CONOSCERE E USARE Node-RED 3

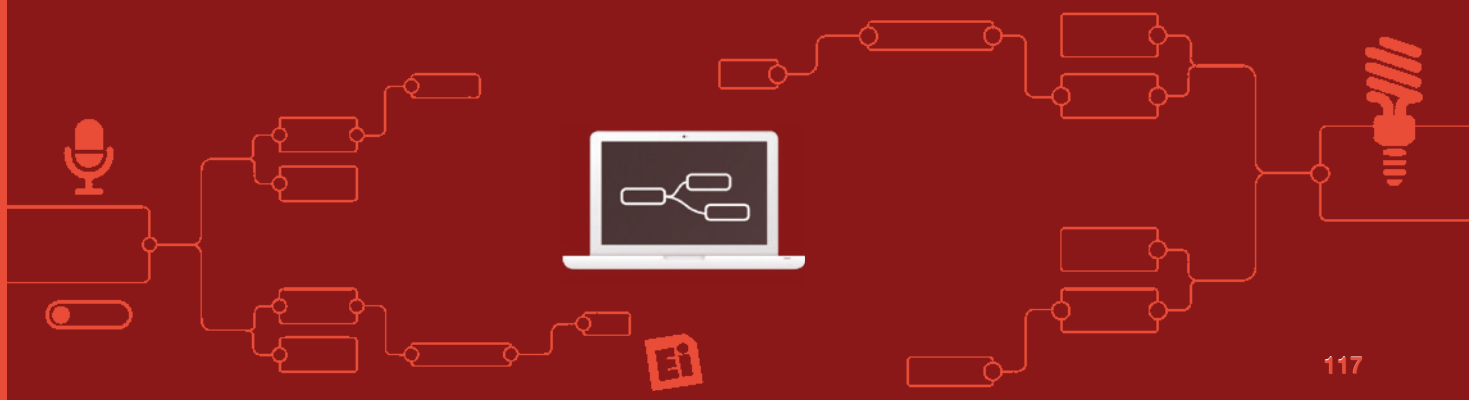
di FRANCESCO FICILI

Proseguiamo lo studio del tool di flow-based programming orientato all'IoT ed alla connettività. In questa terza puntata analizziamo la gestione delle periferiche della Raspberry PI tramite Node-RED, approfittandone per l'implementazione di nuovi esempi pratici. Terza puntata.

Nelle precedenti puntate abbiamo introdotto i concetti fondamentali di Node-RED, presentando l'ambiente grafico ed i principali nodi predefiniti, implementando anche i primi, semplici, esempi. In questa puntata muoviamo un passo avanti e iniziamo a scoprire come gestire le periferiche della Raspberry Pi tramite Node-RED. Come abbiamo già accennato in precedenza, Node-RED è un ambiente di programmazione che supporta una moltitudine di piattaforme, dal momento che le dipendenze hardware vengono completamente astratte. È chiaro, però, che tali dipendenze esistono e la gestione dell'hardware specifico varia a seconda del target utilizzato (ossia è diverso tra una Raspberry Pi e una BeagleBoard).
Tali differenze sono indirizzate in maniera diversa a seconda del nodo utilizzato; in alcuni casi i nodi sono specifici, ossia sono legati alla piattaforma hardware (è il caso di alcuni nodi Raspberry Pi). In altri casi invece i nodi sono strutturati per supportare diverse piattaforme, e fa parte della configurazione del nodo la selezione della piattaforma specifica. Nelle puntate future del corso, utilizzeremo sempre la Raspberry Pi come piattaforma hardware per i nostri test e quindi faremo spesso uso di nodi specifici.
Tenete comunque conto che tali specificità possono o meno essere portate su altre piattaforme, a seconda delle caratteristiche hardware (ad esempio una semplice interfaccia GPIO è spesso facilmente portabile, mentre interfacce più specifiche lo sono di meno o possono non essere supportate da alcune piattaforme).

PERIFERICHE RASPBERRY PI

La dotazione di periferiche della Raspberry Pi è straordinariamente ricca e questo SBC dispone sia delle periferiche evolute tipiche delle piattaforme ad alta potenza computazionale (Ethernet, USB,



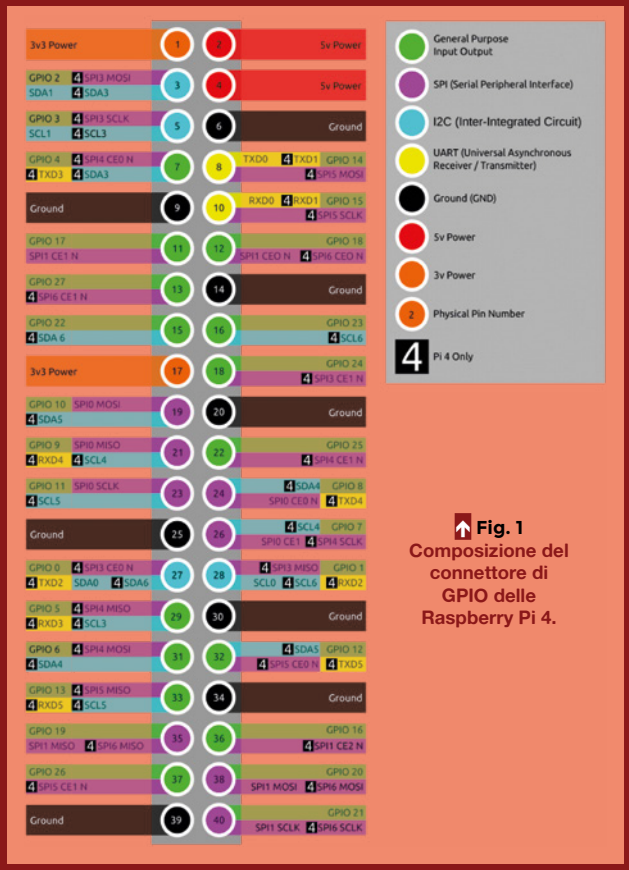


Fig. 1 Composizione del connettore di GPIO delle Raspberry Pi 4.

