

# Progetto di un Capacimetro 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

# Capacimetro

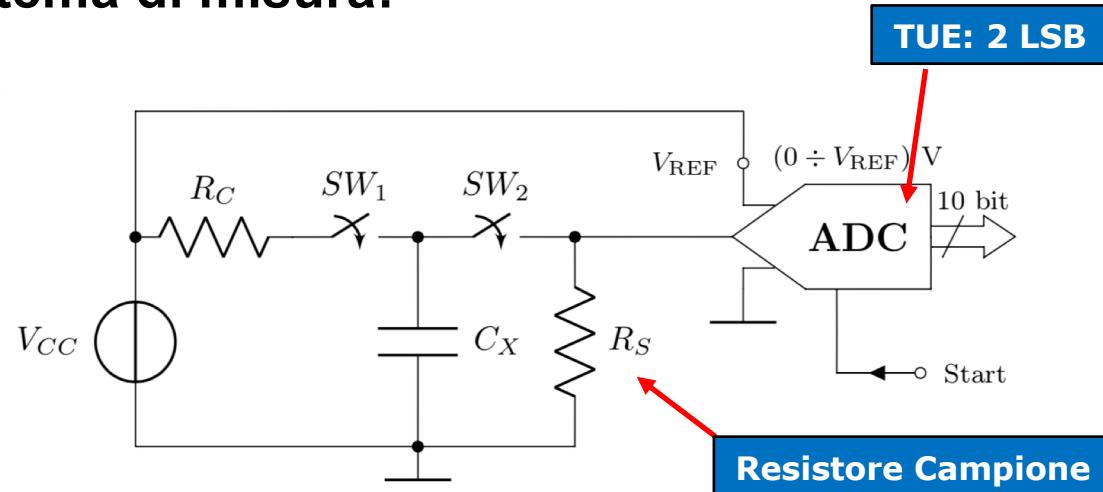
## Obiettivo:

Realizzare un capacimetro con le seguenti specifiche:

- campo di misura: 1 nF - 1  $\mu$ F
- ottimizzato per misurare il campione civetta (3.3 nF)
- metodo di misura: costante di tempo
- incertezza migliore consentita dalle risorse Arduino

# Capacimetro

Sistema di misura:



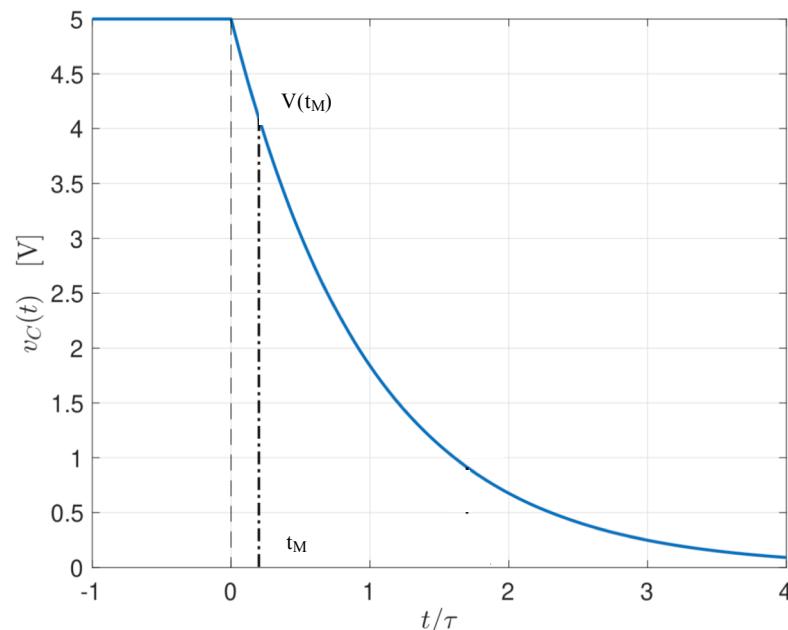
SW1	SW2	Funzione	Cost. tempo
ON	OFF	Carica	$\tau = C_X R_C$
OFF	ON	Scarica	$\tau = C_X R_S$

→ 
$$C_X = \tau / R$$

# Capacimetro

## Sistema di misura:

$$v(t) = V_{CC} \cdot e^{-\frac{t}{R_S \cdot C_X}} = V_{CC} \cdot e^{-t/\tau}$$



All'istante  $t_M$ :

$$-\tau \cdot \log\left(\frac{v(t_M)}{V_{CC}}\right) = t_M$$



$$\tau = t_M / \log\left(\frac{V_{CC}}{v(t_M)}\right)$$

$$C_X = \tau / R_S$$

# Capacimetro

## Sistema di misura:

$$v(t) = V_{CC} \cdot e^{-\frac{t}{R_S \cdot C_X}} = V_{CC} \cdot e^{-t/\tau}$$

**E' possibile calcolare la costante di tempo con una sola misura, ma:**

- Bisogna conoscere  $V_{CC}$ 
  - ❖ Dipende fortemente dal computer
- Bisogna misurare  $t_M$ 
  - ❖ Tempo tra l'inizio scarica e la misura di tensione

**Contributi di incertezza dovuti a:**

→  $V_{CC}$ ,  $t_M$ ,  $v(t_M)$ ,  $R_S$

**All'istante  $t_M$ :**

$$-\tau \cdot \log\left(\frac{v(t_M)}{V_{CC}}\right) = t_M$$



$$\tau = t_M / \log\left(\frac{V_{CC}}{v(t_M)}\right)$$

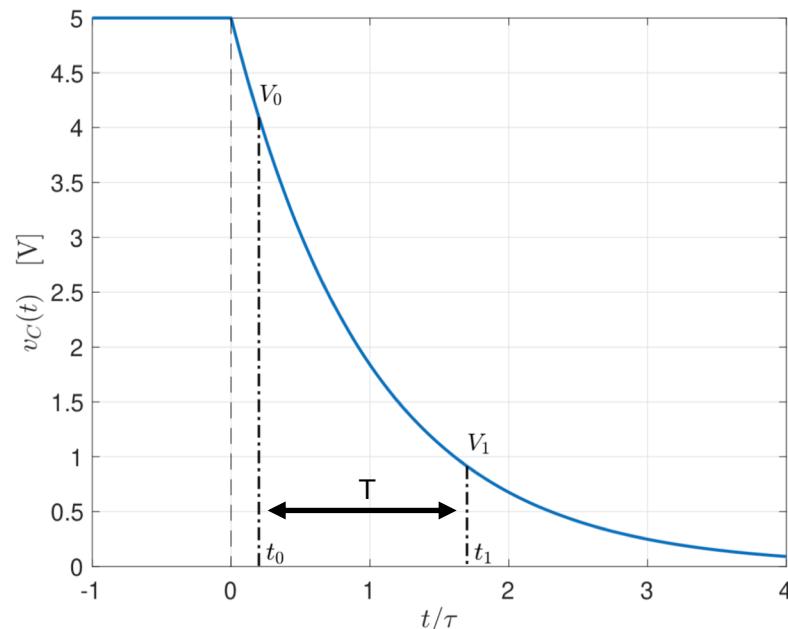
$$C_X = \tau / R_S$$

# Capacimetro

## Sistema di misura:

$$v(t) = V_{CC} \cdot e^{-\frac{t}{R_S \cdot C_X}} = V_{CC} \cdot e^{-t/\tau}$$

Due misure  
istanti  $t_0$  e  $t_1$



$$V_0 = V_{CC} \cdot e^{-t_0/\tau}$$

$$V_1 = V_{CC} \cdot e^{-t_1/\tau}$$



$$V_0/V_1 = e^{T/\tau}$$

$$\tau = T/\log(V_0/V_1)$$

$$C_X = \tau/R_S$$

# Capacimetro

## Sistema di misura:

$$v(t) = V_{CC} \cdot e^{-\frac{t}{R_S \cdot C_X}} = V_{CC} \cdot e^{-t/\tau}$$

Due misure  
istanti  $t_0$  e  $t_1$

### Vantaggi della misura a 2 punti:

- La misura non dipende da  $V_{CC}$  (tensione a cui si è caricata la capacità)
- L'intervallo  $T$  non dipende più dall'istante di inizio scarica
  - ❖ Basta prendere 2 campioni a distanza fissa (accurata)

