

# Progetto di un termometro elettronico con Arduino UNO

© Politecnico di Torino

Questo materiale è distribuito gratuitamente ad esclusivo uso degli allievi del Politecnico di Torino per la preparazione all'esame. Ogni altro uso sia commerciale sia divulgativo è espressamente vietato senza il consenso scritto dell'autore

## Parte preliminare

**Verificare il corretto funzionamento dell'ambiente (IDE) e della scheda Arduino UNO**

Usando esclusivamente le funzioni di Arduino:

- Scrivere un programma che faccia lampeggiare il led «L» presente sulla scheda Arduino UNO (connesso al pin 13) alla frequenza di 2 Hz.

*Facoltativo da fare a casa sul Simulatore:*

- Collegare un pulsante al pin numero 3 e modificare il programma precedente per bloccare il lampeggio mentre il pulsante è premuto
  - Collegate il pulsante tra l'ingresso 3 e GND e attivate il resistore di Pullup del pin 3

## Parte preliminare

Verificare il corretto funzionamento dell'ambiente (IDE) e

de

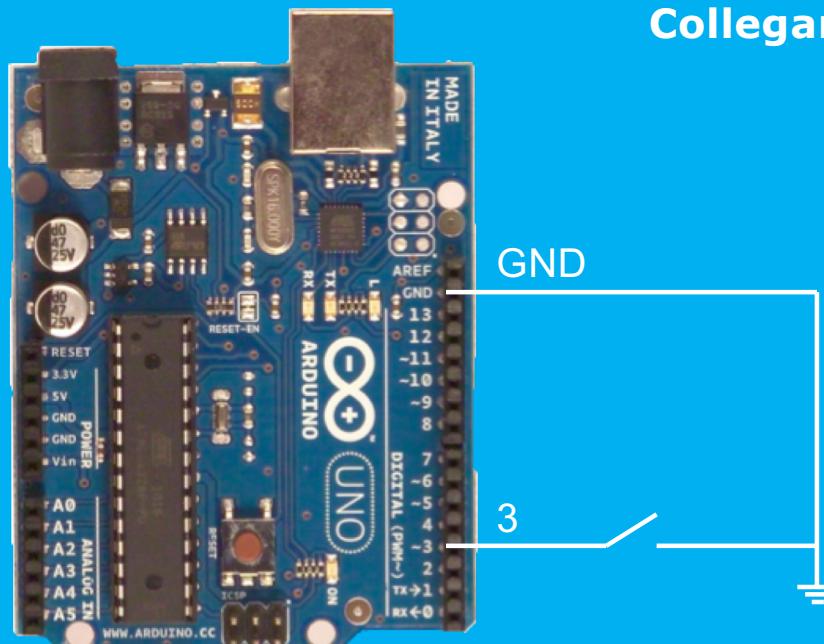
Us

➤

Fa

➤

### Collegamenti



- Collegate il pulsante tra l'ingresso 3 e GND e attivate il resistore di Pullup del pin 3

## Obiettivo

**Si vuole progettare un termometro da usare come sensore in un sistema di controllo della temperatura (LabView)**

Requisiti sistema di misura:

- Incertezza massima nel campo 20 °C – 30 °C: 1 °C
- Risoluzione di misura minima: 0.1 °C
- Velocità di acquisizione: 4 misure/s

Sensore disponibile: sensore analogico LM335

Materiale a disposizione durante il progetto:

- Termo-igrometro del laboratorio (incertezza 0.1 °C)
- Sensore digitale DHT22 (Termo-igrometro AM2303)

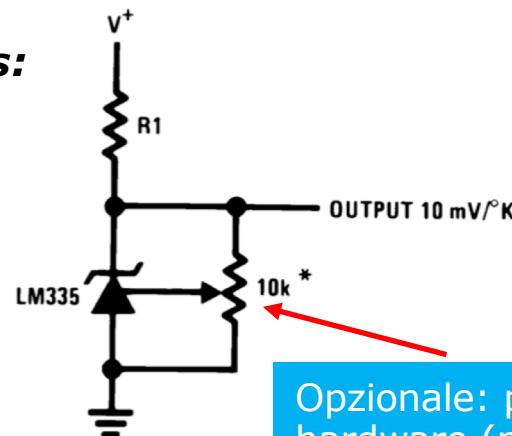
# Sensore analogico LM335

## Caratteristiche principali

- Sensore a 2 terminali (diodo zener)
- Uscita tarata in Kelvin, con sensibilità  $10 \text{ mV/K}$
- Corrente inversa di lavoro: da  $400 \mu\text{A}$  a  $5 \text{ mA}$
- ***Absolute maximum ratings:***

- ❖ Reverse Current:  $15 \text{ mA}$
- ❖ Forward Current:  $10 \text{ mA}$

**Attenzione al valore di R1:**  
Con correnti maggiori si danneggia!



Opzionale: per la messa a punto hardware (noi non lo usiamo)

# Sensore analogico LM335

## Caratteristiche metrologiche

### 6.5 Temperature Accuracy: LM335, LM335A<sup>(1)</sup>

PARAMETER	TEST CONDITIONS	LM335A			LM335			UNIT	
		MIN	TYP	MAX	MIN	TYP	MAX		
Operating Output Voltage	$T_C = 25^\circ\text{C}$ , $I_R = 1 \text{ mA}$	2.95	2.98	3.01	2.92	2.98	3.04	V	
Uncalibrated Temperature Error	$T_C = 25^\circ\text{C}$ , $I_R = 1 \text{ mA}$		1	3		2	6	$^\circ\text{C}$	
Uncalibrated Temperature Error	$T_{\text{MIN}} \leq T_C \leq T_{\text{MAX}}$ , $I_R = 1 \text{ mA}$		2	5		4	9	$^\circ\text{C}$	
Temperature Error with $25^\circ\text{C}$	$T_{\text{MIN}} \leq T_C \leq T_{\text{MAX}}$ , $I_R = 1 \text{ mA}$		0.5	1		1	2	$^\circ\text{C}$	
Calibration	Calibrated Error at Extended	$T_C = T_{\text{MAX}}$ (Intermittent)		2		2		$^\circ\text{C}$	
Temperature	Non-Linearity	$I_R = 1 \text{ mA}$		0.3	1.5		0.3	1.5	$^\circ\text{C}$

(1) Accuracy measurements are made in a well-stirred oil bath. For other conditions, self heating must be considered.

### 6.3 Thermal Information

THERMAL METRIC <sup>(1)</sup>	LM335 / LM335A	LM235 / LM235A	LM135 / LM135A	UNIT	
	SOIC (D)	TO-92 (LP)	TO-46 (NDV)		
	8 PINS	3 PINS	3 PINS		
$R_{\theta JA}$	Junction-to-ambient thermal resistance	165	202	400	
$R_{\theta JC}$	Junction-to-case thermal resistance	—	170	—	$^\circ\text{C}/\text{W}$

(1) For more information about traditional and new thermal metrics, see the *IC Package Thermal Metrics* application report, [SPRA953](#).

# Sensore analogico LM335

## Caratteristiche metrologiche

$$\begin{aligned} T_{\text{MIN}} &= -40 \text{ }^{\circ}\text{C} \\ T_{\text{MAX}} &= 100 \text{ }^{\circ}\text{C} \\ R_{\text{OUT}} &= 0.6 \text{ } \Omega \end{aligned}$$

### 6.5 Temperature Accuracy: LM335, LM335A<sup>(1)</sup>

PARAMETER	TEST CONDITIONS	LM335A			LM335			UNIT
		MIN	TYP	MAX	MIN	TYP	MAX	
Operating Output Voltage	$T_C = 25^{\circ}\text{C}$ , $I_R = 1 \text{ mA}$	2.95	2.98	3.01	2.92	2.98	3.04	V
Uncalibrated Temperature Error	$T_C = 25^{\circ}\text{C}$ , $I_R = 1 \text{ mA}$		1	3		2	6	$^{\circ}\text{C}$
Uncalibrated Temperature Error	$T_{\text{MIN}} \leq T_C \leq T_{\text{MAX}}$ , $I_R = 1 \text{ mA}$		2	5		4	9	$^{\circ}\text{C}$
Temperature Error with 25°C	$T_{\text{MIN}} \leq T_C \leq T_{\text{MAX}}$ , $I_R = 1 \text{ mA}$		0.5	1		1	2	$^{\circ}\text{C}$
Calibration	Calibrated Error at Extended	$T_C = T_{\text{MAX}}$ (Intermittent)			2		2	$^{\circ}\text{C}$
Temperature	Non-Linearity	$I_R = 1 \text{ mA}$	0.3	1.5	0.3	1.5		$^{\circ}\text{C}$

(1) Accuracy measurements are made in a well-stirred oil bath. For other conditions, self heating must be considered.