HDMI, CSI, DSI ecc.), sia delle periferiche di base, più vicine al mondo dei microcontrollori (IO Digitale, PWM, periferiche di comunicazione inter-circuit, eccetera).
In particolare, nella versione 4, con il passaggio dal BCM28xx al BCM2711, è stato notevolmente arricchito il set di funzioni alternative presenti sul connettore di GPIO.
In particolare, la versione 4 dispone delle seguenti periferiche sul connettore GPIO (ovviamente multiplexate sul totale dei 28 pin disponibili):

- Fino a 28x GPIO Pin
- Fino a 6x UART
- Fino a 6x I2C
- Fino a 5x SPI
- Fino a 2x PWM
- Fino a 3x GPCLK outputs
- 1x SDIO
- 1x DPI (Display Parallel Port)
- 1x PCM
- 1x 1-Wire

In Fig. 1 è riportata la disposizione delle varie funzioni alternative sul connettore GPIO a 40-pin della Raspberry Pi 4 Model B. Come si può facilmente notare le possibilità offerte dalle interfacce presenti sul connettore di GPIO sono notevoli; è possibile infatti controllare I/O digitali, come pulsanti o relè, interfacciare una moltitudine di sensori e attuatori usando i vari bus di comunicazione disponibili, interfacciare motori o LED tramite i canali PWM e molto altro ancora. Nei prossimi paragrafi vedremo come gestire in Node-RED due gruppi di interfacce, il GPIO standard (ossia IO Digitali e PWM) e alcune periferiche di comunicazione (UART, I²C, 1-wire).

GESTIONE GPIO RASPBERRY PI

La dotazione di base della Raspberry Pi prevede una serie di standard GPIO (28 in totale, corrispondenti al GPIO bank 0 del BCM2711), che possono essere utilizzati come digital input o digital output.
La dinamica del segnale di queste linee digitali è 0-3,3V e ogni linea è dotata di pull-up e pull-down selezionabili via software, quindi non è necessario avere resistenze di pull-up o pull-down esterne per interfacciare pulsanti ed interruttori. Inoltre alcune linee di GPIO possono essere utilizzate come canali PWM (Pulse-Width Modulation), in particolare il BCM2711 dispone di due canali PWM (PWM 0 e 1) multiplexati con i pin GPIO 12 e 13 (o in alternativa GPIO 18 e 19).
In ogni caso tutti i GPIO possono implementare il PWM software e vedremo che questa modalità è ampiamente supportata in Node-RED. Non ci sono canali analogici nativi all'interno del BCM2711 (come anche nei predecessori), di conseguenza per acquisire segnali analogici è necessario interfacciare un convertitore A/D esterno, tramite un apposito bus di comunicazione (tipicamente SPI o I²C).
In Node-RED, per l'interazione con i GPIO della Raspberry Pi esistono diverse soluzioni; in particolare riportiamo tre moduli che possono essere installati dalla palette di Node-RED e sono tra i più diffusi.

- **node-red-node-pi-gpio**: questo è un set di nodi che troviamo installato di default su Node-RED quando viene utilizzato lo script di installazione suggerito sul

Fig. 2 - Nodi node-red-node-pi-gpio

sito web di Node-RED. Se utilizziamo Raspberry Pi OS, il modulo è già preconfigurato per funzionare correttamente, quindi questa è la via più semplice per poter accedere al GPIO della Raspberry Pi usando Node-RED. Il modulo implementa tutte le funzionalità di base, ossia Digital Input (con gestione pull-up/down e debounce), Digital Output e PWM.

- **node-red-node-pi-gpiod**: questo modulo si basa sul demone PiGPIOd (*http://abyz.me.uk/rpi/pigpio/pigpiod.html*) per interfacciare il GPIO della Raspberry. Di conseguenza, per poter utilizzare i suoi nodi, è prima necessario installare correttamente il demone, seguendo le istruzioni riportate sulla sezione download della pagina web di PiGPIOd (*http://abyz.me.uk/rpi/pigpio/download.html*). Non esiste un vero e proprio installer, ma bisogna scaricare un pacchetto che poi deve essere compilato localmente. Inoltre il demone deve essere preventivamente avviato (tramite riga di comando o allo startup usando rc.local o systemd) affinchè i nodi funzionino correttamente. Si tratta di un sistema più complesso rispetto ai nodi predefiniti, ma che offre migliori performance e funzionalità aggiuntive (ad esempio la possibilità di controllare servo tramite le linee di GPIO).
- **node-red-contrib-gpio**: questo modulo permette di accedere al GPIO su una serie di differenti piattaforme (incluse Raspberry PI, Beagle Bone, Intel Galileo, eccetera).

Nei paragrafi che seguono utilizzeremo i nodi di default (node-red-node-pi-gpio), ma teniamo presente che ci sono anche altre strade che consentono di migliorare le performance ed accedere a funzionalità aggiuntive.

