

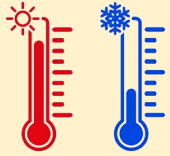
# SENSORI DI TEMPERATURA

*Silvano Sgrignoli*

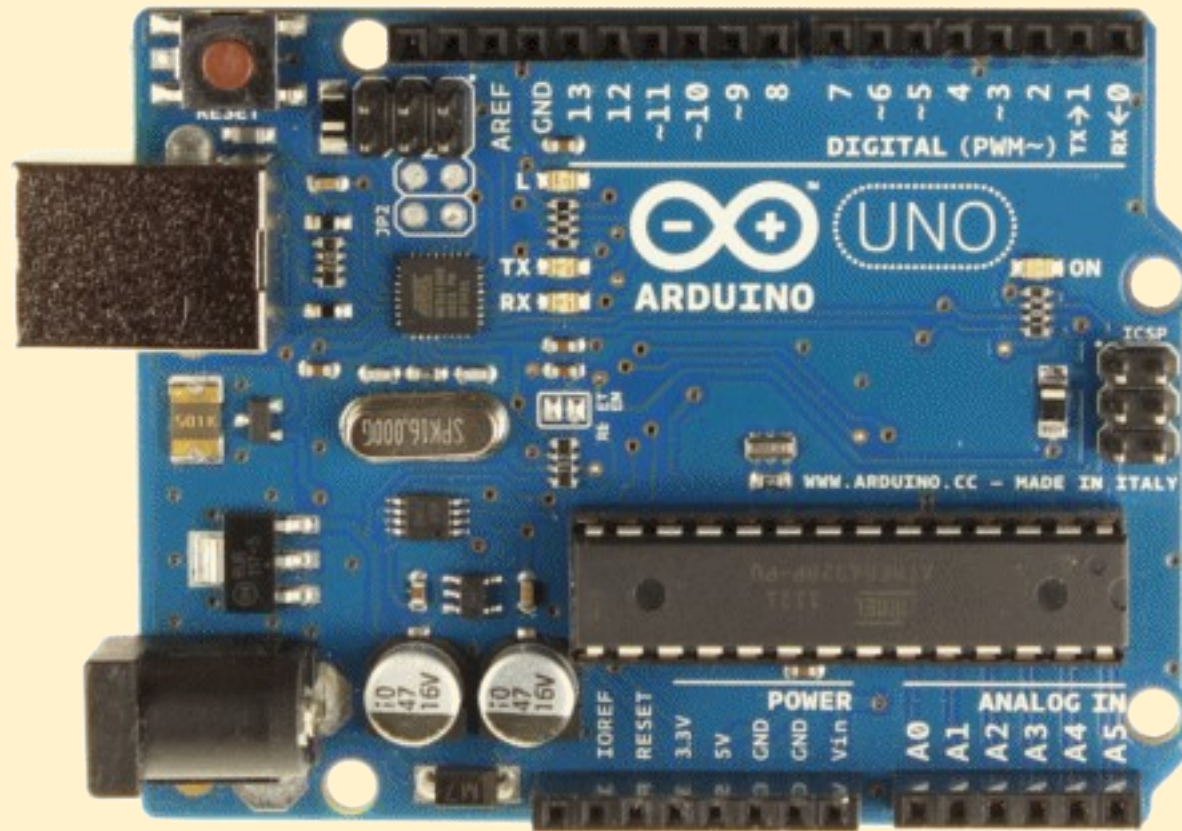
Pavia, 13 ottobre 2021

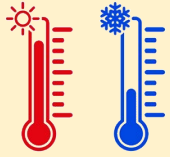


Associazione  
per l'Insegnamento  
della Fisica

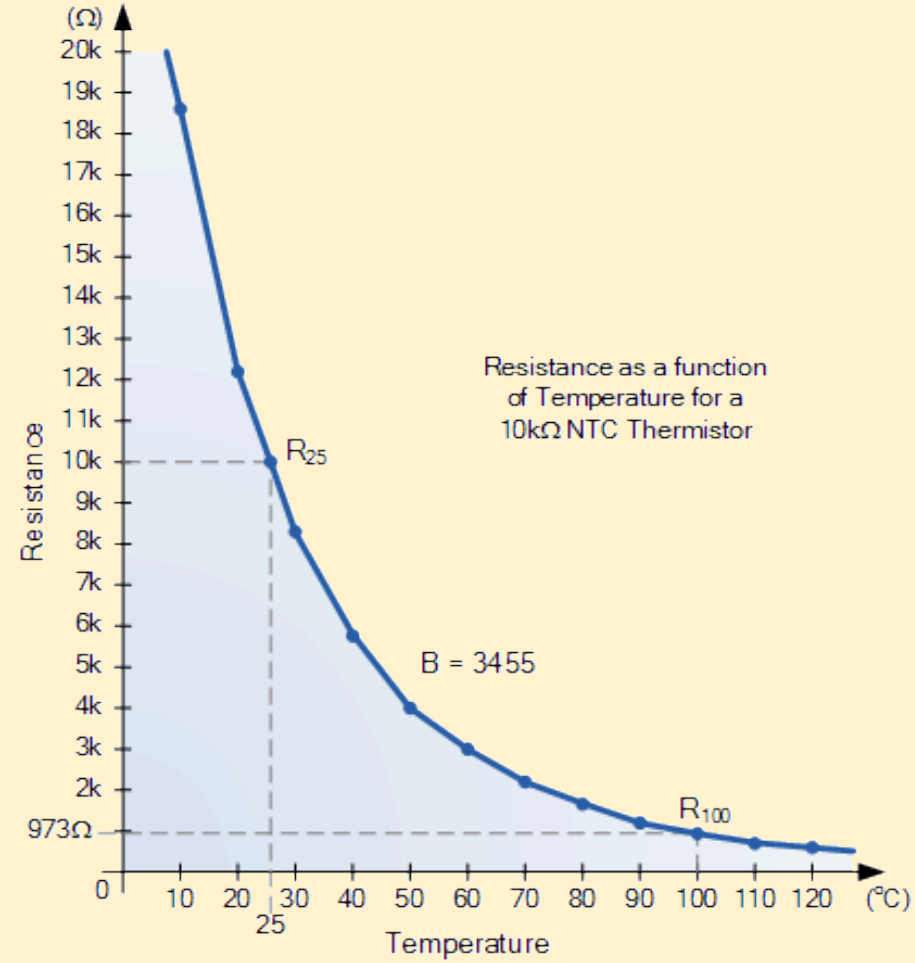
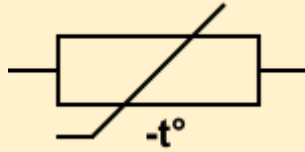


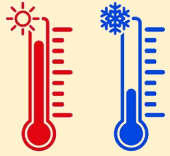
ANALOGICI	DIGITALI
NTC	DS18B20
Diodo	
LM35	



