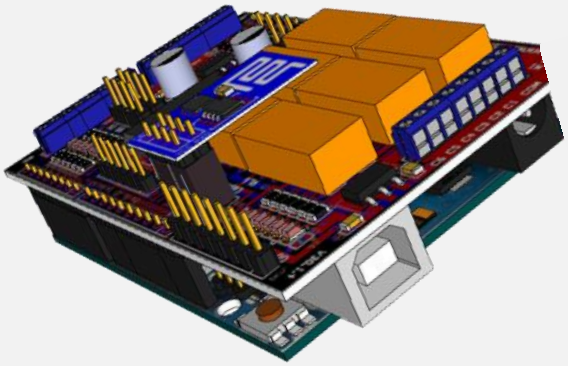
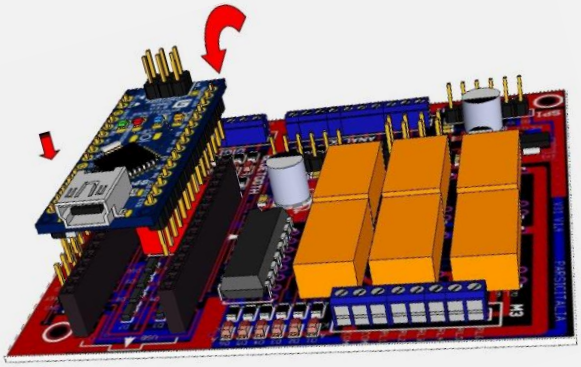
# Home Automation con Arduino Nano





## Introduzione

Da parte del grande pubblico, è sempre più crescente la richiesta di applicazioni mobili: le cosiddette APP. Da alcuni anni, la comparsa del sistema di sviluppo Arduino, ha avvicinato al mondo della microelettronica una grande massa di appassionati, continuamente alla ricerca di novità o progetti che permettano di sviluppare applicazioni in modo semplice e pratico. Ottenere un circuito elettronico che funzioni realmente e che dia all'utilizzatore la sicurezza di avere qualcosa di veramente funzionante, è cosa complicata ed alcune volte difficile, dato che è necessario mettere insieme un bagaglio di conoscenze ed esperienze non sempre alla portata di tutti. In queste pagine descriviamo un sistema automatico domotico, da impiegare con Arduino, per controllare a distanza il proprio impianto di riscaldamento, l'impianto di irrigazione delle piante, il sistema antifurto, il cancello, ed altro ancora. Il progetto è stato sviluppato interamente in ambiente Arduino e può essere impiegato, oltre all'home automation, nel wireless industriale e nei dispositivi di sicurezza.

Lo scopo di questa trattazione sarà quello di guidare il lettore ad un immediato utilizzo delle interfacce V30 e V31, prodotte da Parsic Italia, mediante alcuni esempi pratici e attraverso gli strumenti software messi a disposizione dalla piattaforma Arduino. Le interfacce, sono indicate in quelle situazioni dove c'è la necessità di avere un controllo portatile wireless, con la possibilità di poter accedere da un qualsiasi punto e in qualunque momento all'impianto della propria abitazione o attività commerciale.

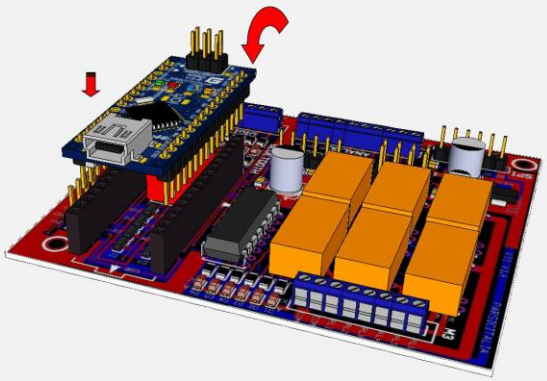
### Arduino Total Control.

Grazie all'APP ATC, il processo di controllo tra la scheda V31 e lo Smartphone, diventa incredibilmente intuitivo, alla portata di chiunque si cimenti con la piattaforma di programmazione Arduino.
Attraverso l'interfaccia grafica di ATC, l'applicazione può essere adattata ai più frequenti impieghi in ambito della domotica, come il controllo di temperatura ambientale, accensione e spegnimento luci, teleallarme, ecc. In ATC sono disponibili funzionalità grafiche di base come pulsanti, caselle di testo, animazioni e funzionalità più complesse come la comunicazione Bluetooth e Wi-Fi.
In avanti, andremo a chiarire, nell'apposito capitolo riservato all'interfaccia grafica, il funzionamento dell'App ATC che è una applicazione dedicata, progettata per la scheda **V31**. Il pacchetto software si divide in due parti: il codice sorgente (sketch), che deve essere trasferito all'interno dell'**AT328,** e l'App che deve essere installata sul vostro Android attraverso **Google Play**.

## Come ottenere risultati immediati

L'hardware di Arduino è realizzato da una serie di componenti elettronici di piccole dimensioni, che insieme a un microcontrollore ATmega, è utile alla creazione di prototipi per scopi didattici e professionali. L'ambiente di sviluppo integrato *(IDE) di Arduino*, idealmente concepito per iniziare alla programmazione i neofiti, è oggi utilizzato anche in campo professionale e permette la stesura di codice sorgente. Attraverso *l'IDE di Arduino*, l'utente è in grado di compilare e lanciare il programma eseguibile con un solo click. *L'IDE*, permette l'adozione di particolari tecniche di programmazione, basate sul linguaggio Wiring, che facilita il programmatore offrendo un modo semplice per accedere alle periferiche di input/output della piattaforma hardware, grazie anche alle librerie software messe a disposizione. Tutto il software Arduino è libero, e gli schemi circuitali sono distribuiti come hardware liberi (Arduino open source).

La scheda *V31*, è realizzata con l'ausilio dell'*Arduino Nano* e può essere collocata su supporti per barra DIN, impiegando un cover plastico da 72mm. Consente di realizzare, in maniera relativamente rapida e semplice, dispositivi di controllo per sensori di temperatura e umidità, azionamento motori, teleallarme, controllo luci, telecontrolli che fanno uso di linee di comunicazione Bluetooth e Wi-Fi, ecc.



**V31**

La scheda **V31**, richiede l'inserimento dell'**Arduino Nano** sull'apposito zoccolo a 32 pin, rispettando la posizione della chiave d'inserzione che all'atto della saldatura sul pcb, sarà collocata nella parte posteriore dello zoccolo. P*orre attenzione* al posizionamento del Nano i cui pin maschi dovranno occupare i relativi connettori femmina, *come si vede in figura, **posizionando la presa USB** della scheda Arduino verso l'esterno del supporto.* La programmazione di Arduino avviene collegando la presa USB al PC ed avviando le procedure di programmazione attraverso gli strumenti dell'IDE di sviluppo. Quando la programmazione è completata, si può scollegare la scheda dalla presa USB del PC. Le schede, impiegate in abbinamento alle **App ANDROID** per **Smartphone e Tablet**, sono indicate anche per la didattica scolastica, nei corsi di apprendimento di **Atmel Studio oppure MIT App Inventor**, e sostituiscono validamente buona parte dei cosiddetti **starter-kit**.

## Caratteristiche tecniche della scheda V31

**Applicazioni:**

- Sistemi di Automazione, Domotica, Robotica
- Antifurti
- Automotive
- Progettazione
- Didattica scolastica

**Alimentazioni**

- Ingresso 9-12V 500 mA ingresso protetto
- Alimentatore ausiliario 3V 500 mA on-board
- Uscite 3V - 5V 200mA per sensori esterni

**Ingressi/Uscite**

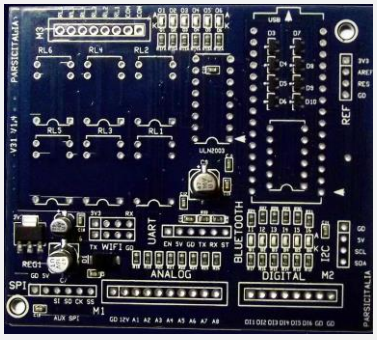
- 6 linee digitali, bidirezionali 20mA, con segnalazioni led
- 6 uscite digitali su relè 1 Amp. con segnalazioni led
- 8 ingressi analogici protetti, risoluzione 10 bit
- 1 dip-switch/Jumper-switch per selezione segnali SPI/I2C
- 1 port SPI/I2C
- 1 port UART

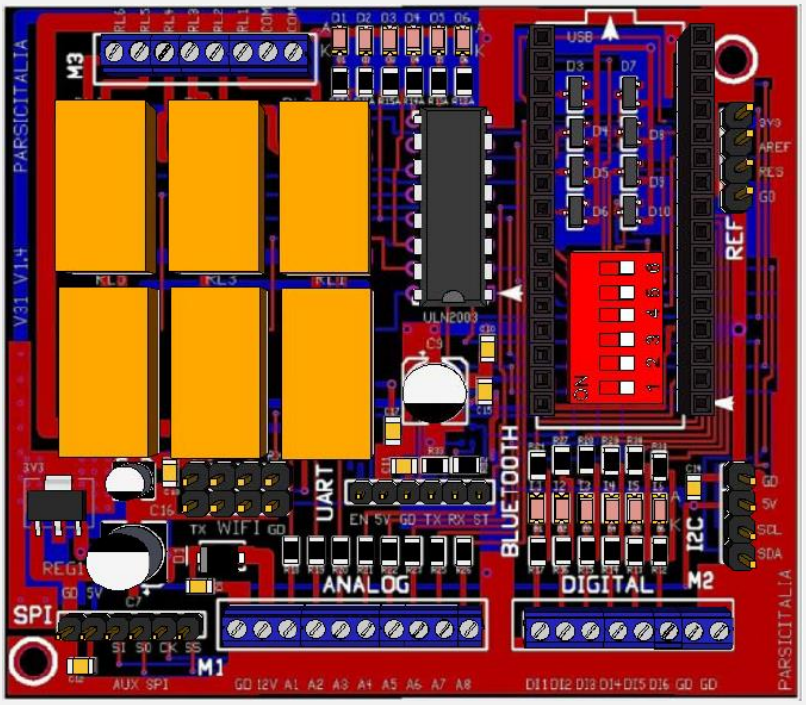
**Segnalazioni Led scheda V31**

| LED | Descrizione | Note |
|---|---|---|
| LD1 Verde | Power input +9 (range 9-12Vcc) | Arduino Nano PWR |
| LD2 Rosso | TX | Arduino Nano UART |
| LD3 Rosso | RX | Arduino Nano UART |
| LDI1 Rosso | Digital Input 1 | Bidirezionale |
| LDI2 Rosso | Digital Input 2 | Bidirezionale |
| LDI3 Rosso | Digital Input 3 | Bidirezionale |
| LDI4 Rosso | Digital Input 4 | Bidirezionale |
| LDI5 Rosso | Digital Input 5 | Bidirezionale |
| LDI6 Rosso | Digital Input 6 | Bidirezionale |
| LDO1 Verde | Digital Output 1 | Relè 1 |
| LDO2 Verde | Digital Output 2 | Relè 2 |
| LDO3 Verde | Digital Output 3 | Relè 3 |
| LDO4 Verde | Digital Output 4 | Relè 4 |
| LDO5 Verde | Digital Output 5 | Relè 5 |
| LDO6 Verde | Digital Output 6 | Relè 6 |

**Piano di montaggio.**

La scheda è fornita con i componenti **SMD** preassemblati, come si vede in figura. L'utilizzatore dovrà saldare al PCB, alcuni componenti discreti e relativi accessori che si trovano nella confezione. L'operazione, molto semplice da eseguire, richiede un minimo di attrezzatura e mezzora di applicazione manuale. Le istruzioni sono contenute nel presente manuale, seguendo le illustrazioni e disegni a colori. Si raccomanda di impiegare *ottimo stagno da laboratorio* ed un saldatore da 25W con punta fine. Procedere con l'inserimento dei componenti nelle apposite piazzuole, rispettando le polarità dove previsto. Inserire prima i componenti a basso profilo e poi, quelli a profilo più alto. A fine lavoro lavare il circuito, per rimuovere le incrostazioni, impiegando detergente liquido alcalino (pH11), oppure solvente chimico non tossico. Asciugare con un getto d'aria calda. Il risultato sarà un circuito con i componenti perfettamente allineati e saldature lucide, esenti da opacità.