### 6.3 Thermal Information

THERMAL METRIC <sup>(1)</sup>	LM335 / LM335A	LM235 / LM235A	LM135 / LM135A	UNIT	
	SOIC (D)	TO-92 (LP)	TO-46 (NDV)		
	8 PINS	3 PINS	3 PINS		
$R_{\theta JA}$	Junction-to-ambient thermal resistance	165	202	400	$^{\circ}\text{C/W}$
$R_{\theta JC}$	Junction-to-case thermal resistance	—	170	—	

(1) For more information about traditional and new thermal metrics, see the IC Package Thermal Metrics application report, [SPRA953](#).

# Sensore digitale DHT22



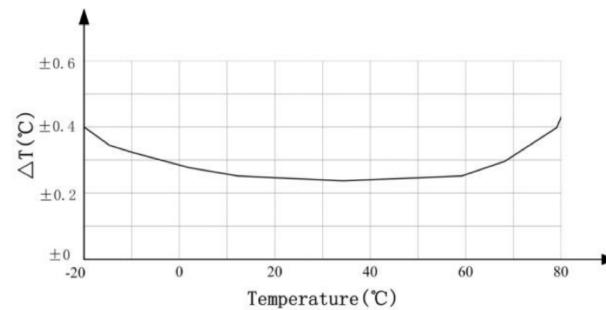
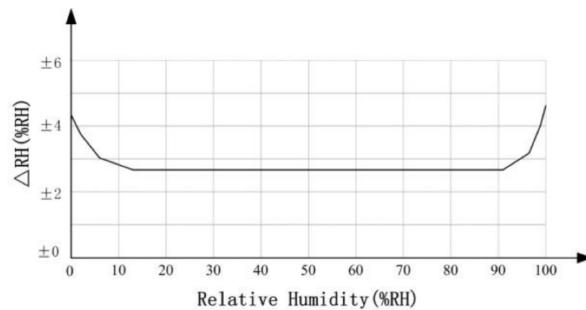
## Caratteristiche Metrologiche

Table 2: AM2302 Relative humidity performance table

Parameter	Condition	min	typ	max	Unit
Resolution			0.1		%RH
Range		0		99.9	%RH
Accuracy <sup>[1]</sup>	25°C		± 2		%RH
Repeatability			± 0.3		%RH
Exchange		Completely interchangeable			
Response <sup>[2]</sup>	1/e(63%)		<5		S
Sluggish			<0.3		%RH
Drift <sup>[3]</sup>	Typical		<0.5		%RH/yr

Table 3: AM2302 Relative temperature performance

Parameter	Condition	min	typ	max	Unit
Resolution			0.1		°C
			16		bit
Accuracy			± 0.5	± 1	°C
Range		-40		80	°C
Repeat			± 0.2		°C
Exchange		Completely interchangeable			
Response	1/e(63%)		<10		S
Drift			± 0.3		°C/yr



➤ Tempo di misura minimo: 500 ms

# Progettazione Termometro

## Riassumendo

- **LM335**
  - ✓ **Velocità lettura (uscita analogica)**
  - ❖ **Incertezza (uncalibrated)**
  - **Risoluzione (dipende dal sistema di misura)**

# Progettazione Termometro

## Riassumendo

### ➤ LM335

- ✓ **Velocità lettura (uscita analogica)**
- ❖ **Incertezza (uncalibrated)**
- **Risoluzione (dipende dal sistema di misura)**



### Soluzione...

Utilizziamo il sensore digitale o il termo-igrometro del laboratorio come riferimento per la messa a punto di quello analogico

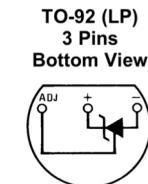
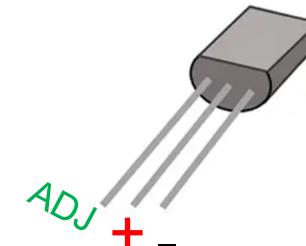
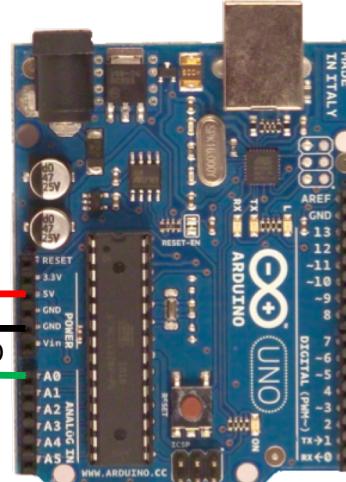
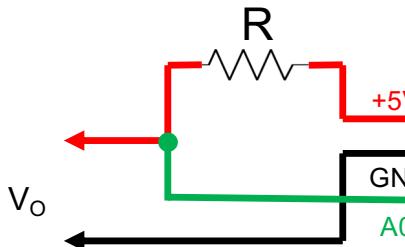
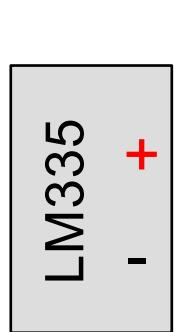
## **PARTE I**

# **Realizzazione termometro elettronico con interfaccia grafica mediante sensore analogico**

# Sensore analogico LM335

## Funzionamento

Il resistore  $R$  di polarizzazione deve limitare la corrente inversa nel diodo tra 0.4 mA e 5 mA



TO-92 (LP)  
3 Pins  
Bottom View

Uscita nominale a  
25 °C pari a  
2.9815 V

$$V_O = T_{(K)} \cdot S \quad S = \frac{10 \text{ mV}}{\text{K}} \quad \Rightarrow \quad T_{\circ\text{C}} = V_O/S - 273.15$$

## Sensore analogico LM335

### Prestazioni metrologiche attese

Modello di misura:

$$T = \frac{V_o}{10 \text{ mV/K}} - 273.15 \quad V_o = N_{ADC} \cdot \frac{V_{REF}}{1024}$$

## Sensore analogico LM335

### Prestazioni metrologiche attese

Modello di misura:

$$T = \frac{V_o}{10 \text{ mV/K}} - 273.15$$

S

Incetezza sensore  
(coefficiente di  
sensibilità, S)

$$V_o = N_{ADC} \cdot \frac{V_{REF}}{1024}$$

Incetezza convertitore  
ADC  
(total unadjusted error)

Incetezza campione di  
tensione  
(riferimento ADC)

# Sensore analogico LM335

## Prestazioni metrologiche attese

Coefficiente di sensibilità:  $S = 10 \text{ mV/K}$

- L'incertezza da assegnare ad  $S$  dipende dall'uso del sensore, se con o senza messa a punto
- Con la messa a punto si corregge l'errore di guadagno:

### 7.3.1 Temperature Calibration Using ADJ Pin

Included on the LM135 chip is an easy method of calibrating the device for higher accuracies. A pot connected across the LM135 with the arm tied to the adjustment terminal (as shown in Figure 12) allows a 1-point calibration of the sensor that corrects for inaccuracy over the full temperature range.

This single point calibration works because the output of the LM135 is proportional to absolute temperature with the extrapolated output of sensor going to 0-V output at 0 K (-273.15°C). Errors in output voltage versus temperature are only slope (or scale factor) errors so a slope calibration at one temperature corrects at all temperatures.