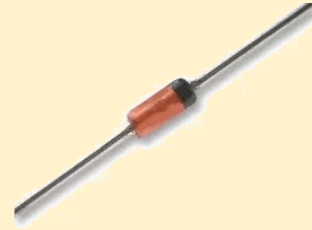


# Termistore NTC - *Negative Temperature Coefficient*

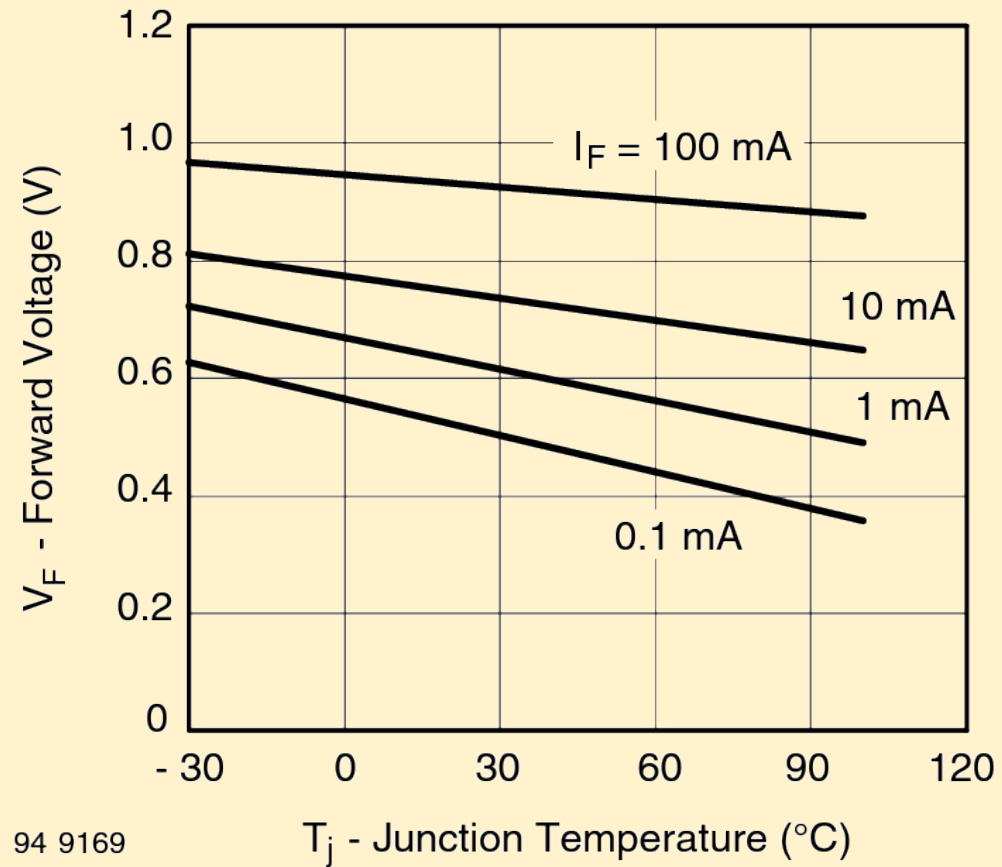