DIGITAL I/O

Come abbiamo già detto il set di nodi node-red-node-pi-gpio è preinstallato nella versione di Node-RED per Raspberry Pi. Questo set è presente nella sezione Raspberry Pi della palette (Fig. 2) e contiene i seguenti nodi:

- **rpi-gpio out**: permette di gestire i pin di GPIO in uscita; i pin possono essere gestiti in modalità digitale o PWM;
- **rpi-gpio in**: permette di gestire i pin di GPIO in ingresso; questo nodo permette anche di controllare i setting delle resistenze di pull-up e pull-down integrate, oltre che impostare il debounce.
- **rpi-mouse**: permette di gestire un mouse USB collegato alla Raspberry (solo gestione pulsanti);
- **rpi-keyboard**: permette di gestire una tastiera USB collegata alla Raspberry Pi.

Per i nostri scopi i nodi di maggiore interesse sono i primi due (GPIO out e in), mentre non ci occuperemo dei nodi mouse e keyboard. Abbiamo già avuto modo di utilizzare il nodo GPIO negli esempi precedenti, senza però entrare nel dettaglio della configurazione ed esaminare le funzionalità aggiuntive. Partiamo quindi dall'analisi di questo nodo, per poi passare al nodo GPIO in.

Fig. 3 - Configurazione nodo GPIO out.

GPIO out

Per posizionare il nodo sul workspace preleviamolo dalla palette e posizioniamolo su un flow, come di consueto. Come si può notare già dall'icona, il nodo ha un singolo ingresso e nessuna uscita, questo ci fa capire che riceve in input un messaggio (contenente lo stato logico o il valore del PWM duty-cycle con il quale vogliamo controllare l'uscita selezionata) e non restituisce nulla in uscita, in quanto attua direttamente la linea fisica. Cliccando due volte con il tasto sinistro del mouse sul nodo si aprirà la schermata di configurazione, riportata in Fig. 3.
Come si può vedere dalla predetta figura, la configurazione del noto GPIO out è estremamente intuitiva. Sostanzialmente abbiamo una schermata con un controllo principale (Pin) che riproduce la composizione del connettore di GPIO a 40 Pin della Raspberry Pi, dal quale possiamo selezionare il Pin che intendiamo utilizzare. Una volta eseguita questa operazione selezioniamo la modalità del Pin (Digital output o PWM), usando il drop-down menu Type. Infine abbiamo la possibilità di impostare il valore iniziale del Pin (selezionando la checkbox "Initialize pin state") e possiamo anche dare un nome più familiare al nodo (ad esempio

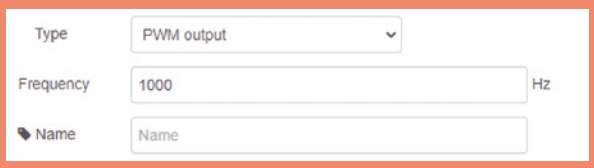


Fig. 4 - Configurazione PWM.

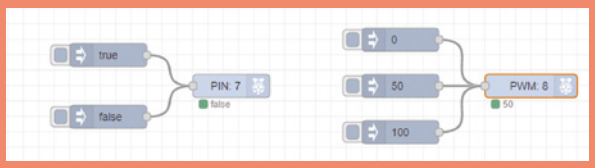


Fig. 5 - Test digital output e PWM.

LED_1). Nel caso in cui si selezioni PWM anziché Digital output come modalità, la schermata cambia leggermente e presenta un nuovo controllo di tipo textbox chiamato Frequency, che serve per impostare la frequenza del PWM in Hz, come si può vedere dalla Fig. 4.
Il controllo di questo nodo è estremamente semplice, in sostanza è sufficiente inviare un messaggio il cui msg.payload contenga lo stato dell'uscita digitale (come boolean o come duty-cycle PWM). Per testare il funzionamento del nodo si possono costruire dei semplici nodi di test, come quelli rappresentati in Fig. 5, che fanno uso del nodo Inject per iniettare i valori digitali e di duty-cycle.

GPIO in

Il nodo GPIO in funziona in maniera del tutto duale rispetto al nodo GPIO out. Prelevandolo dalla palette e posizionandolo sul workspace si nota subito che il nodo ha una singola uscita e nessun ingresso. In maniera similare al nodo GPIO out, l'input di questo nodo e dato dal cambio di stato sulla linea digitale configurata, e il suo output è un messaggio il cui payload contiene il valore digitale che rappresenta lo stato del pin (0 o 1). Come per il nodo GPIO out, anche questo nodo dispone di una finestra delle proprietà (Fig. 6) che può essere aperta tramite doppio click sinistro sull'icona del nodo.
Anche in questo caso abbiamo il selettore Pin che ci permette di selezionare quale GPIO utilizzare dalla lista dei GPIO presenti sul pinout del connettore a 40 pin. I controlli addizionali Resistor e Debounce permettono di configurare la resistenza di pull-up/down e l'intervallo di debounce (in ms). Questi due controlli risultano molto utili nel caso in cui si debbano interfacciare dei pulsanti. Infine la checkbox "Read initial state of pin on deploy/restart?" permette di effettuare una prima lettura all'avvio del flow, così che lo stato del pin non risulti indeterminato all'inizio. Il solito controllo di tipo textbox "Name" permette di dare un nome più appropriato al nodo (ad esempio BUTTON_1).
Se vogliamo testare il funzionamento del nodo è sufficiente (dopo averlo configurato opportunamente) collegare un nodo di debug alla sua uscita, come rappresentato in Fig. 7 e collegare fisicamente un pulsante alla linea digitale selezionata. Con una singola pressione del pulsante il nodo dovrebbe ritornare due messaggi, corrispondenti ai due stati tra cui transita la linea digitale (ad esempio 0 e 1, pulsante premuto e poi rilasciato, se il pulsante si chiude verso massa e viene utilizzata una resistenza di pull-up sulla linea).
Per verificare il funzionamento del GPIO di base, può essere molto pratico utilizzare l'HAT I/O distribuito da Futura Elettronica (www.futurashop.it) con il codice FT1060M. Questo HAT (sostanzialmente uno shield, una cui immagine è visibile in Fig. 8) dispone inoltre di un convertitore AD/DA (PCF8591) I²C, in modo da poter implementare anche alcune funzioni analogiche. Ecco le caratteristiche e la dotazione dell'HAT FT160M:

- 5 LED, collegati ad altrettanti digital output;
- 2 Pulsanti e 1 interruttore, collegati ad altrettanti digital input;

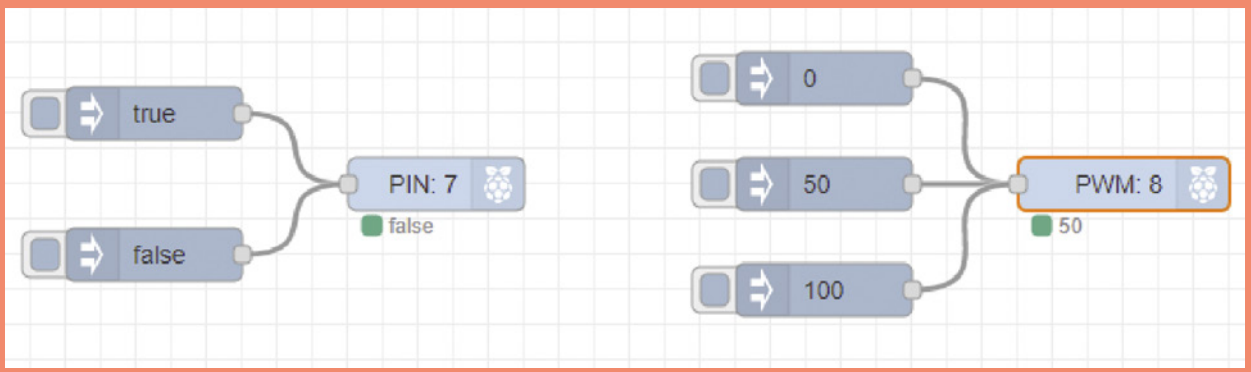


Fig. 6 - Configurazione del nodo GPIO in.

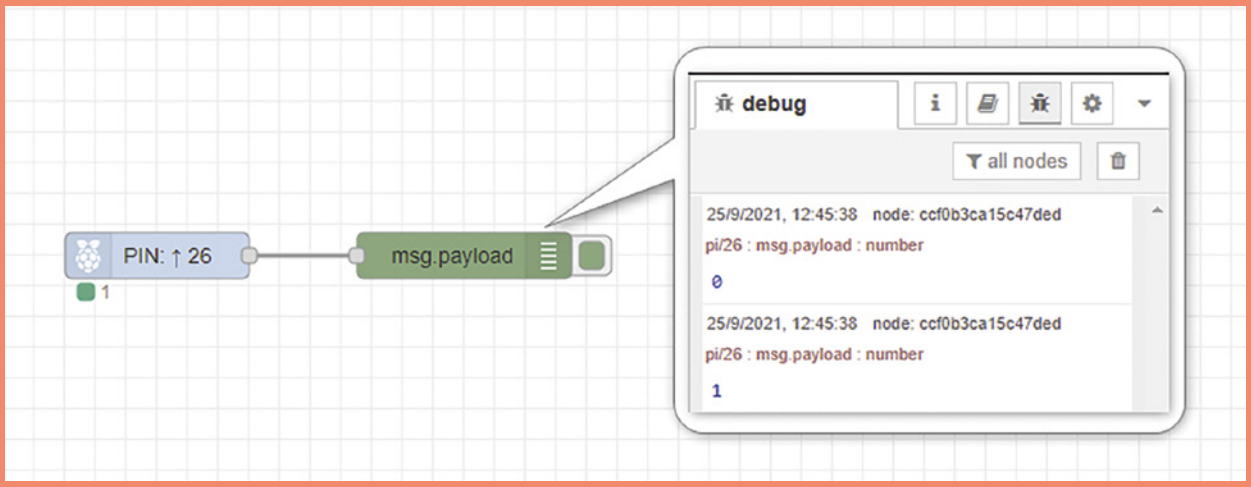


Fig. 7 - Test del nodo GPIO in con visualizzazione della finestra di debug.

- 1 LED, collegato al canale D/A del PCF8591;
- 1 Sensore di temperatura, collegato ad un canale A/D del PCF8591;
- 1 Fotoresistenza, collegata ad un canale A/D dell'integrato di data-acquisition PCF8591 on board.

PERIFERICHE DI COMUNICAZIONE

Dopo aver esaminato in dettaglio i vari nodi per la gestione del GPIO, passiamo adesso all'analisi delle più comuni periferiche della Raspberry Pi che possiamo accedere tramite Node-RED. Come abbiamo visto nella parte introduttiva dell'articolo, la Raspberry Pi dispone di diverse periferiche di comunicazione, che permettono di interfacciare diversi tipi di dispositivi, come ad esempio sensori, attuatori o anche PC o altri SBC.
Per ovvie ragioni di spazio non è possibile effettuare una trattazione esaustiva, quindi ci limiteremo a trattare alcune delle periferiche più comuni e che ci saranno utili per lo sviluppo dei nostri progetti pratici anche nel prosieguo del corso.