**V31**  *prima*  *dopo*



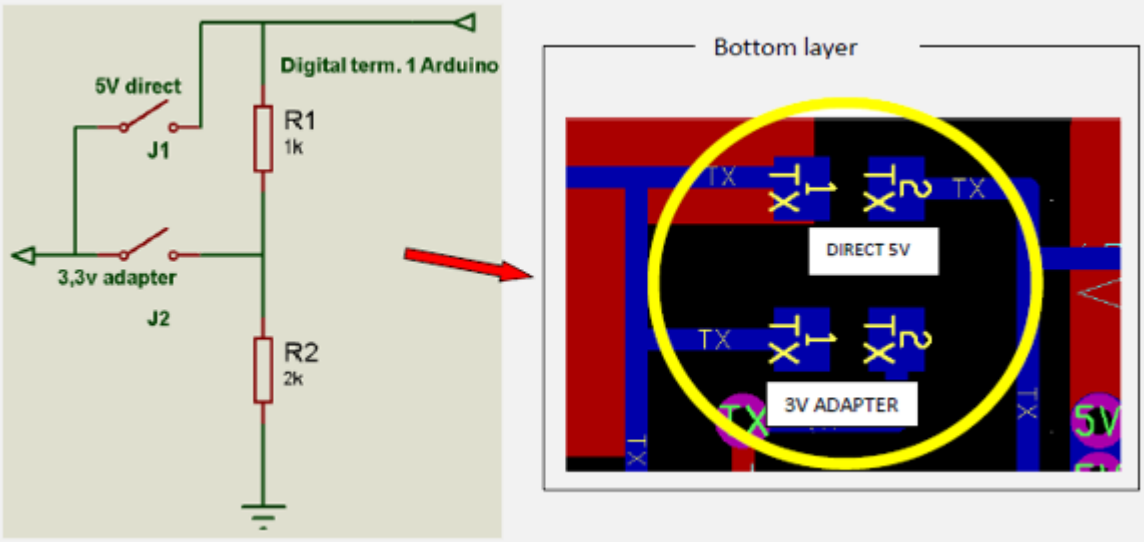

**Posizionamento dello shield V30 sull'Arduino Uno**

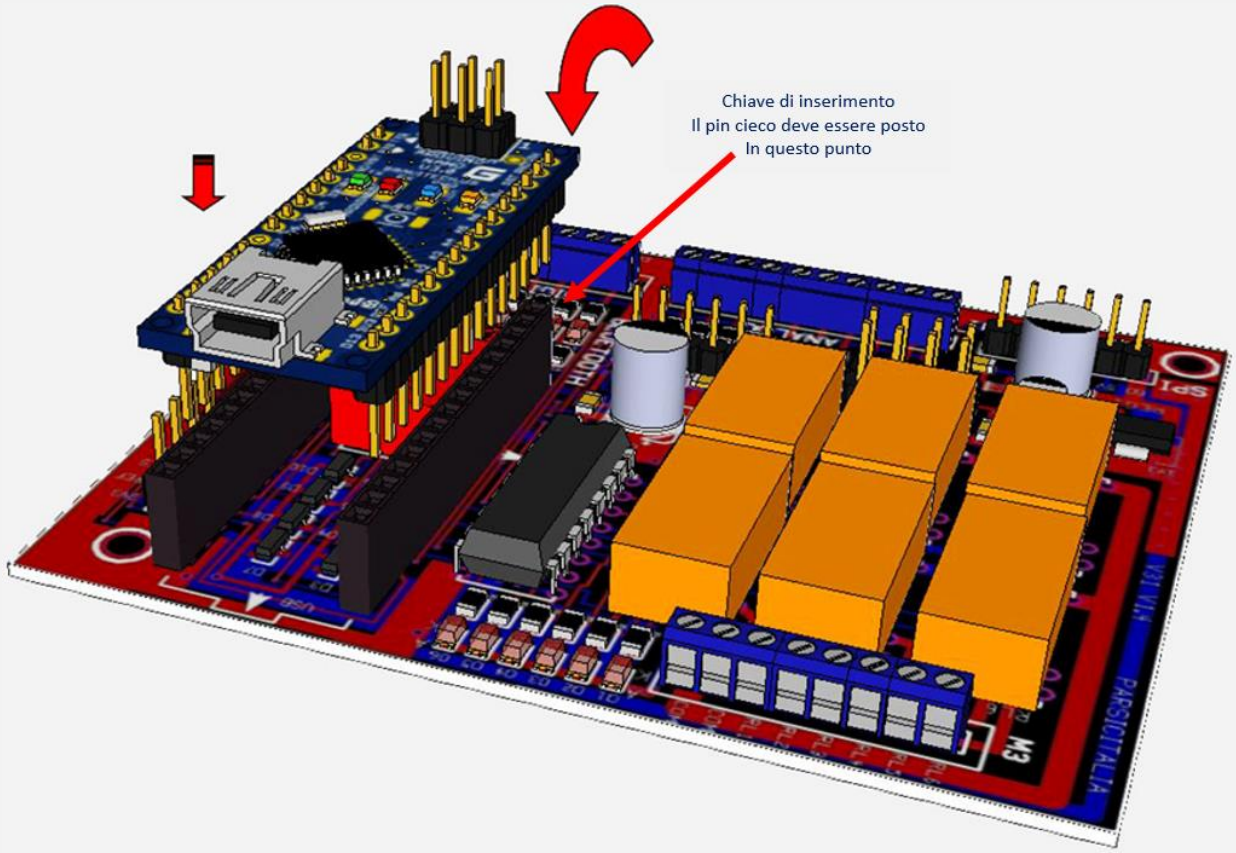
Prima di procedere alla descrizione di alcune applicazioni pratiche, descriveremo l'hardware della scheda, specificandone le caratteristiche tecniche ed il suo funzionamento.

## Settaggi

⚠

Nel layout inferiore del PCB, sono predisposti alcuni **jumper-pad**, individuabili nel riquadro **TXD Direct e TXD Adapter 3V3**. Opportunamente selezionati, chiudendoli con una goccia di stagno, permettono di inserire un partitore di tensione necessario ad adattare il segnale proveniente dal pin **TXD (PD1)** di Arduino, funzionante a **5V,** verso i terminali dei circuiti ausiliari funzionanti a **3,3V** (Wi-Fi – Bluetooth, ecc.). Ovviamente, per evitare conflitti hardware, selezionare un **solo jumper per volta**.



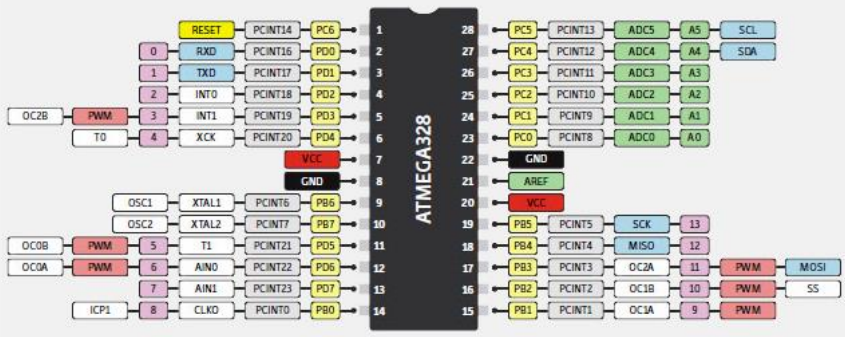

**Posizionamento dell'Arduino Nano sulla V31**

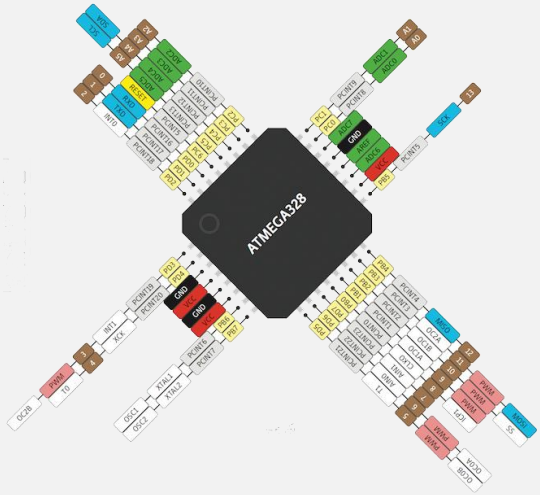
## Le linee digitali ed analogiche

### Ingressi – Uscite Atmega328

In generale, ciascuno dei 14 pin digitali di Arduino può essere impiegato come ingresso o uscita digitale. Questi pin sono di tipo **bidirezionale**, funzionano con un livello di **5V** e possono erogare in uscita massimo **30mA**. Sono dotati di resistenza di pull-up interna, disconnessa di default (attivabile via software), del valore di 20-50KOhm. Alcuni di questi terminali svolgono funzioni specializzate. La scheda *V31* è stata progettata per svolgere funzioni standard di I/O. I port *"B e D" di Arduino*, si collegano attraverso i circuiti ad essi associati, adottando tecniche hardware ampiamente collaudate. La *V31*, dispone di 8 ingressi analogici (A0…A7) ognuno dei quali ha una risoluzione a 10bit. La risoluzione dell'ADC è di 10bit, che si traduce nella restituzione di un numero intero compreso tra 0 e 1023 stati diversi.
Il convertitore analogico-digitale (ADC) interno agli Atmega, è settato di default per acquisire valori tra 0 e 5V. La scheda permette di settare il valore AREF con il quale, con una apposita funzione, si può fissare il valore di riferimento dell'ADC. Per dare connettività, le schede sono equipaggiate di terminazioni specializzate per i collegamenti USART, Wi-Fi, Bluetooth. Gli ambiti di applicazione delle linee seriali sono praticamente infiniti, utilizzabili per ogni tipo di interfaccia uomo-macchina.

