

# Progetto Capacimetro «Advanced» con Arduino UNO e LabVIEW

© 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

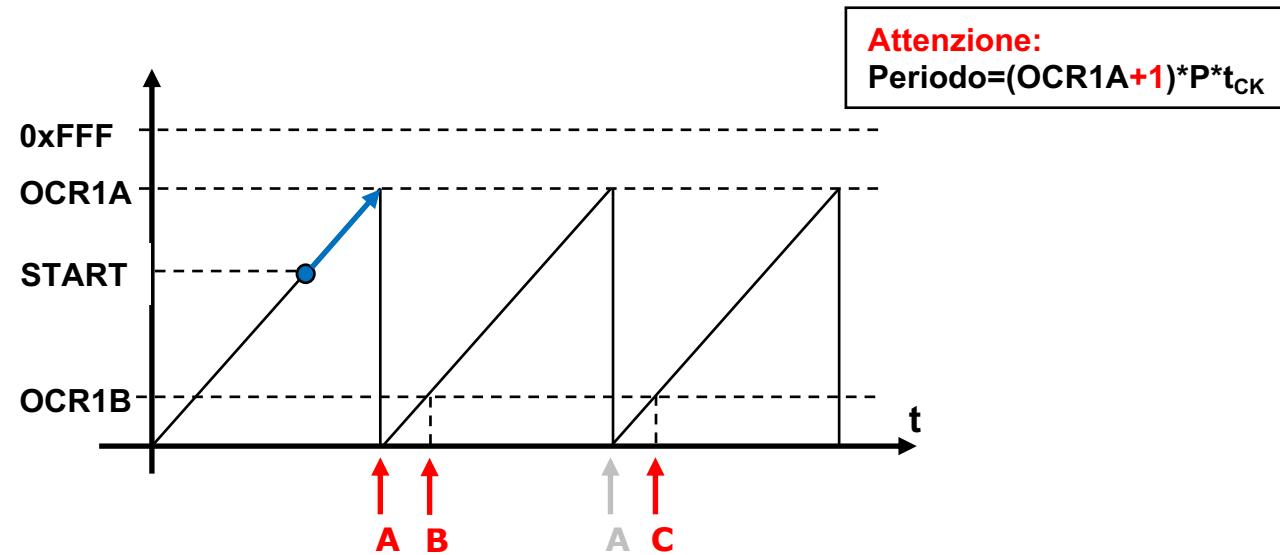
## Difetti principali della versione base:

- Campo di misura limitato verso le basse capacità
  - ❖ Bassa frequenza di campionamento
- Risultato fortemente dipendente dai due punti scelti per fare la misurazione. In ogni punto si ha un effetto diverso dell'errore di:
  - ❖ offset
  - ❖ non linearità

$$C_x = \frac{T}{R_S \cdot \log(V_0/V_1)} = \frac{T}{R_S \cdot \log(N_0/N_1)}$$

# Temporizzazione azioni

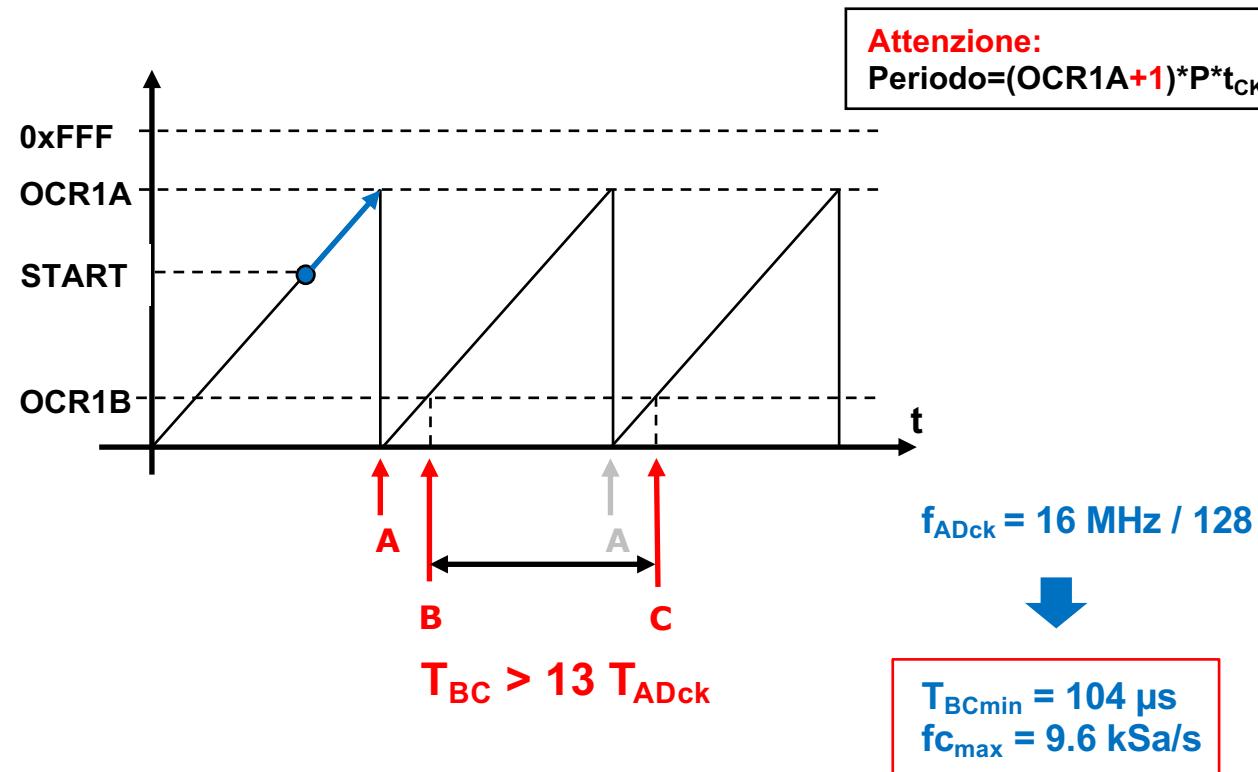
Tecnica usata nella versione base



Qual è la massima frequenza di campionamento  
possibile (minimo tempo  $T_{BC}$ )?