# Diodo al silicio, polarizzazione diretta

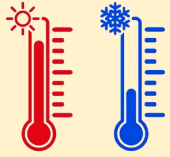


1N4148 Vishay

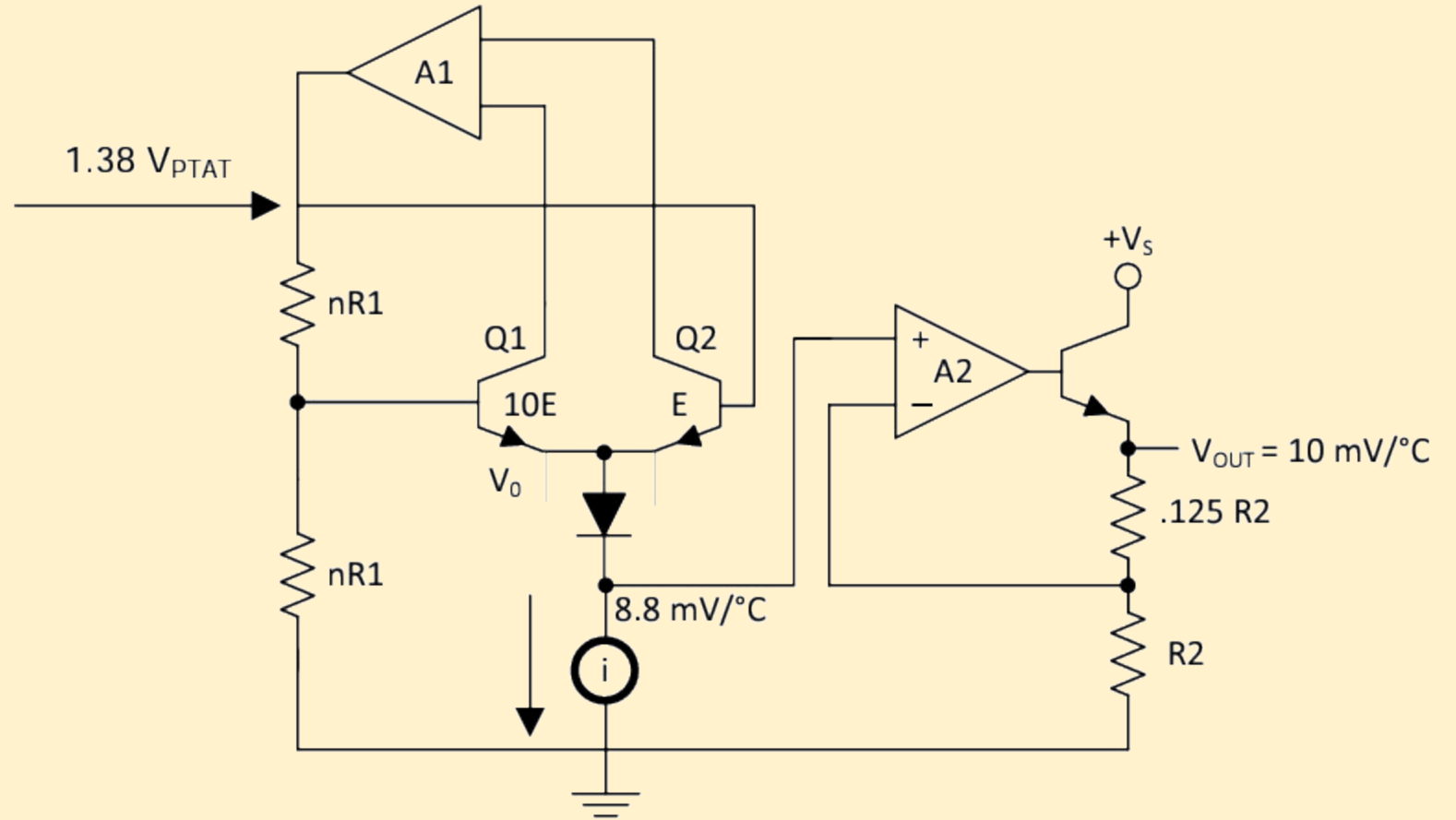
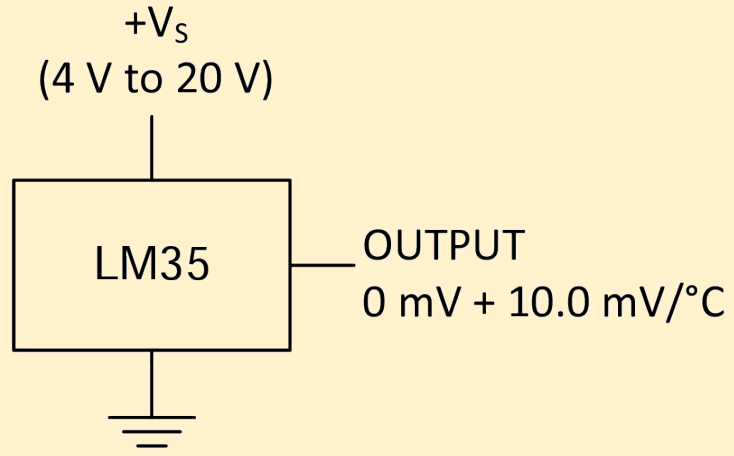