### Contributi di incertezza sono ora:

→  $T$ ,  $V1/V2$ ,  $R_S$

$$V_0 = V_{CC} \cdot e^{-t_0/\tau}$$

$$V_1 = V_{CC} \cdot e^{-t_1/\tau}$$



$$V_0/V_1 = e^{T/\tau}$$

$$\tau = T/\log(V_0/V_1)$$

$$C_X = \tau/R_S$$

# Capacimetro con Arduino

Usando il convertitore A/D:

$$C_X = \tau / R_S = \frac{T}{R_S \cdot \log(V_0/V_1)}$$

nel sistema di misura a 2 punti si ha:

$$\begin{aligned} V_0 &= V_{CC_A} \cdot e^{-t_0/\tau} \\ V_1 &= V_{CC_A} \cdot e^{-t_1/\tau} \end{aligned} \rightarrow \frac{V_0 \cdot V_{CC_A}}{V_1 \cdot V_{CC_A}} = e^{T/\tau}$$

Semplificazione lecita perché  $V_{CC_A}$  è sempre la stessa quantità: la tensione a cui si è caricata la capacità

$$\begin{aligned} V_0 &= V_{REF_0} \cdot N_0 / 1024 \\ V_1 &= V_{REF_1} \cdot N_1 / 1024 \end{aligned} \rightarrow C_X = \frac{T}{R_S \cdot \log \left( \frac{V_{REF_0} \cdot N_0}{V_{REF_1} \cdot N_1} \right)}$$

Il convertitore contribuisce due volte all'incertezza

Semplificazione lecita perché è ragionevole supporre che  $V_{REF}$  resti sufficientemente costante tra le due misure

# Capacimetro con Arduino

Usando il convertitore A/D potrebbe sembrare conveniente la misura a 1 punto, ma:

$$C_X = \tau/R_S = \frac{T}{R_S \cdot \log(V_0/V_1)}$$

Tensione di carica

$$V_0 = V_{CC_A} \quad \Rightarrow \quad V_{CC_A}/V_1 = e^{T/\tau}$$

$$V_1 = V_{CC_A} \cdot e^{-t_1/\tau}$$

Il convertitore contribuisce una volta sola all'incertezza

$$V_1 = V_{REF_1} \cdot N_1/1024 \quad \Rightarrow \quad C_X = \frac{T}{R_S \cdot \log\left(\frac{1024 \cdot V_{CC_A}}{V_{REF_1} \cdot N_1}\right)} = \frac{T}{R_S \cdot \log\left(\frac{1024 \cdot V_{CC_A}}{V_{CC_1} \cdot N_1}\right)}$$

La semplificazione è lecita solo se la capacità si carica esattamente a Vcc e se  $V_{CC} = V_{REF}$

Resta la difficoltà nell'attuare un tempo accurato tra l'inizio scarica e la misura di  $V_1$

# Capacimetro con Arduino

Usando il convertitore A/D potrebbe sembrare conveniente la misura a 1 punto, ma:

$$C_X = \frac{T}{R_S \cdot \log(V_0/V_1)}$$

**Tensione di carica**

$$V_0 = V_{CC_A}$$
$$V_1 = V_{CC_A} \cdot e^{-t_1/\tau}$$
$$V_1 = V_{REF_1} \cdot N_1/1024$$

**Il convertitore contribuisce alla incertezza**

**In generale non possiamo garantire che non ci sia una differenza significativa tra la tensione di carica e Vref**

**Meglio la misura a due punti**

La semplificazione è lecita solo se la capacità si carica esattamente a Vcc e se  $V_{CC} = V_{REF}$

Resta la difficoltà nell'attuare un tempo accurato tra l'inizio scarica e la misura di  $V_1$

## Analisi incertezza

Modello di misura:

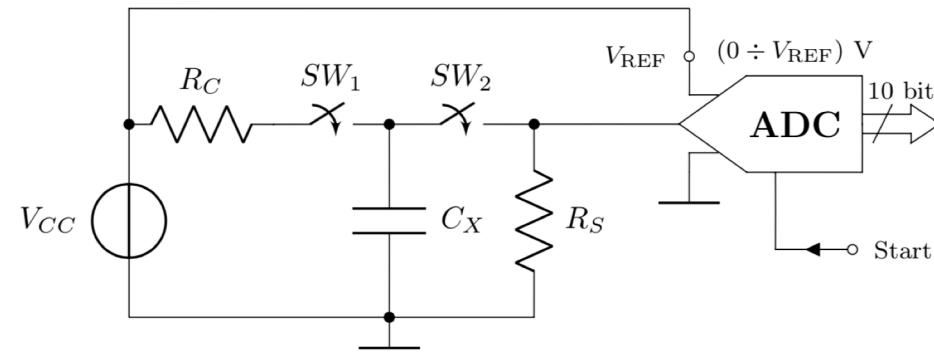
$$\tau = T / \log(V_0/V_1)$$

$$C_x = \tau / R_s = \frac{T}{R_s \cdot \log(V_0/V_1)}$$

Calcolo incertezza  $C_x$ ?

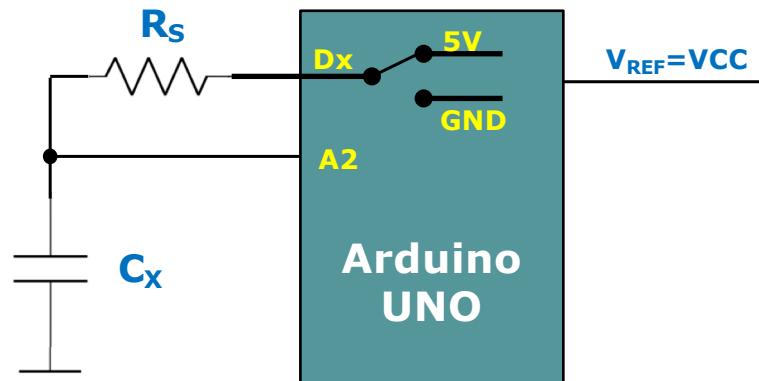
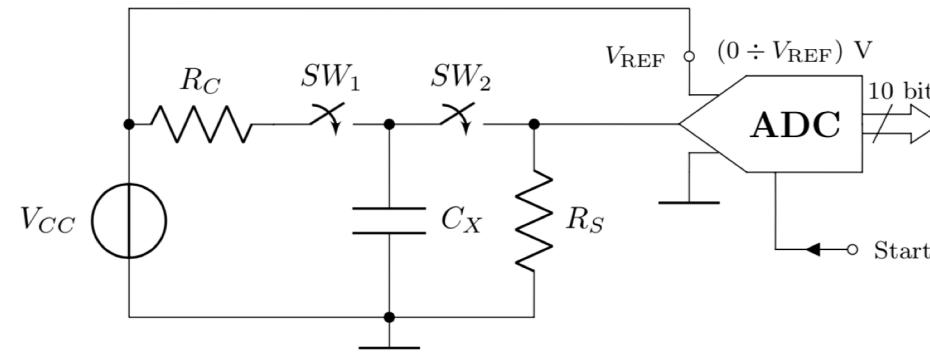
- Dipende dall'intervallo  $T$  (da cui dipende  $V_1$ )
- Quale intervallo usare per minimizzare l'incertezza di misura?
- Quanto vale l'incertezza di misura nel punto di ottimo