Seriale

Cominciamo dalla periferica di comunicazione per eccellenza, ossia la porta seriale o porta UART (Universal Asyncronous Receiver and Transmitter). La UART è una porta di comunicazione asincrona, di tipo full-duplex (ossia è possibile ricevere e trasmettere allo stesso tempo, dal momento che è dotata di due linee di trasmissione, RX e TX), ed opera nativamente a livello TTL (ossia con dinamica 0-5V). La porta seriale viene spesso utilizzata con diversi standard di livello fisico, come RS-232, RS-422 e RS-485, che permettono di ampliare le distanze di comunicazione (ad esempio tramite l'adozione di tecniche differenziali nella trasmissione del segnale) o migliorare l'indirizzamento dei dispositivi. La Raspberry Pi è nativamente dotata di due UART: la mini UART (/dev/ttyS0) e la PL011 UART (/dev/ttyAMA0). Per impostazione predefinita, la mini UART è mappata sui pin GPIO 14 e 15 (rispettivamente TXD ed RXD) mentre la PL011 è dedicata al modulo Bluetooth on-board. Ci serviremo quindi della mini UART come

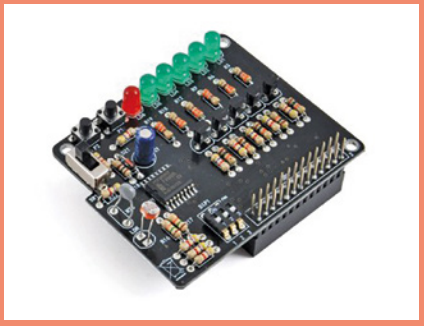


Fig. 8 - Raspberry Pi I/O HAT FT1060M.

Fig. 9 - Nodi del modulo serial port.

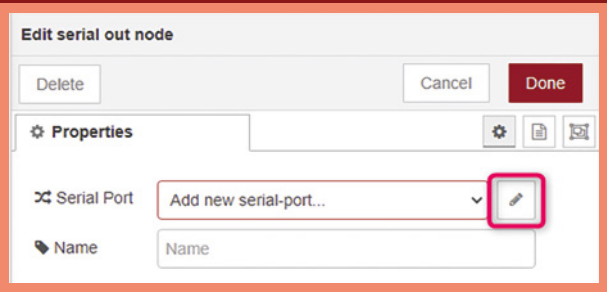


Fig. 10 - Finestra delle proprietà del nodo Serial Out.

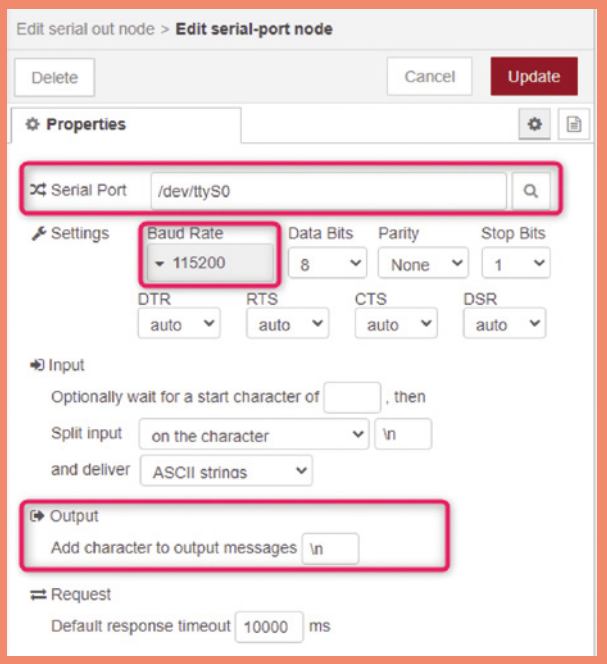


Fig. 11 - Configurazione del nodo serial-port.

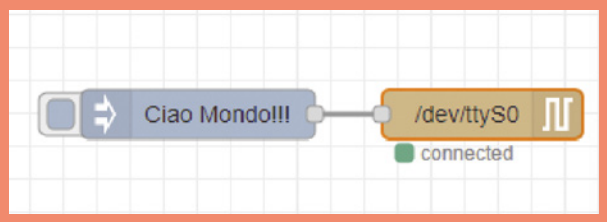


Fig. 12 - Test Seriale.

interfaccia seriale per i nostri test, ma prima di poterla utilizzare con Node-RED occorre disabilitare la console, che per impostazione predefinita è attiva su questa interfaccia di comunicazione e, di conseguenza, ci impedirebbe di utilizzare questo dispositivo da Node-RED. La console sulla mini UART può essere disabilitata utilizzando l'utility raspi-config (sudo raspi-config) o, più semplicemente, editando il file cmdline.txt.
Per editare il file cmdline.txt, aprite una shell (ad esempio tramite connessione SSH) e digitate il comando:

```
sudo nano /boot/cmdline.txt
```

Una volta aperto il file con nano editor, rimuovete la seguente sezione:

```
console=serial0,115200
```

ed effettuate un reboot.

```
sudo reboot
```

A questo punto la mini UART è scollegata dal terminale e possiamo utilizzarla. Node-RED offre il modulo serial port per la gestione della seriale (*https://flows.nodered.org/node/node-red-node-serialport*), composto da tre nodi (Fig. 9):

- **Serial In**: permette di ricevere dati dalla porta seriale.
- **Serial Out**: permette di inviare dati alla porta seriale.
- **Serial Request**: permette di inviare un dato e si mette in attesa della risposta (combinazione del nodo serial out – serial in).