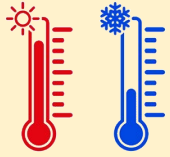
94 9169

Forward Voltage vs. Junction Temperature

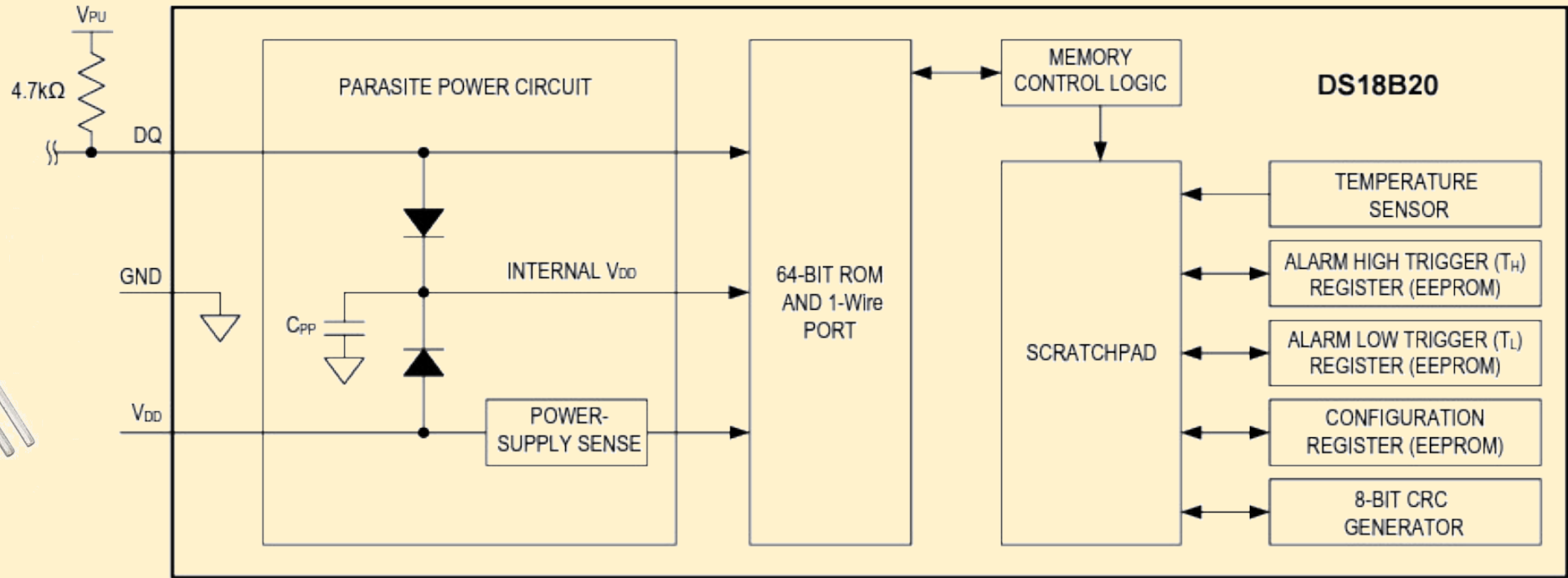
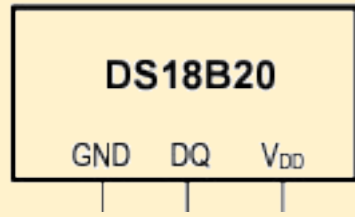