### Pinout dell'ATMEGA328

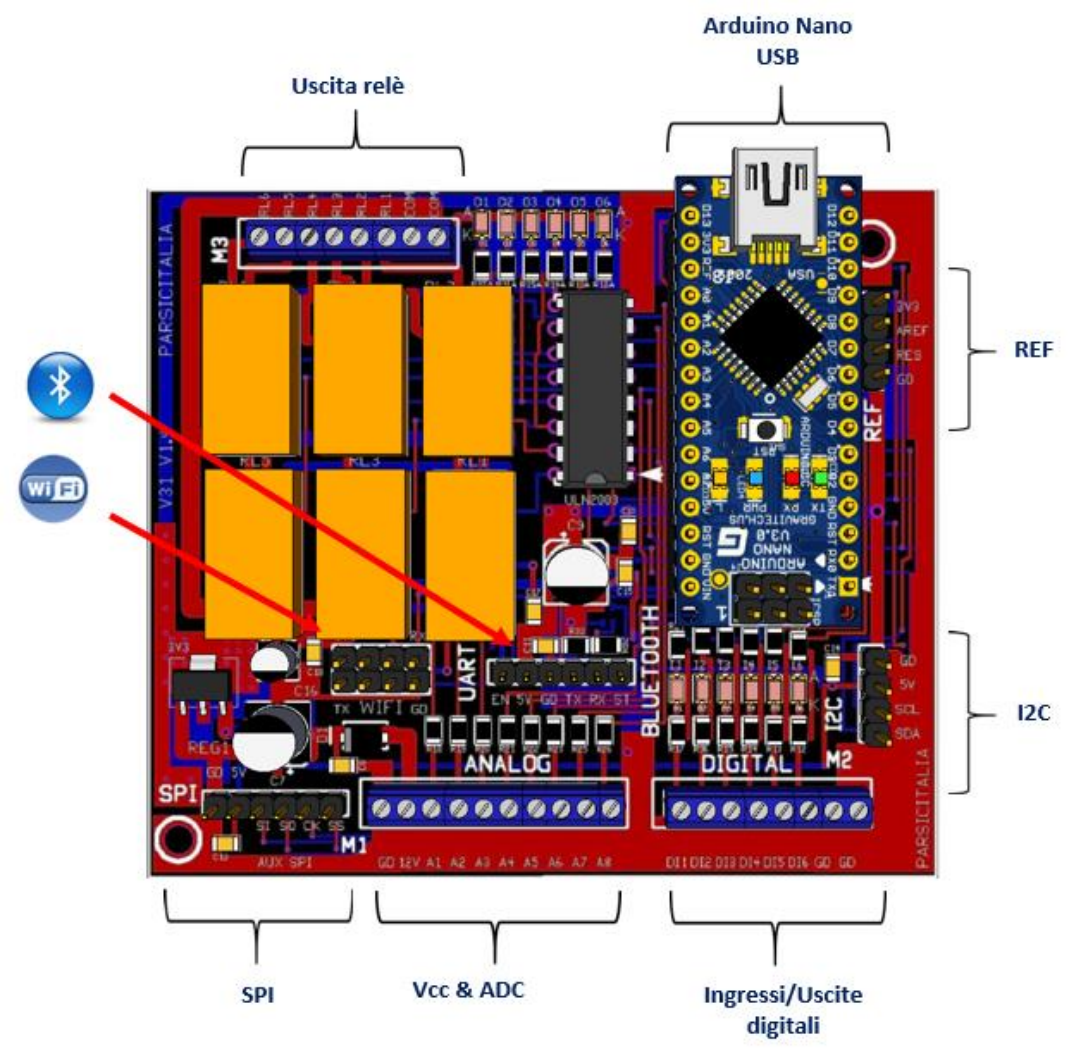


## Linee digitali I/O della scheda V31

La V31 può essere alloggiata in un contenitore per barra DIN e collocata all'interno di un quadro elettrico.
L'*app Parsic Italia App* permette di gestire fino a quattro schede diversamente posizionate in ambiente, in modo da costituire una sorta di sistema di controllo a elaborazione distribuita. Per evitare conflitti di comunicazione, ogni scheda sarà dotata di indirizzo IP diverso, modificabile via software.
Questa soluzione permette di ottenere un funzionamento più flessibile del sistema, aumentandone la sua sicurezza.

## Pinout della scheda V31



**Linee digitali I/O**

Le linee digitali **PortD** sono di tipo bidirezionale. Sono impiegate sia come terminali d'ingresso o come terminali d'uscita: la selezione degli I/O avviene via software. Sono protette da una rete circuitale resistiva che riduce la corrente in uscita a 15mA, circa. Fanno capo ai morsetti **M2**. Ai morsetti **2-4-5** sono disponibili anche i segnali **PWM.**

| Morsetto | Descrizione | Note |
|---|---|---|
| M2-1 | Digital Input 1 PD2 | Bidirezionale D2 |
| M2-2 | Digital Input 2 PD3 - PWM | Bidirezionale D3 - PWM |
| M2-3 | Digital Input 3 PD4 | Bidirezionale D4 |
| M2-4 | Digital Input 4 PD5 - PWM | Bidirezionale D5 - PWM |
| M2-5 | Digital Input 5 PD6 - PWM | Bidirezionale D6 - PWM |
| M2-6 | Digital Input 6 PD7 | Bidirezionale D7 |
| M2-7 | GND | Massa generale |
| M2-8 | GND | Massa generale |

**Linee uscita digitale PortB**

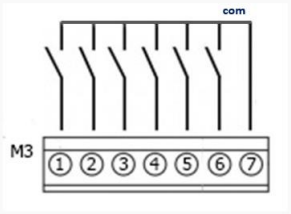
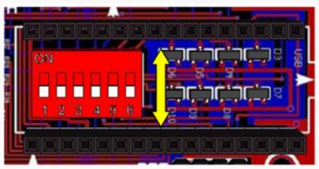
Le uscite digitali del PortB dell'Atmega328, fanno capo ai relè **RL1 – RL6**, la cui portata è di 1 Ampere in AC1. Per carici superiori è opportuno il collegamento esterno a relè opportunamente dimensionati. Sono condivise con i Port di comunicazione SPI/PWM. Le terminazioni dei contatti relè sono disponibili al **morsetto M3**

| Morsetto | Descrizione. Le linee PB1~PB5 sono di tipo condiviso | Note |
|---|---|---|
| M3-1 | COM | Contatto comune |
| M3-2 | COM | Contatto comune |
| M3-3 | Relè 1 PB0 | Contatto NA 1A |
| M3-4 | Relè 2 PB1 – PWM | Contatto NA 1A |
| M3-5 | Relè 3 PB2 – SS/PWM | Contatto NA 1A |
| M3-6 | Relè 4 PB3 – MOSI/PWM | Contatto NA 1A |
| M3-7 | Relè 5 PB4 - MISO | Contatto NA 1A |
| M3-8 | Relè 6 PB5 - SCK | Contatto NA 1A |

**Condivisione delle uscite digitali. Linea di comunicazione SPI**

Per usufruire della periferica di comunicazione **SPI,** le linee digitali d'uscita (relè) sono selezionabili a mezzo **Dip-Switch**, posizionato all'interno del connettore a 30 poli Arduino. Portare a OFF gli interruttori **1 SCK, 2 MOSI, 3 MISO, 4 SS** *se dovrete impiegare un dispositivo SPI esterno***.** Questi switch, aperti, escludono l'azionamento dei relè **3/4/5/6**. La tabella seguente indica i terminali digitali interessati alla comunicazione **seriale SPI**.

| Terminale AUX-SPI | Descrizione | Note |
|---|---|---|
| J1 | GND | GND alimentazione |
| J2 | +5V | Out tensione di alimentazione 5V |
| J3 | SPI MOSI (SI) | SPI MOSI = SW2 |
| J4 | SPI MISO (SO) | SPI MISO = SW3 |
| J5 | SPI SCK (CK) | SPI SCK = SW1 |
| J6 | SPI SS (SS) | SPI SS = SW4 |

| Out Digitali Relè | Sezionamento relè condivisione SPI |
|---|---|
|   |  |

### Ingressi Analogici

Si collegano all'ingresso M1 **8 segnali analogici**, contenuti nel range tra **0 e 10V**. Allo stesso morsetto si collega l'alimentazione esterna compresa tra 9 e 12Vcc. Se è impiegato il bus di comunicazione **I2C**, gli ingressi analogici saranno ridotti a 6: gli ingressi 4 e 5, infatti, non potranno essere utilizzati.

| Morsetto | Descrizione | Note |
|---|---|---|
| M1-1 | GND | Massa generale |
| M1-2 | Ingresso alimentazione generale 9-12Vcc | Alimentazione consigliata 9Vcc |
| M1-3 | ADC0 | |
| M1-4 | ADC1 | |
| M1-5 | ADC2 | |
| M1-6 | ADC3 | |
| M1-7 | ADC4 - SDA | Condiviso I2C SDA SW6 |
| M1-8 | ADC5 - SCL | Condiviso I2C SCL SW5 |
| M1-9 | ADC6 | |
| M1-10 | ADC7 | |

### Condivisione delle linee analogiche. I2C

Selezionare gli switch ***5 SCL*** – ***6 SDA*** se impiegate un dispositivo I2C esterno. Questi switch, escludono gli ingressi analogici **ADC4 – ADC5** come specificato nella tabella precedente.

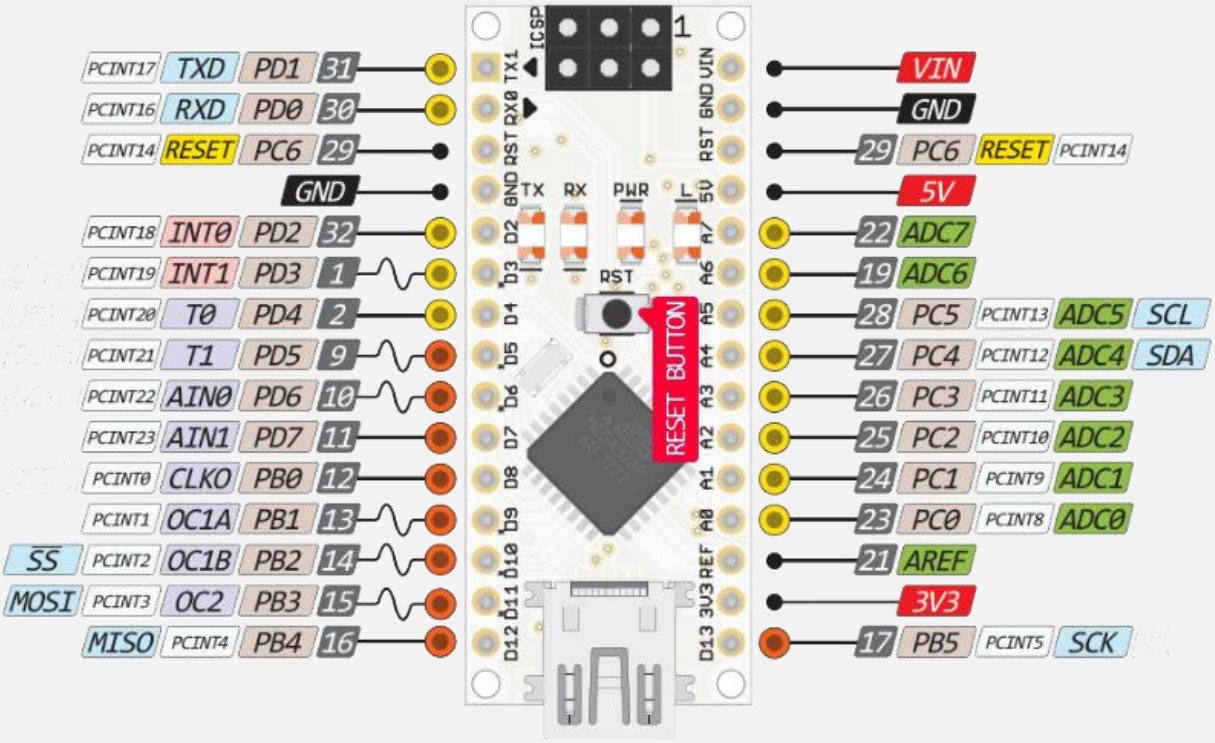
| Terminale I2C | Descrizione | Note |
|---|---|---|
| J1 | GND | Massa generale |
| J2 | Alimentazione ausiliaria 5V | Out alimentazione 5V 100mA |
| J3 | ADC5 - SCL | SCL |
| J4 | ADC4 - SDA | SDA |