- La messa a punto può essere effettuata via hardware (ADJ pin) o via software (ricalcolando  $S$ , e la sua incertezza)

## Sensore analogico LM335

### Prestazioni metrologiche attese

Modello di misura:

$$T = \frac{V_o}{10 \text{ mV/K}} - 273.15 \quad V_o = N_{ADC} \cdot \frac{V_{REF}}{1024}$$

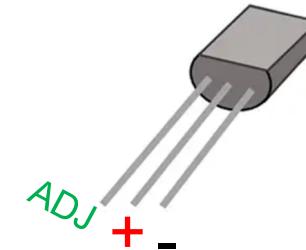
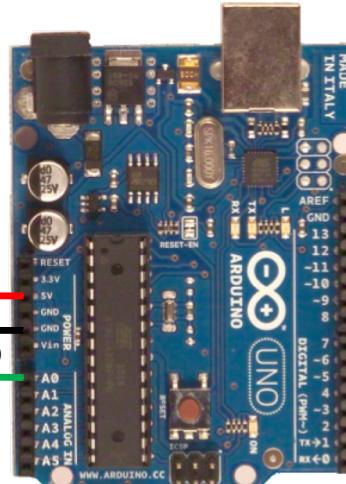
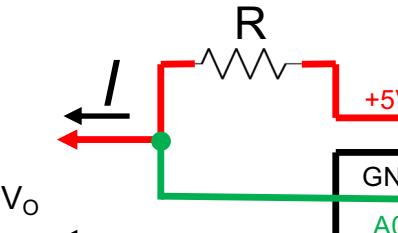
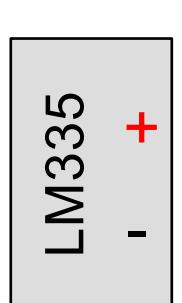


- Usando  $V_{REF} = +5 \text{ V}$  (alimentazione da USB)  $\rightarrow \varepsilon = 5\%$
- Usando  $V_{REF} = 1.1 \text{ V}$  (riferimento interno)  $\rightarrow \varepsilon = 10\%$
- Usando  $V_{REF} = 3.3 \text{ V}$  (regolatore su scheda)  $\rightarrow \varepsilon = 1.5\% ???$
- Usando  $V_{REF} = 4.1 \text{ V}$  (riferimento LM4040A)  $\rightarrow \varepsilon = 0.1\%$

# Sensore analogico LM335

## Funzionamento

Il resistore  $R$  di polarizzazione deve limitare la corrente inversa nel diodo tra 0.4 mA e 5 mA



Uscita nominale a  
25 °C pari a  
2.9815 V

Attenzione all'auto riscaldamento!

$$I = \frac{5V - V_O}{R}$$

# Sensore analogico LM335

## Cosa fare esattamente

- Dimensionare R
- Calcolare l'incertezza del termometro con  $V_{REF} = 5$  V, senza messa a punto, senza auto riscaldamento (misura al power-on)
- Calcolare la risoluzione assoluta del termometro e valutare l'auto riscaldamento
- Scrivere un programma Arduino che legga il sensore 4 volte al secondo e invii su seriale la temperatura in decimi di grado Celsius (numero intero, 1 misura per riga, ad esempio 253 per 25.3 °C)
- Si modifichi il programma in modo che ogni misura sia ottenuta come media di 40 letture acquisite a distanza di circa 1 ms
- Osservate le misure trasmesse e rivalutate sperimentalmente la risoluzione (scaldate il sensore con un dito). E' cambiata rispetto al caso senza medie? [Provate a pensare ad una spiegazione!](#)
- Modificare il programma LabView per leggere i risultati trasmessi dall'ultima versione del programma Arduino ([sostituendo lo knob e il ciclo di ritardo con i blocchi di lettura](#))

## Sensore analogico LM335

### Cosa fare esattamente – *Parte facoltativa*

- Modificare il programma LabView per trasmettere lo stato del controllo (inviare carattere «A» se riscaldamento ON, «B» se condizionatore ON, «C» se entrambi OFF)
- Aggiungere al programma Arduino l'accensione di due LED esterni (caso «A» e «B»)

SUGGERIMENTO:

- ❖ Su Arduino leggere con la funzione `Serial.read()` in un `while` loop in modo da prelevare tutti i caratteri presenti nel buffer di ricezione e garantire di visualizzare l'ultimo stato inviato da LabView

## Comunicazione tra LabVIEW e la strumentazione

La comunicazione con la strumentazione è possibile mediante le **VISA (Virtual Instrumentation Software Architecture) API**

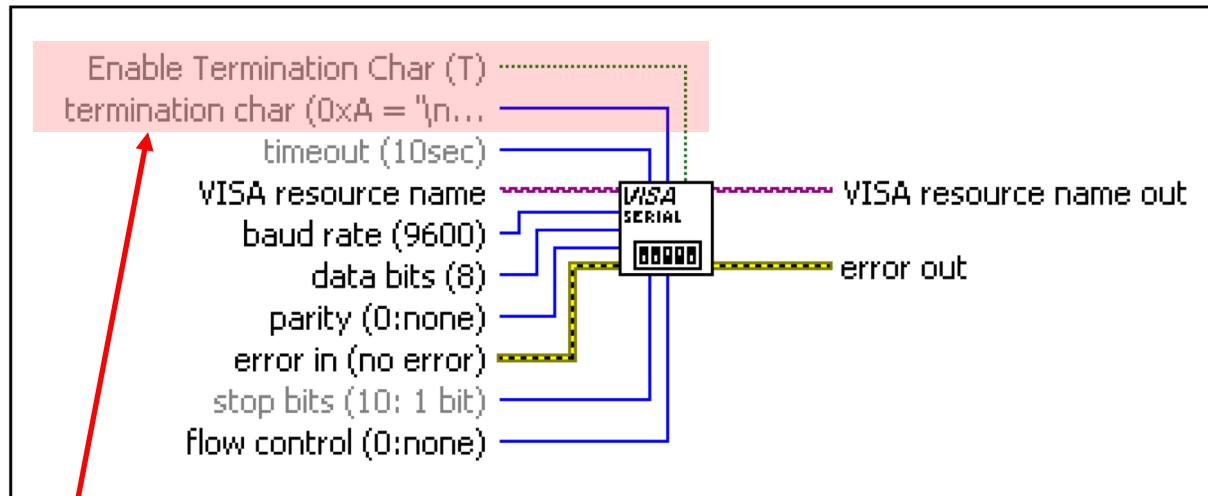
Ossia un layer di API che gestiscono la comunicazione con la strumentazione in modo standard ed indipendente dal bus utilizzato (es. Seriale, USB, GPIB, Ethernet...)

Ogni strumento connesso è un «VISA Resource», con un determinato nome, che descrive la posizione nel sistema  
Es: GPIB0::3::INSTR

E' possibile associare degli alias (*Visa Alias*, es «generatore»)

## Comunicazione su porta seriale

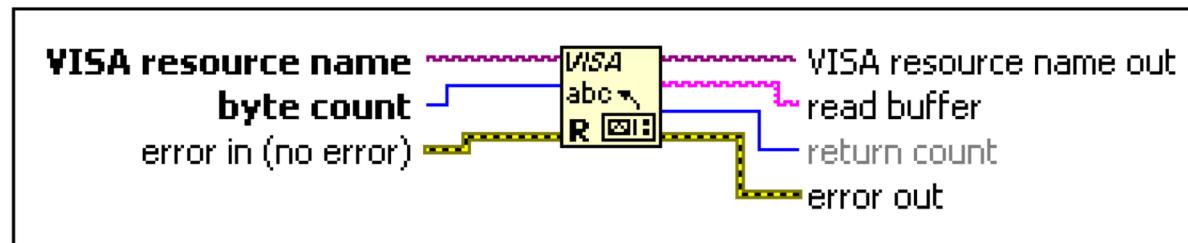
Il blocco **Visa Serial** è specifico per configurare ed aprire la comunicazione con la porta seriale



Attenzione all'uso del terminatore di linea (utile nelle trasmissioni di testo, da non usare nelle trasmissioni binarie)