# Realizzazione



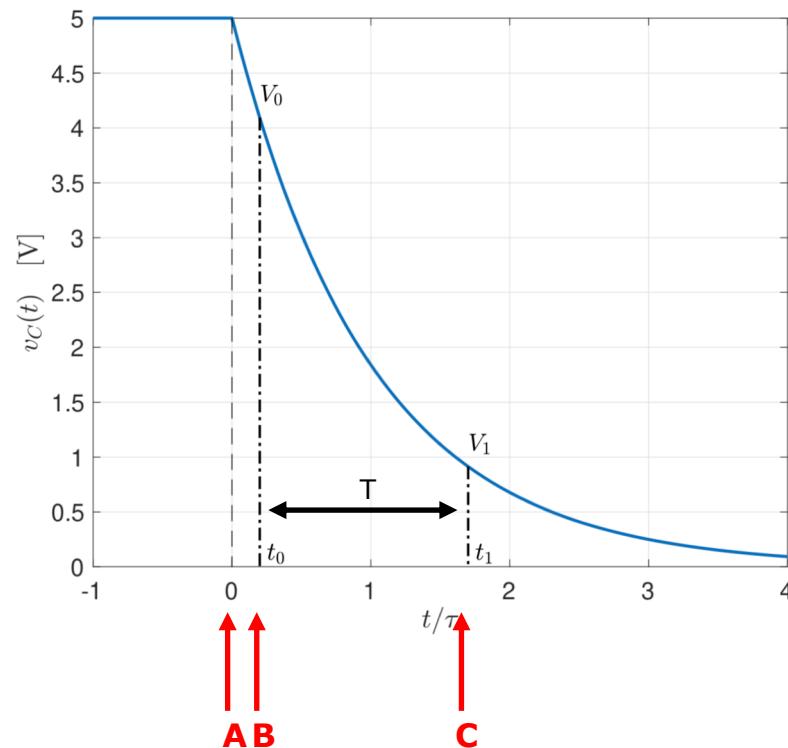
Come realizzare il  
circuito con Arduino?

# Realizzazione



# Realizzazione

Come temporizzare le azioni necessarie?



- A) Comutazione stato logico (inizio scarica)
- B) Misura  $V_0$
- C) Misura  $V_1$



**La distanza tra B e C deve essere accurata!**

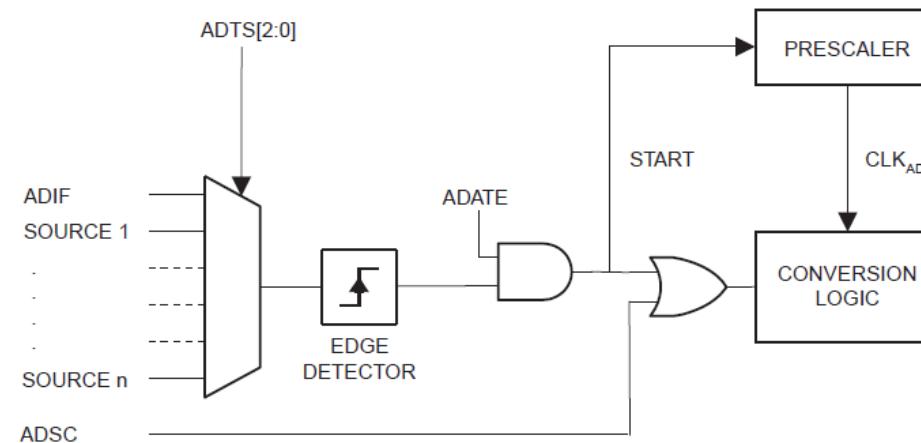
# Il convertitore A/D

Trigger hardware di inizio conversione:

- Software (ADSC)
- Hardware:

- AD interrupt flag di fine conversione (modo free running)
- Comparatore analogico
- Interrupt esterno
- Timer / Counter 0 (overflow, compare soglia A)
- Timer / Counter 1 (overflow, **compare soglia B**, capture)

La conversione può essere comandata dalla **soglia B** del timer 1



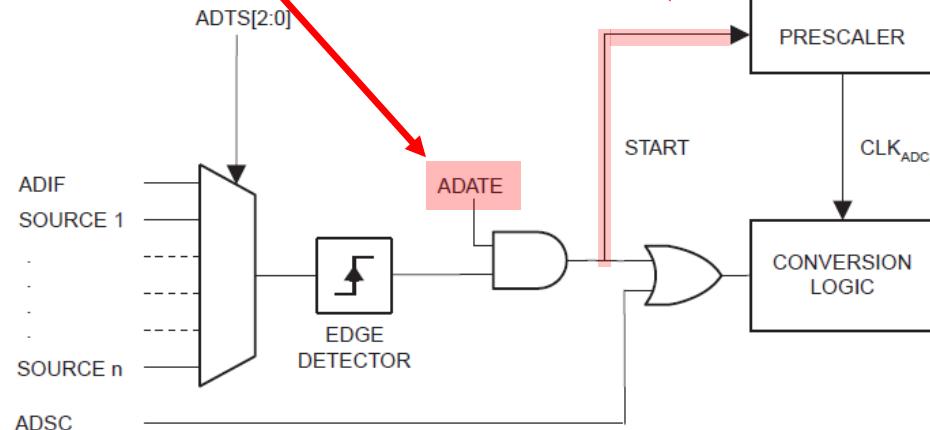
# Il convertitore A/D

Trigger hardware di inizio conversione:

- Software (ADSC)
- Hardware:
  - AD interrupt flag
  - Conversione analogico-digitale
  - Pin analogico
  - Timer 0 (overflow, compare soglia A)
  - Timer 1 (overflow, compare soglia B, capture)

Con i trigger hardware il prescaler viene azzerato → no latenza variabile: il clock AD diventa sincrono con lo START

Abilita il trigger hardware (fronte di salita)



Dai lucidi  
Arduino Parte II

# Il convertitore A/D

**ADC Control and Status Register A**

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

**ADCSRB – ADC Control and Status Register B**

Bit	7	6	5	4	3	2	1	0	
(0x7B)	-	ACME	-	-	-	ADTS2	ADTS1	ADTS0	ADCSRB
Read/Write	R	R/W	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

**ADMUX – ADC Multiplexer Selection Register**

Bit	7	6	5	4	3	2	1	0	
(0x7C)	REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

**DIDR0 – Digital Input Disable Register 0**

Bit	7	6	5	4	3	2	1	0	
(0x7E)	-	-	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	DIDR0
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

# Il convertitore A/D

ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

**ADEN: Accensione convertitore (0 = OFF, 1 = ON)**

**ADSC: Comando di inizio conversione (1 → ADSC)**

- la conversione inizierà al ciclo di clock  $t_{AD}$  successivo
- il bit ritornerà a 0 appena conclusa la conversione

**ADATE: Abilitazione trigger hardware di inizio conversione**

Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

# Il convertitore A/D

ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

**ADIF:** flag di interrupt dell'evento fine conversione

**ADIE:** flag di abilitazione della gestione interrupt dell'evento di fine conversione

Bit	7	6	5	4	3	2	1	0	
(0x7C)	REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

DIDR0 – Digital Input Disable Register 0

Bit	7	6	5	4	3	2	1	0	
(0x7E)	-	-	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	DIDR0
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

# Il convertitore A/D

ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

## Prescaler generazione clock AD

Table 23-5. ADC Prescaler Selections

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

# Il convertitore A/D

ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

ADCSRB – ADC Control and Status Register B

Bit	7	6	5	4	3	2	1	0	
(0x7B)	-	ACME	-	-	-	ADTS2	ADTS1	ADTS0	ADCSRB
Read/Write	R	R/W	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

**ACME:** permette di collegare il canale analogico  
selezionato con l'ingresso negativo del comparatore

Non ha effetto quando il convertitore è acceso

DIDR0 – Digital Input Disable Register 0

Bit	7	6	5	4	3	2	1	0	
(0x7E)	-	-	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	DIDR0
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Dai lucidi  
Arduino Parte II

# Il convertitore A/D

ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

ADCSRB – ADC Control and Status Register B

Bit	7	6	5	4	3	2	1	0	
(0x7B)	-	ACME	-	-	-	ADTS2	ADTS1	ADTS0	ADCSRB
Read/Write	R	R/W	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

## Selezione sorgente trigger hardware

Table 23-6. ADC Auto Trigger Source Selections

ADTS2	ADTS1	ADTS0	Trigger Source
0	0	0	Free Running mode
0	0	1	Analog Comparator
0	1	0	External Interrupt Request 0
0	1	1	Timer/Counter0 Compare Match A
1	0	0	Timer/Counter0 Overflow
1	0	1	Timer/Counter1 Compare Match B
1	1	0	Timer/Counter1 Overflow
1	1	1	Timer/Counter1 Capture Event