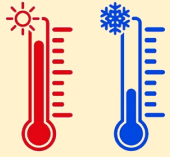
# Circuito integrato LM35



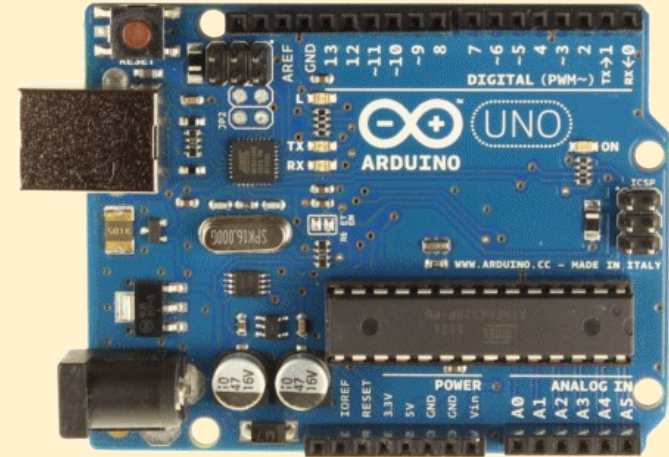
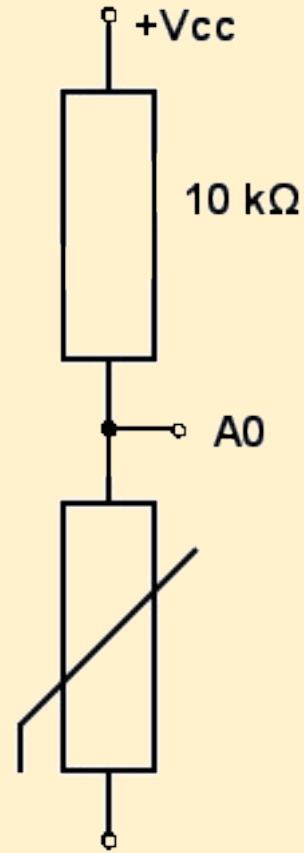


# Circuito integrato DS18B20 (Maxim Integrated)





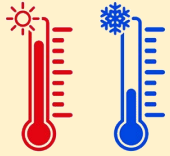
# Termistore NTC - schema di collegamento



Programma: NTC\_prova.ino

$$\frac{R}{R_0} = \exp\left(\beta \left[\frac{1}{T} - \frac{1}{T_0}\right]\right)$$

$$\frac{1}{T} = \frac{1}{\beta} \ln\left(\frac{R}{R_0} \cdot \frac{\sum x}{N \cdot 1023 - \sum x}\right)$$



# Termistore NTC - programma

```
/* NTC measurements with Arduino. Input goes to A0, AREF and NTC
connected to 3.3 V. */
```

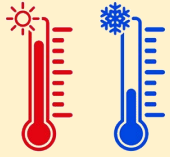
```
#define THERMISTOR A0 // Analog input to connect to
#define N 5 // Number of samples for a single measure
#define R_S 9850 // Series resistor value (Ω), measured
#define T_0 22 // ref. temperature (°C), measured
#define R_0 11220 // Resistance at ref. temperature (Ω), measured
#define BETA 3869 // Thermistor beta coeff. (K^(-1)), computed.
// 3950 K^(-1) is the standard value
#define K_C 273.15 // Conversion factor K <--> °C

float v_divider = 1.0*R_S/R_0;
float rev_T0 = 1.0/(T_0 + K_C);

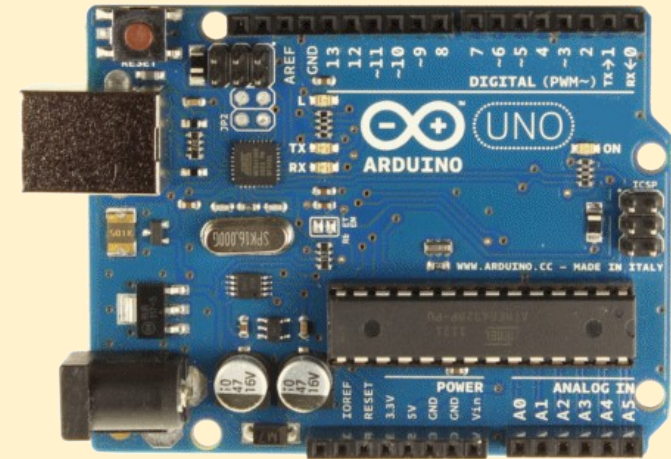
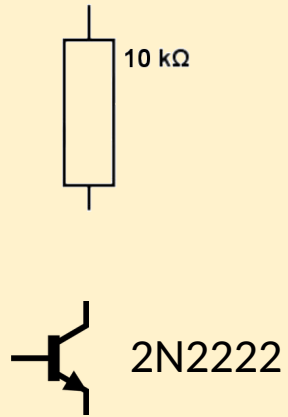
void setup(void) {
  Serial.begin(9600);
  analogReference(EXTERNAL); // Remember to connect ADC and AREF
  // to 3.3 V for greater stability *****
  delay(10);
  analogRead(THERMISTOR); // Adjust ADC for new reference
}
```

```
void loop(void) {
  uint8_t i;
  int adc = 0;
  // take N samples
  for (i=0; i< N; i++) {
    adc += analogRead(THERMISTOR);
    delay(10);
  }
  float temp = log(v_divider*adc/(N*1023-adc))/BETA + rev_T0;
  float t_NTC = 1/temp - K_C;
  Serial.print("Temperatura: t = ");
  Serial.print(t_NTC, 1);
  Serial.println(" °C");
  delay(1000);
}
```

$$\frac{1}{T} = \frac{1}{\beta} \ln \left( \frac{R}{R_0} \cdot \frac{\sum x}{N \cdot 1023 - \sum x} \right)$$

