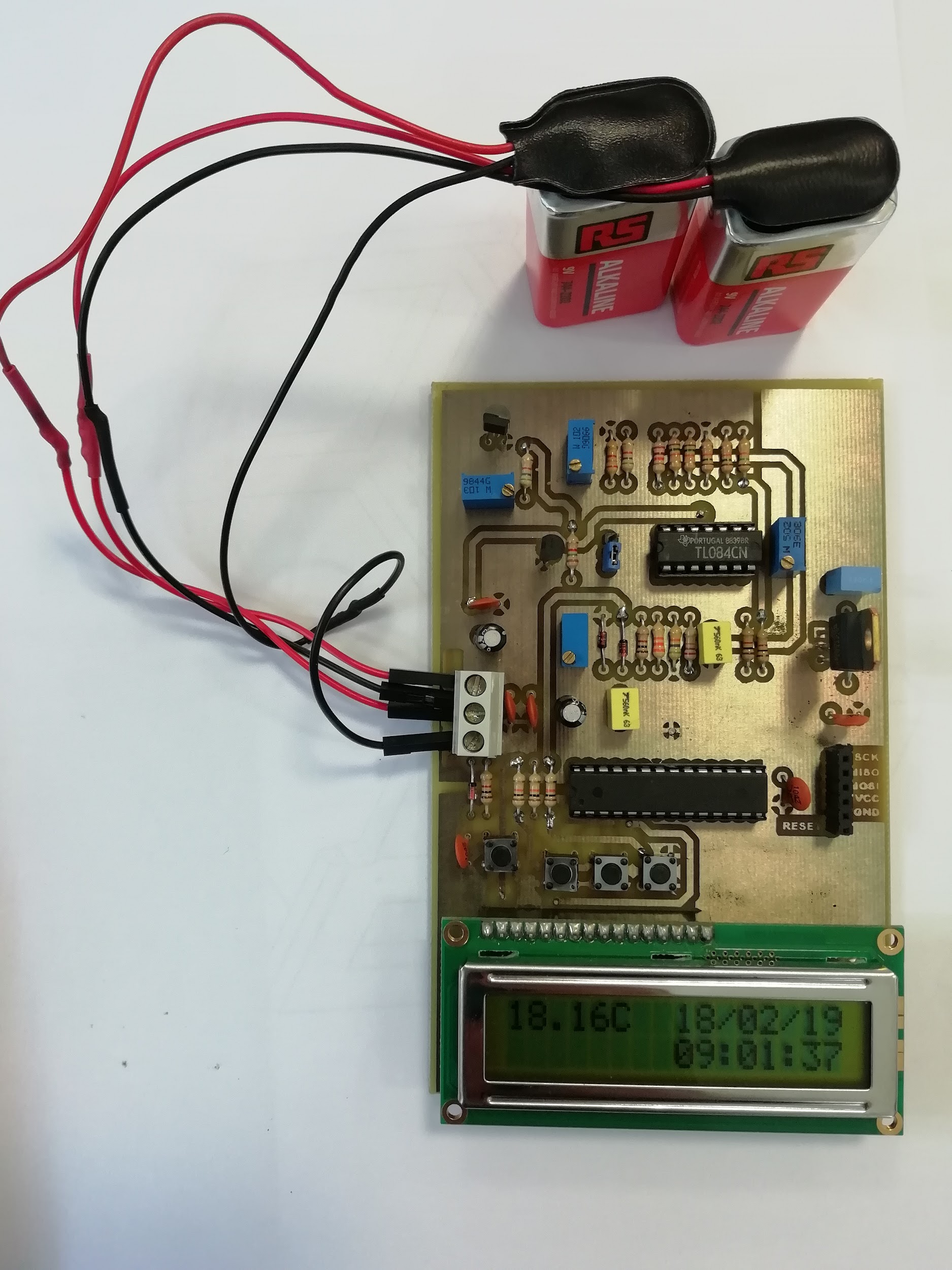
**PROGETTO DI TPSEE N°1**

Progettazione e realizzazione

di un

termometro digitale

TEMPERATURA DI LAVORO 0°C - 50°C

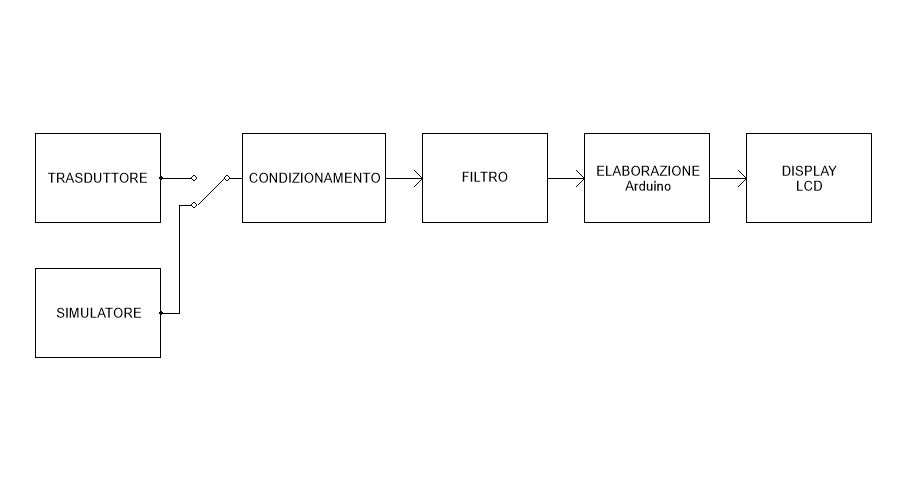


Marigo Andrea

Introduzione:

Realizzazione di un termometro digitale con range di temperature di lavoro 0°C – 50°C con visualizzazione di: temperatura rilevata, data e ora su display LCD.

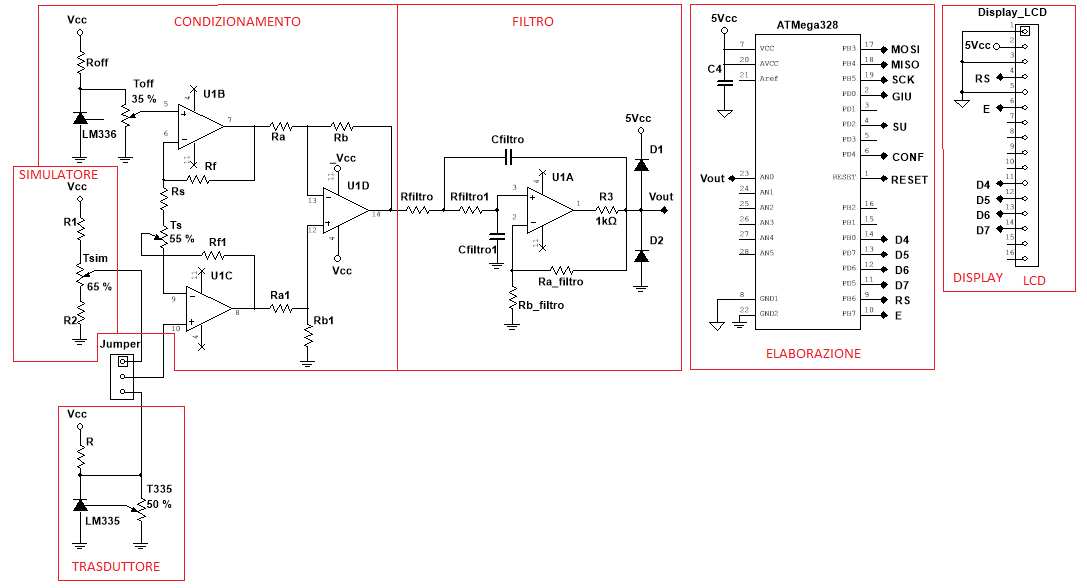
Schema a blocchi:



Il termometro digitale è costituito da:

* Trasduttore, il quale serve per rilevare la temperatura ambientale, può essere sostituito da un simulatore;
* Condizionamento, questo blocco ha lo scopo di aumentare la precisione delle rilevazioni di temperatura, facendo corrispondere il range di tensione fornito dal sensore (2,73V – 3,23V) con quello richiesto in ingresso da Arduino (0-5V), in modo da utilizzare tutti i 10 bit del suo convertitore;
* Filtro, serve per eliminare eventuali disturbi del segnale;
* Elaborazione, permette di gestire i dati rilevati;
* Display, serve per visualizzare data, ora e temperatura.