# Il convertitore A/D

## Selezione tensione di riferimento AREF

Table 23-3. Voltage Reference Selections for ADC

REFS1	REFS0	Voltage Reference Selection
0	0	AREF, Internal $V_{ref}$ turned off
0	1	$AV_{CC}$ with external capacitor at AREF pin
1	0	Reserved
1	1	Internal 1.1V Voltage Reference with external capacitor at AREF pin

### ADMUX – ADC Multiplexer Selection Register

Bit	7	6	5	4	3	2	1	0	
(0x7C)	REFS1	REFS0	ADLAR	–	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

### DIDR0 – Digital Input Disable Register 0

Bit	7	6	5	4	3	2	1	0	
(0x7E)	–	–	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	DIDR0
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Dai lucidi  
Arduino Parte II

## Il convertitore A/D

### Allineamento risultato conversione (a sinistra)

15	14	13	12	11	10	9	8
-	-	-	-	-	-	ADC9	ADC8
ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0
7	6	5	4	3	2	1	0

ADLAR=0  
(Right)

15	14	13	12	11	10	9	8
ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2
ADC1	ADC0	-	-	-	-	-	-
7	6	5	4	3	2	1	0

ADLAR=1  
(Left)

### ADMUX – ADC Multiplexer Selection Register

Bit	7	6	5	4	3	2	1	0
(0x7C)	REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

### DIDR0 – Digital Input Disable Register 0

Bit	7	6	5	4	3	2	1	0
(0x7E)	-	-	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

# Il convertitore A/D

Selezione canale analogico Arduino:

0-5 → A0-A5  
 8 → Termometro  
 14 → 1.1 V  
 15 → GND

MUX 3..0	Canale microc.
0000	ADC0
...	ADC1-ADC4
0101	ADC5
1000	Termometro
1110	1.1 V
1111	GND

ADMUX – ADC Multiplexer Selection Register

Bit	7	6	5	4	3	2	1	0	
(0x7C)	REFS1	REFS0	ADLAR	–	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

DIDR0 – Digital Input Disable Register 0

Bit	7	6	5	4	3	2	1	0	
(0x7E)	–	–	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	DIDR0
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Dai lucidi  
Arduino Parte II

# Il convertitore A/D

ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

ADCSRB – ADC Control and Status Register B

Bit	7	6	5	4	3	2	1	0	
(0x7B)	–	ACME	–	–	–	ADTS2	ADTS1	ADTS0	ADCSRB
Read/Write	R	R/W	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

ADMUX – ADC Multiplexer Selection Register

Registro per disabilitare l'ingresso digitale associato al pin. Diminuisce il rumore ed il consumo

DIDR0 – Digital Input Disable Register 0

Bit	7	6	5	4	3	2	1	0	
(0x7E)	–	–	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	DIDR0
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

# Introduzione ad Arduino

## Timer 1 – Registri

TCCR1A – Timer/Counter1 Control Register A

Bit	7	6	5	4	3	2	1	0	
(0x80)	COM1A1	COM1A0	COM1B1	COM1B0	–	–	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

La soglia A può portare a livello basso un pin digitale (9)!

### Funzione Output Compare: azione sui pin

Table 15-1. Compare Output Mode, non-PWM

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	Toggle OC1A/OC1B on Compare Match.
1	0	Clear OC1A/OC1B on Compare Match (Set output to low level).
1	1	Set OC1A/OC1B on Compare Match (Set output to high level).

# Introduzione ad Arduino

## Timer 1 – Funzione Output Compare

Arduino function			Arduino function
reset	(PCINT14/RESET)	PC6	1 28 □ PC5 (ADC5/SCL/PCINT13)
digital pin 0 (RX)	(PCINT16/RXD)	PD0	2 27 □ PC4 (ADC4/SDA/PCINT12)
digital pin 1 (TX)	(PCINT17/TXD)	PD1	3 26 □ PC3 (ADC3/PCINT11)
digital pin 2	(PCINT18/INT0)	PD2	4 25 □ PC2 (ADC2/PCINT10)
digital pin 3 (PWM)	(PCINT19/OC2B/INT1)	PD3	5 24 □ PC1 (ADC1/PCINT9)
digital pin 4	(PCINT20/XCK/T0)	PD4	6 23 □ PC0 (ADC0/PCINT8)
VCC	VCC	7	22 □ GND GND
GND	GND	8	21 □ AREF analog reference
crystal	(PCINT6/XTAL1/TOSC1)	PB6	9 20 □ AVCC VCC
crystal	(PCINT7/XTAL2/TOSC2)	PB7	10 19 □ PB5 (SCK/PCINT5) digital pin 13
digital pin 5 (PWM)	(PCINT21/OC0B/T1)	PD5	11 18 □ PB4 (MISO/PCINT4) digital pin 12
digital pin 6 (PWM)	(PCINT22/OC0A/AIN0)	PD6	12 17 □ PB3 (MOSI/OC2A/PCINT3) digital pin 11(PWM)
digital pin 7	(PCINT23/AIN1)	PD7	13 16 □ PB2 (SS/OC1B/PCINT2) digital pin 10 (PWM)
digital pin 8	(PCINT0/CLK0/ICP1)	PB0	14 15 □ PB1 (OC1A/PCINT1) digital pin 9 (PWM)

Pin controllabili

# Introduzione ad Arduino

## Timer 1 – Registri

TCCR1A – Timer/Counter1 Control Register A

Bit	7	6	5	4	3	2	1	0	
(0x80)	COM1A1	COM1A0	COM1B1	COM1B0	–	–	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Il pin è controllabile  
(digitalWrite) solo nel  
modo normale

### Funzione Output Compare: azione sui pin

Table 15-1. Compare Output Mode, non-PWM

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	Toggle OC1A/OC1B on Compare Match.
1	0	Clear OC1A/OC1B on Compare Match (Set output to low level).
1	1	Set OC1A/OC1B on Compare Match (Set output to high level).

Attenzione: quando si cede il controllo del pin (9) al timer, non è possibile cambiarne il livello logico con il controllo del pin  
→ noi però dobbiamo portare il pin a livello logico 1 per la carica

# Introduzione ad Arduino

## Timer 1 – Registri

digitalWrite(9, 1) diventa:  
**TCCR1A = 0b11000000;**  
**TCCR1C = 0b10000000;**

TCCR1A – Timer/Counter1 Control Register A

Bit	7	6	5	4	3	2	1	0	
(0x80)	COM1A1	COM1A0	COM1B1	COM1B0	–	–	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

**TCCR1A** – Timer/Counter1 Control Register A

**Scrivendo 1 si forza l'evento soglia A**

**Funzioni**

**15.11.3 TCCR1C – Timer/Counter1 Control Register C**

Bit	7	6	5	4	3	2	1	0	
(0x82)	FOC1A	FOC1B	–	–	–	–	–	–	TCCR1C
Read/Write	R/W	R/W	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

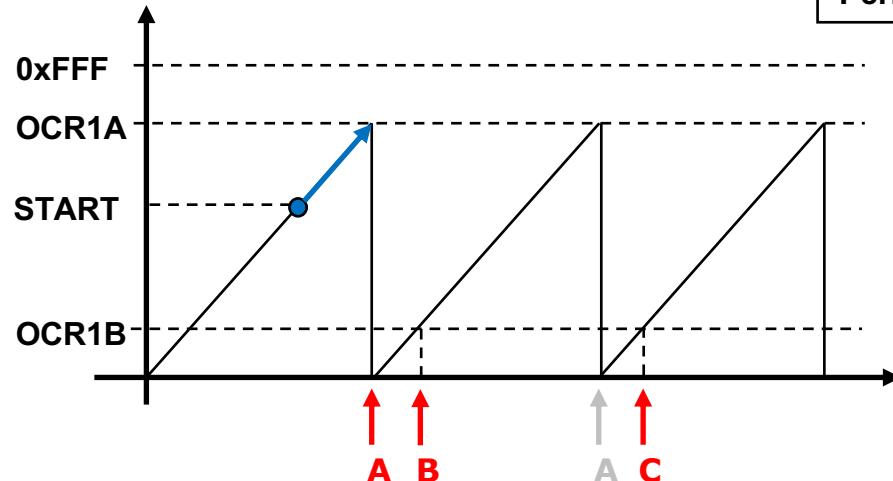
**TIMSK1** – Timer/Counter1 Mask Register