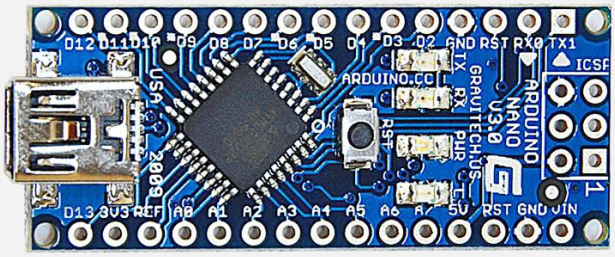
### Connettore ausiliario AREF

Al connettore **REF** sono riportati alcuni collegamenti, utili per la gestione degli ingressi analogici e altre utilità, come descritto nella tabella seguente:

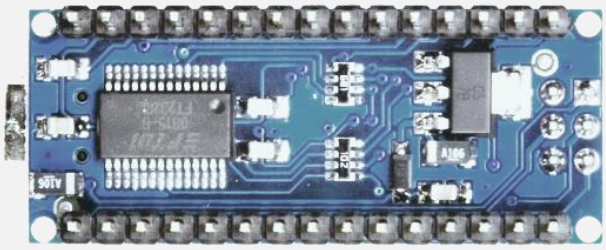
| Terminale REFI2C | Descrizione | Note |
|---|---|---|
| J1 | 3V3 Tensione ausiliaria 3,3V | Out alimentazione 3,3V 250mA |
| J2 | AREF | Tensione di riferimento analogico |
| J3 | Reset | Pulsante di reset esterno |
| J4 | GND | Massa generale |

## Pinout dell'Arduino Nano



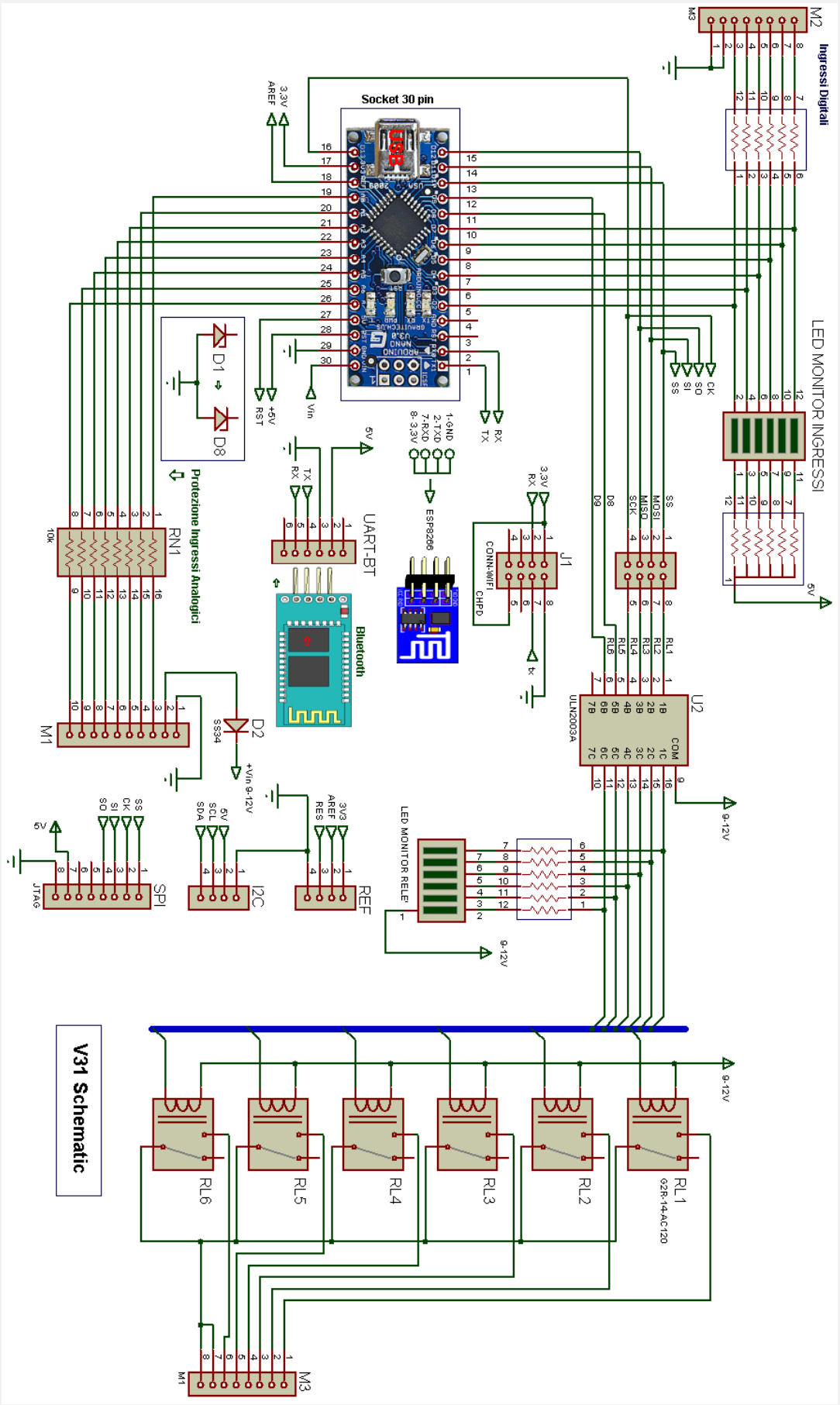



Upper layer



Bottom layer

## Schema elettrico V31 Arduino Nano



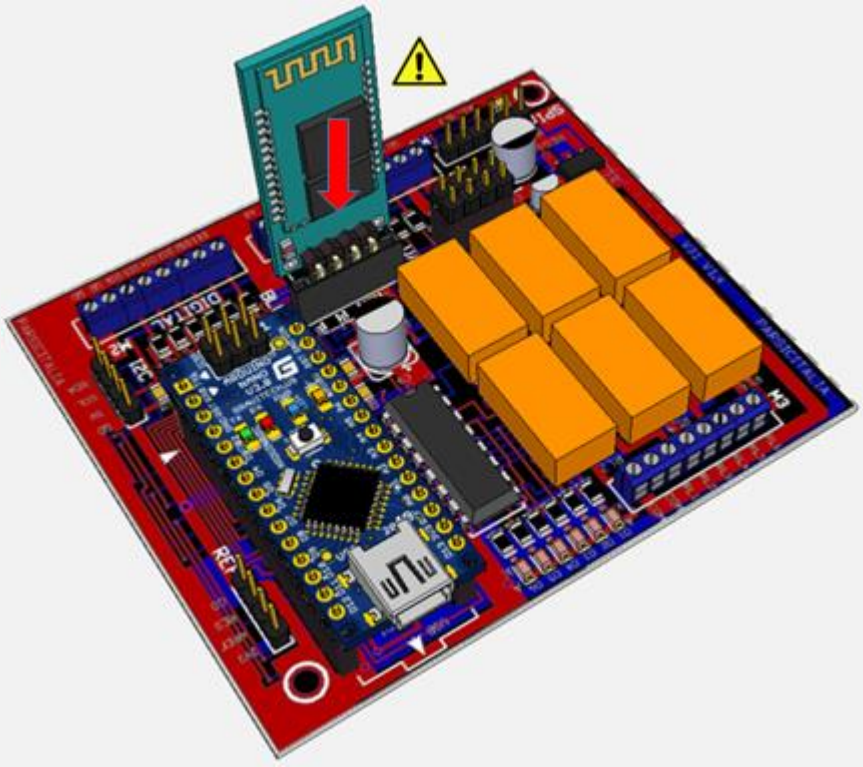
## Le comunicazioni seriali USART

Per tutte le applicazioni in cui serva instaurare una connessione di tipo **Bluetooth o Wi-F**i, la sched*a V31* è dotata di connettori specializzati sui quali possono essere innestati sia la scheda Bluetooth, tipo **HC05(06)** che quella Wi-Fi, tipo **ESP8266**. Si tratta di moduli di comunicazione molto economici, e sono prodotti in vari formati. I più comuni sono appunto quelli di seguito menzionati.

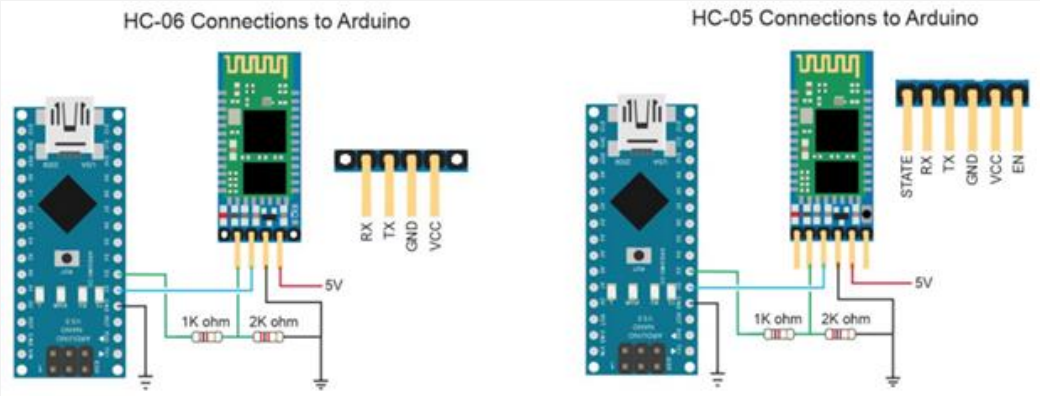
## Bluetooth

**V31 inserimento della scheda Bluetooth**

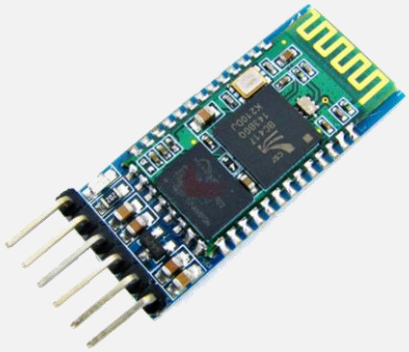
Il modulo Bluetooth è un modulo che permette di trasformare una porta USART, comunemente conosciuta come porta seriale, in una porta con profilo SPP, Serial Port Profile. Durante la fase di programmazione dell'Arduino, la scheda Bluetooth deve essere *temporaneamente scollegata*, per evitare conflitti hardware. Si noti che il modulo Bluetooth appena collegato alla scheda, lampeggia velocemente, indicando così che è pronto a ricevere una comunicazione esterna di attivazione. Il collegamento con il modulo Arduino è già predisposto sulla scheda in modo che il terminale TX del modulo HC si colleghi al terminale RX di Arduino e il terminale RX al terminale TX. L'app Parsic Italia, riconosce autonomamente la presenza della comunicazione Bluetooth proveniente dalla scheda e la procedura di attivazione per lo scambio dei dati è semplificata dalla facilità d'uso dei suoi controlli.