Schema elettrico:

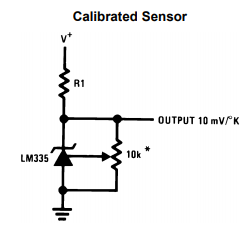


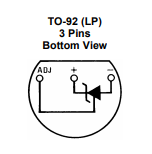
Nello schema elettrico sono evidenziati i vari blocchi in cui è suddiviso il circuito:

* Nel blocco del trasduttore, c’è il sensore di temperatura LM335, che può essere simulato da un generatore di tensione formato da un partitore di tensione e un trimmer;
* Nel blocco del condizionamento è presente una parte di offset, costituita da un reference diode LM336, e una di amplificazione, creata mediante l’uso di un amplificatore per strumentazione;
* Il blocco del filtro è costituito da un passa basso del secondo ordine VCVS;
* Il blocco dell’elaborazione è formato dal programmatore ATMEGA 328P;
* Il display è un LCD a 2 linee con 32 caratteri.

Componenti utilizzati:

LM335:

questo componente ci permette di misurare la temperatura ambientale, la sua tensione di breakdown direttamente proporzionale alla temperatura assoluta è di 10mV/°K. Prevede un terzo pin (adj) che permette la calibrazione del sensore, utilizzando lo schema in figura:



TL084:

è un amplificatore operazionale con ingressi JFET che nel nostro caso ci permette di creare l’amplificatore per strumentazione.

Caratteristiche elettriche:

Tensione di alimentazione: 5V<+Vcc<15V -15V<-Vcc<-5V

Tensione di offset dell’ingresso: 3mV

Questa è la differenza di potenziale tra gli ingressi, quando si ha un ideale corto virtuale tra essi.

Corrente assorbita dagli ingressi:

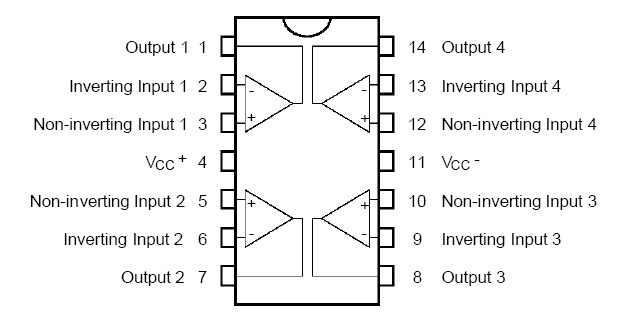
dipende dalla corrente di offset e di bias: IIO =|I+ - I- |=5pA

IIB =(I+ + I- )/2=30pA

Slew rate: 13V/us

è la velocità di variazione della tensione di uscita

CMRR: 86dB

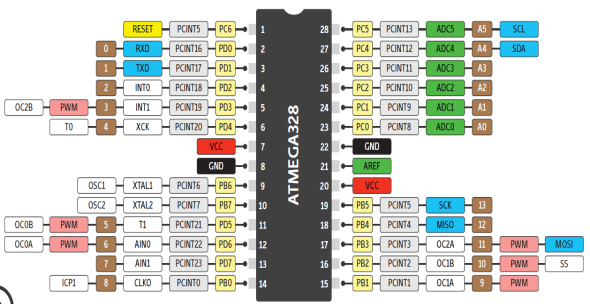
è il rapporto tra il guadagno differenziale e il guadagno in modo comune



ATMega 328:

è un micro controllore creato da ATMEL con una architettura Harvard modificata a 8 bit RISC

La piedinatura del circuito integrato è la seguente:



PIN ATMega e scopo nel nostro progetto:  
17 MOSI programmazione dell’ATMega

18 MISO programmazione dell’ATMega

19 SCK programmazione dell’ATMega

2 GIU’ gestione dei bottoni che controllano il display

4 SU gestione dei bottoni che controllano il display

6 CONFERMA gestione dei bottoni che controllano il display

14 D4 programmazione e gestione del display

13 D5 programmazione e gestione del display

12 D6 programmazione e gestione del display

11 D7 programmazione e gestione del display

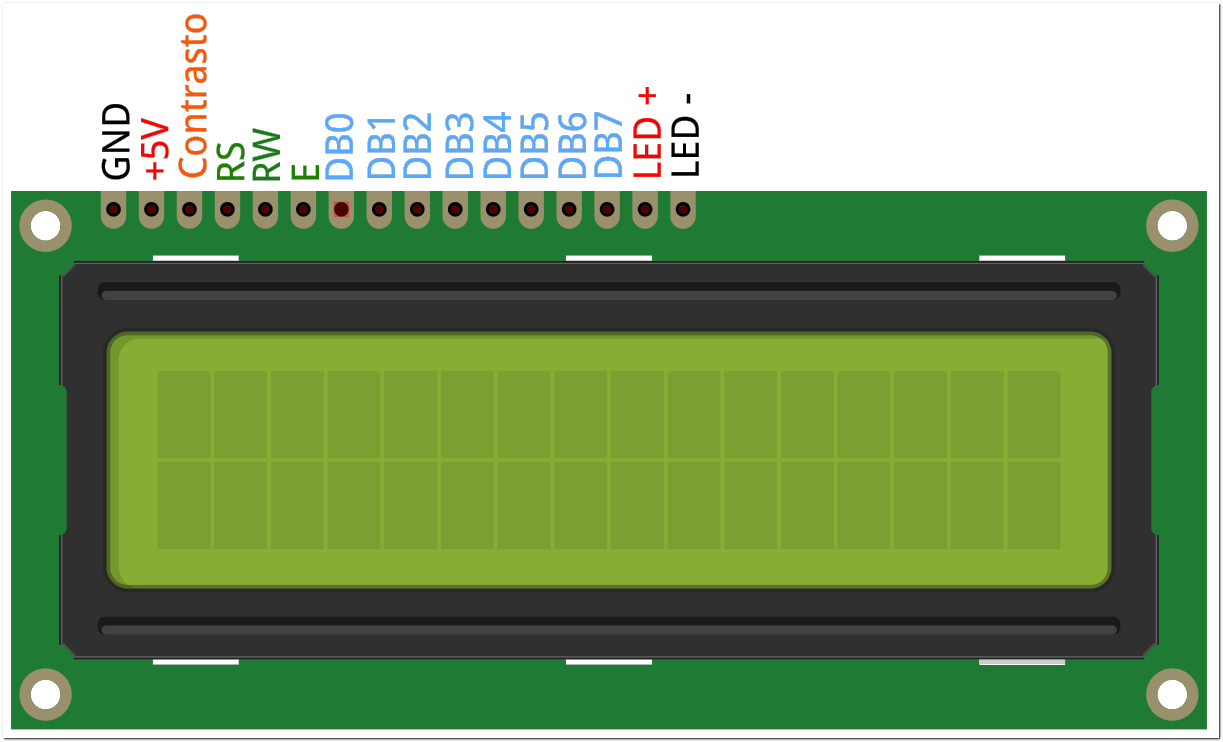
9 RS programmazione e gestione del display

10 E programmazione e gestione del display

Hitachi hd44780:

questo display ci permette di visualizzare tutte le informazioni che inseriamo su 2 righe e 16 colonne così facendo noi possiamo decidere su ogni casella cosa visualizzare

| **Numero del pin** | **Funzione del pin** |
| --- | --- |
| 1 | Vss (Massa) |
| 2 | Vcc (Genericamente 5V) |
| 3 | Vee (Controllo contrasto, da 2V a -3V secondo i tipi) |
| 4 | R/S (0 per selezionare l'invio di un comando, 1 per i dati) |
| 5 | R/W (0 per selezionare la scrittura di dati o comandi, 1 per la lettura dei dati o dello stato) |
| 6 | E (inizia il ciclo di scrittura o lettura, secondo R/S e R/W) |
| 7 | D0 (Bus dati) |
| 8 | D1 (Bus dati) |
| 9 | D2 (Bus dati) |
| 10 | D3 (Bus dati) |
| 11 | D4 (Bus dati) |
| 12 | D5 (Bus dati) |
| 13 | D6 (Bus dati) |
| 14 | D7 (Bus dati) |
| 15 | A (Vcc retroilluminazione, se presente) |
| 16 | K (Vss retroilluminazione, se presente |