A questi si aggiunge il nodo di configurazione serial-port. Il nodo di configurazione serial-port permette di configurare il comportamento della porta seriale, impostando valori come il baud rate, il numero di data bits e stop bits, la parità, ecc. Per testare il funzionamento della porta seriale dobbiamo dotarci di un convertitore USB-Seriale e collegarlo alla Raspberry Pi; Occorre collegare i pin 14 e 15 (con una connessione di tipo crossed), oltre ad un pin di GND.
Una volta fatto il collegamento possiamo posizionare il nodo Serial out sul workspace e cliccare due volte sul tasto sinistro per aprine la finestra delle proprietà (Fig. 10).
A questo punto clicchiamo sull'icona a forma di matita a fianco del drop down menu "Serial Port".
Questo permetterà di aprire la schermata di Fig. 11, dalla quale è possibile configurare la porta seriale. Selezioniamo come porta /dev/ttyS0 (è possibile cliccare sul pulsante a forma di lente di ingrandimento per ottenere una lista delle interfacce seriali disponibili), impostiamo il baud-rate a 115200 baud e nella sezione output indichiamo '\n' come carattere da aggiungere ad ogni stringa (in modo da avere un ritorno a capo automatico).
Infine clicchiamo su Update e Done per confermare la configurazione e l'associazione del nodo Serial out con il nodo di configurazione serial-port appena creato.
Ora la nostra porta seriale è configurata, per utilizzarla in trasmissione è sufficiente collegare un nodo inject con la stringa che vogliamo inviare sulla porta seriale. Nell'esempio di Fig. 12 abbiamo inviato la stringa "Ciao Mondo!!!" sulla porta seriale, se apriamo una finestra di terminale su un PC collegato al convertitore USB-seriale che abbiamo

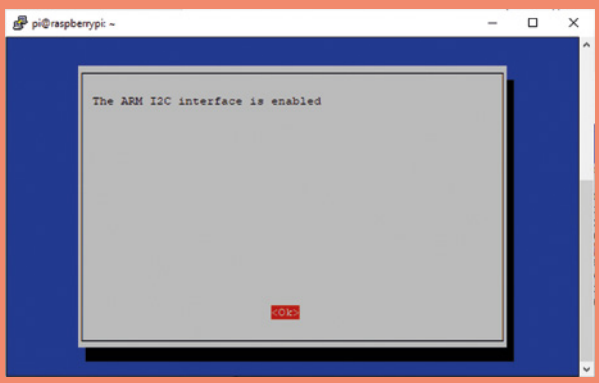


Fig. 13 - Attivazione della periferica I2C su Raspberry Pi.

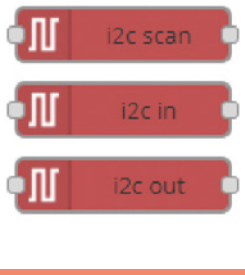


Fig. 14 - Nodi per la gestione dell'interfaccia I2C tramite Node-RED.

Fig. 15 - Nodo per la gestione dei sensori di temperatura DS18B20.

usato per interfacciare la porta UART della Raspberry Pi dovremmo ricevere la stringa inviata.

I²C

La Raspberry PI è dotata anche di una periferica I²C, assegnata per impostazione predefinita ai GPIO 2 e 3 (SDA e SCL, rispettivamente). L'I²C è una interfaccia molto utile, infatti si tratta di un protocollo seriale sincrono half-duplex, di tipo master-slave multi-drop. Questo significa che, a differenza della seriale asincrona, può indirizzare diversi dispositivi, agendo proprio come un piccolo bus di comunicazione. Tramite questa interfaccia possiamo collegare alla nostra Raspberry Pi una moltitudine di sensori ed attuatori, controllandoli facilmente tramite Node-RED.
Per utilizzare l'I²C la prima operazione da eseguire è quella di abilitare l'interfaccia. Questo può essere fatto semplicemente dall'utility raspi-config (suro raspi-config), selezionando l'opzione "Interface Options" e successivamente l'opzione "I2C". Ci verrà chiesto se vogliamo abilitare la periferica, domanda alla quale dovremo rispondere affermativamente e ad operazione conclusa dovremmo vedere un messaggio come quello di Fig. 13 che conferma l'attivazione della periferica I²C (al termine dell'operazione è necessario eseguire un reboot).
Una volta abilitata la periferica dovremo installare un modulo per la gestione dell'interfaccia I²C-Bus sotto Node-RED. Uno dei moduli più versatili, in quanto del tutto generico è il modulo node-red-contrib-i2c (*https://flows.nodered.org/node/node-red-contrib-i2c*), che come di consueto potremo installare tramite l'opzione "Manage palette → Install". Questo modulo contiene tre nodi, come evidenziato in Fig. 14, che sono:

- I2c Scan: permette di effettuare una scansione del bus I2C alla ricerca di dispositivi presenti, elencandoli per indirizzo in un array fornito come risposta.
- I2c In: permette di effettuare una richiesta di lettura sul bus I2C.
- I2c Out: permette di effettuare una richiesta di scrittura sul bus I²C.

Oltre a questo set di nodi generico sono presenti diversi altri nodi per la gestione di dispositivi specifici, che semplificano ulteriormente la gestione, sollevando l'utilizzatore anche dall'onere di inviare la sequenza di comandi necessari all'interrogazione di un determinato dispositivo. Tuttavia, data la quantità di dispositivi I²C presenti sul mercato, risulta molto utile avere a disposizione anche questo set generico.
Avremo modo di utilizzare il bus I²C in seguito nel corso delle altre puntate, per adesso è sufficiente sapere come attivarlo e quali sono i nodi per gestirlo.