⚠ La scheda Bluetooth fornita, nei Kit è di tipo **HC05-06** compatibile con i livelli a 5V. Non necessita di partitore resistivo esterno



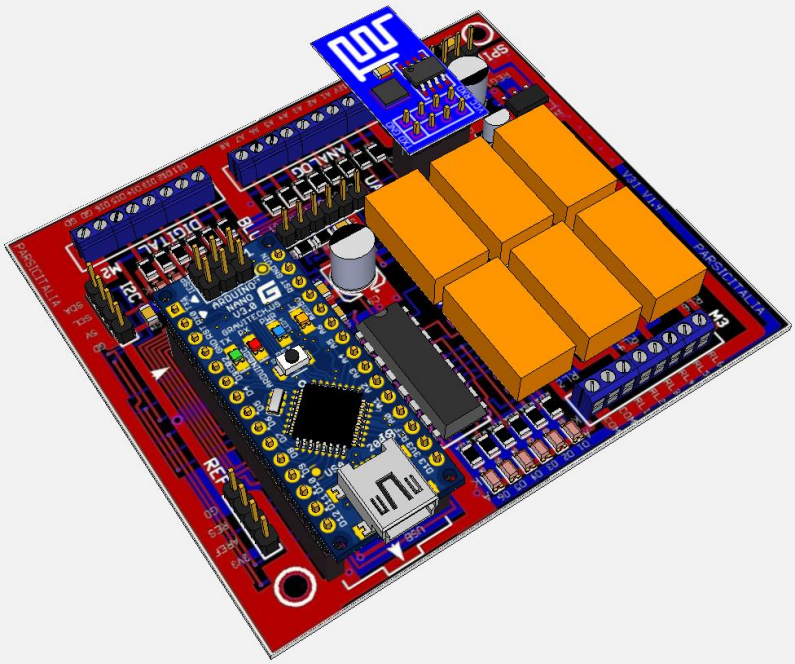

**Altri schemi di connessione Bluetooth**



## UART – Wi-Fi

**Connessione Wi-Fi ed impiego della scheda ESP8266SoC**

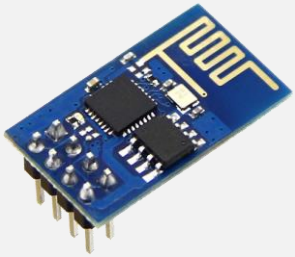
La scheda ESP8266 si inserisce direttamente sul connettore a 8 poli, come visibile in figura:



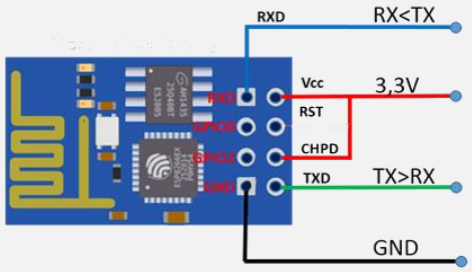
Le connessioni **GPI0 e Reset**, non necessarie al funzionamento del modulo, sono esclusi dai collegamenti. Il terminale **CH_PD** è collegato al terminale di alimentazione 3,3V.

**Nota tecnica importante** ⚠

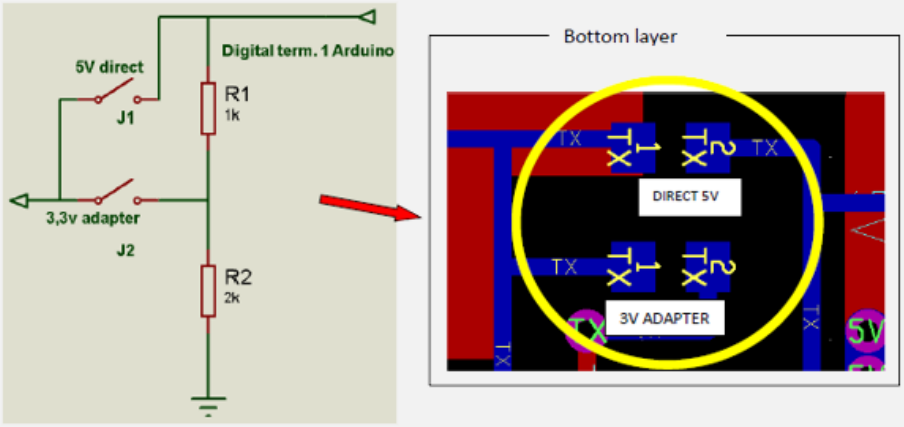
***Prima** di inserire la scheda è necessario saldare il ponte **TXD Adapter**, posizionato nel lato saldature del PCB. Il ponte **TXD Direct** deve restare aperto. Assicurarsi della corretta operazione se non volete vedere andare in fumo la vostra interfaccia Wi-Fi. La **ESP8266** è alimentata dal regolatore ausiliario a 3V installato a bordo delle schede V30 e V31.*



**ESP8266**



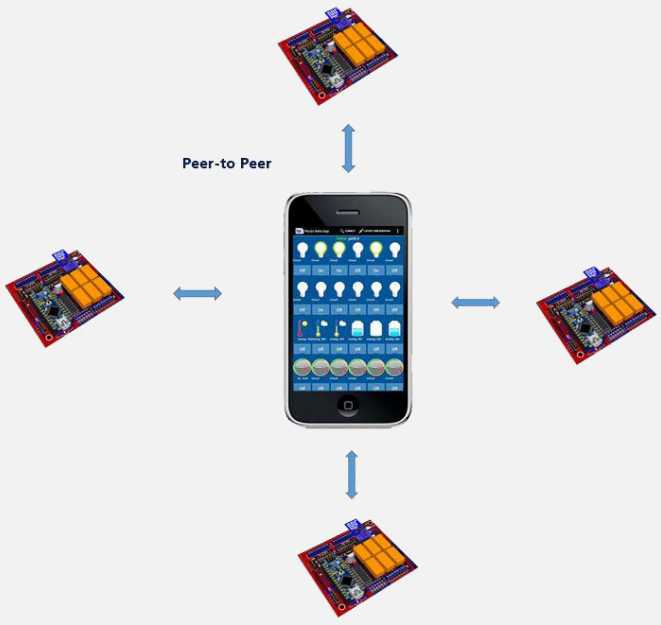
**Descrizione dei collegamenti al connettore Wi-Fi**




**Predisposizione dei jumper Direct – Adapter nella scheda V31**


*port WiFi V31*

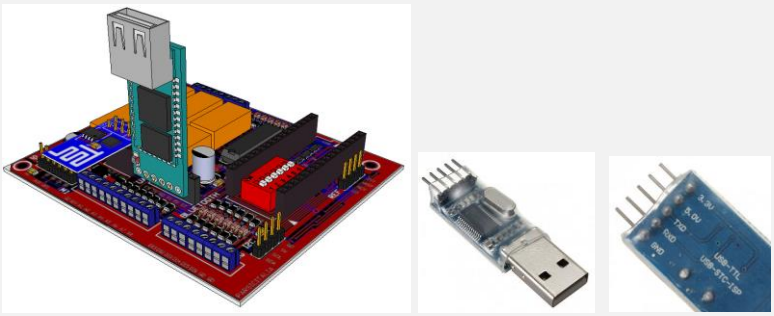
Nelle pagine seguenti introdurremo alcuni semplici argomenti didattici relativi al modulo ESP8266 e delle sue modalità di connessione. Spiegheremo inoltre come modificare il firmware dello stesso per la programmazione Wiring in ambiente IDE Arduino. *Nell'applicazione pratica con la scheda V31 il modulo ESP* ***non necessita di alcuna modifica software*** *e va usato così come reperibile in commercio.*



## La connessione Wi-Fi

La connessione wireless, Wi-Fi, assicura il trasferimento dati, su aree locali impiegando piccoli ricetrasmettitori operanti alla frequenza di 2,4 GHz (gigahertz). Nel campo delle telecomunicazioni, la tecnologia Wi-Fi consente a due terminali di collegarsi tra loro attraverso una rete locale. A sua volta la rete locale può essere allacciata alla rete Internet, tramite un router ed usufruire dei servizi di connettività offerti da un provider. Qualunque dispositivo fisso o portatile può connettersi ad una rete Wi-Fi adottando le specifiche tecniche del protocollo Wi-Fi.

L'*ESP8266 SoC (System-On-Chip),* è un modulo Wi-Fi dotato di un processore ARM 32bit e interfaccia GPIO per usi generali. Permette ai microcontrollori (MCU) di connettersi a una rete Wi-Fi rendendo semplici le connessioni TCP/IP, utilizzando i comandi AT (Hayes). Questo è possibile grazie al firmware residente nell'ESP8266. L'ESP8266, di fatti è autonomo, può essere programmato come un qualsiasi microcontrollore e per scopi diversi. Dato che l'ESP è dotato di CPU propria è pensabile interfacciare le porte GPIO a sensori e attuatori, senza l'ausilio dell'Arduino. Chi desidera eseguire test di comunicazione o verificare il funzionamento dell'ESP attraverso i comandi AT, può scollegare l'Arduino Nano o il chip Atmega 328 dalla scheda, e collegare alla porta UART un'interfaccia tipo *serial-to-usb*, PL2303HX o similare. Le connessioni, di questa interfaccia, si prestano bene per essere collegate direttamente alla presa UART della scheda V31 e V30, senza apportare alcuna modifica al circuito.



Predisposto il circuito, collegare l'interfaccia alla presa USB del PC e adoperare l'IDE Arduino per la programmazione.

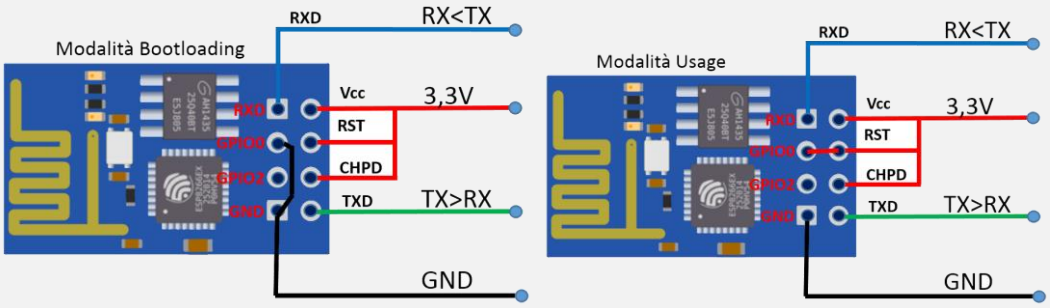
| UART Wi-Fi **V31** | Descrizione | Note |
|---|---|---|
| J1 | 3V3 tensione alimentazione 3V | Alimentazione scheda |
| J2 | RX | Collegamento RX-TX UART |
| J3 | RS Reset | Non connesso Floating |
| J4 | GPIO | Non connesso Floating |
| J5 | CHPD | Connesso +3V |
| J6 | GPIO | Non connesso Floating |
| J7 | TX | Collegamento TX-RX UART |
| J8 | GND | Massa generale |

## Modalità Bootloader

Alimentando il modulo ESP8266, questi si avvierà secondo le seguenti modalità di *start-up*:

- *UART - Bootloading*
- *FLASH - Usage*
- *SD_Card Boot – SDIO*

| Modalità | GPIO-0 | GPIO-1 | GPIO-15 |
|---|---|---|---|
| Bootloading | L | H | L |
| Usage | H | H | L |
| SDIO | Floating | Floating | H |

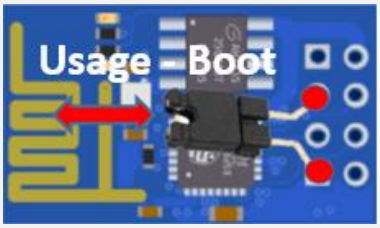


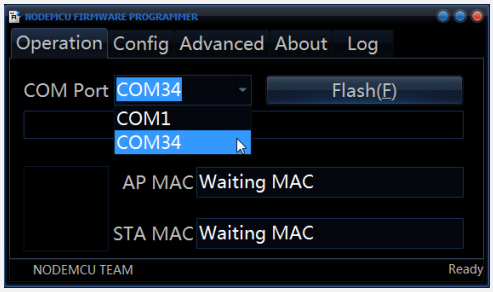

La terza modalità, *SDIO*, non è applicabile al modulo a cui facciamo riferimento.