# Temporizzazione azioni

Tecnica usata nella versione base



# Capacimetro

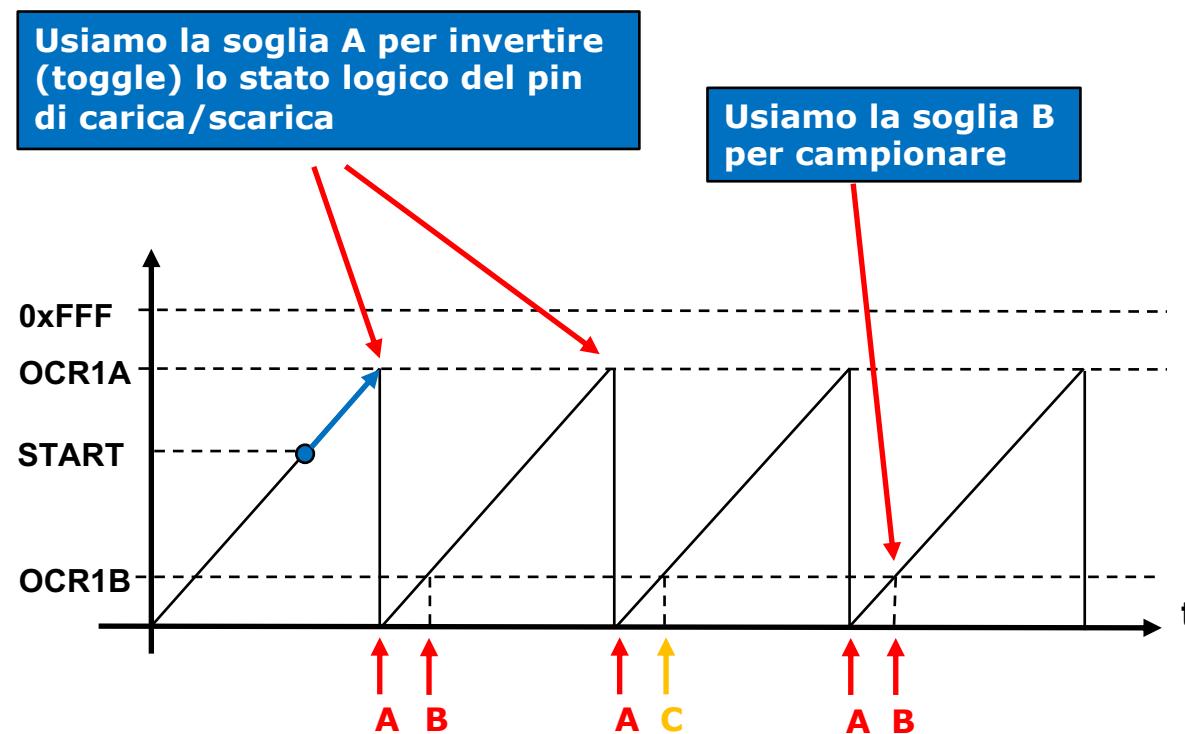
## Obiettivo:

Migliorare la versione base del capacimetro ed estendere il campo di misura per piccole capacità (1 pF - 10 nF), tramite:

- Acquisizione di molti punti della scarica
- Campionamento alla massima frequenza possibile (mantenendo la massima risoluzione,  $f_{AD} < 200$  kHz)
- Fitting esponenziale per compensare l'errore di offset
- Applicazione del dithering per aumentare la risoluzione
- Controllo grafico attraverso LabVIEW
- Compensazione effetti parassiti significativi

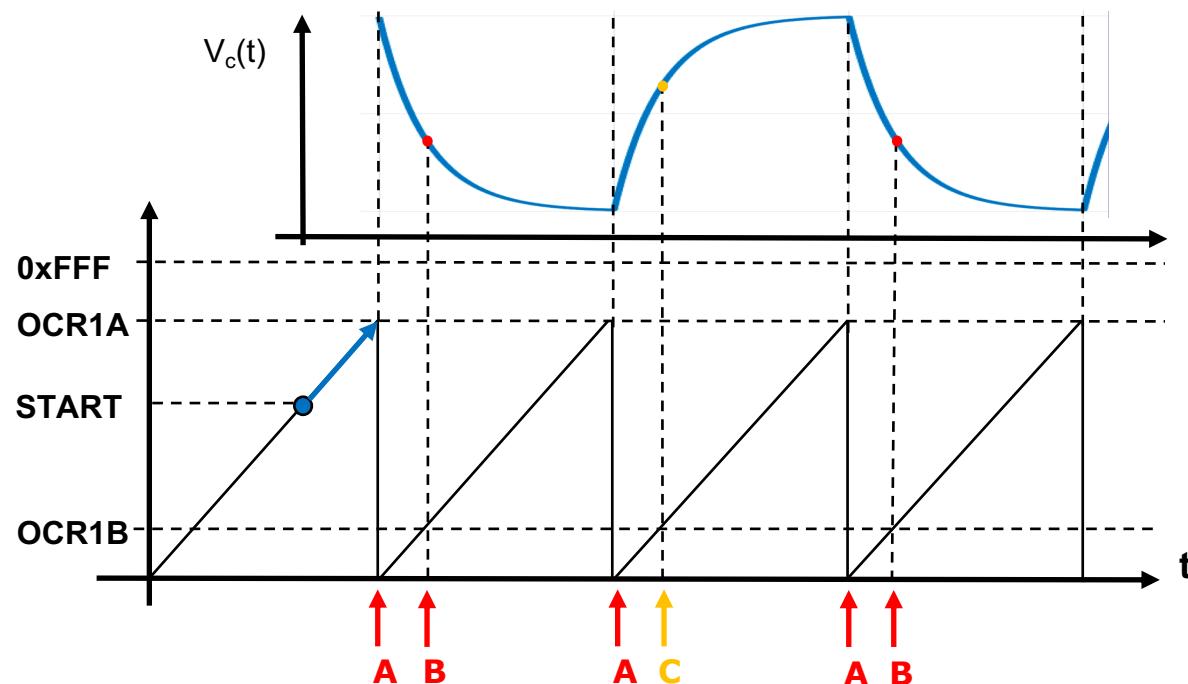
# Campionamento

Tecnica di campionamento:



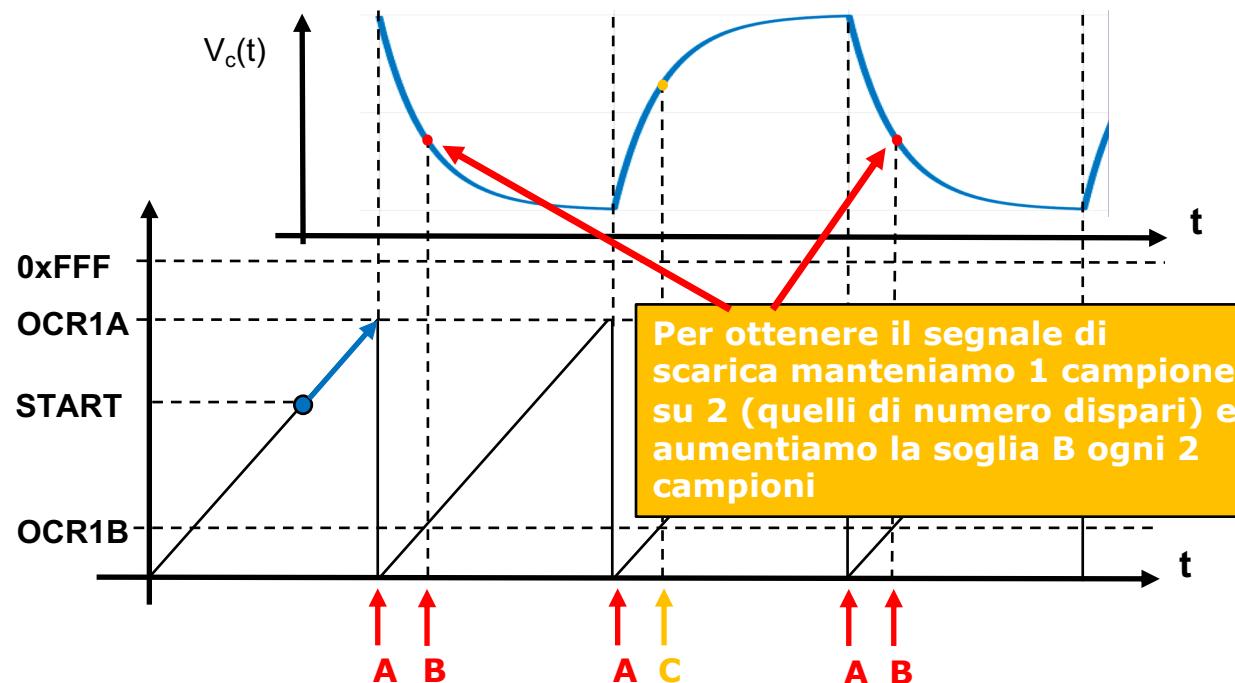
# Campionamento

Tecnica di campionamento:



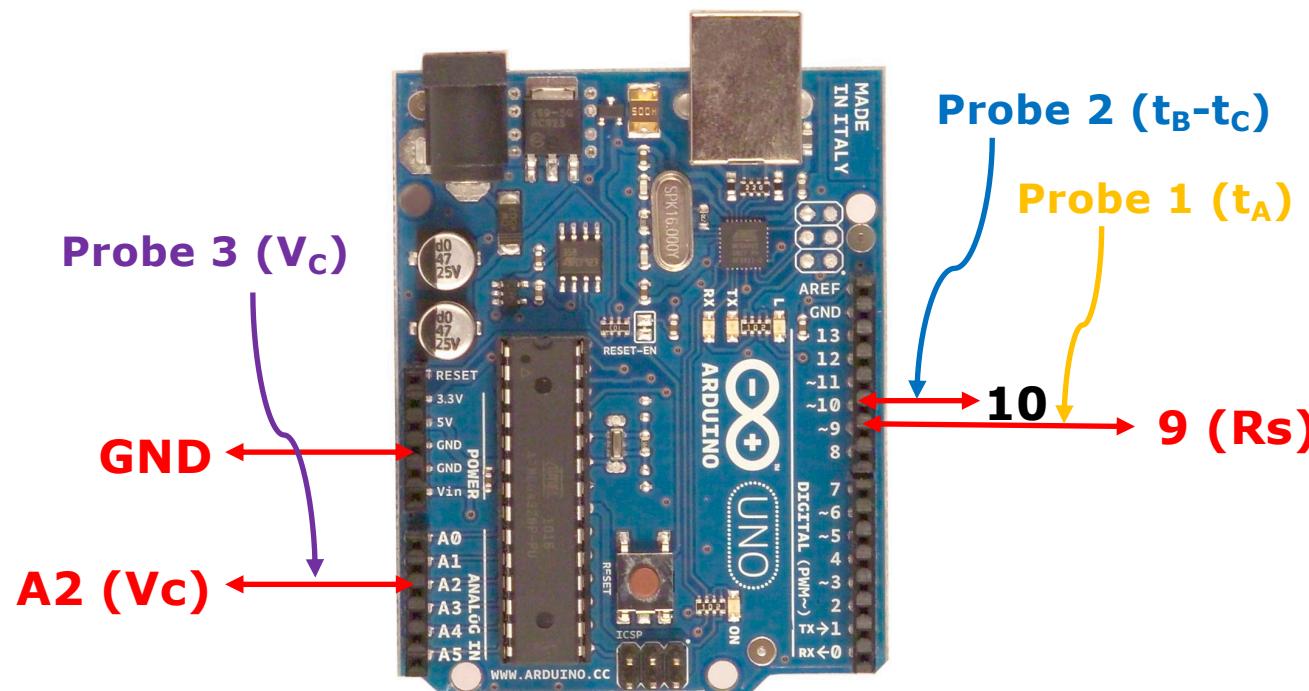
# Campionamento

Tecnica di campionamento:



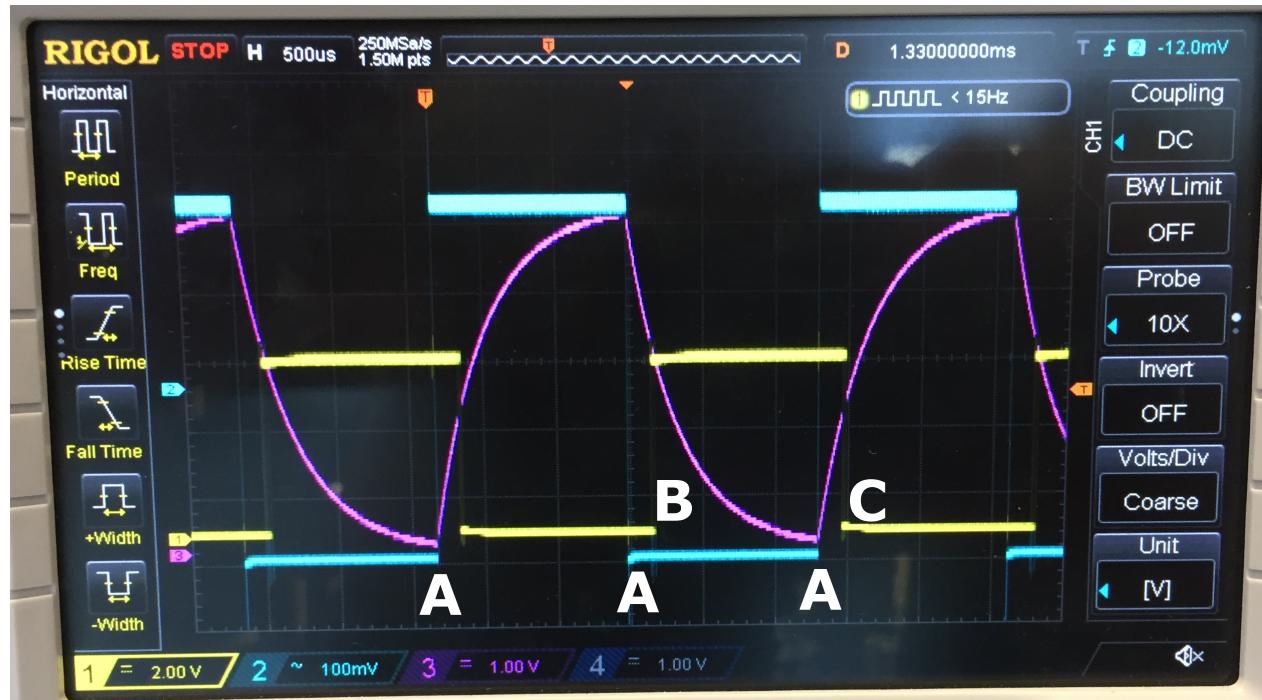
# Temporizzazione azioni

DEBUG Azioni (stessi collegamenti)