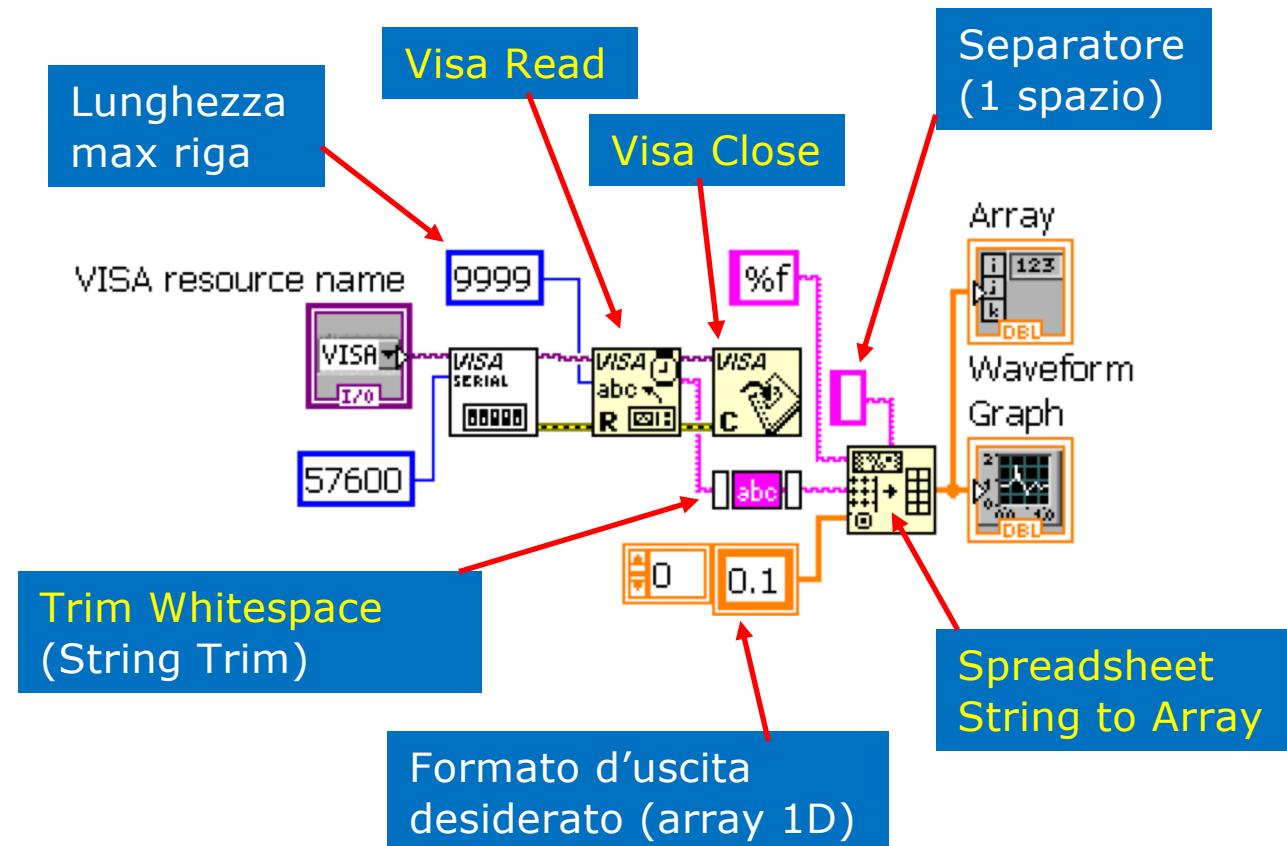
## Comunicazione su porta seriale

Il blocco **Visa Read** permette di leggere un certo numero di caratteri dallo strumento



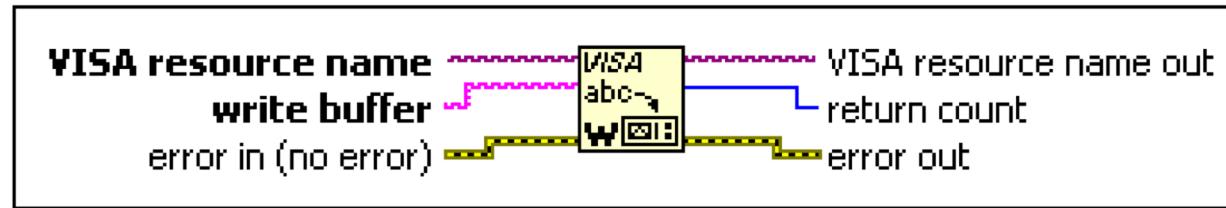
Nel caso della porta seriale, se è stata aperta specificando il carattere terminatore, la lettura termina appena si riceve il terminatore (es. «\n»)

## Esempio di comunicazione con Arduino - Modalità Testo



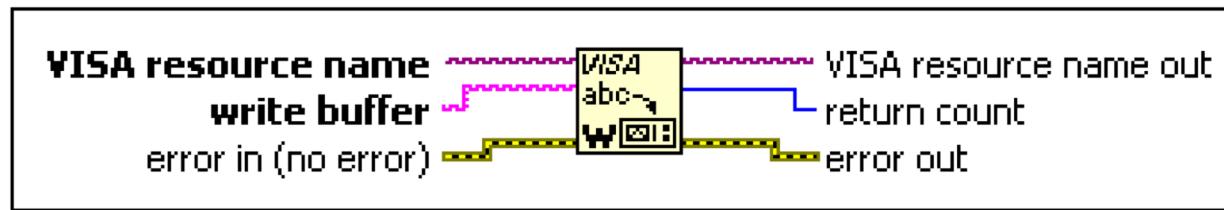
## Comunicazione su porta seriale

Il blocco **Visa Write** permette di inviare dei dati allo strumento



## Comunicazione su porta seriale

Il blocco **Visa Write** permette di inviare dei dati allo strumento



*Per inviare ad Arduino lo stato del sistema di controllo, collegare al cavo «write buffer» una stringa contenente uno tra i caratteri A,B,C in base allo stato dei led sul Front Panel*

*Per ricevere il carattere con Arduino dalla porta seriale usate: **Serial.read()** → se non sono presenti caratteri restituisce -1*

## Comunicazione su porta seriale

**Regole fondamentali per la corretta gestione della porta seriale**  
(per non impazzire in laboratorio...):

La porta seriale deve essere utilizzata da Arduino durante la programmazione del programma e successivamente da LabView per la ricezione delle misure; **è fondamentale che**:

**un utilizzatore rilasci (chiuda) la porta seriale dopo l'uso:**

- in Arduino assicurarsi che il serial monitor sia chiuso prima di far partire il programma LabView
- in LabView assicurarsi di eseguire il *Visa Close* prima di riutilizzare la IDE di Arduino (**può essere utile creare un VI che contenga solo la Visa Close, da eseguire per chiudere la porta**)

## **PARTE II**

# **Misura della tensione di riferimento del convertitore A/D con DMM**

## Sensore analogico LM335

### Cosa fare esattamente

- Misurate con il multimetro da banco HP34401 la tensione di riferimento del convertitore A/D (accessibile dall'esterno attraverso il pin AREF)
- Modificate il programma Arduino per utilizzare la tensione misurata
- Rivalutate l'incertezza del termometro dopo la modifica

Attenzione all'effetto della corrente assorbita dai LED sulla tensione di alimentazione (se avete aggiunto i LED reali su breadboard)

## **PARTE III**

**Messa a punto del sensore  
analogico tramite il sensore di  
riferimento (digitale)**

# Sensore digitale DHT22

## Cosa fare (messa a punto del LM335)

- Leggere la temperatura in prossimità del sensore LM335 mediante il termometro di riferimento «DHT» (il termo-igrometro del laboratorio oppure il sensore digitale DHT22)
- Utilizzando le letture del termometro di riferimento e del termometro analogico calcolare il rapporto Vcc/S (usate due misure al power-on per mitigare l'autorisaldamento)

$$T_K = \frac{n}{1024} \cdot \frac{V_{CC}}{S} \quad \Rightarrow \quad \frac{V_{CC}}{S} = \frac{T_K \cdot 1024}{n} = \frac{(T_{^{\circ}C}^{DHT} + 273.15) \cdot 1024}{n}$$

- Utilizzate il nuovo rapporto Vcc/S nel programma Arduino e verificate la corrispondenza tra le temperature fornite dai due sensori (scaldate i due sensori con le mani e resettate Arduino tramite il pulsante di reset per far ripartire il programma)
- **Provate a stimare l'incertezza del termometro dopo la messa a punto**

## Sensore digitale DHT22

### Cosa fare (messa a punto del LM335)

- Leggere la temperatura in prossimità del sensore LM335 mediante il termometro di riferimento «DHT» (il termo-igrometro del laboratorio oppure il sensore digitale DHT22)
- Utilizzando le letture del termometro di riferimento e del termometro analogico calcolare il rapporto Vcc/S (usate due misure al power-on per mitigare l'autorisaldamento)

$$T_K = \frac{n}{1024} \cdot \frac{V_{CC}}{S} \quad \beta \Rightarrow \beta \frac{V_{CC}}{S} = \frac{T_K \cdot 1024}{n} = \frac{(T_C^{DHT} + 273.15) \cdot 1024}{n}$$

$$T_K = \frac{n}{1024} \cdot \beta \quad \beta = \frac{(T_C^{DHT} + 273.15) \cdot 1024}{n}$$