*Il primo utilizzo dell'ESP* è come semplice modem di comunicazione Wi-Fi (Modalità Usage). Si collega alla porta seriale di un qualsiasi MCU esterno ed è gestito per mezzo di comandi AT. Ciò è possibile grazie al firmware di serie, residente nell'ESP8266.

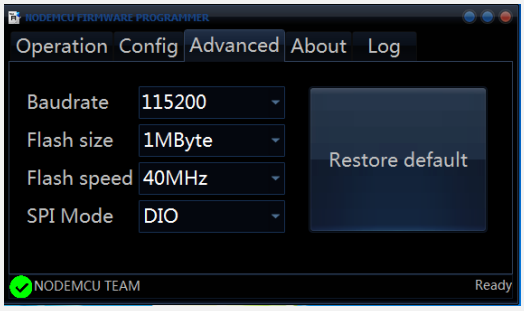
*Il secondo utilizzo* usa l'ESP come controllore autonomo abilitato al Wi-Fi, programmabile in modo interattivo mediante linguaggio di *scripting* (linguaggio di programmazione interpretato). Nella versione modificata di Arduino IDE, la programmazione dei moduli ESP avviene grazie all'estensione del linguaggio *Wire*. In pratica è possibile programmare il microcontrollore interno all'ESP come se fosse una scheda Arduino.

La programmazione del modulo ESP8266 in modalità stand-alone, operando in ambiente IDE Arduino, avviene installando il firmware opensource *NodeMcu,* per mezzo dell'applicazione ESP8266 Flasher , reperibili all'indirizzo web: *https://github.com/nodemcu/nodemcu-flasher*. Eseguito il download posizionare i files in una cartella denominata NodeMcu.
Collegare la scheda ESP in modalità *Bootloading,* ponticellando i teminali *GND e GPIO-0*, come visibile in figura. Si consideri che grazie alle predisposizioni della scheda V30-V31 non è necessario l'apporto di un alimentatore esterno a 3,3V, tantomeno di collegamenti elettrici aggiuntivi, ad esclusione del ponte *GND-GPIO-0*. Dopo aver alimentato il circuito, lanciare il programma ESP8266 Flasher.

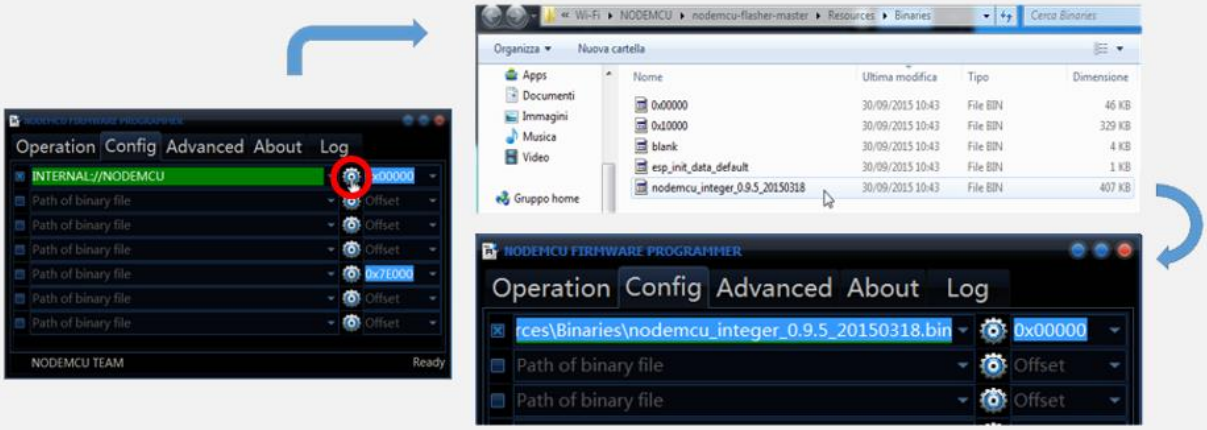


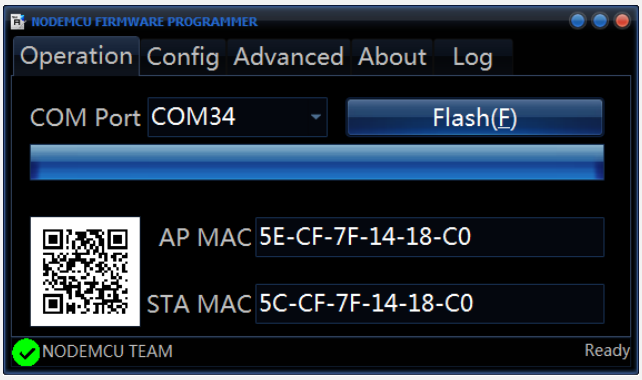
Selezionare la porta COM, a cui è collegata la scheda. Portarsi su Advanced, e selezionare i valori come indicati nella figura seguente:




Portarsi poi su *Config*; un click sul primo elemento della lista, ed aprire la cartella *Binaries*, posizionata *nel percorso > NodeMcu* (*NodeMcu/Flasher/Master/Resources/Binaries)*; selezionare il firmware *modemcu_integer-0.9.5_20150318*




In Operation, azionare Flash(F) per avviare la programmazione della scheda ESP; attendere che il bargraph esaurisca il suo percorso. Durante la connessione può accadere che l'ESP non sia riconosciuto. Riscontrato il problema di connessione, ricontrollare i collegamenti o disconnettere e riconnettere il modulo.




La procedura di programmazione dura pochi secondi. Al termine saranno evidenziati i codici MAC e il codice QR dell'ESP8266. Chiudere al termine l'applicazione.

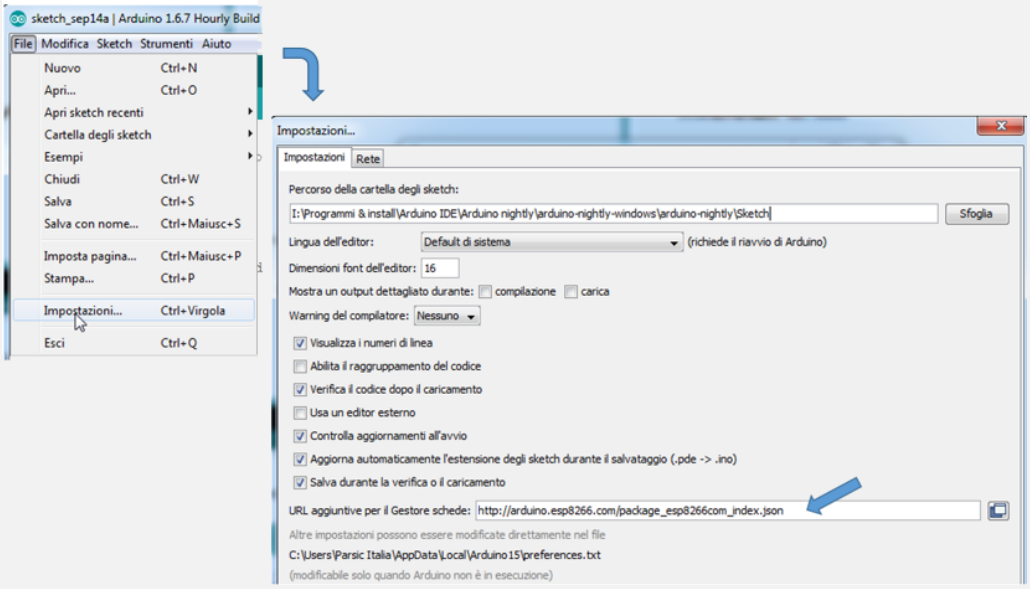
**Preparazione dell'IDE Arduino.**

È richiesta l'ultima versione dell'Arduino IDE, distribuita tramite il sito Arduino.org. Lanciare il programma e, da *"File"*, selezionare *"Impostazioni"*.

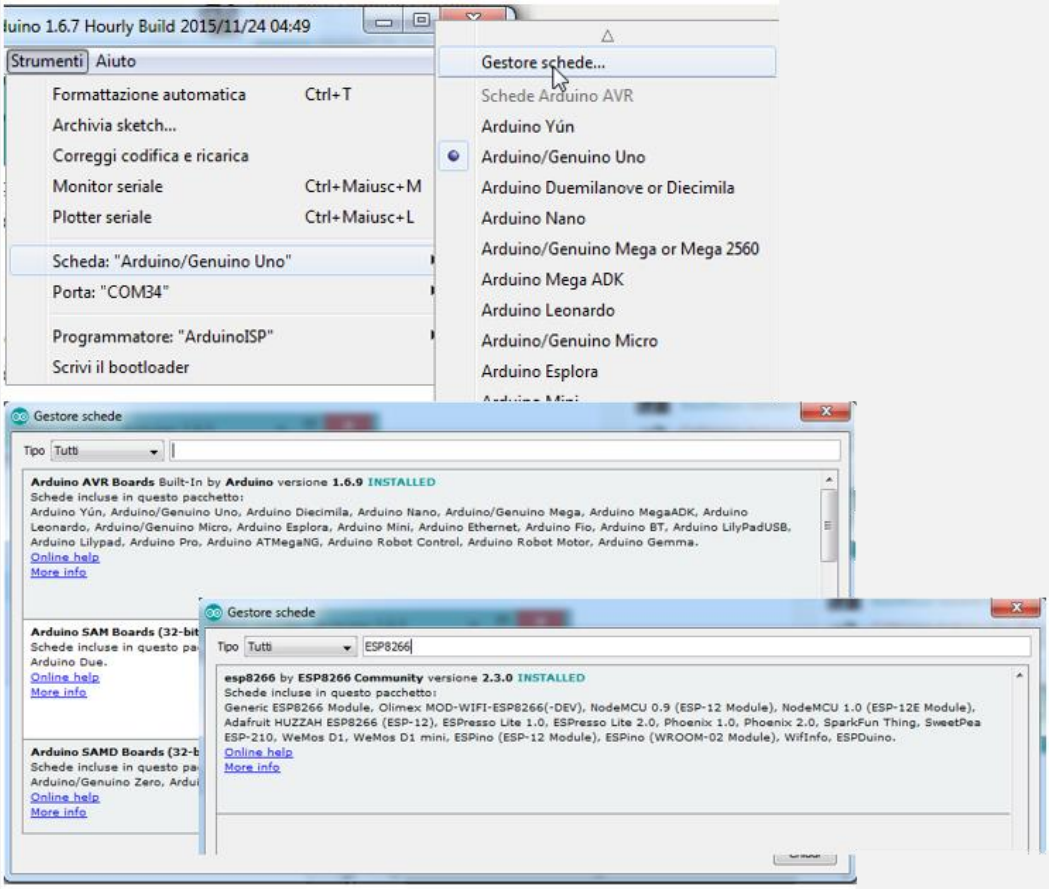
Nella finestra di configurazione, nel campo URL aggiuntive inserire il percorso:

http://arduino.esp8266.com/package_esp8266com_index.json

(reperibile all'indirizzo web: https://github.com/nodemcu/nodemcu-firmware)