**TIFR1** – Timer/Counter1 Flag Register

**Si seleziona la funzione SET e si comanda la commutazione immediata agendo sul registro TCCR1C**

# Temporizzazione azioni

## Timer 1 in modalità «CTC»



Attenzione:

$$\text{Periodo} = (\text{OCR1A} + 1) * P * t_{\text{CK}}$$

Soglia A:

- Azzerza il contatore (Periodo =  $T_{BC}$ )
- Inizia la scarica (pin low)

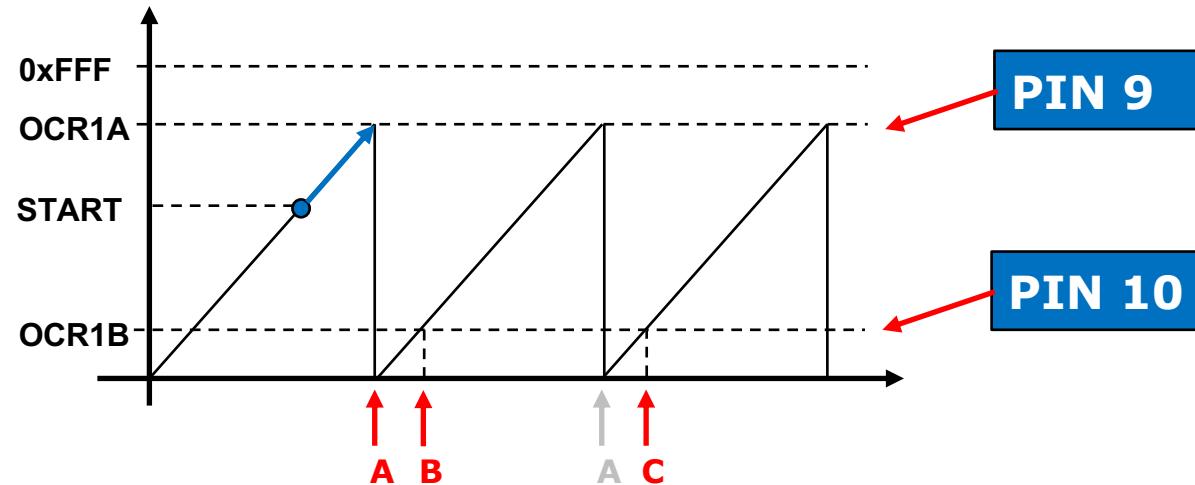
Soglia B:

- Inizia la conversione A/D
- OCR1B definisce  $T_{AB}$

La soglia A deve essere attraversata prima della soglia B → Inizializziamo il contatore a START compreso tra soglia A e B  
Esempio: **START = Soglia A - 1**

# Temporizzazione azioni

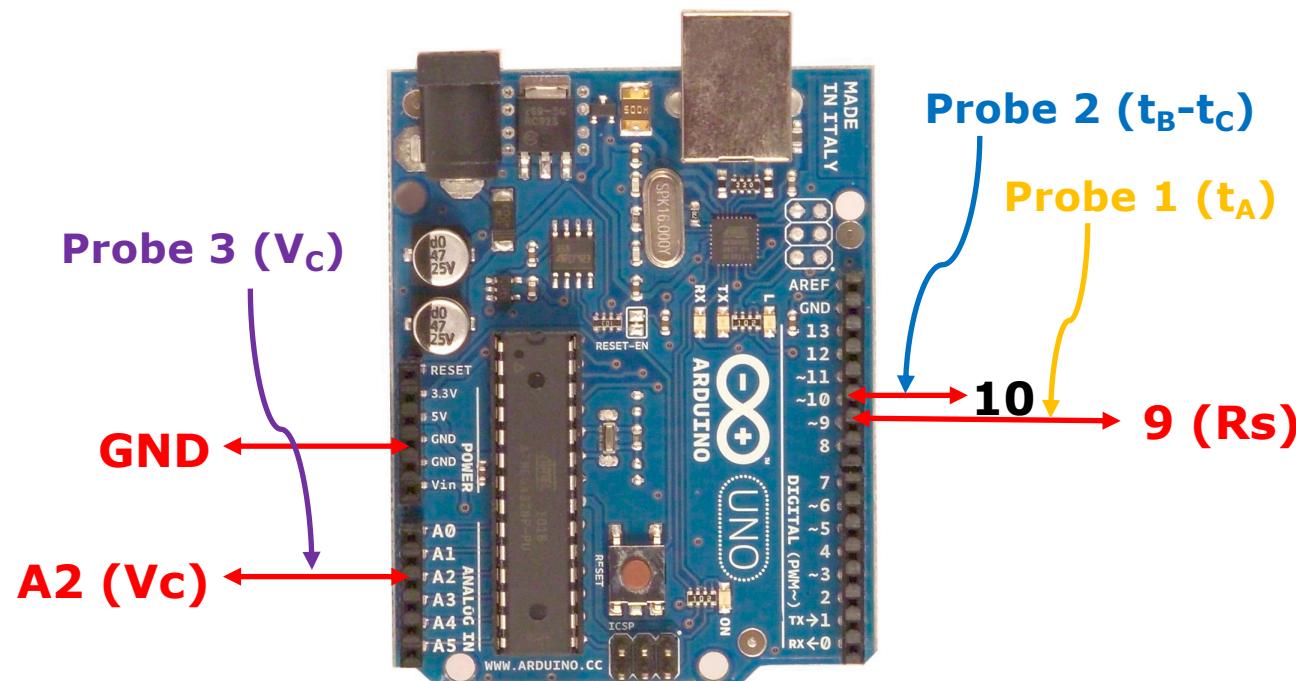
## DEBUG Azioni



Per poter controllare con l'oscilloscopio anche l'istante esatto di campionamento (B e C) possiamo impostare il Timer per fare il PIN TOGGLE sulla soglia B ☺