CALCOLI

### LM335:

Calcoli per dimensionare la resistenza che regola la corrente entrante nel sensore:

dal datasheet del sensore di temperatura ricaviamo le seguenti informazioni:

-Operating temperature: -40°C < T < +100°C

-Operating current: 0.4mA < I < 5mA ITYP=1mA

Il range di temperature a cui vogliamo far lavorare il nostro termometro sarà:

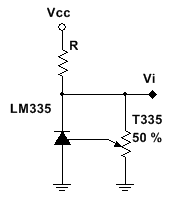
0°C ≤ T ≤ 50°C

Considerando che il sensore ha una tensione di rottura direttamente proporzionale alla temperatura assoluta a 10mV/°K, le tensioni fornite in uscita da esso saranno:

273°K ≤ T ≤ 323°K

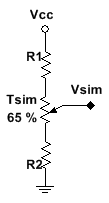
2.73V ≤ VOUT ≤ 3.23V

Inoltre è necessario collegare al pin adjustment del LM335 un trimmer da 10k𝝮, il quale permette di tarare il sensore per ridurre le inaccuratezze nella lettura della temperatura.



### SIMULATORE:

Calcoli per dimensionare le resistenze del simulatore in modo che fornisca le stesse tensioni che può fornire il sensore:



### CONDIZIONAMENTO:

Per creare il blocco di condizionamento viene usato un amplificatore per strumentazione:

Mediante questo blocco bisogna far coincidere il range di tensioni fornito dal trasduttore/simulatore con quello che può accettare in ingresso l’Arduino: 0÷5V

Sottraendo una tensione di offset si fanno coincidere gli estremi inferiori dei due range:

0V ≤ (V335-VOFF) ≤ 0.5V

Moltiplicando per un certo guadagno si fanno coincidere gli estremi superiori:

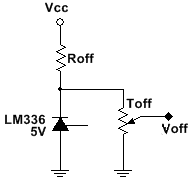
0V ≤ VOUT ≤ 5V

Questo guadagno è costituito dal guadagno dell’amplificatore e da quello del filtro:

#### OFFSET:

Calcoli per dimensionare la resistenza del circuito che genera la tensione di offset

si decide di usare TOFF=10k𝝮



Si dovrà impostare il trimmer in modo da avere VOFF=2.73V

#### AMPLIFICAZIONE:

L’amplificatore è diviso in due stadi ognuno dei quali ha un proprio guadagno e un errore che si somma al segnale di ingresso, per questo bisogna cercare di ridurre il più possibile il guadagno del secondo stadio, infatti il rumore che viene introdotto nel primo stadio viene amplificato del guadagno totale, mentre il rumore del secondo stadio viene amplificato solo del secondo guadagno, quindi se esso è uguale a 1 il rumore totale sarà minore.

Fissando RF=15kΩ:

É necessario un valore molto preciso di RS quindi si utilizza un trimmer, con una resistenza fissa in serie per evitare corto circuiti:

### FILTRO:

Calcoli per dimensionare gli elementi che costituiscono il filtro, il quale è un butterworth passa-basso VCVS del 2°ordine:

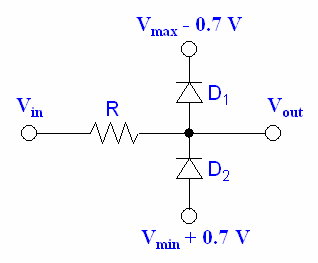
essendo butterworth, ξ=0.707

se vengono utilizzati e :

AO=guadagno in banda passante=1.586

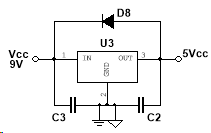
ELABORAZIONE E DISPLAY:

In cascata al filtro sono stati posti due diodi clamp in modo che la tensione entrante nell’ATMega non lo danneggi:

nel nostro circuito i due diodi sono stati collegati a 5V e massa, in questo modo:

-se Vin supera i 5V, D2 è polarizzato inversamente e non conduce, mentre D1 conduce e devia la corrente verso l’alimentazione, così Vout, collegata al pin dell’ATMega, non supera il valore limite;

-se Vin scende al di sotto dei 0V, D1 è polarizzato inversamente, mentre D2 conduce e dirige la corrente verso massa nel nostro circuito e non verso il pin del controllore.