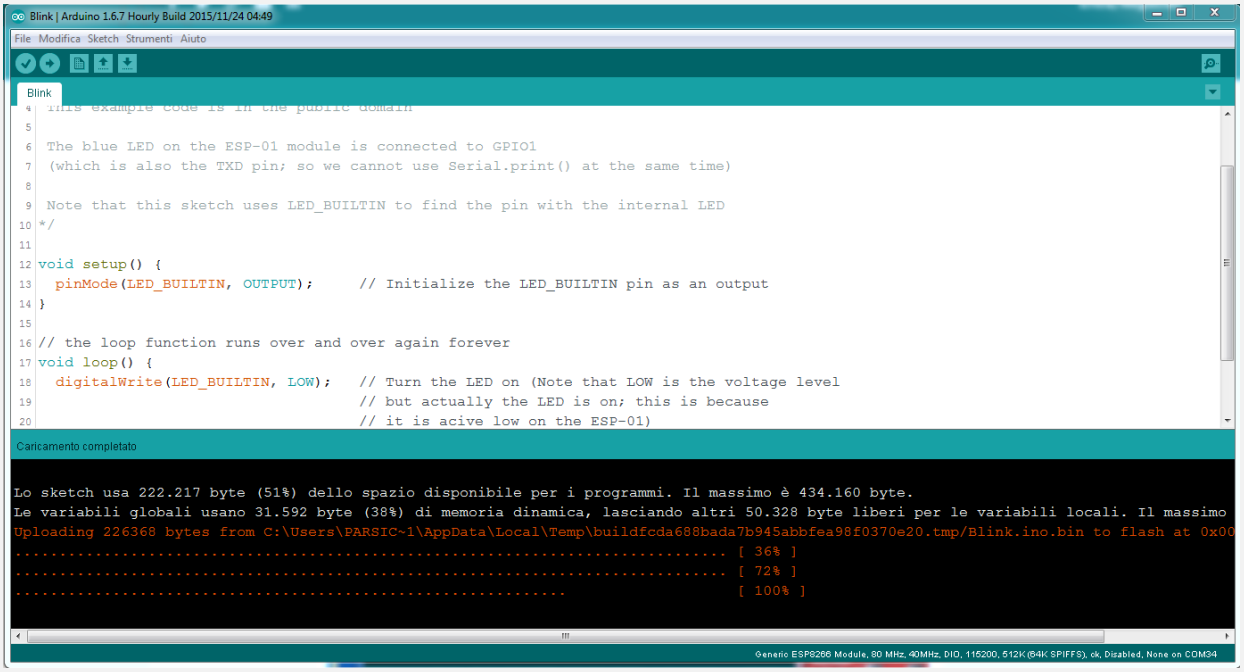

In Strumenti, seguire il percorso *"Scheda"* > *"Gestore Schede "*.
Accanto al riquadro *"Tipo"* scrivere *ESP8266, dare invio* e procedere all'installazione.



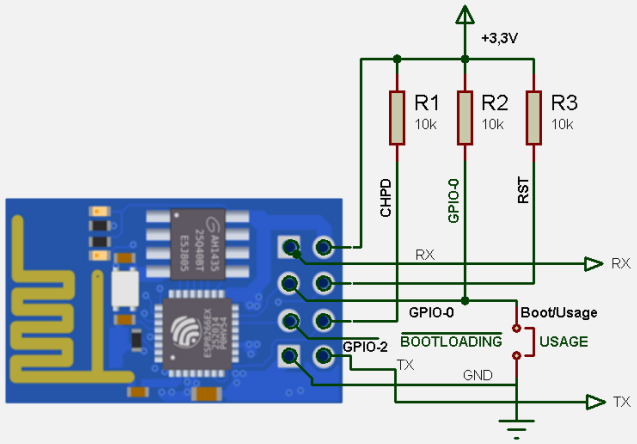
Attendere il tempo necessario affinché l'aggiornamento sia concluso (qualche minuto), poi controllate che l'installazione sia andata a buon fine.

## Eseguire il primo test

Seguire il percorso Strumenti > Scheda e selezionare *"Generic ESP8266 Module"*
Collegare ora la scheda in *modalità "Usage" aprendo il ponte, precedentemente collegato fra il terminale GND e GPIO-0*. Scollegare per qualche secondo la presa USB dal modulo e riconnetterla.
*Dal percorso "File" > "Esempi" > "ESP8266" > caricare l'esempio "Blink".*
*In "Strumenti" accertarsi che la "Porta" selezionata corrisponda a quella su cui si è connessi.* Caricare lo sketch, attendendo per alcuni secondi che il ciclo di programmazione vada a termine. A programmazione conclusa si vede, sulla scheda ESP8266, il led blu lampeggiare un secondo ON, due secondi OFF.




Per migliorare la stabilità del modulo ESP8266, in caso di installazione definitiva, è consigliabile collegare alcune resistenze di pull-up del valore di 10KΩ ai terminali GPIO e Reset. Il ponte Boot/Usage, può risultare di utilità in fase di programmazione.




Da questo momento in poi, potrete programmare il modulo impiegando l'IDE Arduino oppure utilizzando l'interprete Lua NodeMcu (nodemcu.com)

| ESP8266 AT Command Set | | |
|---|---|---|
| Function | AT Command | Response |
| Working | A | OK |
| Restart | AT+RST | OK [System Ready, Vendor:www.ai-thinker.com] |
| Firmware version | AT+GMR | AT+GMR 0018000902 OK |
| List Access Points | AT+CWLAP | AT+CWLAP +CWLAP:(4,"RochefortSurLac",-38,"70:62:b8:6f:6d:58",1)<br>+CWLAP:(4,"LiliPad2.4",-83,"f8:7b:8c:1e:7c:6d",1)<br>OK |
| Join Access Point | AT+CWJAP?<br>AT+CWJAP=SSID",”Password” | Query AT+CWJAP? +CWJAP:"RochefortSurLac" OK |
| Quit Access Point | AT+CWQAP=?<br>AT+CWQAP | Query<br>OK |
| Get IP Address | AT+CIFSR | AT+CIFSR 192.168.0.105<br>OK |
| Set Parameters of Access Point | AT+ CWSAP?<br>AT+ CWSAP= <ssid>,<pwd>,<chl>, <ecn> | Query<br>ssid, pwd<br>chl = channel, ecn = encryption |
| WiFi Mode | AT+CWMODE?<br>AT+CWMODE=1<br>AT+CWMODE=2<br>AT+CWMODE=3 | Query<br>STA<br>AP<br>BOTH |
| Set up TCP or UDP connection | AT+CIPSTART=?<br>(CIPMUX=0) AT+CIPSTART = <type>,<addr>,<port><br>(CIPMUX=1) AT+CIPSTART= <id><type>,<addr>, <port> | Query<br>id = 0-4, type = TCP/UDP, addr = IP address, port= port |
| TCP/UDP Connections | AT+ CIPMUX?<br>AT+ CIPMUX=0<br>AT+ CIPMUX=1 | Query<br>Single<br>Multiple |
| Check join devices' IP | AT+CWLIF | |
| TCP/IP Connection Status | AT+CIPSTATUS | AT+CIPSTATUS? no this fun |
| Send TCP/IP data | (CIPMUX=0) AT+CIPSEND=<length>;<br>(CIPMUX=1) AT+CIPSEND= <id>,<length> | |
| Close TCP / UDP connection | AT+CIPCLOSE=<id> or AT+CIPCLOSE | |
| Set as server | AT+ CIPSERVER= <mode>[,<port>] | mode 0 to close server mode; mode 1 to open; port = port |
| Set the server timeout | AT+CIPSTO?<br>AT+CIPSTO=<time> | Query<br><time>0~28800 in seconds |
| Baud Rate | AT+CIOBAUD?<br>Supported: 9600, 19200, 38400, 74880, 115200, 230400, 460800, 921600 | Query AT+CIOBAUD? +CIOBAUD:9600 OK |
| Check IP address | AT+CIFSR | AT+CIFSR 192.168.0.106<br>OK |
| Firmware Upgrade (from Cloud) | AT+CIUPDATE | 1. +CIPUPDATE:1 found server<br>2. +CIPUPDATE:2 connect server<br>3. +CIPUPDATE:3 got edition<br>4. +CIPUPDATE:4 start update |
| Received data | +IPD | (CIPMUX=0): + IPD, <len>:<br>(CIPMUX=1): + IPD, <id>, <len>: <data> |
| Watchdog Enable | AT+CSYSWDTENABLE | Watchdog, auto restart when program errors occur: enable |
| Watchdog Disable | AT+CSYSWDTDISABLE | Watchdog, auto restart when program errors occur: disable |

**La programmazione di Arduino**

Le operazioni da seguire sono le seguenti:

1. Collegare l'Arduino al PC utilizzando un cavo USB con terminazione tipo Micro usb;
2. Aprire l'IDE Arduino, selezionare "Strumenti > Gestore schede" e selezionare il modello su cui operate. Selezionate ancora "Strumenti > Porta seriale": spuntare il flag della porta COM collegata;
3. Copiare il programma o gli esempi, forniti assieme alla scheda, nell'area Editing dell'IDE.
4. Azionare il tasto verifica e correggere eventuali errori.
5. Lanciare la compilazione e programmazione azionando il tasto carica.

Qualche volta, al primo utilizzo, possono apparire segnalazioni di errore a causa dell'errato collegamento USB. In questo caso provare a sconnettere e riconnettere il cavo USB. Durante la fase di caricamento del codice, il modulo ***Bluetooth o WiFi devono essere temporaneamente disconnessi dalla scheda***. Se esistono problemi di caricamento del codice, potrebbe verificarsi un conflitto hardware sulla scheda V31. Per risolverlo, aprire il dip-switch 1 (CK). Se il problema persiste, *scollegare temporaneamente l'Arduino Nano* e procedere alla programmazione.
Come primo programma, conviene installare, lo sketch Parsic V31 WiFi example. L'operazione richiede pochi minuti di applicazione. Se l'APP è stata installata nello smartphone, attivare la comunicazione seriale Wi-Fi come descritto in precedenza ed eseguire i test di funzionamento. Dopo aver sperimentato l'applicazione si può procedere, se necessario ad eventuali modifiche, seguendo uno stile standard di codifica. Questo permette una facile manutenzione e aggiornamento del programma, anche dopo molto tempo dal suo utilizzo. Le istruzioni sono quelle impiegate per la struttura di un programma Arduino