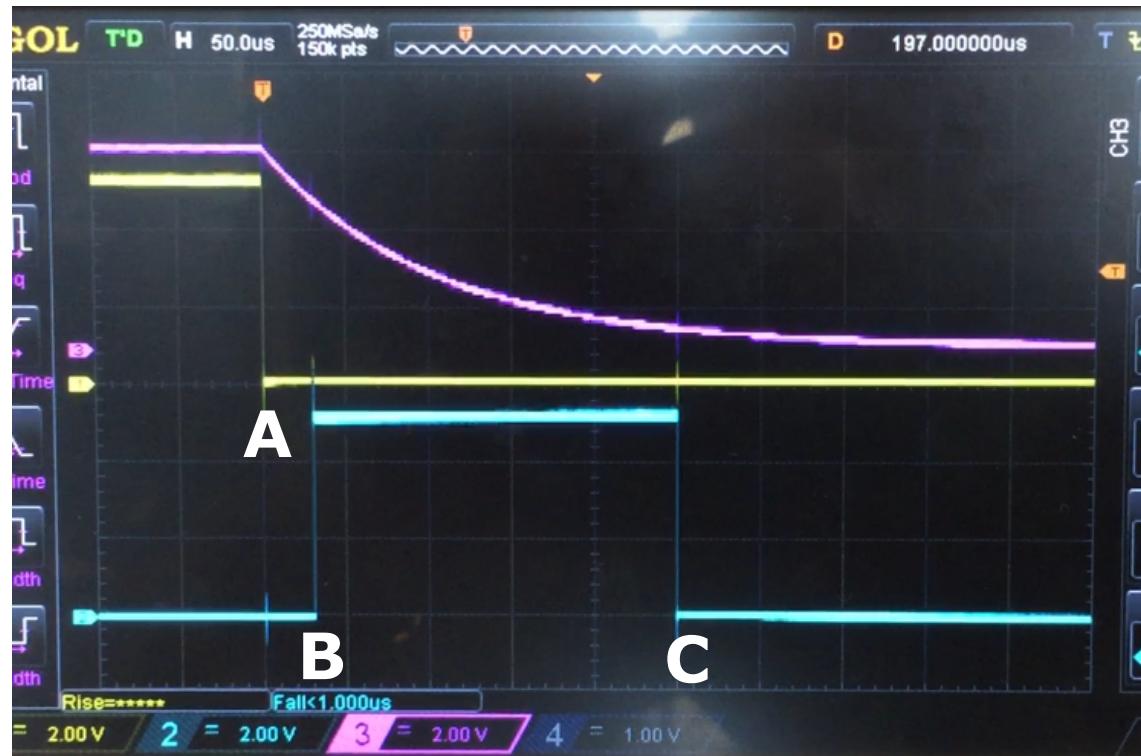
# Temporizzazione azioni

## DEBUG Azioni



# Temporizzazione azioni

## DEBUG Azioni



# Architettura del firmware

## Setup

### Configurazione Seriale

### Configurare Pin 9 e 10 in uscita

### Configurazione Timer 1

- CTC Mode
- STOP (prescaler 0)
- Compare Soglia A – Pin Hi
- Forza transizione
- Compare Soglia A – Pin Low
- Compare Soglia B – Pin Toggle
- Soglia A  $\leftarrow$  tempo  $T_{BC}$
- Soglia B  $\leftarrow$  tempo  $T_{AB}$  (es 10)
- TCNT1  $\leftarrow$  valore START (soglia A-1)
- Abilitare interrupt soglia B

### Configurazione AD

- Eseguire analogRead(A2)
- Abilitare trigger hardware
- Prescaler 128
- Selezionare Timer 1 Soglia B

# Architettura del firmware

## Setup

Configurazione Seriale

Configurare Pin 9 e 10 in uscita

Configurazione Timer 1

- CTC Mode
- STOP (prescaler 0)
- Compare Soglia A – Pin Hi
- Forza transizione
- Compare Soglia A – Pin Low
- Compare Soglia B – Pin Toggle
- Soglia A  $\leftarrow$  tempo  $T_{BC}$
- Soglia B  $\leftarrow$  tempo  $T_{AB}$  (es 10)
- TCNT1  $\leftarrow$  valore START (soglia A-1)
- Abilitare interrupt soglia B

Configurazione AD

- Eseguire analogRead(A2)
- Abilitare trigger hardware
- Prescaler 128
- Selezionare Timer 1 Soglia B

Impostare registro  
TCCR1B

# Architettura del firmware

## Setup

### Configurazione Seriale

### Configurare Pin 9 e 10 in uscita

### Configurazione Timer 1

- CTC Mode
- STOP (prescaler 0)
- Compare Soglia A – Pin Hi
- Forza transizione
- Compare Soglia A – Pin Low
- Compare Soglia B – Pin Toggle
- Soglia A  $\leftarrow$  tempo  $T_{BC}$
- Soglia B  $\leftarrow$  tempo  $T_{AB}$  (es 10)
- TCNT1  $\leftarrow$  valore START (soglia A-1)
- Abilitare interrupt soglia B

### Configurazione AD

- Eseguire analogRead(A2)
- Abilitare trigger hardware
- Prescaler 128
- Selezionare Timer 1 Soglia B

Registri  
TCCR1A, TCCR1B,  
TCCR1C

# Architettura del firmware

## Setup

### Configurazione Seriale

### Configurare Pin 9 e 10 in uscita

### Configurazione Timer 1

- CTC Mode
- STOP (prescaler 0)
- Compare Soglia A – Pin Hi
- Forza transizione
- Compare Soglia A – Pin Low
- Compare Soglia B – Pin Toggle
- Soglia A  $\leftarrow$  tempo  $T_{BC}$
- Soglia B  $\leftarrow$  tempo  $T_{AB}$  (es 10)
- $TCNT1 \leftarrow$  valore START (soglia A-1)
- Abilitare interrupt soglia B

### Configurazione AD

- Eseguire analogRead(A2)
- Abilitare trigger hardware
- Prescaler 128
- Selezionare Timer 1 Soglia B

### Impostare Registri OCR1A, OCR1B, TCNT1

Per il campo del  
campione civetta (3.3 nF)  
è adatto il **prescaler 1:1**

# Architettura del firmware

## Setup

### Configurazione Seriale

### Configurare Pin 9 e 10 in uscita

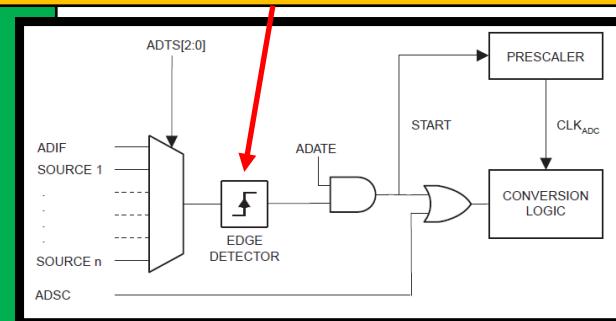
### Configurazione Timer 1

- CTC Mode
- STOP (prescaler 0)
- Compare Soglia A – Pin Hi
- Forza transizione
- Compare Soglia A – Pin Low
- Compare Soglia B – Pin Toggle
- Soglia A  $\leftarrow$  tempo  $T_{BC}$
- Soglia B  $\leftarrow$  tempo  $T_{AB}$  (es 10)
- TCNT1  $\leftarrow$  valore START (soglia A-1)
- **Abilitare interrupt soglia B**

### Configurazione AD

- Eseguire analogRead(A2)
- Abilitare trigger hardware
- Prescaler 128
- Selezionare Timer 1 Soglia B

I'IF deve tornare a 0 per garantire i fronti (il modo più semplice è gestire l'interrupt con una ISR vuota)



### Registri TIMSK1, TIFR1:

```
TIFR1 |= 4;  
TIMSK1 |= 4;
```

E definizione ISR vuota:

# Architettura del firmware

## Setup

### Configurazione Seriale

### Configurare Pin 9 e 10 in uscita

### Configurazione Timer 1

- CTC Mode
- STOP (prescaler 0)
- Compare Soglia A – Pin Hi
- Forza transizione
- Compare Soglia A – Pin Low
- Compare Soglia B – Pin Toggle
- Soglia A  $\leftarrow$  tempo  $T_{BC}$
- Soglia B  $\leftarrow$  tempo  $T_{AB}$  (es 10)
- TCNT1  $\leftarrow$  valore START (soglia A-1)
- Abilitare interrupt soglia B

### Configurazione AD

- Eseguire `analogRead(A2)`
- Abilitare trigger hardware
- Prescaler 128
- Selezionare Timer 1 Soglia B

Metodo «furbo» per far configurare il grosso dei registri ad Arduino

# Architettura del firmware

## Setup

Configurazione Seriale

Configurare Pin 9 e 10 in uscita

Configurazione Timer 1

- CTC Mode
- STOP (prescaler 0)
- Compare Soglia A – Pin Hi
- Forza transizione
- Compare Soglia A – Pin Low
- Compare Soglia B – Pin Toggle
- Soglia A  $\leftarrow$  tempo  $T_{BC}$
- Soglia B  $\leftarrow$  tempo  $T_{AB}$  (es 10)
- TCNT1  $\leftarrow$  valore START (soglia A-1)
- Abilitare interrupt soglia B

Configurazione AD

- Eseguire analogRead(A2)
- Abilitare trigger hardware
- Prescaler 128
- Selezionare Timer 1 Soglia B

Registro  
ADCSRA



# Architettura del firmware

## Setup

### Configurazione Seriale

### Configurare Pin 9 e 10 in uscita

### Configurazione Timer 1

- CTC Mode
- STOP (prescaler 0)
- Compare Soglia A – Pin Hi
- Forza transizione
- Compare Soglia A – Pin Low
- Compare Soglia B – Pin Toggle
- Soglia A  $\leftarrow$  tempo  $T_{BC}$
- Soglia B  $\leftarrow$  tempo  $T_{AB}$  (es 10)
- TCNT1  $\leftarrow$  valore START (soglia A-1)
- Abilitare interrupt soglia B

### Configurazione AD

- Eseguire analogRead(A2)
- Abilitare trigger hardware
- Prescaler 128
- **Selezionare Timer 1 Soglia B**

Registro  
ADCSR

# Architettura del firmware

## Loop

Attesa carica capacità (delay)