1-Wire

L'ultima interfaccia che analizziamo in questa puntata è l'interfaccia 1-wire. Come il nome suggerisce, la caratteristica principale di questa semplice interfaccia è quella di consentire la gestione di dispositivi esterni tramite un singolo collegamento (più ovviamente gli estremi della tensione di alimentazione). Sebbene non esistano moltissimi dispositivi compatibili con questo standard, il basso costo, la semplicità di gestione e le esigue risorse necessarie per gestirlo (un singolo pin da sacrificare per la connessione), lo rendono spesso un'ottima scelta implementativa.
Come nel caso precedente, per impostazione predefinita la Raspberry Pi ha un'interfaccia 1-wire disponibile, allocata sul GPIO 4, che va attivata, esattamente come

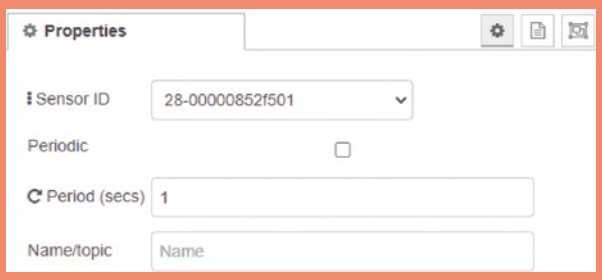


Fig. 16 - Finestra delle proprietà del nodo sensor-ds18b20.

abbiamo visto fare prima per l'I2C. Lanciamo quindi raspi-config, e selezioniamo nuovamente "Interface Option", ma stavolta selezioniamo l'opzione "1-Wire" dal menu di gestione delle interfacce, confermando l'abilitazione. Anche in questo caso, al termine della procedura avremo un messaggio di conferma e sarà necessario eseguire un reboot del sistema.
Dal punto di vista di Node-RED, non esiste ad oggi un set di nodi generico per la gestione della 1-wire, ma piuttosto una serie di nodi specifici per i singoli dispositivi.
Ad esempio, uno dei dispositivi 1-wire più diffusi è il sensore di temperatura Dallas Seminconductor DS18B20. Questo dispositivo, incapsulato in un case plastico tipo TO-92 (a tre terminali) è facilmente reperibile, ha un bassissimo costo ed una elevata precisione di misura; inoltre se ne possono collegare diversi sullo stesso bus 1-wire, in quanto ogni dispositivo è identificato da uno specifico serial number univoco. Esistono svariati nodi per la gestione di questo dispositivo in Node-RED; uno molto efficace è quello fornito dal modulo node-red-contrib-sensor-ds18b20 (*https://flows.nodered.org/node/node-red-contrib-sensor-ds18b20*), illustrato in Fig. 15.
Il nodo è di semplicissimo utilizzo, una volta installato e portato sul workspace, facendo doppio click possiamo accedere alla finestra delle proprietà (Fig. 16). Qui verranno già elencati, ordinati per serial number, tutti i sensori DS18B20 che la scheda Raspberry Pi è riuscita a rilevare (ovviamente dovremo averne collegato almeno uno per vedere qualcosa).
A questo punto il nodo può essere configurato per eseguire letture periodiche ad un certo rate, oppure naturalmente esegue una lettura ogni volta che viene ricevuto un messaggio sulla porta di ingresso. Il risultato della lettura, già convertito in Celsius, verrà inserito nel campo payload del messaggio in output, come di consueto.

ESEMPIO PRATICO: LETTURA DI UN SENSORE DS18B20 TRAMITE INTERFACCIA SERIALE

Arrivati a questo punto, possiamo sfruttare le conoscenze acquisite per implementare una semplice applicazione: ciò che vogliamo realizzare stavolta è un dispositivo che acquisisca una lettura da un sensore di temperatura, nel nostro caso il popolare integrato DS18B20, e la invii ad un terminale seriale in esecuzione su Personal Computer ogni volta che quest'ultima viene richiesta.
Allo scopo utilizzeremo la stringa "Temp?" per identificare una richiesta valida, mentre qualsiasi altra stringa verrà ritenuta non valida.
Per l'implementazione ci occorrono due blocchi serial: serial in (per ricevere la richiesta) e serial out (per fornire la risposta). Inoltre utilizzeremo un nodo function e un nodo switch per verificare che la richiesta ricevuta sia effettivamente formattata correttamente e altri 2 nodi function per formattare le due risposte (positiva con lettura di temperatura e negativa). Infine ci occorre il nodo sensor-ds18b20 visto in precedenza. In Fig. 17 è rappresentato il flow che implementa il nostro semplice programma: analizziamo nodo per nodo il flow al fine di comprenderlo meglio. Il Nodo Serial IN riceve la stringa in ingresso dalla linea seriale. Il configuration node di questo blocco è del tutto simile a quello visto nel paragrafo dedicato alla porta seriale. Il nodo function GetRequest ha il compito di verificare se la richiesta sia stata correttamente formattata: ci aspettiamo di ricevere "Temp?\n", in quanto il con-

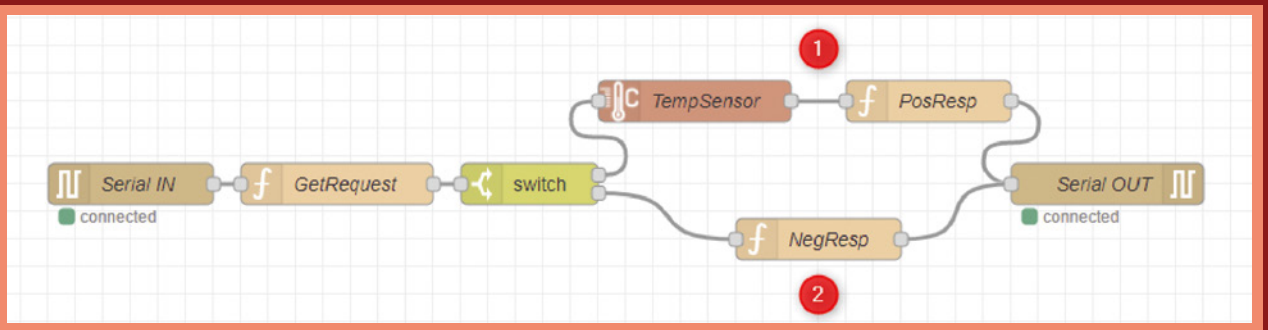


Fig. 17 - Flow dell'esempio pratico.