- Variabili, Costanti e Tipo di dato;
- Funzioni (parametri d'ingresso e parametri di ritorno);
- Operazioni Aritmetiche e Operatori Logici e Operatori di Confronto;
- Istruzioni condizionali;
- Istruzioni di preprocessore;
- Costanti di Arduino
- Funzioni di Arduino

parsic_v31_wifi_example

parsic_v31_wifi_example §

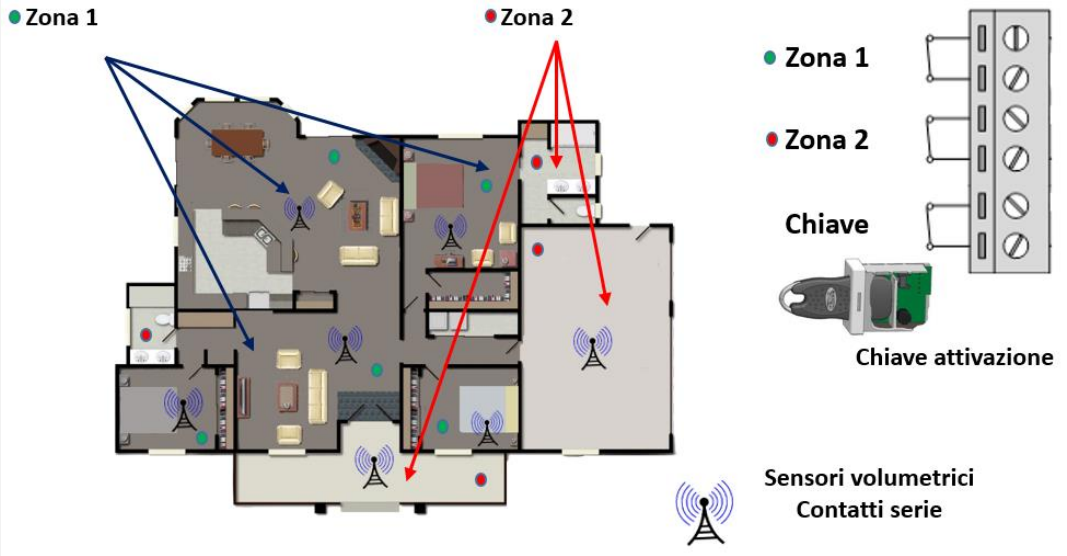
```
// Relay 1 is at pin 8, relay 2 is at pin 9 and so on.
int RelayPins[MAX_RELAYS]  = {8, 9, 10, 11, 12, 13};
// Relay 1 will report status to toggle button and image 00, relay 2 to button 01 and so on.
String RelayAppId[MAX_RELAYS] = {"00", "01", "02", "03", "04", "05"};
// Command list (turn on - off for eachr relay)
const char CMD_ON[MAX_RELAYS] = {'A', 'B', 'C', 'D', 'E', 'F'};
const char CMD_OFF[MAX_RELAYS] = {'a', 'b', 'c', 'd', 'e', 'f'};
// Used to keep track of the relay status in eeprom
int RelayStatus = 0;
int STATUS_EEADR = 20;

/*
 * Digital input config
 */
#define MAX_D_INPUTS 6
boolean DigitalLatch[MAX_D_INPUTS] = {false, false, false, false, false, false};
boolean DIStatus[MAX_D_INPUTS] = {false, false, false, false, false, false};
int DigitalInputs[MAX_D_INPUTS] = {2, 3, 4, 5, 6, 7};
String DIAppId[MAX_D_INPUTS] = {"06", "07", "08", "09", "10", "11"};
```

## Software

### Antifurto home. Sketch Arduino V31

L'applicazione che descriviamo qui di seguito, è utile in tutti quei casi sia necessario installare un sistema antifurto, che proteggerà l'abitazione o un magazzino dalle intrusioni "*non autorizzate*". Il sistema di allarme è basato sulla scheda V31, facilmente installabile in una scatola elettrica commerciale e provvista di guida omega, a cui sono connessi le linee di allarme di tipo NC (normalmente chiuso). Le linee provengono da due zone diverse: la Zona 1 e la Zona 2. In pratica, dopo aver collocato i vostri sensori di tipo volumetrico e/o perimetrale, riunire la serie dei contatti in un unico filo, collegato ai morsetti di ingresso digitale. La prima serie di contatti riguarderà la Zona 1, la seconda serie di contatti la Zona 2. Si deciderà quale linea assegnare, in caso di una abitazione, alla zona notte e alla zona giorno. Lo schema generale dell'impianto sarà simile a quello rappresentato in figura:




**Nota:**

Alla prima applicazione dell'App, per quanto riguarda la connessione seriale WiFi, è richiesto l'inserimento di una ***password*** sostituibile, con altra simile, modificando il rigo 34 dello sketch:

***password parsicApp***

```
34  String pass = "parsicAPP";              //Password Wi-Fi
```

L'inserimento del circuito di allarme avviene attraverso una chiave di attivazione (qualunque tipo wireless o elettromeccanico). Il contatto della chiave, sarà mantenuto normalmente chiuso (N.C.) per tutta la durata del controllo.

**Inserimento e spegnimento allarme**

Ogni qualvolta si alimenta la scheda, è necessario attendere 10 secondi circa, affinché il ciclo software sia perfettamente stabilizzato. Al termine di questa pausa un relè si aziona per un secondo. A connessione WiFi attiva, un messaggio vocale emesso dallo smartphone, avverte che il collegamento è ripristinato.
Il controllo della zona 1 – zona 2, avviene soltanto a chiave di attivazione inserita (contatto chiave normalmente chiuso). Azionando i pulsanti Zona 1 – Zona 2, portandoli a ON, un messaggio vocale avverte che i circuiti sono stati attivati.


Se una delle due zone è sottoposta ad allarme, intrusione non autorizzata, si attiva:

- Il relè di allarme, temporizzato per un periodo limitato di funzionamento della sirena;
- Il relè di lampada lampeggiante, con azione permanente, fino al reset manuale;
- Si accende la lampada di zona nel pannello sinottico;
- Viene riprodotto un messaggio vocale di avvertimento per tre volte consecutive.



**Reset dell'allarme.**
Portando tasto ON della zona soggetta ad allarme, a OFF, si resetta il ciclo.
I restanti tasti del pannello di controllo sono programmati per svolgere le seguenti funzioni:

- Accensione luci interne/esterne con relè bistabile
- Accensione sirena (panico) oppure altro accessorio elettrico con relè bistabile
- Comando apriporta con relè monostabile (impulso)
- Comando chiusura tapparelle con relè monostabile (impulso)

Lo sketch Arduino, relativo al circuito di allarme, è scritto evitando l'impiego di algoritmi complicati ed è comprensibile a chiunque abbia un minimo di esperienza di programmazione.
Chi desidera personalizzare l'applicazione, oppure ottimizzarla, può operare attraverso l'Editor di Arduino, conservando inalterato lo sketch originale, salvando le eventuali modifiche impiegando estensioni numerate: App V1.1.2, App V1.1.3, ecc.

Allarme_V1.1_solo_Wi-Fi_it

Allarme_V1.1_solo_Wi-Fi_it§

```
28
29
30 #define BAUD_RATE 115200              //baud rate Wi-Fi
31 #define RX_PIN       0                //Pin RX seriale
32
33 String SS_ID = "Parsic_Italia_Allarm";   //SSID Wi-Fi
34 String pass = "parsicAPP";             //Password Wi-Fi
35 String IPAP = "\"192.168.1.60\"";      //Indirizzo ip statico del modulo (Variabile a scelta)
36
37 String sPort = "80";                   //Porta di comunicazione standard HTTP (Porta 80)
38
39
40 //predisposizione ingressi digitali di tipo pull-down
41 int Zona1 = 2;             //ingresso sensori N.C. pull-down zona 1
42 int Zona2 = 3;             //ingresso sensori N.C. pull-down zona 2
43
44 int Chiave = 7;            //ingresso interruttore a chiave abilitazione settaggi APP ATC
45
46 int Sirena = 8;            //uscita relè¨ sirena
47 int Lampada = 9;           //uscita relè¨ lampeggiante sirena
48 int Buzzer = 10;           //buzzer interno
49 int CombTel = 11;          //uscita Relè¨ abilitazione messaggio allarme GSM
50
51 //Logica controllo lampeggiante
52 int LampState = LOW;
53 unsigned long prevMillisLamp = 0;
54 const long intervalLamp = 500;      //intervallo lampeggiante in millisecondi
```

Chi desidera ottenere un controllo ambientale di temperatura, può prendere spunto dallo sketch raffigurato di seguito. Il software permette di impostare due soglie di intervento relè, consentendo di regolare l'isteresi su un determinato valore.

Analogico_1

Analogico_2 | Arduino 1.6.6

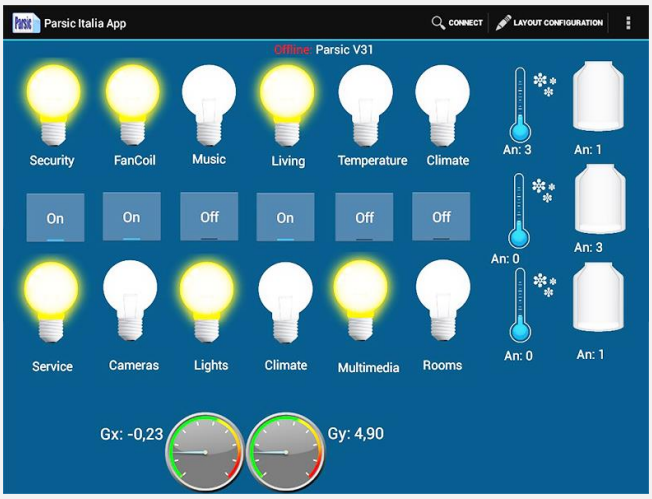
File Modifica Sketch Strumenti Aiuto

Analogico_2

```
1  void TX_RX_Loc();
2
3  int S1=0; //stato relè 1
4  int S2=0; //stato relè 2
5
6  int rel1 = 8;
7  int rel2 = 9;
8
9  int Pin_in_rel1_2 = A0; //pin ingresso analogico relativo al relè 5_6 (Input)
10 int Soglia_INF_rel1_2 = 200; //quando l'ingresso analogico supera la Soglia con trigger di schmitt --> il relè si accende (Soglia Inferiore)
11 int Soglia_SUP_rel1_2 = 800; //quando l'ingresso analogico supera la Soglia con trigger di schmitt --> il relè si accende (Soglia Superiore)
12 int TR_soglia_rel1_2 = 10; //quando l'ingresso analogico = Soglia - TR --> il relè si spegne (Isteresi)
```

**Conclusioni**

Il progetto Parsic Italia App contiene molti elementi didattici, utili e pratici, per chi desidera un approccio immediato con il sistema di sviluppo Arduino. Per coloro che desiderano l'impiego immediato di un modulo di automazione, la scheda V31 si presta bene per essere impiegata in quei progetti di automazione con un limitato budget di spesa. Per coloro che si avvicinano per la prima volta a questa materia, attraverso gli esempi software, le schede sono utilissime per l'apprendimento delle tecniche di programmazione, passando dalla teoria alla pratica con un semplice click del mouse.
Tutto ciò che è necessario, dal punto di vista hardware, è disponibile a bordo delle schede, senza doversi munire di numerosi accessori per il prolungamento dei collegamenti elettrici.
I requisiti tecnici di base per l'impiego delle schede sono un PC ed un alimentatore. Il computer dovrà essere dotato di sistema operativo Windows, con a bordo installato l'IDE Arduino, come ampiamente spiegato nei precedenti capitoli.






**Indirizzi internet utili:**
ATC Arduino Total Control Google Play Parsic Italia APP
ATC Arduino Total Control GIGTHUB ATC-Release-Codes
Arduino® Nano https://www.arduino.cc/en/Main/ArduinoBoardNano
Arduino® UNO R3 http://arduino.cc/en/main/arduinoBoardUno
Arduino® Reference Wire http://arduino.cc/en/Reference/Wire
Bluetooth http://arduino.cc/en/Main/ArduinoBoardBT?from=Main.ArduinoBoardBluetooth
ESP8266 https://espressif.com/en/products/hardware/esp8266ex/overview
NodeMcu Open Source Software http://nodemcu.com/index_en.html

**Bibliografie consultate:**
Wikipedia Arduino
Wikipedia ESP8266 NodeMcu
Arduino.cc sito ufficiale Arduino

Si desidera ringraziare il Prof. Vincenzo Curcetti e il sig. Matteo Ricciotti per la loro fattiva collaborazione.