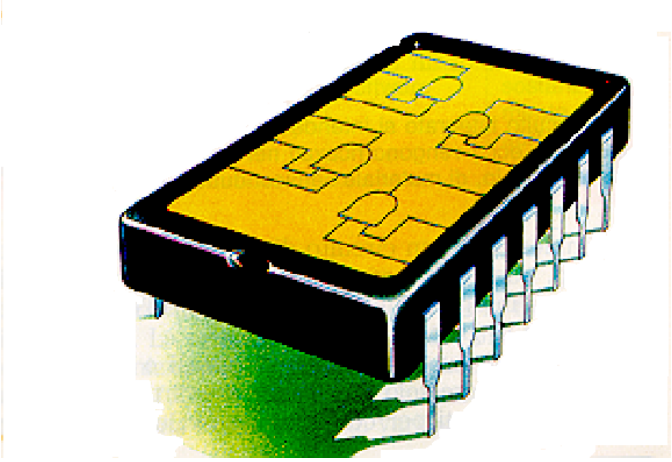
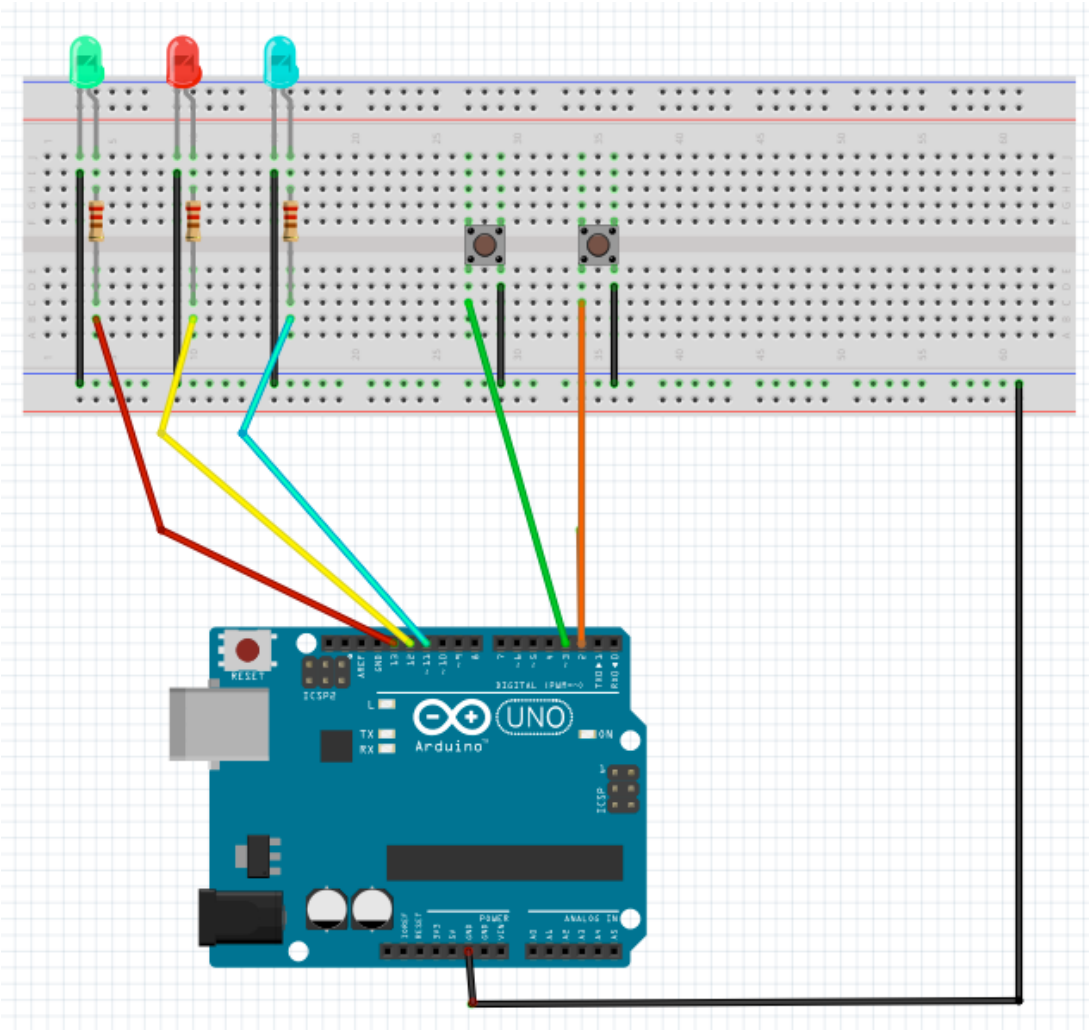
Lez. N. 07 Accendere in sequenza tre led con i button



Materiali

Arduino
3 Led + 2 button
3 resistenze da 220 Ω
Jumper vari

Tanta pazienza!