Per avere a disposizione nel circuito i 5V necessari ad alimentare l’ATMega e il display è stato utilizzato un LM7805:

Infine per gestire i pulsanti del display sono state utilizzate delle resistenze di pull-up da 10kΩ, in modo che i pin fossero attivi bassi.

Programmazione:

per utilizzare il display LCD abbiamo dovuto programmare l’ATMega utilizzando l’IDE di Arduino

Il programma che abbiamo realizzato è il seguente:

#include <LiquidCrystal.h>

LiquidCrystal lcd(12,11,5,4,3,2); //Funzione per indicare i collegamenti

//tra display LCD e Arduino

//lcd(RS, E, D3, D2, D1, D0)

const float Tmax=50; //Temperatura massima raggiungibile

const int ConvertionBits=10; //Numero di bit del convertitore di Arduino

int DigitalValue=0;

float Temperature=0;

/\*\*

\* Gli array sottostanti contengono le informazioni di ogni variabile usata, nell'ordine:

\* secondi, minuti, ore, giorno, mese, anno

\*/

int X[6]={14,11,8,8,11,14}; //Numero della colonna sul display di ogni variabile

int Y[6]={1,1,1,0,0,0}; //Numero della riga sul display di ogni variabile

int Minimum[6]={0,0,0,1,1,0}; //Valore minimo di ogni variabile

int Maximum[6]={59,59,23,0,12,99}; //Valore massimo di ogni variabile

int Array[6]={0,0,0,1,1,0}; //Valore iniziale di ogni variabile

int initial\_seconds=0;

int previousSecond=0;

/\*\*

\* Pin e valori dei pulsanti usati per gestire il display

\*/

int confirmationPin=6;

int Pin\_Up=7;

int Pin\_Down=8;

int confirmationButton=1;

int Button\_Up=1;

int Button\_Down=1;

void setup() {

pinMode(confirmationPin,INPUT);

pinMode(Pin\_Down,INPUT);

pinMode(Pin\_Up,INPUT);

//Inizializzazione display

lcd.begin(16,2);

lcd.clear();

delay(1000);

initial\_SettingTime();

initial\_seconds=millis()/1000;

}

/\*\*

\* Funzione che viene chiamata ogni volta che bisogna

\* stampare la data e l'ora

\*/

void Print\_Time\_Date (unsigned int settingMode){

for(unsigned int i=0; i<6; i++){

//settingMode=1 se si è nella parte iniziale del programma

//in cui si imppostano data e ora iniziali

if(settingMode==1){

if(i==3){

lcd.setCursor(0,0);

lcd.print("Date:");

}

if(i==2){

lcd.setCursor(0,1);

lcd.print("Time:");

}

}

lcd.setCursor(X[i],Y[i]);

//se la variabile ha un valore <10 verrà stampata con

//uno 0 davanti

if(Array[i]<10){

lcd.print("0");

lcd.setCursor((X[i]+1),Y[i]);

}

lcd.print(Array[i]);

if(i!=0)

lcd.setCursor((X[i]+2),Y[i]);

if(i==3 || i==4)

lcd.print("/");

if(i==2 || i==1)

lcd.print(":");

}

return;

}

/\*\*

\* Funzione per impostare il giorno massimo in base al mese e all'anno in cui si è

\* gestisce anche gli anni bisestili

\*/

void Set\_MaximumDay(int Month, int Year){

int Max;

if(Month==11 || Month==9 ||Month==4 || Month==6)

Max=30;

if(Month==1 || Month==3 || Month==5 || Month==7 || Month==8 || Month==10 || Month==12)

Max=31;

int leapYear=Year%4;

if(Month==2 && leapYear==0)

Max=29;

if(Month==2 && leapYear!=0)

Max=28;

Maximum[3]=Max;

}

/\*\*

\* Funzione per impostare l'ora e la data iniziali

\*/

void initial\_SettingTime(){

lcd.blink();

for(unsigned int i=5; i>0; i--){

while(confirmationButton!=0){

//Legge se i pulsanti sono stati premuti

confirmationButton=digitalRead(confirmationPin);