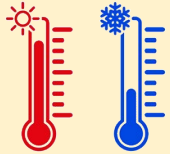


# Diodo - schema di collegamento



3,3 V

Programma: Temp\_diodo.ino



# Diode - programma

```
/* Diode temperature sensor. Remember to use 3.3 V supply for greater
stability */
```

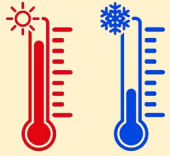
```
#define analogPin A0 // pin connected to sensor
#define V_0 640.6 // voltage drop at 0 °C (mv)
#define V_100 440.6 // voltage drop at 100 °C (mv)
#define deltaT -2.3 // temperature correction, if needed
#define replies 5 // repeated sampling
```

```
void setup() {
  Serial.begin(9600);
  analogReference(INTERNAL);
  analogRead(analogPin); // adjust ADC for new ref.
  delay(10);
}
```

```
void loop() {
  unsigned int sensorRead = 0;
  for (int N = 0; N < replies; N++) {
    sensorRead += analogRead(analogPin);
    delay(5);
  }
  float vdrop = map(sensorRead, 0, 1023*replies, 0, 1100*8);
  float tempC = fmap((float)vdrop, V_0*8, V_100*8, 0, 100);
  Serial.print("t = ");
  Serial.print(tempC + deltaT, 1);
  Serial.println(" °C");
  delay(1000);
}
```

```
// fmap() is a map() function for float
float fmap(float x, float in_min, float in_max, float out_min, float out_max) {
  return (x - in_min)*(out_max - out_min)/(in_max - in_min) + out_min;
}
```





# LM35 – programma

```
/* LM35 measurements with Arduino */
```

```
#define LM35 A0
```

```
#define REF 1.1
```

```
void setup() {
```

```
  Serial.begin(9600);
```

```
  analogReference(INTERNAL);
```

```
}
```

```
void loop() {
```

```
  int aRead = 0;
```

```
  for (int N = 0; N < 10; N++) {
```

```
    aRead += analogRead(LM35);
```

```
    delay(5);
```

```
  }
```

```
  Serial.print("N = ");
```

```
  Serial.print(aRead);
```

```
  float tempC = aRead*REF/102.3;
```

```
  float voltage = aRead*REF/10230;
```

```
  Serial.print("; Vo = ");
```

```
  Serial.print(voltage, 3);
```

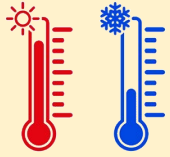
```
  Serial.print(" V; T = ");
```

```
  Serial.print(tempC, 1);
```

```
  Serial.println(" °C");
```

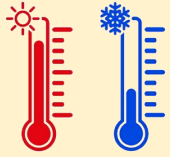
```
  delay(1000);
```

```
}
```

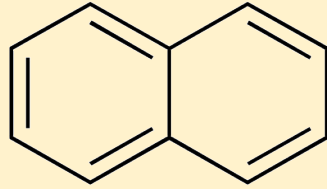


# DIGRESSIONE...

Vedremo dopo l'uso del sensore **DS18B20**

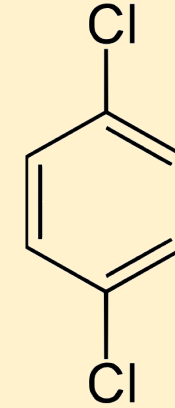
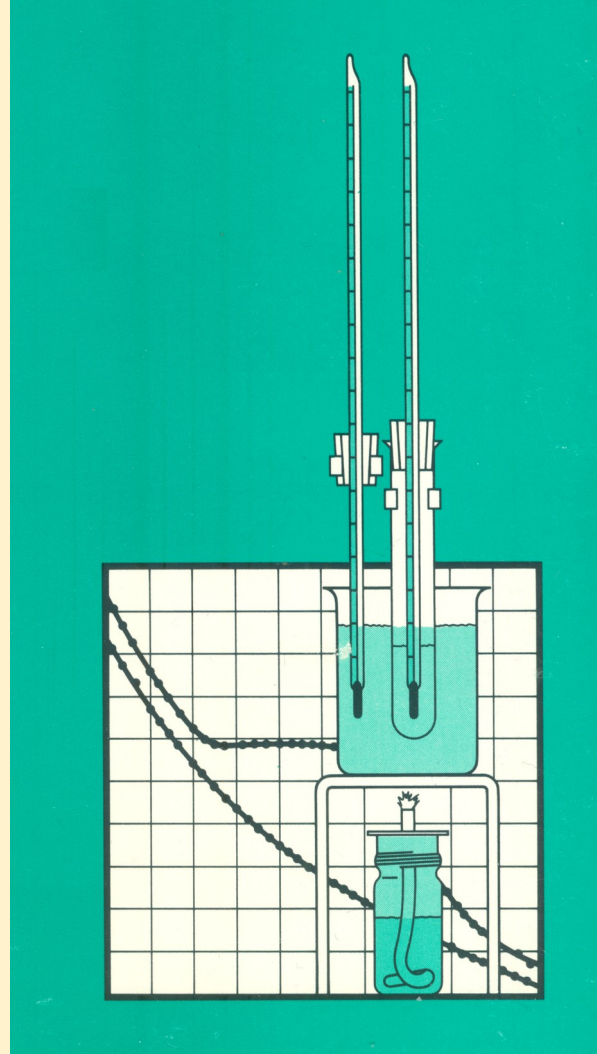