# 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 Toggle**
- Compare Soglia B – Pin Toggle
- **Soglia A ← tempo carica/scarica**
- **Soglia B ← tempo campione #1 (0)**
- TCNT1 ← valore START (soglia A-1)
- Abilitare interrupt soglia B

### Configurazione AD

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

Praticamente invariata,  
tranne 3 modifiche  
(in colore giallo)

# 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 Toggle**
- Compare Soglia B – Pin Toggle
- **Soglia A ← tempo carica/scarica**
- **Soglia B ← tempo campione #1 (0)**
- TCNT1 ← valore START (soglia A-1)
- Abilitare interrupt soglia B

### Configurazione AD

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

Per sfruttare il dithering i disturbi della rete (50 Hz) dovrebbero essere scorrelati dal periodo di campionamento:

Scelgo un T usando un multiplo *brutto* (primo) del clock di sistema:

→ Usiamo 21013

# Architettura del firmware

```
#define NPOINTS    151
#define DT         120

void setup() {

    Serial.begin(57600);
    pinMode(9, OUTPUT);
    pinMode(10, OUTPUT);

    // Configurazione TIMER 1
    TCCR1B = 0b00001000; // Modalita CTC, no clock
    TCCR1A = 0b11010000; // Configura PIN HI su soglia A
    TCCR1C |=0b10000000; // Forza l'output (HI) simulando l'evento soglia A
    TCCR1A = 0b01010000; // Configura PIN Toggle su soglia A

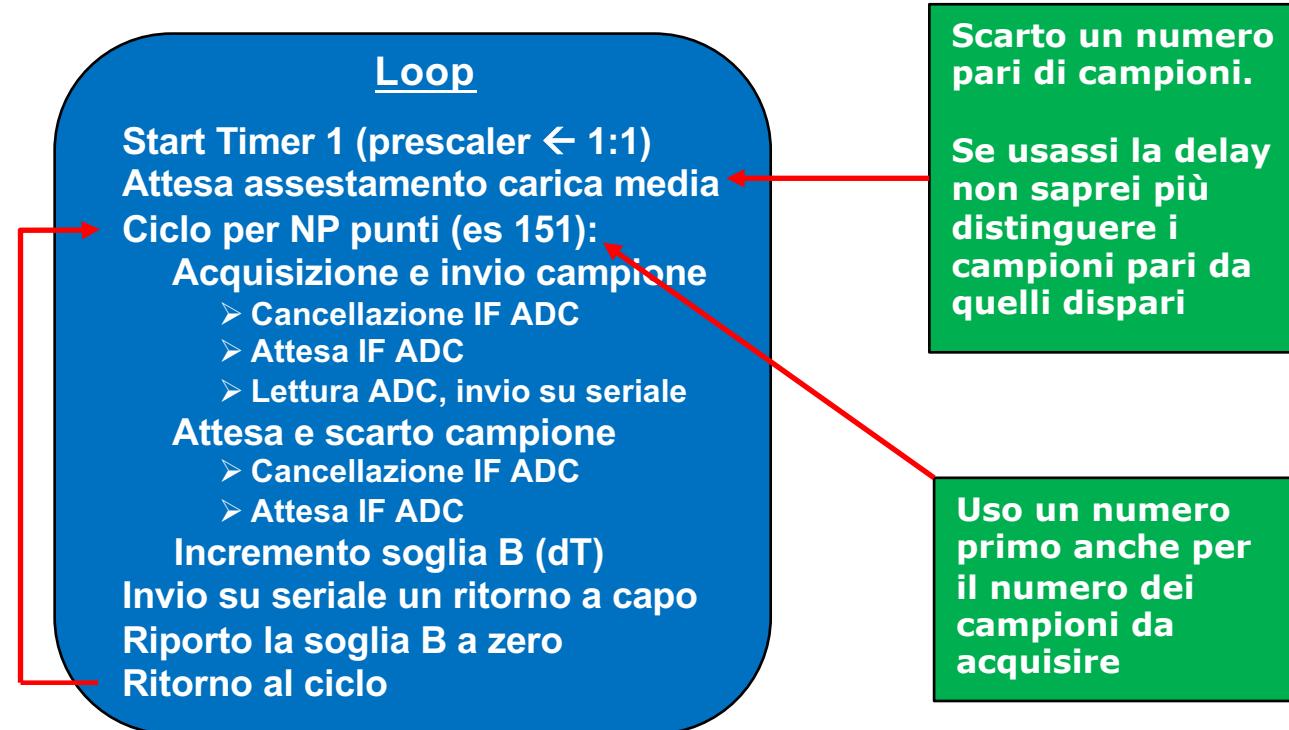
    OCR1A = 21013-1;      // Usa un periodo "asincrono" con 20 mS (1/50 Hz)
    OCR1B = 0;             // TA = 0 us
    TCNT1 = OCR1A-1;      // Inizia quasi subito la scarica

    TIFR1 |= 4;            // Cancella interrupt soglia B
    TIMSK1 |= 4;           // Abilita interrupt soglia B

    // Configurazione ADC
    analogRead(A2);
    ADCSRA = 0b10110111; // ON, ADATE=1, IF Clear, IE = 0, Prescaler = 111 (1:128)
    ADCSRB = 0b00000101; // Trigger hardware: Timer 1 soglia B
}

ISR(TIMER1_COMPB_vect) {}
```