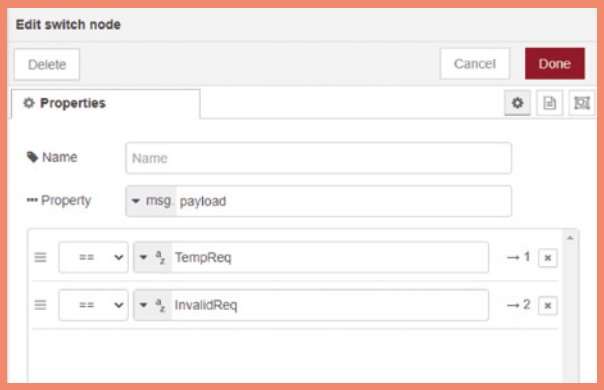


Fig. 18 - Regole del nodo switch.

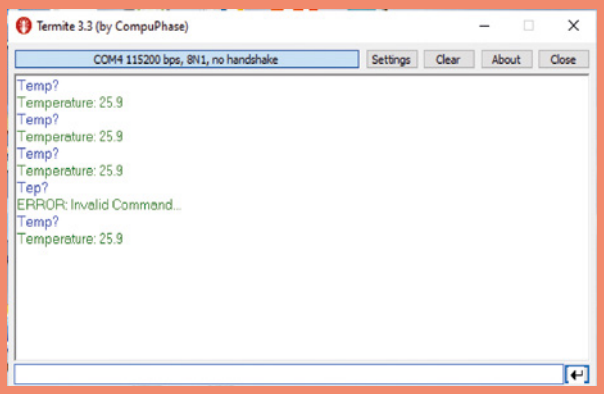


Fig. 19 - Letture di temperatura tramite terminale seriale.

figuration node della porta seriale si aspetta "\n" come carattere di terminazione di una stringa. Il controllo della formattazione è fatto con il codice javascript illustrato qui di seguito:

```
if (msg.payload == "Temp?\n")
{
    msg.payload = "TempReq";
}
else
{
    msg.payload = "InvalidReq";
}
return msg;
```

che riguarda il blocco GetRequest.
Quindi il nodo valuta la stringa e ritorna msg.apyload contenente "TempReq" se la richiesta è corretta o "InvalidReq" altrimenti. A questo punto ci serviamo di un nodo Switch per creare due path alternativi nei due casi. Il nodo switch è stato configurato con le regole visibili in Fig. 18.
A questo punto il flow ha una biforcazione: la parte marcata con '1' è il flusso di risposta positivo, quindi qui viene letta la temperatura dal sensore DS18B20 e il nodo function PosResp formatta la risposta tramite il codice javascript:

```
msg.payload = "Temperature: " + msg.payload;
return msg;
```

che riguarda il blocco PosResp.
Siccome non è necessaria alcuna periodicità nella lettura (viene fornita la temperatura solo su richiesta), l'opzione periodic read è stata disabilitata nel nodo sensor-ds18b20. In questa configurazione il nodo fornisce la temperatura solo quando viene triggerato da un messaggio in ingresso.
La parte marcata invece con '2' è il flusso di risposta negativo, quindi non viene letto il sensore e il nodo function NegResp ritorna una stringa contenente un messaggio di errore, tramite il seguente codice javascript:

```
msg.payload = "ERROR: Invalid Command...";
return msg;
```

relativo al blocco NegResp.
I flussi si ricongiungono convergendo sul nodo Serial OUT, che invia la risposta sulla linea seriale (il configiration node referenziato è lo stesso del nodo Serial IN).
In Fig. 19 è possibile vedere le schermate relative ad alcune letture effettuate tramite il terminale seriale.
Per la realizzazione dell'esempio abbiamo utilizzato la sonda DS18B20 2846-SONDADS18B20 e il convertitore seriale 2846-CAVOUSBSERIALE, entrambi distribuiti da Futura Elettronica. Nel collegare il convertitore USB-Seriale si tenga presente che è sufficiente collegare solo i pin TX, RX e GND, se la Raspberry Pi è già alimentata.
La sonda DS18B20 va collegata al GPIO 4 ed alimentata tramite uno dei pin a 3,3V della Raspberry Pi. Inoltre sul GPIO 4 va attivata la resistenza di pull-up interna.

CONCLUSIONI

Bene, siamo così arrivati alla conclusione di questa terza puntata, nella quale abbiamo visto come controllare le periferiche della scheda Raspberry Pi tramite Node-RED. Siamo partiti dalla semplice gestione del singolo GPIO per arrivare fino alle periferiche di comunicazione, come UART, I²C e 1-wire, per le quali Node-RED offre diversi moduli di gestione.
Nella prossima puntata introdurremo un importantissimo elemento di Node-RED: si tratta della Dashboard, che permette di creare user interfacce (UI) accessibili via web per implementare i nostri progetti.