# Solidificazione. IPS (*Introductory Physical Science*)



naftalina

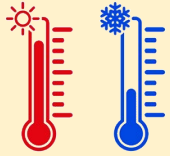
$t_f = 80^\circ\text{C}$



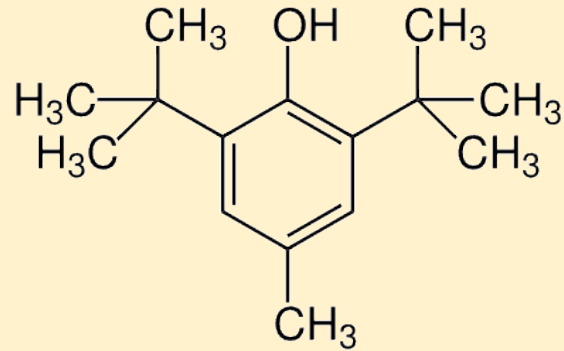
para-diclorobenzolo

$t_f = 53^\circ\text{C}$





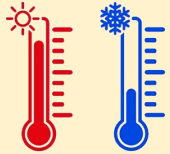
# Solidificazione. IPS (*Introductory Physical Science*, VI ed.)



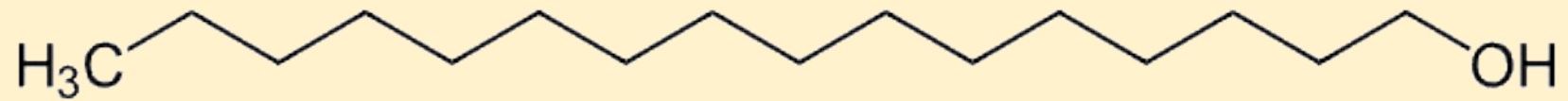
BHT, butilidrossitoluene, E321  
2,6-bis(1,1-dimetiletil)-4-metilfenolo