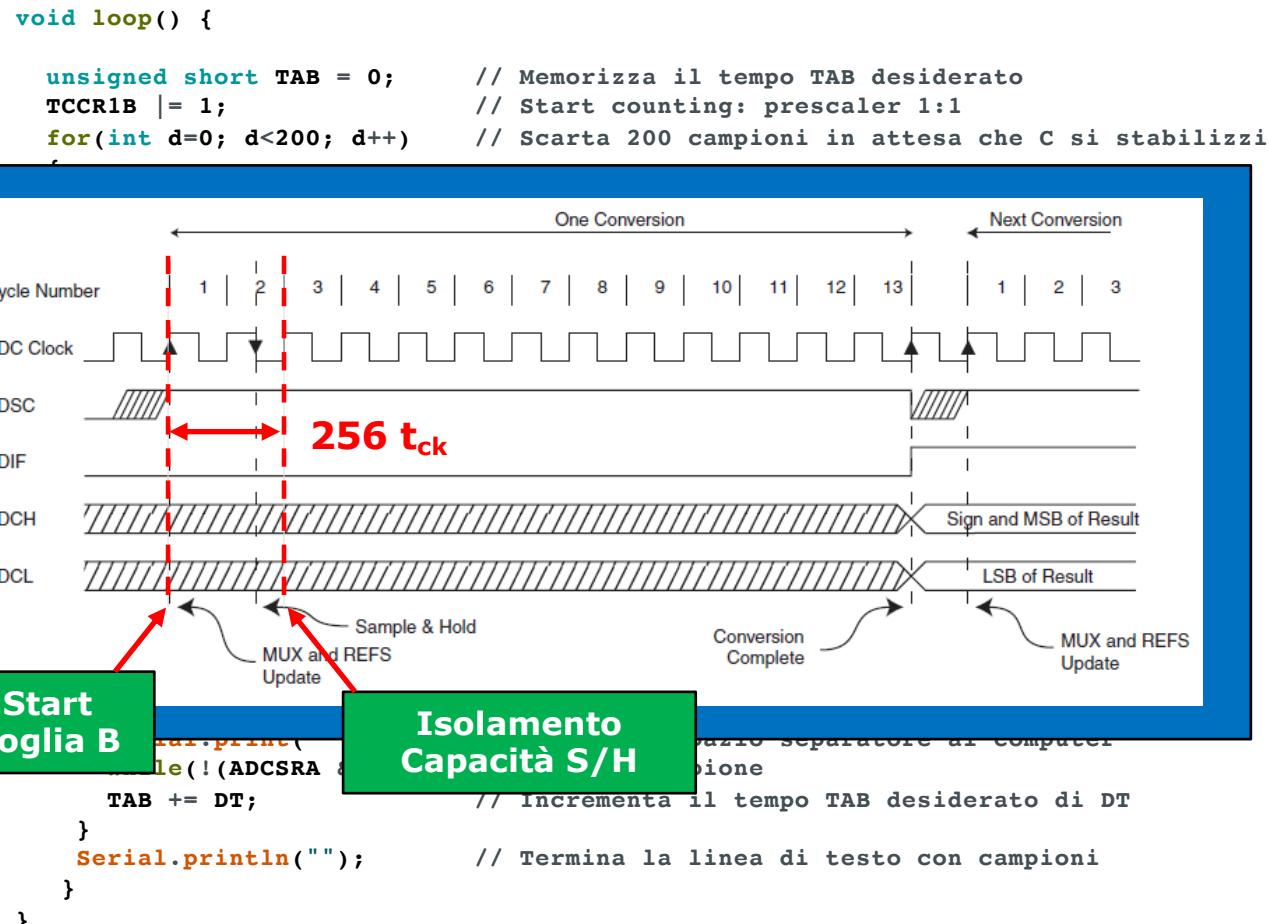
# Architettura del firmware



# Architettura del firmware

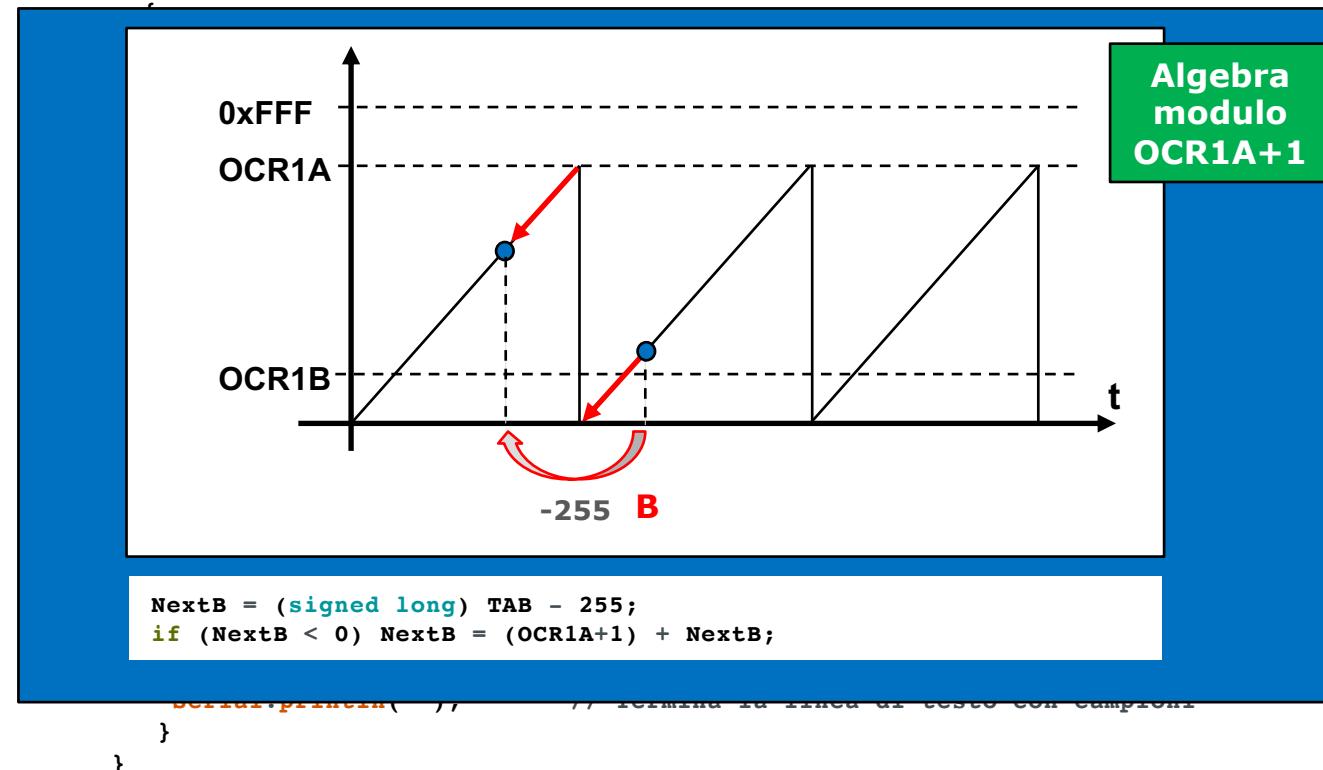
```
void loop() {  
  
    unsigned short TAB = 0;          // Memorizza il tempo TAB desiderato  
    TCCR1B |= 1;                   // Start counting: prescaler 1:1  
    for(int d=0; d<200; d++)       // Scarta 200 campioni in attesa che C si stabilizzi  
    {  
        ADCSRA |= 16;             // Clear IF  
        while(!(ADCSRA & 16));   // Attende campione  
    }  
  
    while(1)                      // Ciclo di acquisizione infinito  
    {  
        TAB = 0;                  // Azzera il tempo TAB desiderato  
        signed long NextB;        // Variabile di supporto  
        for(int r=0; r < NPOINTS; r++) // Ciclo di acquisizione di NPOINTS punti  
        {  
            NextB = (signed long) TAB - 255; // Calcola la soglia B reale  
            if (NextB < 0) NextB = (OCR1A+1) + NextB; // Corregge la soglia B reale  
            OCR1B = NextB;                // Rende effettiva la soglia B reale  
            ADCSRA |= 16;                // Clear IF ADC  
            while(!(ADCSRA & 16));   // Attende primo campione  
  
            ADCSRA |= 16;             // Clear IF ADC  
            Serial.print(ADC);        // Invia il campione al computer  
            Serial.print(" ");        // Invia uno spazio separatore al computer  
            while(!(ADCSRA & 16));   // Attende campione  
            TAB += DT;                // Incrementa il tempo TAB desiderato di DT  
        }  
        Serial.println("");        // Termina la linea di testo con campioni  
    }  
}
```