Button\_Up=digitalRead(Pin\_Up);

Button\_Down=digitalRead(Pin\_Down);

Print\_Time\_Date(1);

lcd.setCursor(X[i]+2,Y[i]);

//se i pulsanti sono stati premuti agisce di conseguenza

if(Button\_Up==0)

Array[i]++;

if(Button\_Down==0)

Array[i]--;

if(i==3)

Set\_MaximumDay(Array[i+1], Array[i+2]);

//se la variabile raggiunge il suo massimo+1

//viene portata al suo valore minimo

if(Array[i]==Maximum[i]+1)

Array[i]=Minimum[i];

//se la variabile raggiunge il suo minimo-1

//viene portata al suo valore massimo

if(Array[i]==Minimum[i]-1)

Array[i]=Maximum[i];

delay(200);

}

confirmationButton=1;

}

lcd.noBlink();

lcd.clear();

return;

}

/\*\*

\* Funzione per aggiornare, una volta impostata, la data e l'ora

\*/

void Update\_Date\_Time(){

for(unsigned int i=0; i<5; i++){

if(i==3)

Set\_MaximumDay(Array[i+1], Array[i+2]);

//quando la variabile raggiunge il suo massimo viene messa al

//suo minimo e la variabile seguente viene aumentata di 1

//Es. se minuti=61 --> minuti=0 e ore+1

if(Array[i]==Maximum[i]+1){

Array[i]=Minimum[i];

Array[i+1]++;

if(i==0)

initial\_seconds=millis()/1000;

}

}

return;

}

void loop() {

//lettura del valore analogico di tensione della temperatura

DigitalValue=analogRead(A0);

//conversione del valore di tensione in un valore di temperatura in Celsius

//Temperatura\_rilevata : temperatura\_massima = valore\_digitale\_rilevato : 1024

Temperature=(DigitalValue\*Tmax)/(pow(2,ConvertionBits));

lcd.setCursor(0,0);

lcd.print(Temperature);

lcd.print("C");

Print\_Time\_Date(0);

//Aggiornamento dei secondi mediante la funzione millis()

previousSecond=Array[0];

while(previousSecond==Array[0])

Array[0]=(millis()/1000)-initial\_seconds;

Update\_Date\_Time();

}

Collaudo:

dopo aver svolto tutti i calcoli, realizzato lo schema elettrico su Multisim e il PCB su Ultiboard, creato la basetta e saldato tutti i componenti, bisogna effettuare il collaudo:

1. Tarare il circuito

* Spostare il Jumper per scegliere l’utilizzo del simulatore;
* Impostare il trimmer del simulatore in modo da avere sul pin 10 dell’operazionale 3,23V;
* Impostare il trimmer del offset in modo da avere sul pin 5 dell’operazionale 2,73V;
* Verificare che sul pin 23 (AN0) dell’ATMega ci siano 5V e nel caso non ci fossero bisogna impostare il trimmer dell’amplificazione in modo da raggiungerli;
* Impostare il simulatore a 2,23V e verificare se sul pin 23 ci sono 0V, in questo modo si sono calibrati i trimmer del blocco di condizionamento;
* Spostare il jumper per scegliere l’utilizzo del sensore;
* Impostare il trimmer del sensore in modo che si visualizzi sul display la temperatura ambientale esatta, utilizzando come riferimento un altro termometro.

1. Impostare data e ora sul display tramite l’utilizzo dei pulsanti dedicati impostando successivamente anno, mese, giorno, ora e minuti utilizzando il bottone di conferma ogni volta;
2. Visualizzare sul display la variazione della temperatura alle variazione delle condizioni ambientali;
3. Verificare l’aggiornamento automatico di data e ora, lasciando in funzione l’orologio per un po’ di tempo e controllando che l’orario si aggiorni correttamente

Conclusioni:

Il progetto è risultato funzionante nonostante vi sia stata qualche difficoltà lungo la progettazione come capire ed attuare un buon piano di massa e la programmazione dell’ATMega 328. Anche scrivere il programma di Arduino ha richiesto un adeguato lavoro ed impegno visto che siamo voluti essere il più precisi possibile.