$$t_f = 69^\circ\text{C}$$





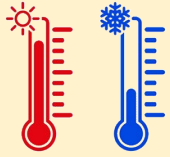
# Solidificazione



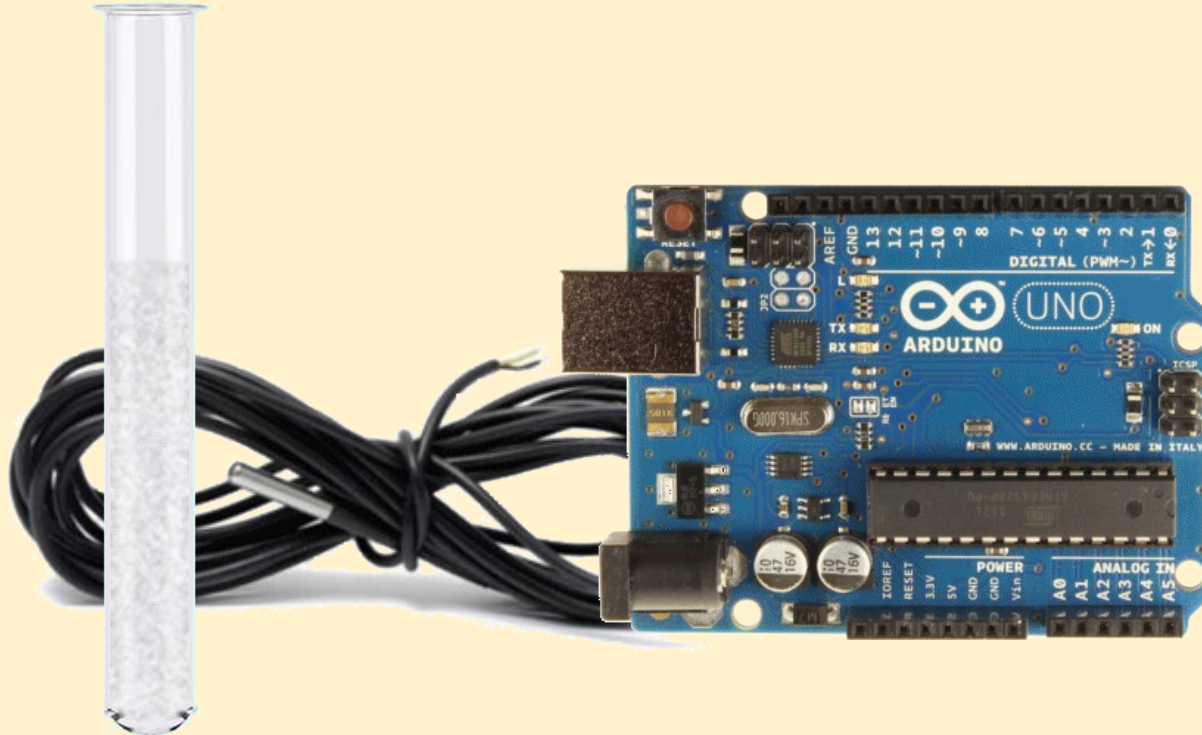
alcol cetilico

$t_f = 49,3 \text{ } ^\circ\text{C}$

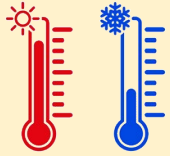




# Solidificazione

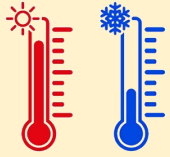


Ora faremo qui l'esperimento...

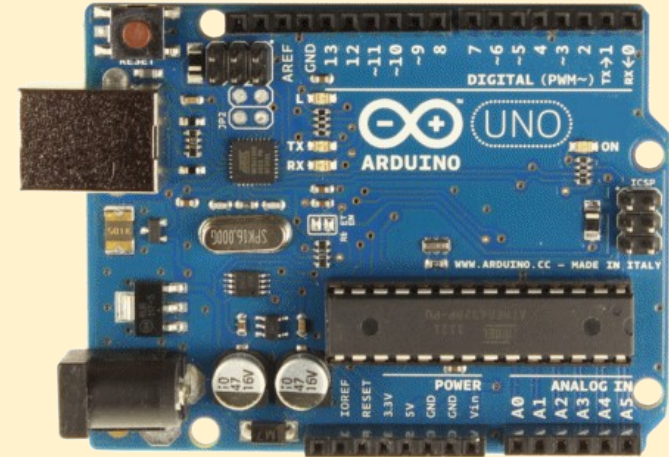
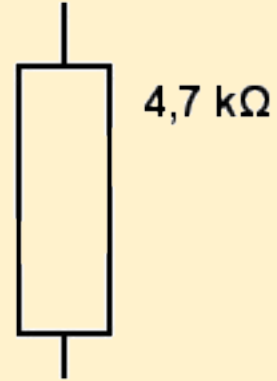
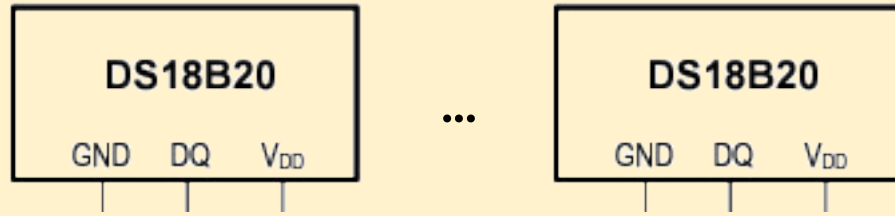


# RIPRENDIAMO...

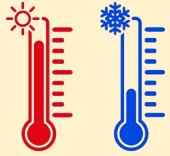
Mentre l'alcol cetilico solidifica, ci occupiamo del sensore **DS18B20**



# DS18B20 - schema d'uso



Programma: 4sensori.ino



# DS18B20 – programma

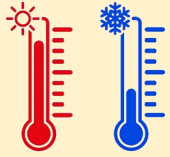
```
/* DS18B20 multiple sensors with Arduino */
```

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2 on the Arduino
#define ONE_WIRE_BUS 2
// Setup a oneWire instance to communicate with
// any OneWire device
OneWire oneWire(ONE_WIRE_BUS);
// Pass oneWire reference to DallasTemperature library
DallasTemperature sensors(&oneWire);
int deviceCount = 0;
float tempC;

void setup(void) {
  sensors.begin(); // Start up the library
  Serial.begin(9600);
  // locate devices on the bus
  Serial.print("Locating devices...");
  deviceCount = sensors.getDeviceCount();
  Serial.print("Found ");
  Serial.print(deviceCount, DEC);
```

```
Serial.println(" devices.");
Serial.println("");
sensors.setResolution(12);
}

void loop(void) {
  // Send command to all the sensors for temperature conversion
  sensors.requestTemperatures();
  // Display temperature from each sensor
  for (int i = 0; i < deviceCount; i++) {
    Serial.print("Sensor ");
    Serial.print(i+1);
    Serial.print(" : ");
    tempC = sensors.getTempCByIndex(i);
    Serial.print(tempC);
    Serial.print((char)176); //shows degrees character
    Serial.print("C | ");
    Serial.print(DallasTemperature::toFahrenheit(tempC));
    Serial.print((char)176); //shows degrees character
    Serial.println("F");
  }
  Serial.println("");
  delay(1000);
}
```



# Riferimenti



<https://bit.ly/3mRbfd>