# Architettura del firmware

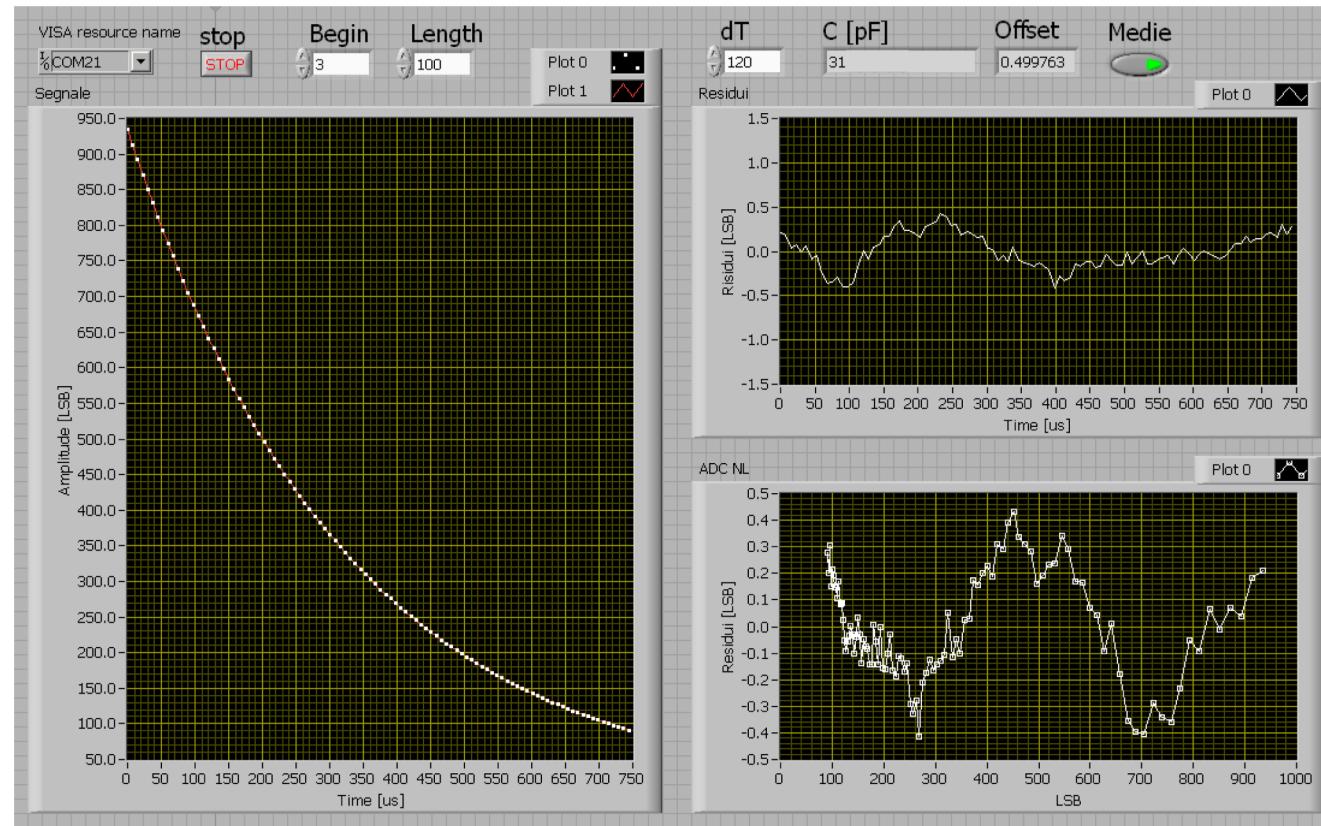


# Architettura del firmware

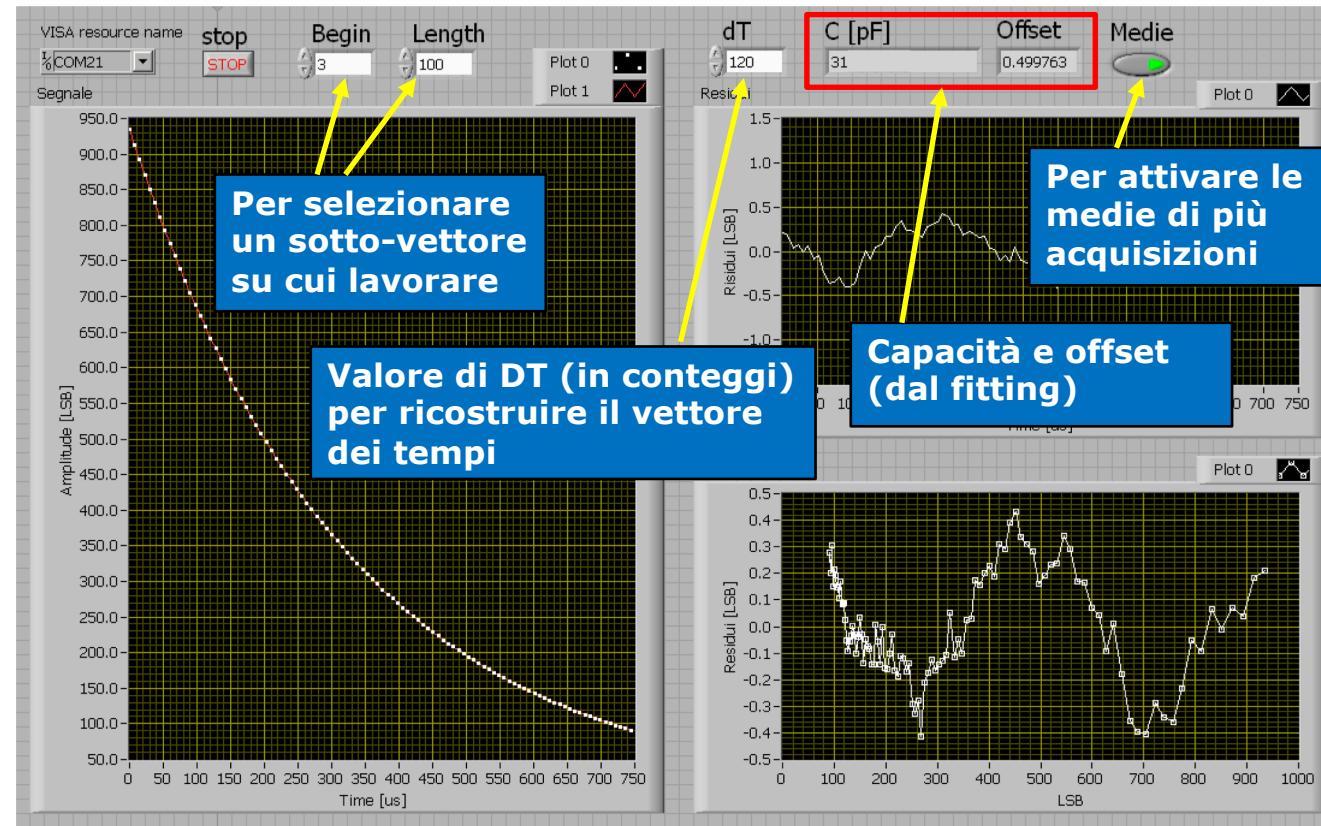
```
void loop() {
    unsigned short TAB = 0;          // Memorizza il tempo TAB desiderato
    TCCR1B |= 1;                   // Start counting: prescaler 1:1
    for(int d=0; d<200; d++)      // Scarta 200 campioni in attesa che C si stabilizzi
    {
        ...
    }
}
```