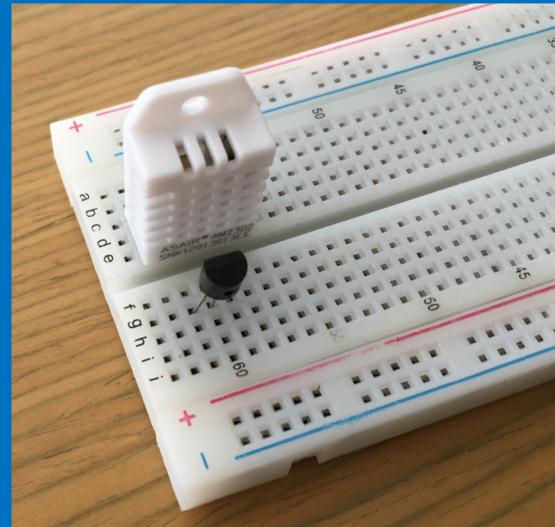
- Utilizzate il nuovo rapporto Vcc/S nel programma Arduino e verificate la corrispondenza tra le temperature fornite dai due sensori (scaldate i due sensori con le mani e resettate Arduino tramite il pulsante di reset per far ripartire il programma)
- **Provate a stimare l'incertezza del termometro dopo la messa a punto**

## Sensore digitale DHT22

**Cosa fare (messa a punto del LM335)**

**Suggerimento:**

**Avvicinate i due sensori**



tra le temperature fornite dai due sensori (scaldate i due sensori con le mani e resettate Arduino tramite il pulsante di reset per far ripartire il programma)

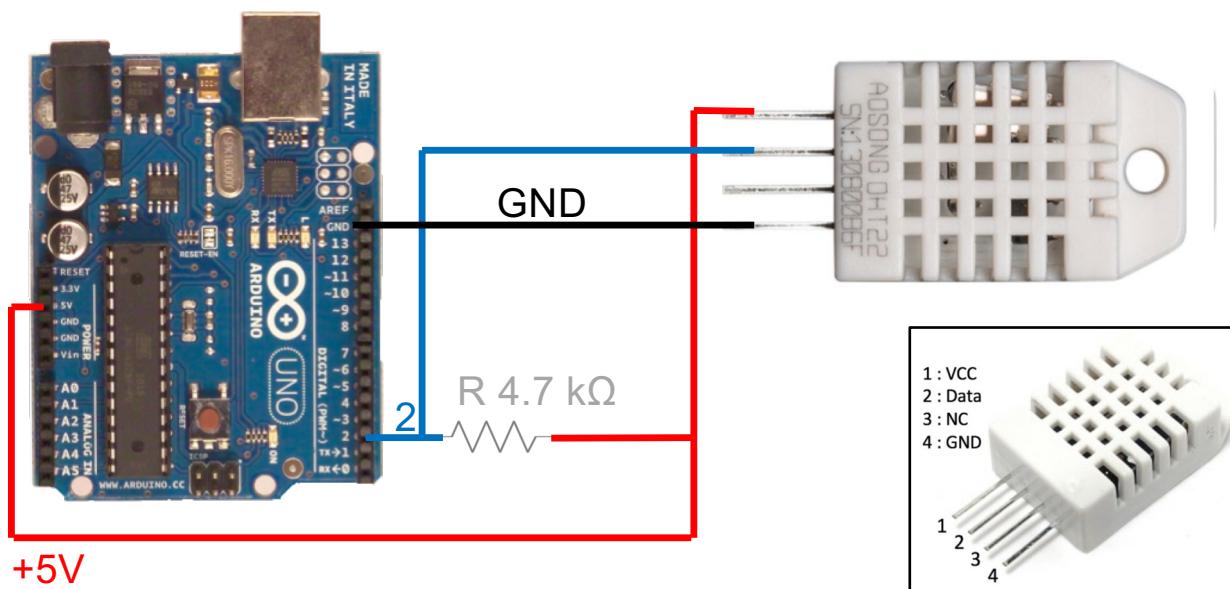
- Provate a stimare l'incertezza del termometro dopo la messa a punto

**Per i gruppi che hanno il sensore  
DHT22 e vogliono utilizzarlo**

# Sensore digitale DHT22

## Funzionamento

I dati transitano (bidirezionale) su una singola linea (1-wire) che deve gestire Arduino in modalità «open-collector» (R è inclusa nel sensore)

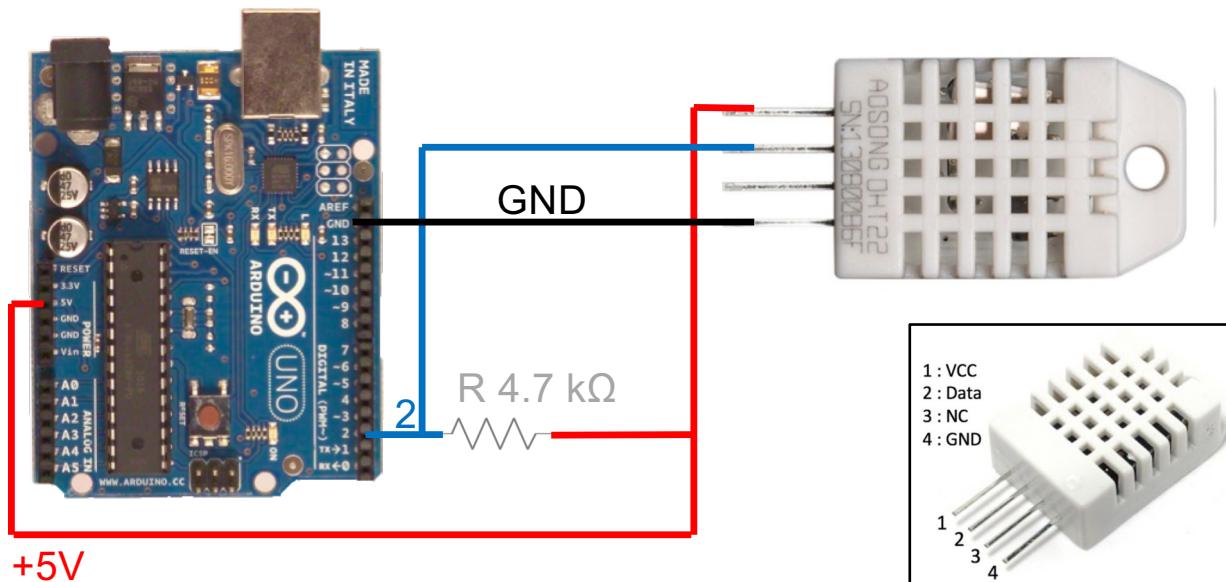


# Sensore digitale DHT22

## Funzionamento

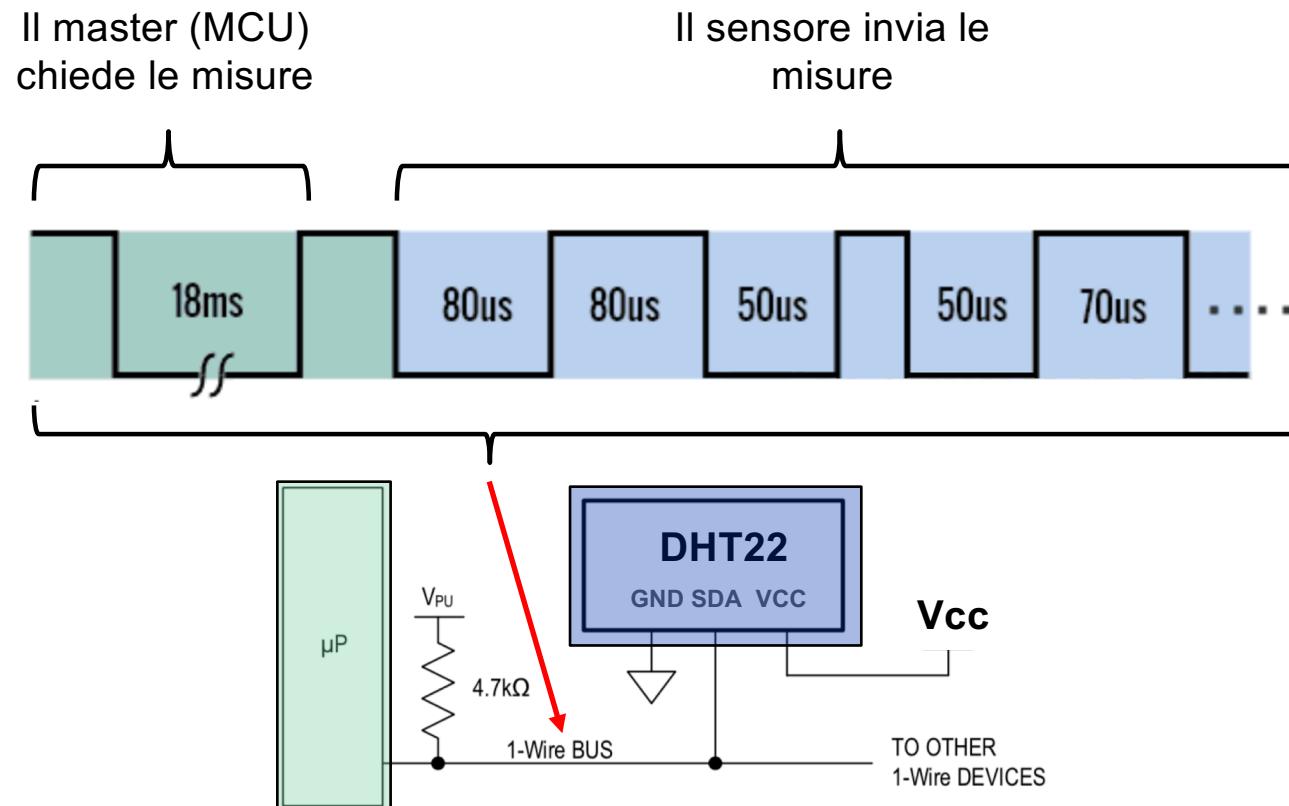
**Attenzione:**  
se invertite l'alimentazione si danneggia!

