Ingresso analogico uscita analogica [PWM]

INGRESSO ANALOGICO USCITA ANALOGICA [PWM]

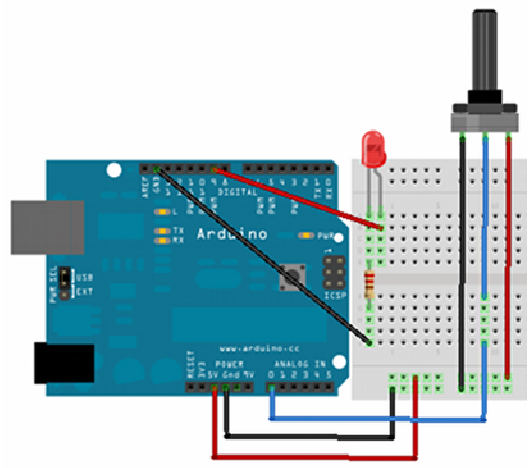
Questo esempio mostra il procedimento di acquisizione di un ingresso analogico e la sua emissione analogica in uscita con il formato PWM.

REQUISITI HARDWARE

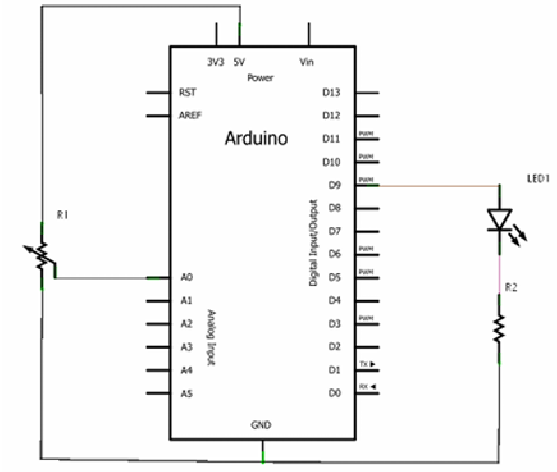
- Scheda Arduino Uno
- Breadboard
- LED
- Resistenza da 220 ohm
- Potenziometro rotativo
- Cavo USB

CIRCUITO

Per costruire il circuito, montare sulla breadboard il potenziometro rotativo e collegare il pin sinistro al pin GND della morsettiera POWER, il pin destro al pin 5V della morsettiera POWER ed il pin centrale al pin A0 della morsettiera ANALOG.
Montare sulla breadboard il LED e la resistenza. Il catodo del LED (polo negativo = piedino più corto) va direttamente collegato ad un lato della resistenza. L'anodo del LED (polo positivo = piedino più lungo) va collegato attraverso un filo di protipizzazione al pin 9 (PWM) della morsettiera DIGITAL di Arduino. L'atro lato della resistenza va collegato al pin GND.

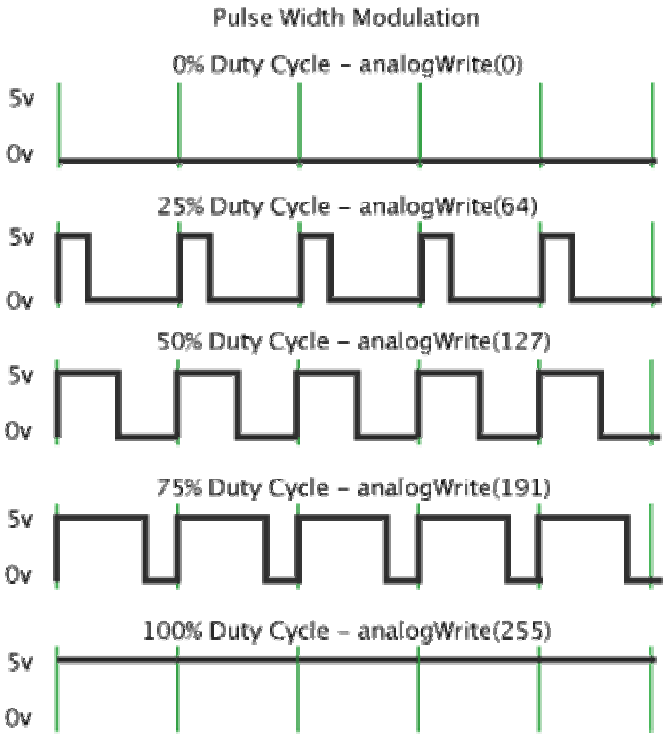


Lo schema elettrico del circuito è il seguente:



USCITE PWM

I pin 3, 5, 6, 9, 10 ed 11 della morsettiera DIGITAL possono essere usati come uscite PWM.
PWM (Pulse Width Modulation) è una tecnica per ottenere una uscita analogica su un canale digitale. Il canale digitale viene utilizzato per creare un'onda quadra a frequenza fissa, un segnale che commuta tra on e off. Questa commutazione on-off genera tensioni che passano bruscamente dal livello basso (0 V) al livello alto (5 Volt). Modificando la parte di tempo che la tensione passa al livello alto rispetto al livello basso si ottengono diversi valori dell'uscita.
La durata di tempo che il segnale passa al livello alto (on time) è chiamata la larghezza di impulso (Pulse Width). Per ottenere diversi valori analogici, si cambia la larghezza dell'impulso senza medicare il periodo dell'onda quadra.
Di conseguenza si possono ottenere "on time" che vanno da 0:il segnale rimane a zero per tutto il periodo) al valore massimo (255) in cui il segnale rimane a 1 per tutto il periodo passando attraverso 256 valori intermedi in cui la proporzione tra "on time" ed "off time" cambia.
Questa proporzione tra "on time" ed "off time" viene chiamata "duty cycle" (ciclo di lavoro) e si esprime in percentuale.
Nel grafico sottostante che mostra la tensione (Y) di uscita in funzione del tempo (X) , le linee verdi dividono i periodi dell'onda quadra. Tale durata è l'inverso della frequenza PWM.
Ad esempo, con frequenza PWM di circa 500Hz, le linee verdi distano circa 2 millisecondi.
Una chiamata alla funzione analogWrite (valore) può impostare il duty cicle ad uno dei 256 valori della scala da 0 - 255.
Ad esempio una analogWrite (255) genera un duty cycle del 100% (sempre acceso), e analogWrite (127) un duty cycle del 50% (acceso metà del tempo) ed una analogWrite(0) genera un duty cicle del 0% (sempre spento).



Sebbene l'uscita sia ancora digitale ha una natura analogica espressa dalla energia emessa dal segnale che è proporzionale al'area dell' "on time".
Con questa tecnica è possibile quindi pilotare dispositivi che siano sensibili all'energia che viene erogata.
Un impiego tipico è il pilotaggio di motori in corrente continua usati spesso in robotica.
In questo esercizio viene usata per pilotare un LED la cui intensità di accensione è determinata dalla energia ricevuta.
La frequenza di default è di 490 Hertz

SVILUPPO DEL PROGRAMMA

Collegare la scheda Arduino Uno al computer mediante i cavo USB ed avviare l'ambiente di sviluppo Arduino.
In uno Sketch vuoto inserire il seguente programma:

```
/*
 * Ingresso analogico uscita analogica PWM
 *
 * Questo è l'esempio gestisce un ingresso analogico
 * ed invia il risultato ad una uscita analogica PWM
 */

int analogValue;
int brightness;

void setup(){
  pinMode(9, OUTPUT); //uscita PWM
}

void loop(){
  analogValue=analogRead(A0);
  brightness=analogValue/4;
  analogWrite(9, brightness);
}
```

FUNZIONE SETUP

Il pin 9 viene impostato come uscita per consentire l'emissione di un segnale PWM che piloti l'intensità di accensione di un LED.

FUNZIONE LOOP

In questa applicazione vengono utilizzate due variabili intere etichettate "analogValue" e "brightness". Entrambe sono di tipo intero. Nel linguaggio Arduino il tipo int è allocato in due bytes quindi può assumere valori da -32768 a +32767.
Le due variabile hanno però un range limitato.
La variabile analogValue è destinata a contenere la conversione del valore analogico letto dal canale A0 che è a 10 bit quindi assume valori da 0 a 1023.
La variabile brightness è destinata a contenere il valore della uscita PWM che è ad 8 bit quindi può assumere valori da 0 a 255.

Nella funzione loop() viene acquisito il valore della tensione partizionata dal potenziometro che dipende dall'angolo di rotazione e viene convertito in digitale con una risoluzione di 10 bit.

```
analogValue=analogRead(A0);
```

Il risultato della conversione A/D viene archiviato nella variabile analogValue con una risoluzione di 10 bit (valori da 0 a 1023).

Il risultato della conversione adattato per entrare nel range di uscita del canale PWM.

```
brightness=analogValue/4;
```

Poiché il segnale di ingresso è a 10 bit mentre quello di uscita è a 8 bit si deve dividere il segnale di ingresso per 4 per farlo entrare nel range di quello di uscita.

Il valore così ottenuto viene inviato in forma PWM al pin di uscita.

```
analogWrite(9, brightness);
```

La funzione analogWrite() avvia l'emissione di una onda quadra a 490 Hertz sul pin 9.
Il valore del duty cycle dipende dal valore della variabile "brightness" che può andare da 0 a 255 provocando una brillantezza del led proporzionale al valore della variabile.
La variabile brightness dipende proporzionalmente attraverso la variabile analogValue dalla rotazione del potenziometro quindi la brillantezza del LED è proporzionale alla rotazione del potenziometro.