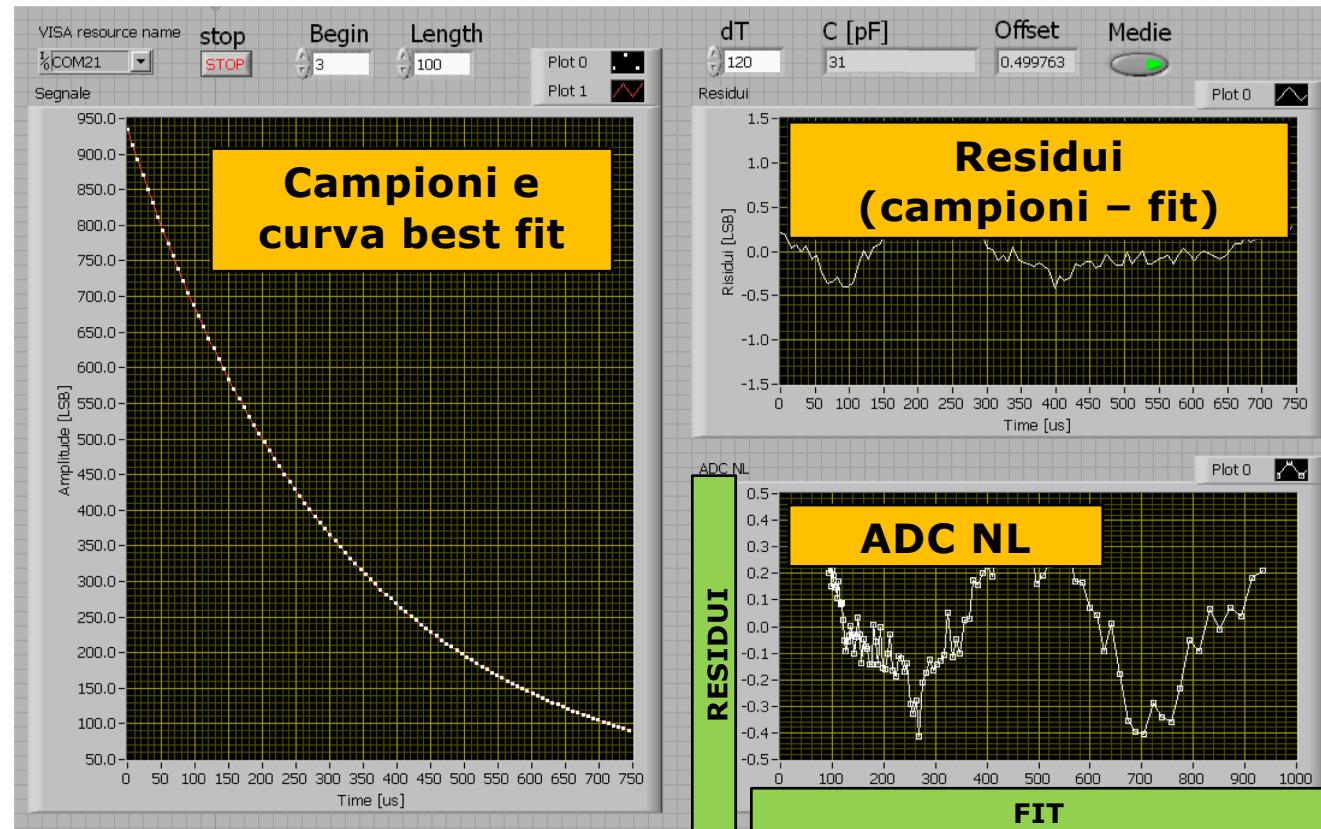
# Programma LabVIEW



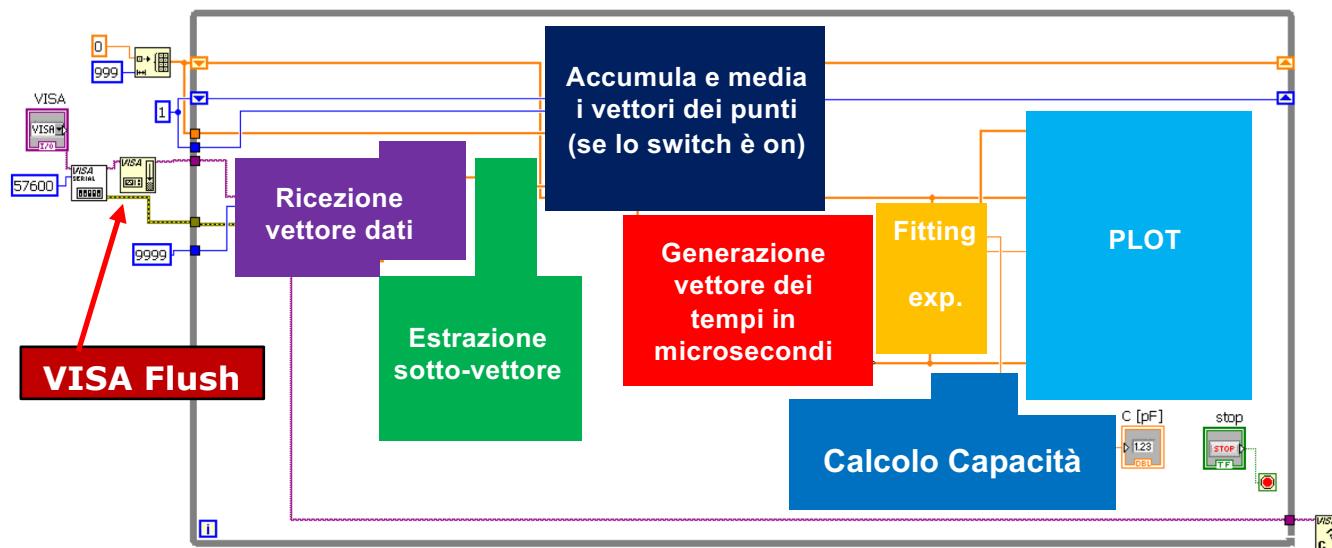
# Programma LabVIEW



# Programma LabVIEW

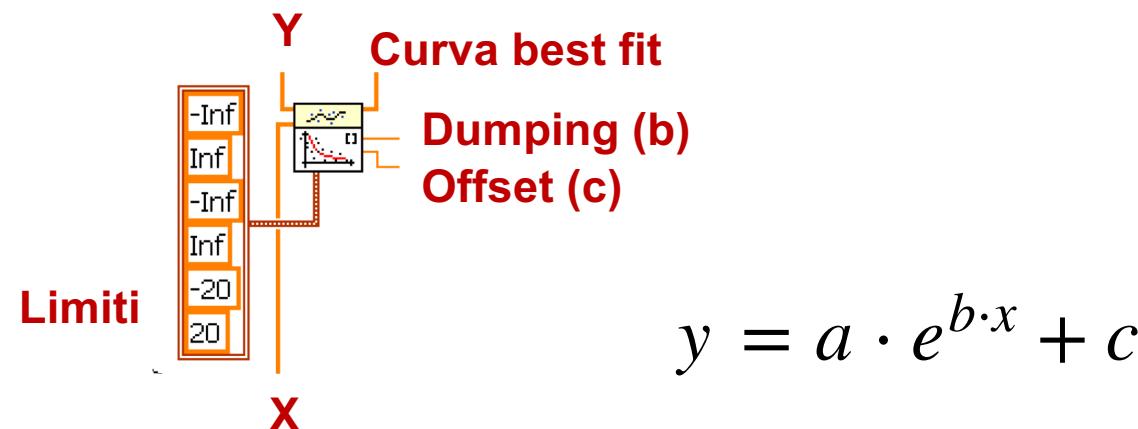
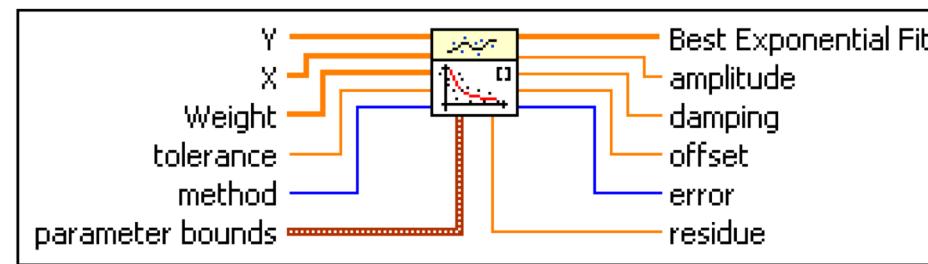


# Programma LabVIEW



# Programma LabVIEW

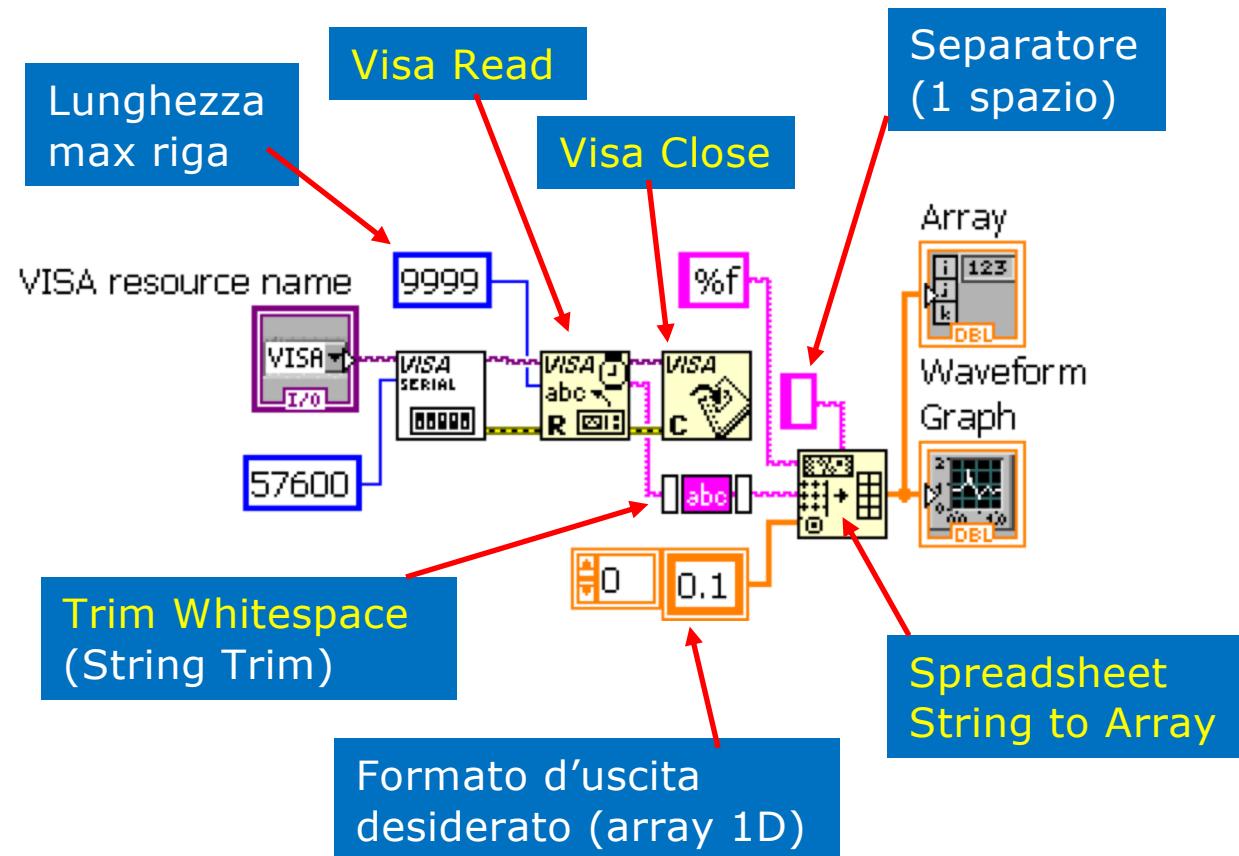
## Fit esponenziale



## **Suggerimento su come procedere**

- Riprodurre l'esempio costruito in aula per ricevere un vettore di dati in formato testo (aggiungere il VISA Flush).
- Ricostruire il vettore con i tempi dei campioni (in microsecondi)
- Visualizzare i punti acquisiti
- Aggiungere il blocco per estrarre un sotto-vettore
- Eseguire il fit esponenziale e aggiungere sul grafico la curva ottenuta
- Calcolare e visualizzare i residui
- Calcolare e visualizzare la capacità
- Inserire i blocchi per accumulare e mediare i vettori
- Compilare il report

## Esempio di comunicazione con Arduino - Modalità Testo



## Per salvare su File i risultati