Start Timer 1 (prescaler  $\leftarrow 1:1$ )

Misura V0

- Cancellazione IF ADC
- Attesa IF ADC
- Lettura V0  $\leftarrow$  ADC

Misura V1

- Cancellazione IF ADC
- Attesa IF ADC
- Lettura V1  $\leftarrow$  ADC

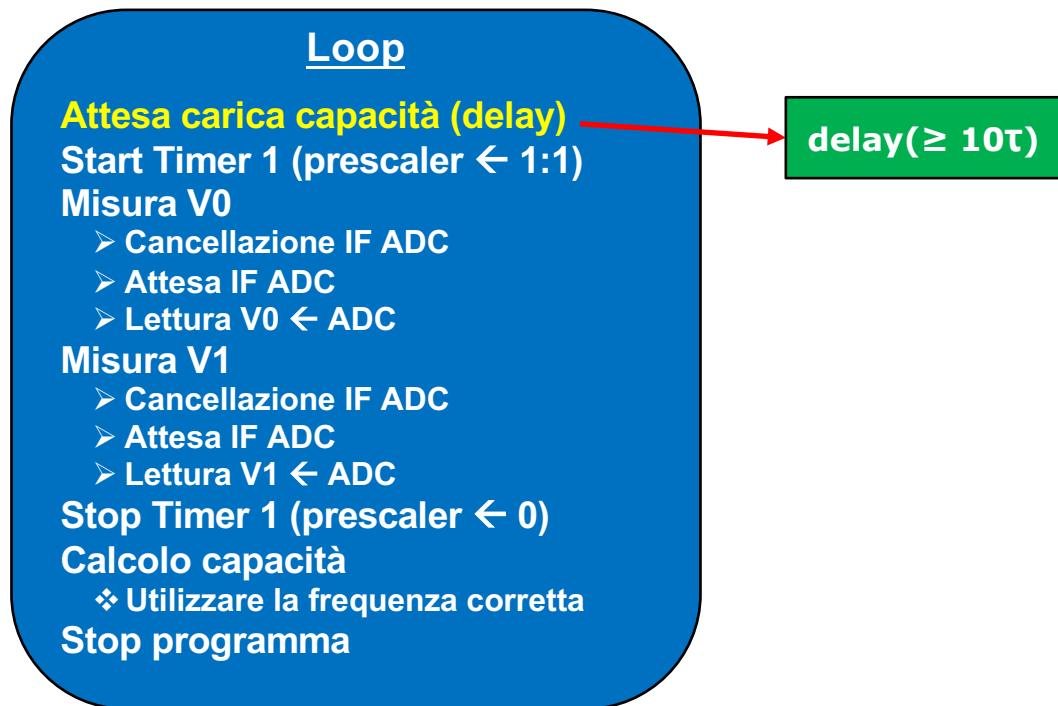
Stop Timer 1 (prescaler  $\leftarrow 0$ )

Calcolo capacità

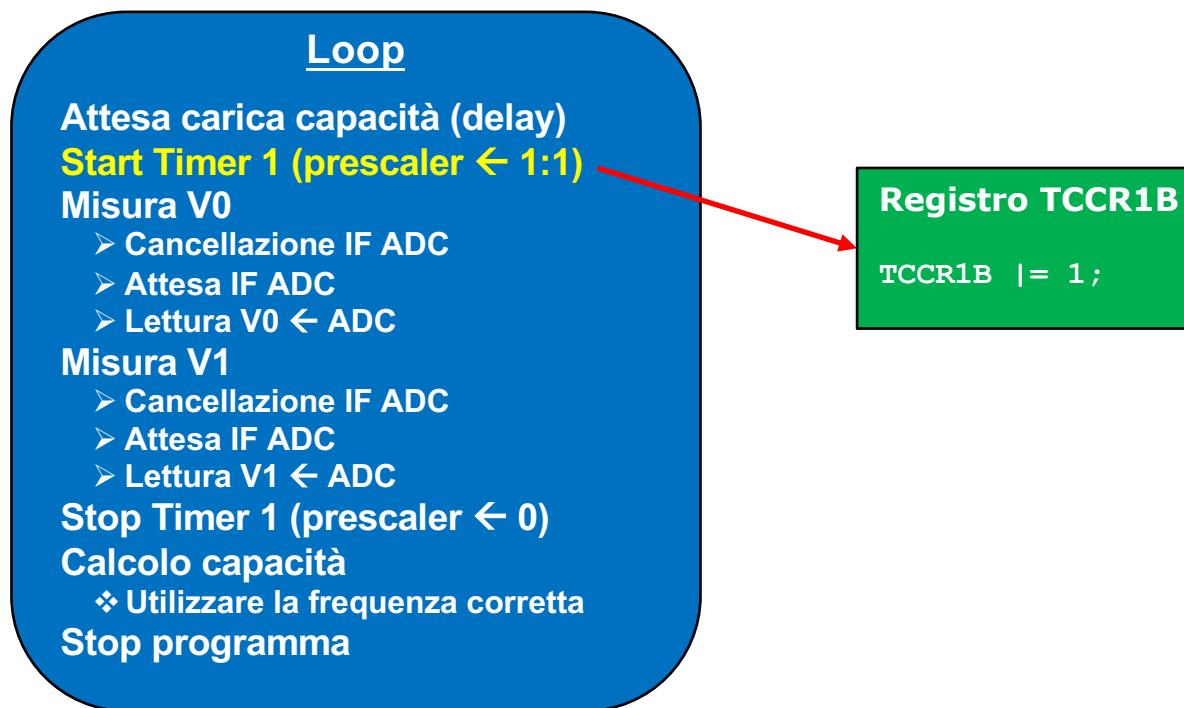
- ❖ Utilizzare la frequenza corretta

Stop programma

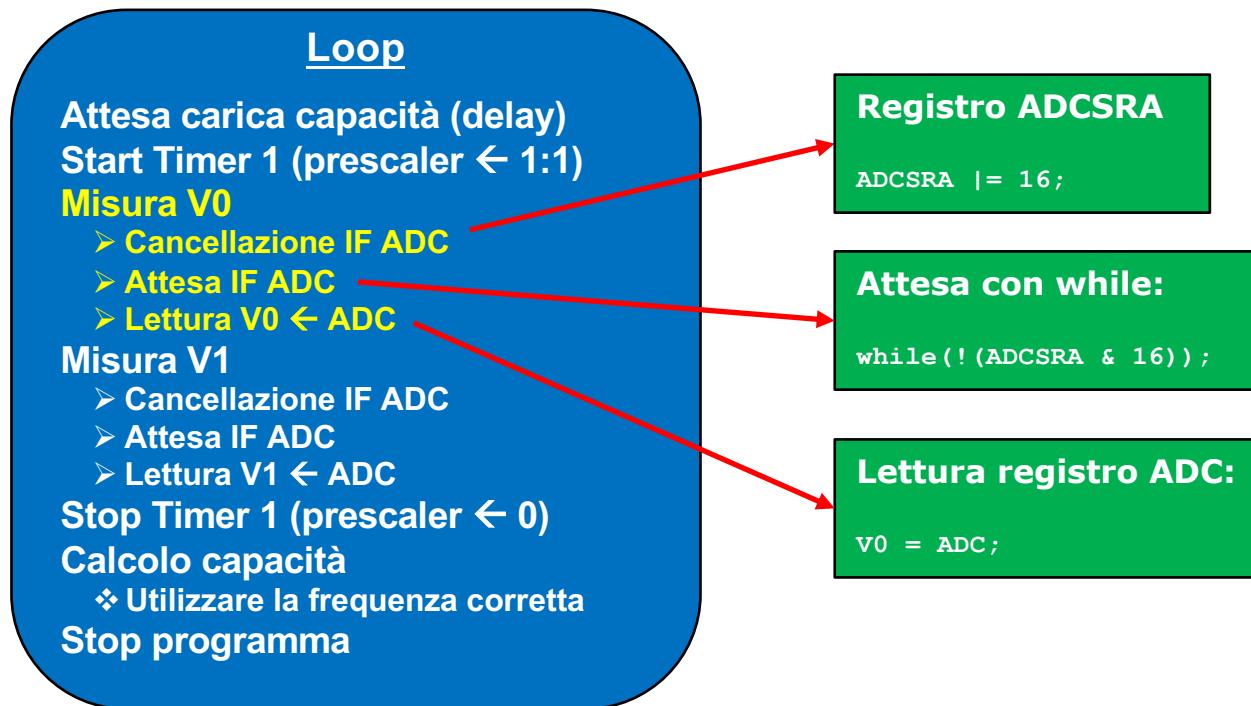
# Architettura del firmware



# Architettura del firmware



# Architettura del firmware



# Architettura del firmware

## Loop

Attesa carica capacità (delay)