I dati transitano (bidirezionale) su una singola linea (1-wire) che deve gestire Arduino in modalità «open-collector» (R è inclusa nel sensore)



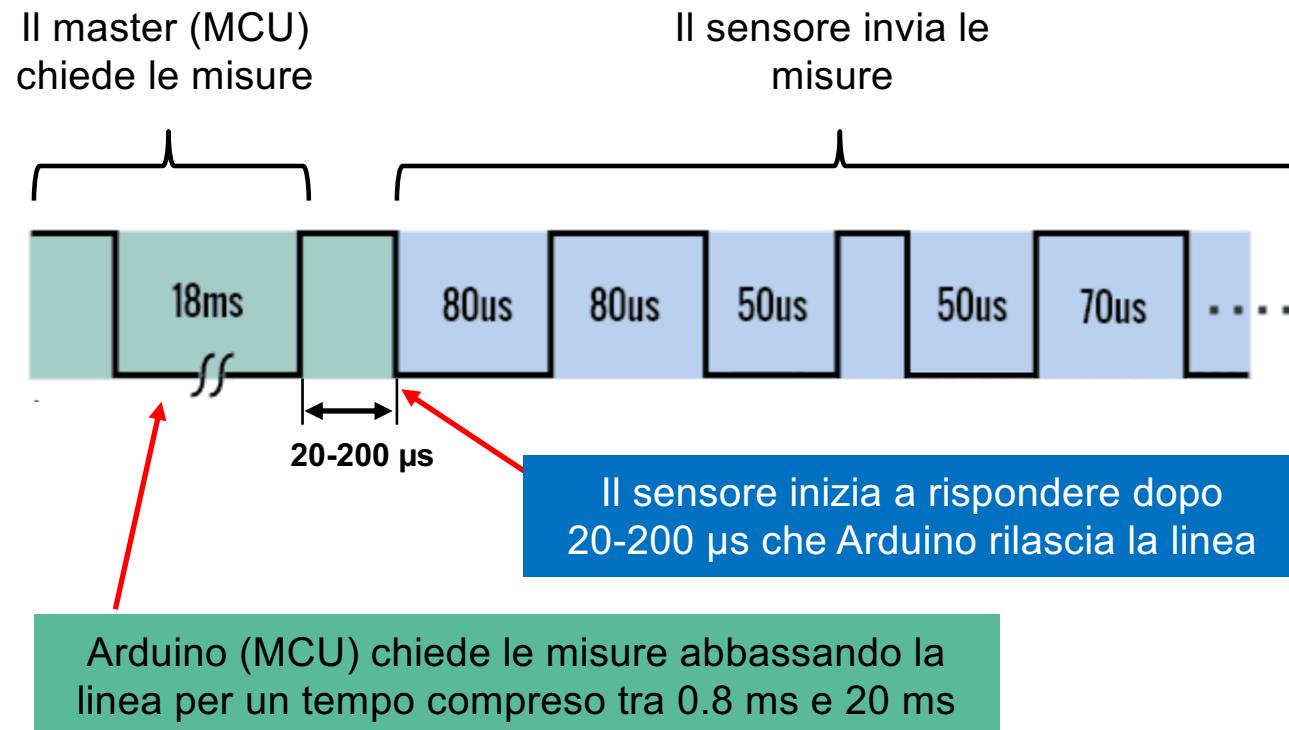
# Sensore digitale DHT22

## Protocollo di comunicazione



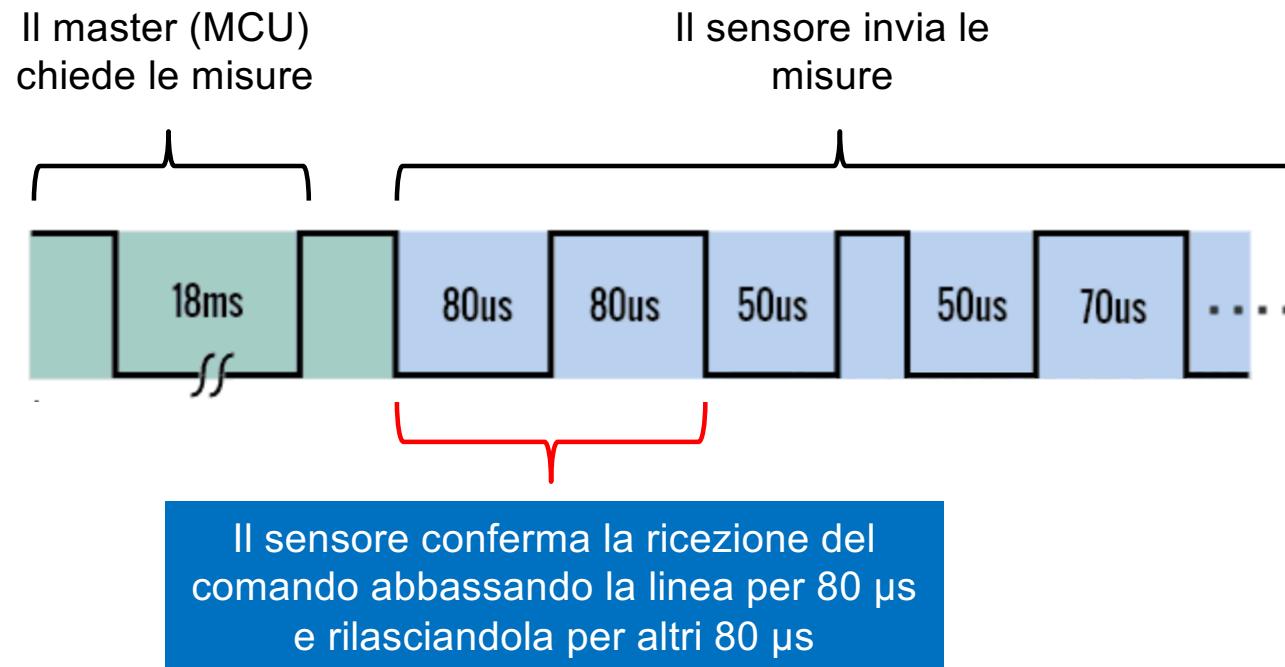
# Sensore digitale DHT22

## Protocollo di comunicazione



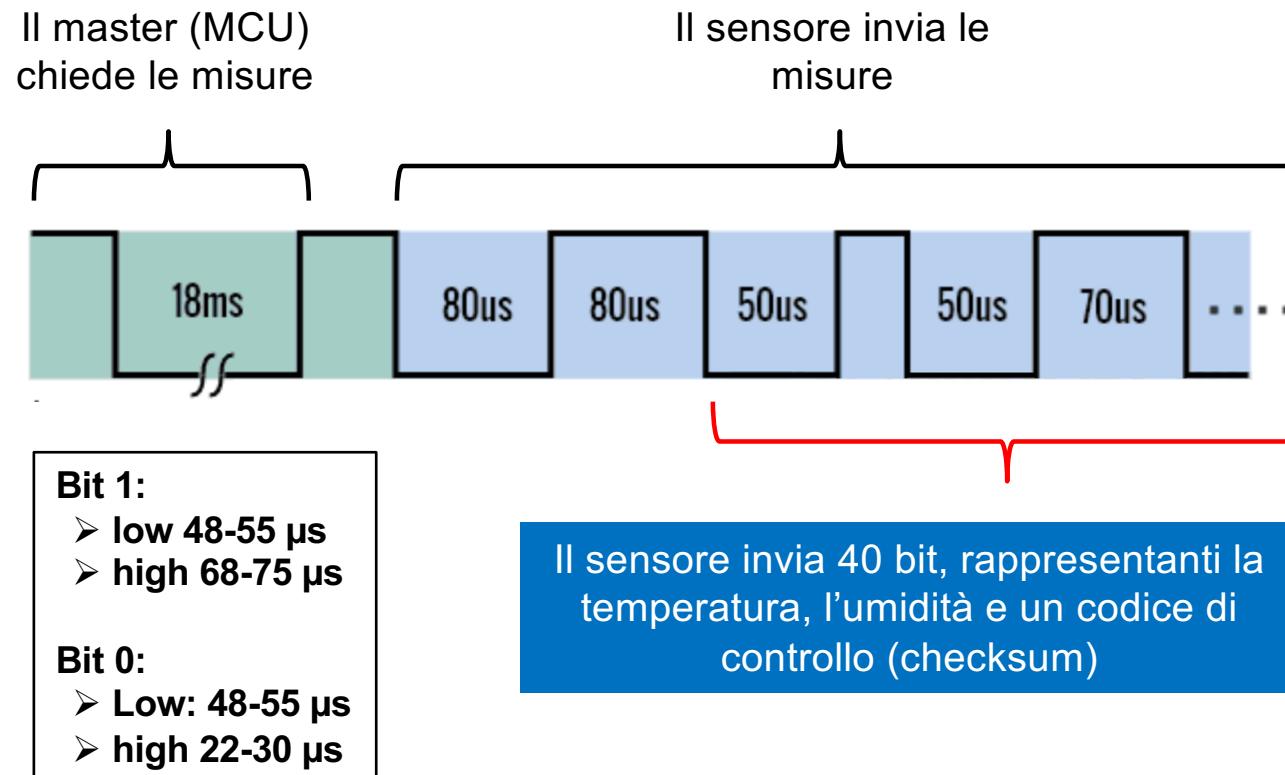
# Sensore digitale DHT22

## Protocollo di comunicazione



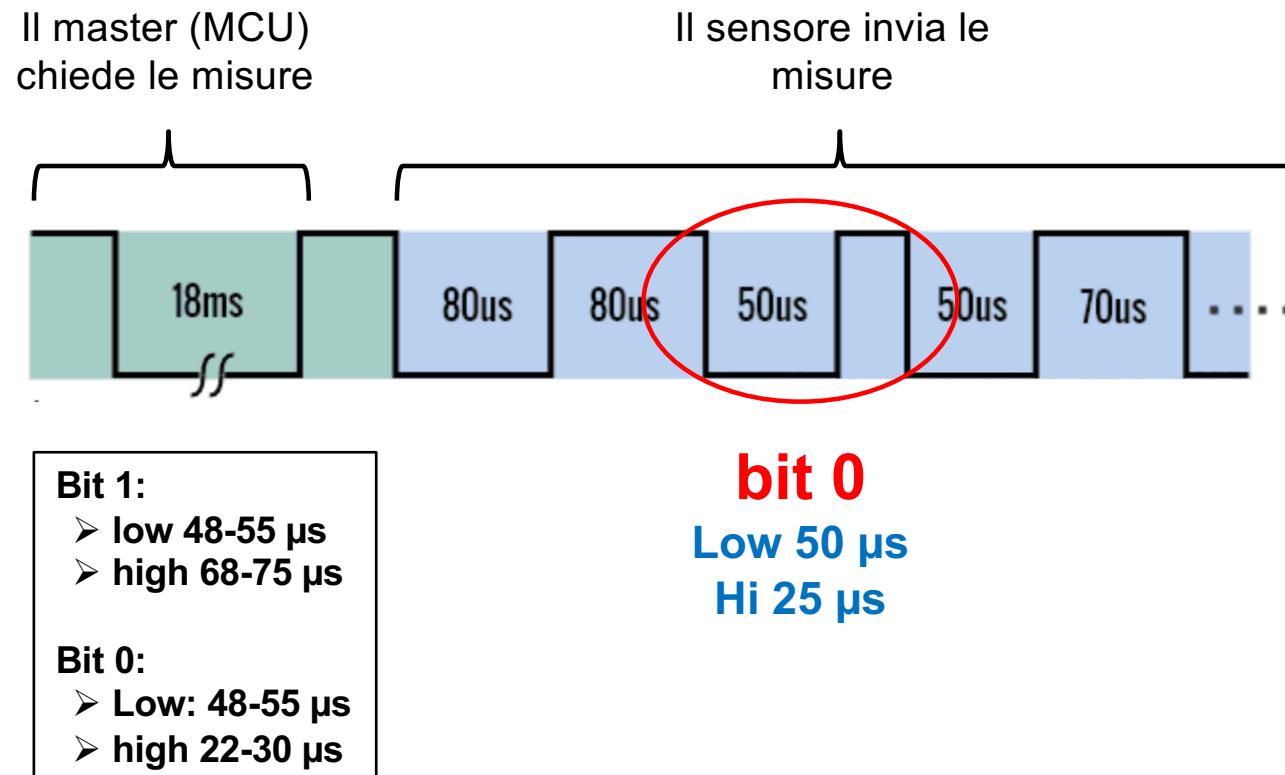
# Sensore digitale DHT22

## Protocollo di comunicazione



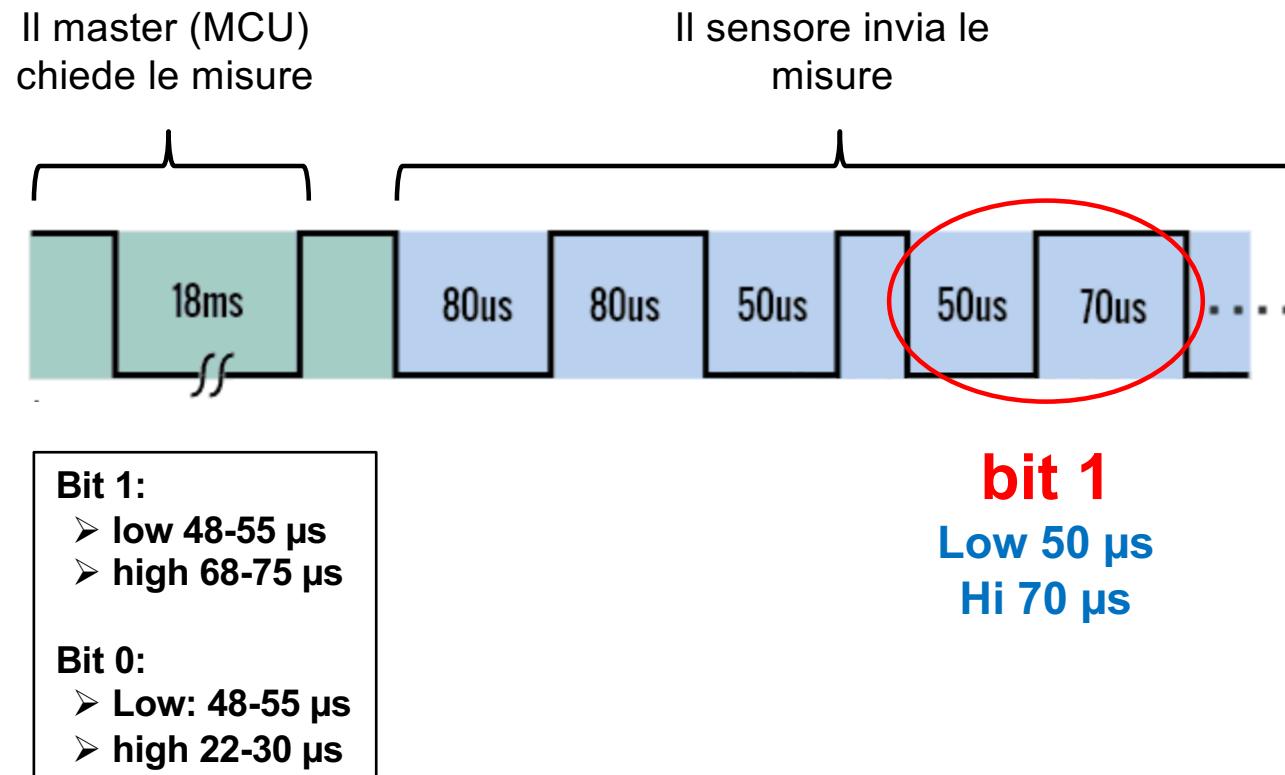
# Sensore digitale DHT22

## Protocollo di comunicazione



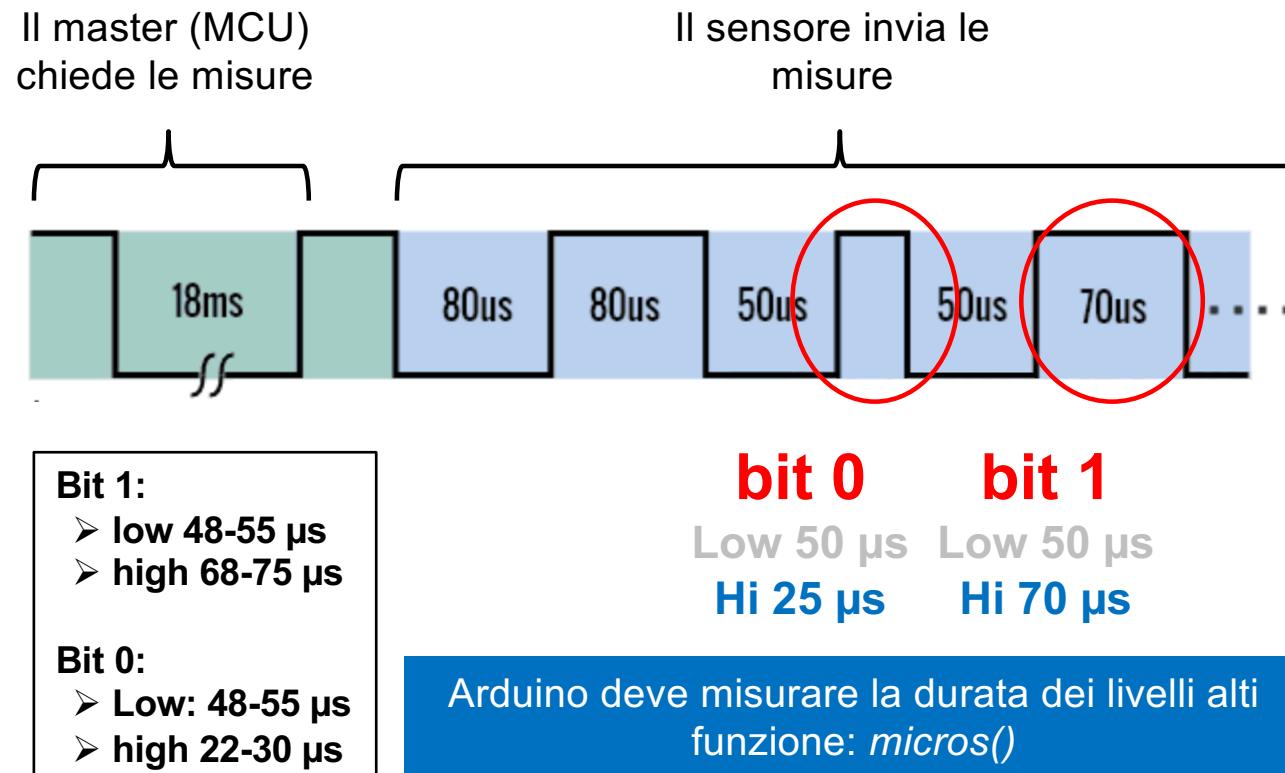
# Sensore digitale DHT22

## Protocollo di comunicazione



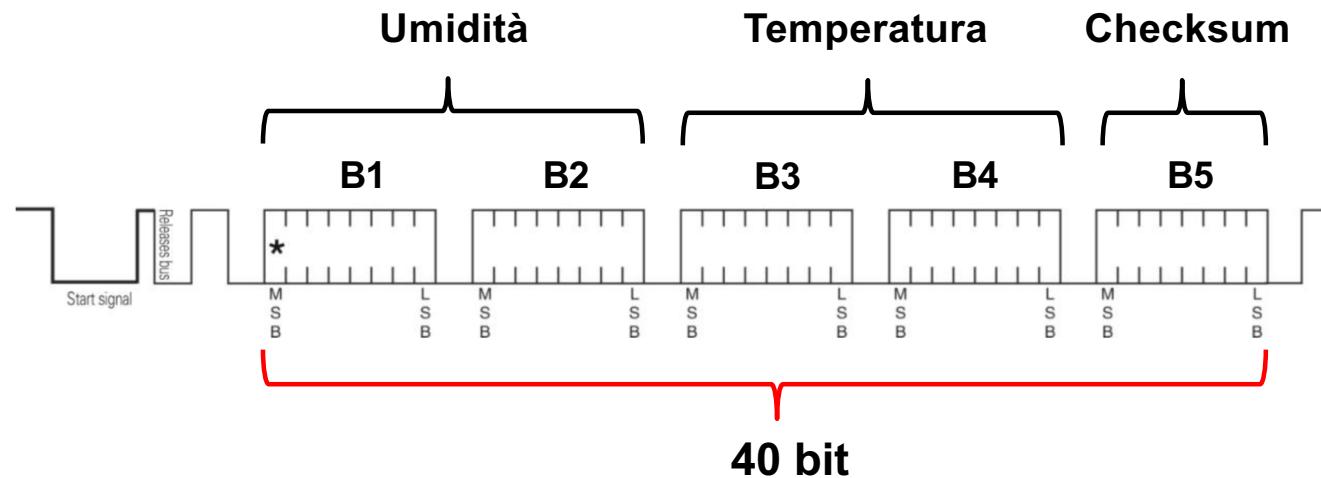
