Gli approfondimenti di

L@BOROBOTICA

L'INTEGRATO MCP 4131-103

Un potenziometro digitale

Potrebbe esserci richiesto di pilotare via software una variazione continua di alcuni parametri, come ad esempio la tensione un alimentatore o il volume di un amplificatore audio. Parlando di auto-motive, una volta, per regolare il volume c'era una manopola, così pure per i toni o per la sintonia.

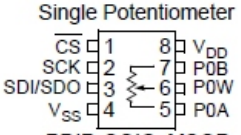
Negli ultimi anni però si è assistito ad una digitalizzazione molto spinta di tutti i dispositivi per automobili: ora con un solo controllo è possibile regolare tutto quello che prima richiedeva una selva di manopole. Come è possibile realizzare una simile diavoleria?

Esistono dei circuiti integrati che al loro interno nascondono dei sofisticati circuiti di regolazione e dei potenziometri analogici, comandati digitalmente. Indirizzando tramite opportuni codici questi potenziometri digitali è possibile con un solo encoder svolgere molteplici funzioni.

L'integrato MCP 4131-103 della Motorola è un componente a 8 pin che si potrà comandare tramite protocollo seriale SPI. Al suo interno è presente un potenziometro da 10K.

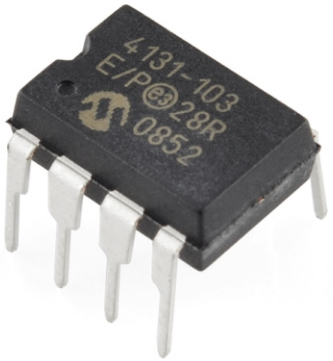
Package Types (top view)

MCP41X1
Single Potentiometer



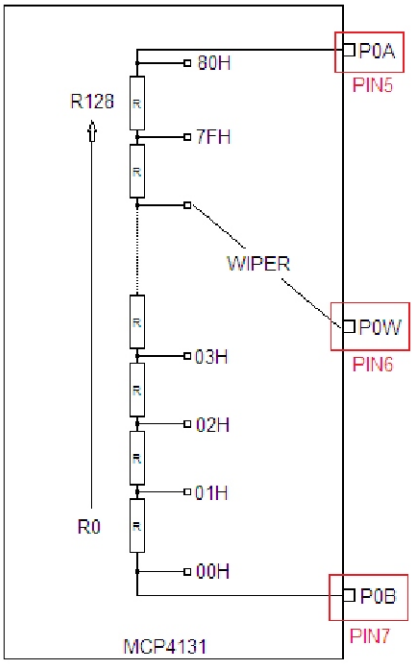
PDIP, SOIC, MSOP

Cs=chip select
Sck=serial data clock
Sdi=serial data input
Vss=Gnd
P0A=potenziometro(1)
P0W=Wiper
P0B= potenziometro(2)
Vdd=Alimentazione.



Il potenziometro è composto da una rete resistiva composta da 128 elementi il cui valore totale è, nel nostro modello, di 10Kohm. Tramite software possiamo spostare la posizione dello **Wiper**, che corrisponde al contatto strisciante di un potenziometro meccanico. Gli spostamenti dello **Wiper** vengono definiti **steps** e vanno dalla posizione 0 **(Wiper collegato al pin P0A)** alla posizione 128 **(Wiper collegato al pin P0B).**

Il potenziometro digitale **MCP4131** permette un massimo di **129 steps**.



POTENZIOMETRO

ENCODER

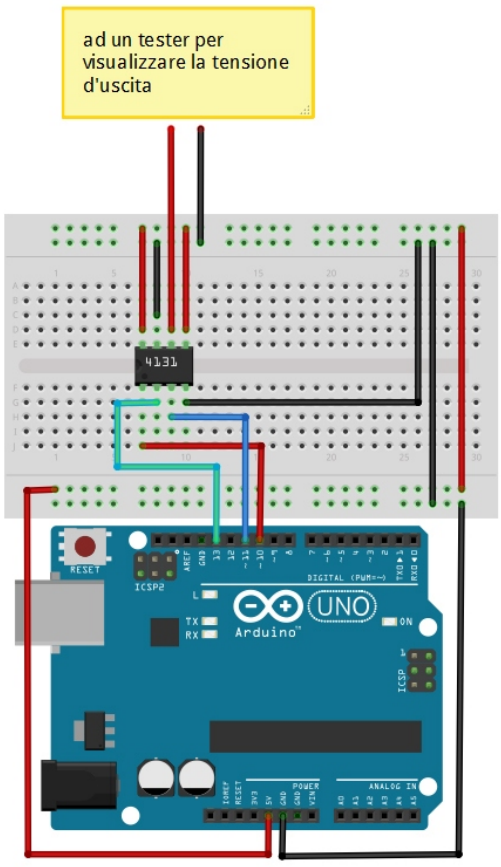
Vediamo ora qualche caratteristica del nostro potenziometro commentando il foglio-dati del costruttore

Device Features

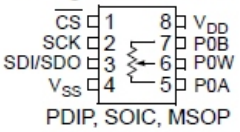
Device	# of POTs	Wiper Configuration	Control Interface	Memory Type	WiperLock Technology	POR Wiper Setting	Resistance (typical)		# of Steps	V_{DD} Operating Range (2)
							R_{AB} Options (kΩ)	Wiper - R_W (Ω)		
MCP4131 (3)	1	Potentiometer (1)	SPI	RAM	No	Mid-Scale	5.0, 10.0, 50.0, 100.0	75	129	1.8V to 5.5V
MCP4132 (3)	1	Rheostat	SPI	RAM	No	Mid-Scale	5.0, 10.0, 50.0, 100.0	75	129	1.8V to 5.5V

Da questa tabella deduciamo che il dispositivo ha un singolo potenziometro, che i valori della posizione del wiper sono **memorizzati in RAM** (piuttosto che sulla EEPROM). Il wiper ha resistenza 75 ohm, l'interfaccia di comunicazione è **SPI**, i numeri di **Steps sono 129** e il range con cui possiamo alimentare l'integrato va da 1.8Vcc a 5.5Vcc.

Ma cos'è allora questa SPI? La **SPI** (Serial Peripheral Interface) è un protocollo di comunicazione seriale usato per la comunicazione tra microcontrollore e una o più periferiche. Può anche essere usata per far comunicare tra di loro due microcontrollori. Utilizzando Arduino, esiste una libreria dedicata, (SPI.h) che ci farà risparmiare molto tempo. Con una connessione SPI c'è sempre un master (il microcontrollore) e degli slave (le periferiche). La libreria di Arduino dedica alla comunicazione SPI alcuni pin: **MOSI** (Master Out Slave In) **11**, **MISO** (Master In Slave Out) **12**, **SCK** Serial data ClocK **13**



Package Types (top view)



Vediamo ora il cablaggio

Come si può notare, il pin 1 dell'integrato (CS) è collegato al pin 10 di Arduino. Con uno stato logico su questo piedino abiliteremo/disabiliteremo la trasmissione dei dati verso l'IC.
Il pin 2 è collegato al pin 13 di Arduino che è dedicato dalla libreria SPI al Serial Data Clock (SCK).
Il pin 3 è collegato al pin 11 di Arduino che è dedicato dalla libreria SPI al Serial Data input/output (SDI/SDO) in questo caso MOSI.
Il pin 4 va a Gnd.
Il pin 7 va a Vcc.
I pin 5 e 7 sono collegati tra Vcc e Gnd .
Il pin 6 (centrale del potenziometro) viene portato all'uscita.
Lo sketch che scriveremo muoverà il potenziometro facendo variare la tensione sul pin 6 da 0 a 5Vcc.

```
//Usiamo un potenziometro digitale con SPI

#include <SPI.h> //carica la libreria SPI
int enable = 10;//pin di enable
int address=0; //indirizzo del wiper
int level;

void setup()
{
   pinMode (enable, OUTPUT);
   SPI.begin();   //inizializza SPI
   Serial.begin(9600);
}

void loop() {
     for (level = 0; level < 129; level++)
{
       digitalWrite(enable,LOW);    //abilita la trasmissione
       SPI.transfer(address);
       SPI.transfer(level); //scrivi level sul wiper
      digitalWrite(enable,HIGH);   // chiude la trasmissione
      Serial.println("aumento");
       Serial.println(level);
       delay(150);
     }

   delay(3000);


   for (level = 129; level > 0; level--)
{
                  digitalWrite(enable,LOW);
                  SPI.transfer(address);
                  SPI.transfer(level);
                  digitalWrite(enable,HIGH);
                  Serial.println("diminuisco");
                  Serial.println(level);
                  delay(150);
   }
   delay(3000);
 }
```

I pin 13 e 11 non vengono dichiarati perché già dedicati dalla libreria SPI.

L'indirizzo del wiper è utile quando si deve comandare più di un potenziometro.

Con un ciclo for si incrementa il valore "level".

Seleziona il wiper

Scrive il valore "level" sulla seriale