Introduzione

Con il seguente circuito ci si propone di far accendere sequenzialmente tre led alla pressione di un button. Con un button il verso dell'accensione è da dx a sx, con la pressione dell'altro button da sx a dx.
Quando si preme un button si hanno dei "rimbalzi di corrente che possono falsare il comportamento del circuito. Nei circuiti precedenti ci siamo protetti con l'uso di resistenze molto forti da 10k Ω. In questo caso, come si vede dalle linee di codice, abbiamo attivato le resistenze di PullUp (esistono anche quelle di PullDown) che permettono la stabilizzazione del circuito. Up verso valori alti come 5V o 1, il Down invero agisce per valori bassi come GD o 0.

Codice

```
/ / LED & BUTTON
// Definizione dei pin
int led1 = 11;
int led2 = 12;
int led3 = 13;

int buttonUp = 2;
int buttonDown = 3;

// Definizione delle variabili
int led = 1; // led da accendere

void setup () {
// Impostazione dei pin come uscite
pinMode (led1, OUTPUT);
pinMode (led2, OUTPUT);
pinMode (led3, OUTPUT);
// Impostazione dei pin come ingressi
pinMode (buttonUp, INPUT);
pinMode (buttonDown, INPUT);
// Abilitazione delle resistenze di PullUp interne sugli ingressi
digitalWrite (buttonUp, HIGH);
digitalWrite (buttonDown, HIGH);
}

void loop () {

if (digitalRead (buttonUp) == 0){ //se il pulsante è premuto
led++; // Incrementa la variabile led
if (led>3) { // Se la variabile supera il valore 3
led=1; // Riporta la variabile a 1
}
delay (500); // Pausa per antirimbalzo
}

if (digitalRead (buttonDown) == 0){ //se il pulsante è premuto
led--; // Decrementa la variabile led
if (led<1) { // Se la variabile è inferiore al valore 1
led=3; // Riporta la variabile a 3
}
delay (500); // Pausa per antirimbalzo
}
```

```
if (led==1) {
digitalWrite (led1, HIGH); // Accendi il led1
digitalWrite (led2, LOW); // Spegni il led2
digitalWrite (led3, LOW); // Spegni il led3
}

if (led==2) {
digitalWrite (led1, LOW); // Spegni il led1
digitalWrite (led2, HIGH); // Accendi il led2
digitalWrite (led3, LOW); // Spegni il led3
}

if (led==3) {
digitalWrite (led1, LOW); // Spegni il led1
digitalWrite (led2, LOW); // Spegni il led2
digitalWrite (led3, HIGH); // Accendi il led3
}
}
```

Pin o Perni digitali

I pin su Arduino possono essere configurati come ingressi o uscite. Il documento si riferisce ai pin digitali, è importante notare che la maggior parte dei pin analogici di Arduino (Atmega), può essere configurata e utilizzata, esattamente nello stesso modo dei pin digitali.
Proprietà dei pin configurati come INPUT
I pin Arduino (Atmega) sono predefiniti come input, quindi non è necessario dichiararli esplicitamente come input con pinMode () quando li si utilizza come input. I pin configurati in questo modo si dice che siano in uno stato di **alta impedenza** . I pin di ingresso fanno richieste estremamente piccole sul circuito che stanno campionando, equivalente a un resistore in serie di 100 megaohm davanti al pin. Ciò significa che richiede pochissima corrente per spostare il pin di input da uno stato all'altro e può rendere i pin utili per compiti come l'implementazione di un sensore tattile capacitivo , la lettura di un LED come fotodiodo o la lettura di un sensore analogico con uno schema come RCTime.
Ciò significa anche che i pin configurati come pinMode (pin, INPUT) senza niente collegato a loro, o con fili collegati ad essi che non sono collegati ad altri circuiti, segnaleranno cambiamenti apparentemente casuali nello stato dei pin, rilevando rumore elettrico dal ambiente, o accoppiando capacitivamente lo stato di un pin vicino.
Resistenze di pull-up con pin configurati come INPUT
Spesso è utile pilotare un pin di input su uno stato noto se non è presente alcun input. Questo può essere fatto aggiungendo un resistore pullup (a + 5V), o un resistore pulldown (resistore a massa) sull'ingresso. Un resistore da 10 K è un buon valore per un pull-up o un resistore pulldown.
Proprietà dei pin configurati come INPUT_PULLUP
Ci sono resistori pullup 20K **integrati** nel chip Atmega a **cui è possibile accedere dal software**. È possibile accedere a questi resistori di pull-up incorporati impostando pinMode () come INPUT_PULLUP. Ciò inverte efficacemente il comportamento della modalità INPUT, dove HIGH significa che il sensore è spento e LOW significa che il sensore è acceso.
Il valore di questo pullup dipende dal microcontrollore utilizzato. Sulla maggior parte delle schede basate su AVR, il valore è garantito tra 20kΩ e 50kΩ. Sull'Arduino Due, è compreso tra 50kΩ e 150kΩ. Per il valore esatto, consultare la scheda tecnica del microcontrollore sulla scheda.

Quando si collega un sensore a un pin configurato con INPUT_PULLUP, l'altra estremità deve essere collegata a terra. Nel caso di un semplice interruttore, questo fa sì che il pin legga HIGH quando l'interruttore è aperto, e LOW quando viene premuto l'interruttore.
I resistori pullup forniscono abbastanza corrente per illuminare debolmente un LED collegato a un pin che è stato configurato come ingresso. Se i LED in un progetto sembrano funzionare, ma in modo molto debole, è probabile che ciò accada.
I resistori di pull-up sono controllati dagli stessi registri (posizioni di memoria interna del chip) che controllano se un pin è ALTO o BASSO. Di conseguenza, un pin configurato per avere resistenze pullup attivate quando il pin è un INPUT, avrà il pin configurato come HIGH se il pin viene quindi commutato su OUTPUT con pinMode (). Questo funziona anche nell'altra direzione, e un pin di uscita che viene lasciato in uno stato HIGH avrà i resistori di pullup impostati se passati a un ingresso con pinMode ().
Prima di Arduino 1.0.1, era possibile configurare i pull-up interni nel seguente modo:

pinMode (pin, INPUT); // imposta il pin da inserire digitalWrite (pin, HIGH); // attiva i resistori pullup

NOTA: il pin digitale 13 è più difficile da utilizzare come ingresso digitale rispetto agli altri pin digitali perché ha un LED e un resistore ad esso collegato che è saldato alla scheda sulla maggior parte delle schede. Se abiliti il suo resistore di pull-up interno da 20k, esso si bloccherà intorno a 1,7 V invece dei 5V previsti perché il LED integrato e il resistore di serie abbassano il livello di tensione, il che significa che restituisce sempre LOW. Se è necessario utilizzare il pin 13 come ingresso digitale, impostarne il pinMode () su INPUT e utilizzare un resistore di abbassamento esterno.
Proprietà dei pin configurati come OUTPUT
I pin configurati come OUTPUT con pinMode () si dice che siano in uno stato a bassa impedenza. Ciò significa che possono fornire una notevole quantità di corrente ad altri circuiti. I pin Atmega possono generare (fornire corrente positiva) o assorbire (fornire corrente negativa) fino a 40 mA (milliampere) di corrente verso altri dispositivi / circuiti. Questa corrente è sufficiente per illuminare brillantemente un LED (non dimenticare il resistore di serie), o eseguire molti sensori, per esempio, ma non abbastanza corrente per far funzionare la maggior parte dei relè, solenoidi o motori.
Cortocircuiti sui pin Arduino, o il tentativo di eseguire dispositivi ad alta corrente da loro, possono danneggiare o distruggere i transistor di uscita nel pin, o danneggiare l'intero chip Atmega. Spesso questo si tradurrà in un pin "morto" nel microcontrollore, ma il chip rimanente funzionerà ancora adeguatamente. Per questo motivo è una buona idea collegare i pin OUTPUT ad altri dispositivi con resistori da 470Ω o 1k, a meno che non sia richiesto il massimo assorbimento di corrente dai pin per una particolare applicazione.