# Sensore digitale DHT22

## Protocollo di comunicazione



# Sensore digitale DHT22

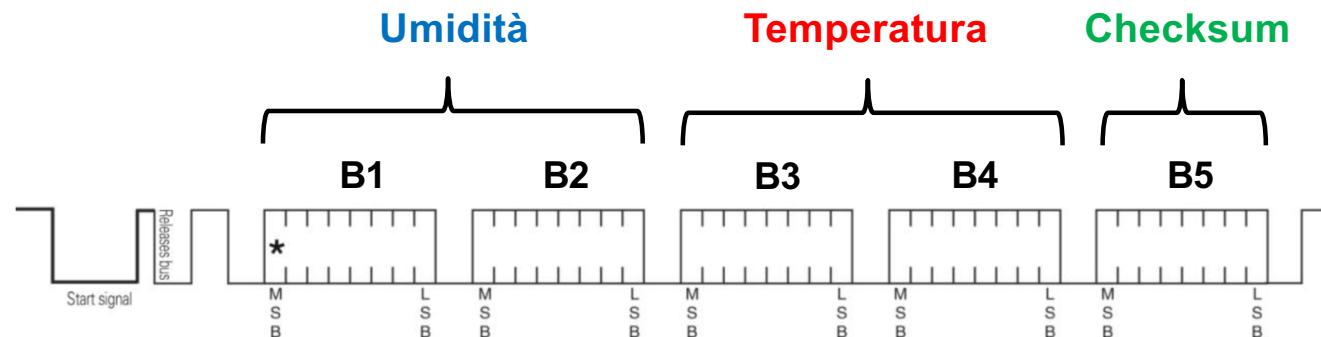
## Protocollo di comunicazione



Il sensore invia 5 byte (40 bit). Ogni byte è inviato a partire dal bit più significativo (MSB)

# Sensore digitale DHT22

## Protocollo di comunicazione



$$\text{Umidità} = (B1 * 256 + B2) / 10 \longrightarrow \text{in \% RH}$$

$$\text{Temperatura} = (B3 * 256 + B4) / 10 \longrightarrow \text{in gradi Celsius}$$

$$\text{Checksum} = B1+B2+B3+B4 \longrightarrow \text{8 bit meno significativi della somma}$$

# Sensore digitale DHT22

## Note implementative con Arduino UNO

- La funzione *digitalWrite* di Arduino è lenta e non è quindi utilizzabile per gestire impulsi larghi pochi microsecondi

### **Soluzione:**

- Manipoliamo direttamente i registri:
  - ❖ Funzione *Pin\_Down*
    - Pin in output (bit corrispondente in DDR a 1)
    - Valore in uscita *LOW* (bit corrispondente in PORT a 0)
  - ❖ Funzione *Pin\_Up*
    - Pin in input (bit corrispondente in DDR a 0)
    - Abilitazione pull-up (bit corrispondente in PORT a 1)

## Sensore digitale DHT22

### Note implementative con Arduino UNO

- Durante le operazioni di lettura il programma non deve essere interrotto da interrupt (la temporizzazione a livello del microsecondo è importante)

#### **Soluzione:**

- Disabilitare gli interrupt all'inizio della lettura con: `cli()`
- Riabilitare gli interrupt al termine della lettura con: `sei()`