Start Timer 1 (prescaler  $\leftarrow 1:1$ )

Misura V0

➢ Cancellazione IF ADC

➢ Attesa IF ADC

➢ Lettura V0  $\leftarrow$  ADC

Misura V1

➢ Cancellazione IF ADC

➢ Attesa IF ADC

➢ Lettura V1  $\leftarrow$  ADC

Stop Timer 1 (prescaler  $\leftarrow 0$ )

Calcolo capacità

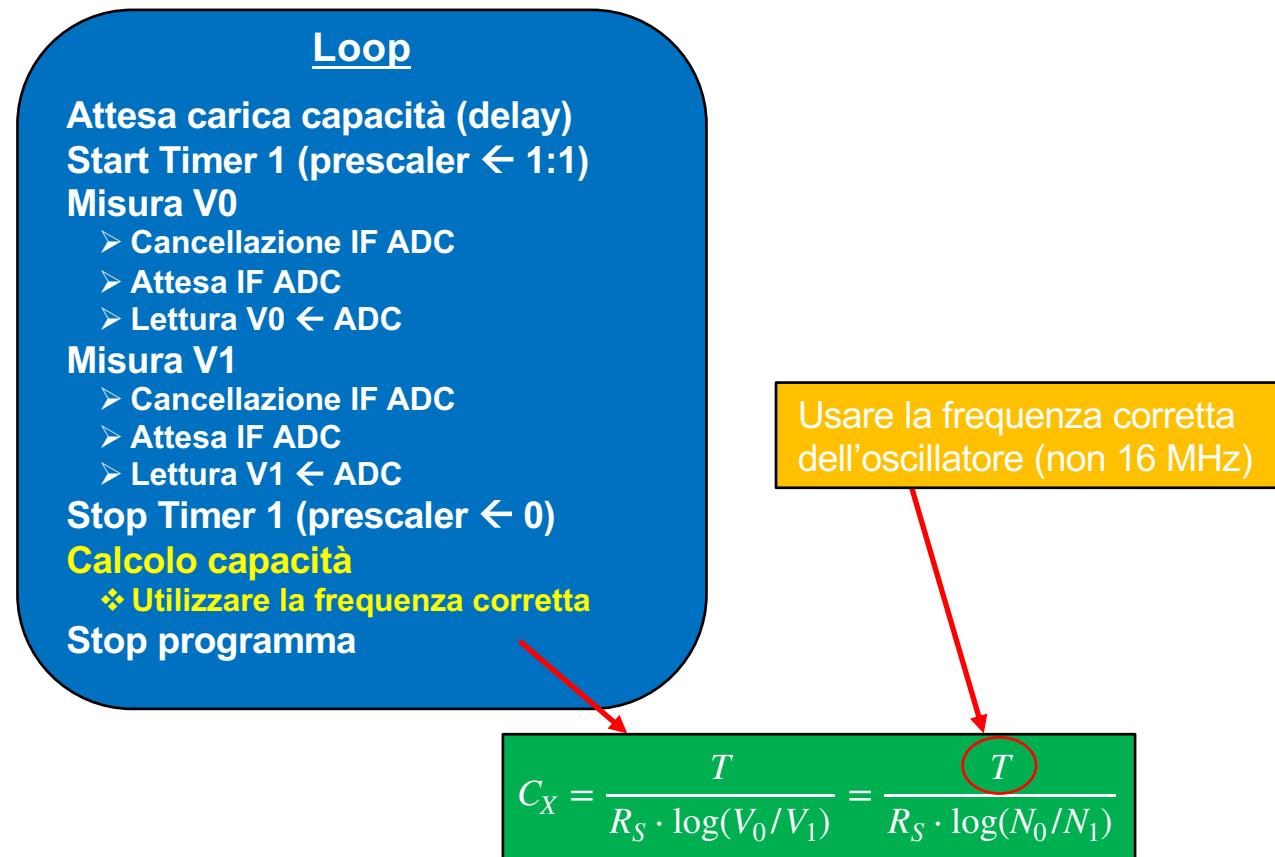
❖ Utilizzare la frequenza corretta

Stop programma

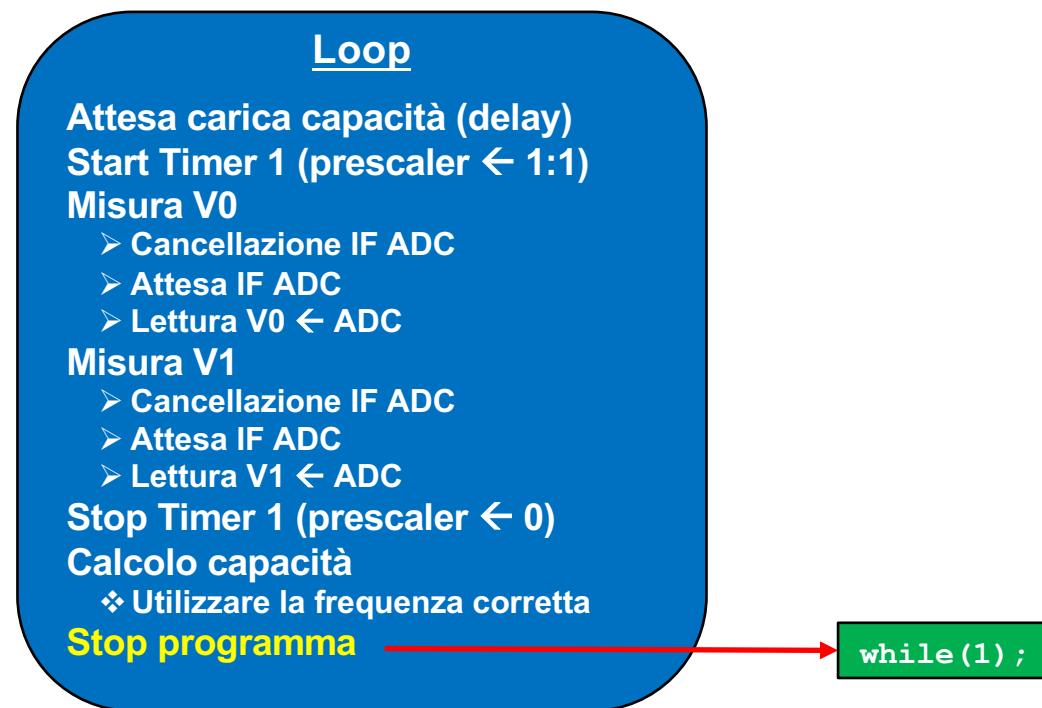
## Registro TCCR1B

```
TCCR1B &= ~1;
```

# Architettura del firmware



# Architettura del firmware



# Cosa bisogna fare

## Operazioni preliminari di progetto:

- Calcolare l'incertezza e ottimizzare il dimensionamento ( $C_x=3.3 \text{ nF}$ ):
  - ❖ Scelta del resistore  $R_s$  e del tempo  $T$  (usare prescaler 1:1)
- Calcolare la risoluzione di misura

## Prima sul simulatore e poi in laboratorio:

- Scrivere il programma e verificarne il funzionamento con l'oscilloscopio, misurando una capacità da 3.3 nF
- Modificare il programma per fare letture ripetute ed eseguire 1000 misure della capacità.
- Stimare valore medio e deviazione standard delle misure,  $u_A(C_x)$
- Misurare il campione civetta (tempo disponibile 30 s)

# Architettura del firmware

Per ripetere più volte la misurazione copiare le istruzioni di configurazione dalla setup all'inizio del Loop e togliere lo stop:

