**Modulo H-Bridge L9110
*rif. www.adrirobot.it***



Questo modulo utilizza  2 chip H-bridge tipo[**L9110**](https://win.adrirobot.it/motor_driver/L9110_HG7881/motor_driver_L9110-HG7881.htm#L9110)(oppure tipo HG7881) indipendenti ognuno dei quali può pilotare un motore con una corrente di esercizio fino 800mA, con una corrente massima di picco di 1.5-2°.

La scheda può essere pilotata con livelli logici TTL compresi tra 2,5 V e 12 V questo permette l'utilizzo con microcontrollori alimentati a 3.3V oppure 5V.

La velocità dei motori può venire controllata tramite PWM, mentre la direzione è data da un livello logico.

Questo modulo può anche essere utilizzato per pilotare un motore passo-passo a due fasi.
La scheda dispone di 6 pin maschio e 2 terminali doppi. I motori si collegano ai 2 terminali. I pin di alimentazione VCC e GND sono posti al centro del connettore.

Il modulo, a differenza degli H-bridge come la [**L293**](https://win.adrirobot.it/arduino/shield_l293/arduino_shield_l293.htm) e [**L298**](https://win.adrirobot.it/motor_driver/l298/L298N_Dual_H-Bridge_Motor_Controller.htm)**,** non richiede una tensione separata per la parte logica e per l'alimentazione dei motori ma sarà sufficiente fornire uno tensione compresa tra i 2,5 e i 12V.

**Caratteristiche:**

* Alimentazione motore e modulo: da 2.5V a 12V
* Pilota 2 Motori in CC oppure un singolo motore passo passo
* Massimo assorbimento per canale: 800mA continui
* Dimensioni: 31mm x 22mm x 12mm
* Peso: 7 gr.

|  |
| --- |
| **L9110 -  Motor control driver** |
| https://win.adrirobot.it/datasheet/speciali/immagini/l9110s_pin.jpg | **https://win.adrirobot.it/images/pdficon.gif** | https://win.adrirobot.it/datasheet/speciali/immagini/l9110s_foto.gif |
| **Piedinatura** | [**Datasheet**](https://win.adrirobot.it/datasheet/speciali/pdf/L9110%20Motor%20control%20driver.pdf) | **Foto dell'integrato** |



**Schema di principio del collegamento dell'integrato**



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|

|  |
| --- |
| **Tabella della verità** |
| **Ingresso** | **Uscita** |   |
| **IA** | **IB** | **OA** | **OB** |  **Descrizione** |
| L |  L |  L |  L |  Off |
| H |  L |  H |  L |  Forward |
| L |  H |  L |  H |  Reverse |
| H |  H |  L |  L |  Off |

 |

**Schema elettrico del modulo**

Lo schema elettrico del modulo è molto semplice, il collegamento del'integrato [**L9110**](https://win.adrirobot.it/motor_driver/L9110_HG7881/motor_driver_L9110-HG7881.htm#L9110)è quello tipico riportato nel datasheet, sugli ingressi di controllo sono presenti delle resistenze da 10K per mantenere stabile il segnale di controllo, la tensione di alimentazione è filtrata dai due condensatori C1 e C2, è inoltre presente il diodo led D1 con la sua resistenza di limitazione R1 da 1K la cui accensione segnala la presenza della tensione di alimentazione. Sulla morsettiera d'uscita verso i motori sono presenti due ulteriori condensatori di filtro C3 e C4 da 0,1 uF.



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|

|  |  |
| --- | --- |
| **Pin** | **Descrizione** |
| **B-IA** |  Motore B Input A (IA) |
| **B-IB** |  Motore B Input B (IB) |
| **GND** |  Negativo alimentazione |
| **VCC** |  Positivo alimentazione 2.5-12V |
| **A-IA** |  Motore A Input A (IA) |
| **A-IB** |  Motore A Input B (IB) |

 |

**Gestione di un motore**



[Listato programma ](https://win.adrirobot.it/motor_driver/L9110_HG7881/Test_L9110_singolo.zip)

|  |  |
| --- | --- |
|

|  |
| --- |
| /\*This example shows how to drive a motor with using HG7881 (L9110)Connections:    Arduino digital output D10 to motor driver input B-IA.Arduino digital output D11 to motor driver input B-IB.Motor driver VCC to operating voltage 5V.Motor driver GND to common ground.Motor driver MOTOR B screw terminals to a small motor.\*/// wired connections#define HG7881\_B\_IA 10 // D10 --> Motor B Input A --> MOTOR B +#define HG7881\_B\_IB 11 // D11 --> Motor B Input B --> MOTOR B -// functional connections#define MOTOR\_B\_PWM HG7881\_B\_IA // Motor B PWM Speed#define MOTOR\_B\_DIR HG7881\_B\_IB // Motor B Direction// the actual values for "fast" and "slow" depend on the motor#define PWM\_SLOW 50  // arbitrary slow speed PWM duty cycle#define PWM\_FAST 200 // arbitrary fast speed PWM duty cycle#define DIR\_DELAY 1000 // brief delay for abrupt motor changesvoid setup(){  **Serial**.begin( 9600 );  pinMode( MOTOR\_B\_DIR, OUTPUT );  pinMode( MOTOR\_B\_PWM, OUTPUT );  digitalWrite( MOTOR\_B\_DIR, LOW );  digitalWrite( MOTOR\_B\_PWM, LOW );}void loop(){  boolean isValidInput;  // draw a menu on the serial port  **Serial**.println( "-----------------------------" );  **Serial**.println( "MENU:" );  **Serial**.println( "1) Avanti veloce" );  **Serial**.println( "2) Avanti" );  **Serial**.println( "3) Soft Stop (Ruote libere)" );  **Serial**.println( "4) Indietro" );  **Serial**.println( "5) Indietro veloce" );  **Serial**.println( "6) Hard stop (Ruote bloccate)" );  **Serial**.println( "-----------------------------" );  do  {    byte c;    // get the next character from the serial port    **Serial**.print( "?" );    while ( !**Serial**.available() )      ; // LOOP...    c = **Serial**.read();    // execute the menu option based on the character recieved    switch ( c )    {      case '1': // 1) Fast forward        **Serial**.println( "Avanti veloce..." );        // always stop motors briefly before abrupt changes        digitalWrite( MOTOR\_B\_DIR, LOW );        digitalWrite( MOTOR\_B\_PWM, LOW );        delay( DIR\_DELAY );        // set the motor speed and direction        digitalWrite( MOTOR\_B\_DIR, HIGH ); // direction = forward        analogWrite( MOTOR\_B\_PWM, 255 - PWM\_FAST ); // PWM speed = fast        isValidInput = true;        break;      case '2': // 2) Forward        **Serial**.println( "Avanti..." );        // always stop motors briefly before abrupt changes        digitalWrite( MOTOR\_B\_DIR, LOW );        digitalWrite( MOTOR\_B\_PWM, LOW );        delay( DIR\_DELAY );        // set the motor speed and direction        digitalWrite( MOTOR\_B\_DIR, HIGH ); // direction = forward        analogWrite( MOTOR\_B\_PWM, 255 - PWM\_SLOW ); // PWM speed = slow        isValidInput = true;        break;      case '3': // 3) Soft stop (preferred)        **Serial**.println( "Soft Stop (Ruote libere)..." );        digitalWrite( MOTOR\_B\_DIR, LOW );        digitalWrite( MOTOR\_B\_PWM, LOW );        isValidInput = true;        break;      case '4': // 4) Reverse        **Serial**.println( "Indietro..." );        // always stop motors briefly before abrupt changes        digitalWrite( MOTOR\_B\_DIR, LOW );        digitalWrite( MOTOR\_B\_PWM, LOW );        delay( DIR\_DELAY );        // set the motor speed and direction        digitalWrite( MOTOR\_B\_DIR, LOW ); // direction = reverse        analogWrite( MOTOR\_B\_PWM, PWM\_SLOW ); // PWM speed = slow        isValidInput = true;        break;      case '5': // 5) Fast reverse        **Serial**.println( "Indietro veloce..." );        // always stop motors briefly before abrupt changes        digitalWrite( MOTOR\_B\_DIR, LOW );        digitalWrite( MOTOR\_B\_PWM, LOW );        delay( DIR\_DELAY );        // set the motor speed and direction        digitalWrite( MOTOR\_B\_DIR, LOW ); // direction = reverse        analogWrite( MOTOR\_B\_PWM, PWM\_FAST ); // PWM speed = fast        isValidInput = true;        break;      case '6': // 6) Hard stop (use with caution)        **Serial**.println( "Hard stop (Ruote bloccate)..." );        digitalWrite( MOTOR\_B\_DIR, HIGH );        digitalWrite( MOTOR\_B\_PWM, HIGH );        isValidInput = true;        break;      default:        // wrong character! display the menu again!        isValidInput = false;        break;    }  } while ( isValidInput == true );  // repeat the main loop and redraw the menu...} |

 |

**Gestione di due motori**



[Listato programma ](https://win.adrirobot.it/motor_driver/L9110_HG7881/Test_L9110_doppio.zip)

|  |  |
| --- | --- |
|

|  |
| --- |
| /\* L9110 motor driver controlling 2 small DC motors\*/const int AIA = 5;  // (pwm) pin 5 connected to pin A-IAconst int AIB = 6;  // (pwm) pin 6 connected to pin A-IBconst int BIA = 10; // (pwm) pin 10 connected to pin B-IAconst int BIB = 11;  // (pwm) pin 11 connected to pin B-IBbyte speed = 150;  // change this (0-255) to control the speed of the motorsvoid setup() {  pinMode(AIA, OUTPUT); // set pins to output  pinMode(AIB, OUTPUT);  pinMode(BIA, OUTPUT);  pinMode(BIB, OUTPUT);  **Serial**.begin( 9600 );}void loop() {  **Serial**.println( "Avanti..." );  forward();  delay(2000);  STOP();  **Serial**.println( "Indietro..." );  backward();  delay(2000);  STOP();  **Serial**.println( "Sinistra..." );  left();  delay(2000);  STOP();  **Serial**.println( "Destra..." );  right();  delay(2000);  STOP();}void backward(){  analogWrite(AIA, 0);  analogWrite(AIB, speed);  analogWrite(BIA, 0);  analogWrite(BIB, speed);}void forward(){  analogWrite(AIA, speed);  analogWrite(AIB, 0);  analogWrite(BIA, speed);  analogWrite(BIB, 0);}void left(){  analogWrite(AIA, speed);  analogWrite(AIB, 0);  analogWrite(BIA, 0);  analogWrite(BIB, speed);}void right(){  analogWrite(AIA, 0);  analogWrite(AIB, speed);  analogWrite(BIA, speed);  analogWrite(BIB, 0);}void STOP(){  analogWrite(AIA, 0);  analogWrite(AIB, 0);  analogWrite(BIA, 0);  analogWrite(BIB, 0);  delay (2000);} |

 |

***Fonte: www.adrirobot.it***