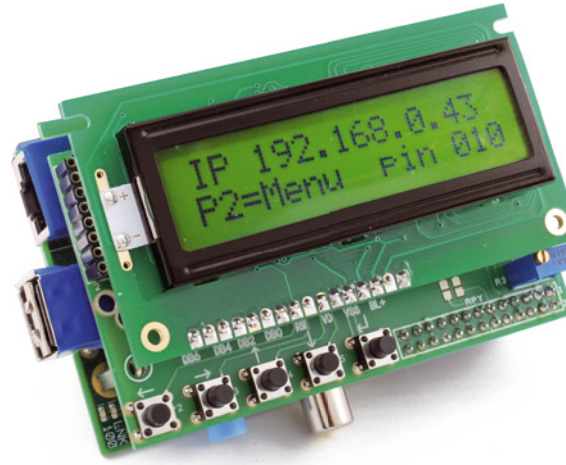
SHIELD DI ESPANSIONE LCD PER RASPBERRY PI (cod. FT1074K)



Shield appositamente realizzata per Raspberry Pi (non inclusa) che permette di realizzare un'interfaccia di controllo esterna per le proprie applicazioni senza necessità di tenere collegati costantemente video, tastiera e mouse. L'unità che utilizza l'integrato MCP23017 per interfacciarsi al GPIO di Raspberry Pi (sfruttando unicamente i pin necessari alla comunicazione I²C) è progettata in modo da poter essere impilata su un'altra shield consentendo l'utilizzo in contemporanea. Sullo shield sono presenti 5 pulsanti ed un connettore a pettine che rende disponibile oltre alle linee 5V, 3V3 e GND anche i pin GPA5, GPA6 e GPA7, del banco GPA dell'integrato MCP23017. Le uscite INTA e INTB sono collegate mediante due ponticelli (JB e JA) ai GPIO27 e GPIO17 di Raspberry Pi, in modo da renderli disponibili ad eventuali applicazioni.

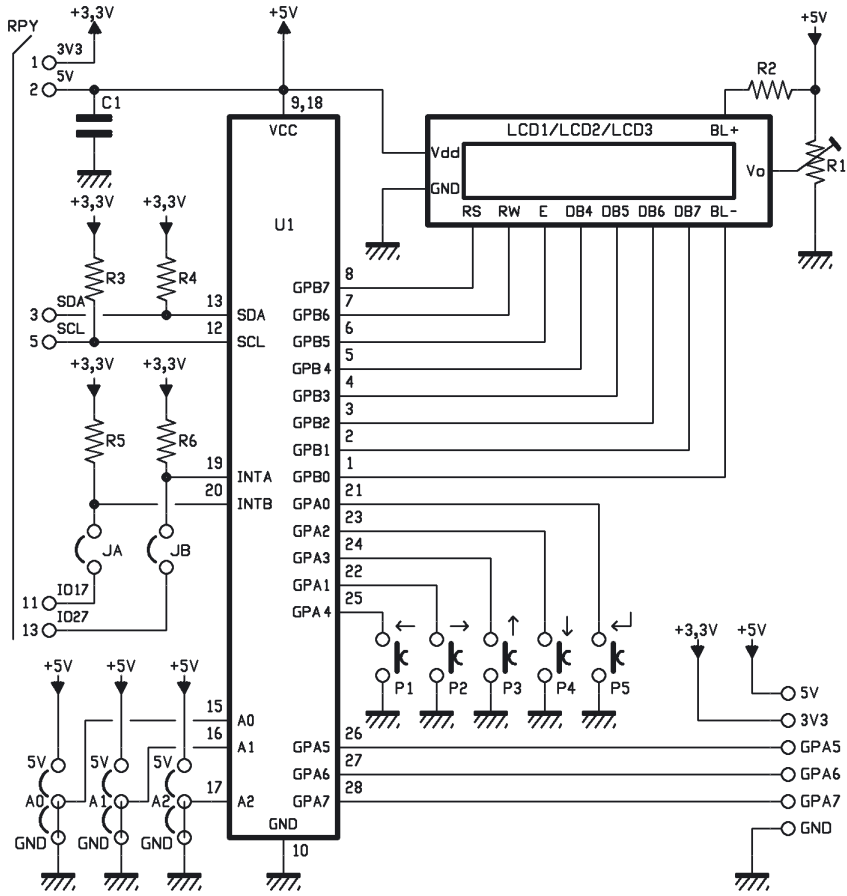
L'alimentazione a 5V viene prelevata dal piedino 2 del connettore GPIO di Raspberry Pi mentre la massa è collegata al piedino 6. Il trimmer multigiri presente sulla scheda permette di regolare il contrasto del display LCD.

Il kit comprende un display LCD 16x2 retroilluminato con caratteri neri su sfondo verde (cod. Futura ACM1602B-FL-YBW) tuttavia lo shield è provvisto di diversi connettori in grado di accogliere differenti tipi di LCD e in particolare i seguenti modelli (acquistabili sempre da Futura Elettronica): LCD 16x2 retroilluminato bianco/nero (cod. LCD16x2WB), LCD 8x2 retroilluminato blu (cod. LCD8x2BN).

Realizzazione pratica

Lo shield è di facile costruzione poiché è costituito da un semplice stampato contenente pochi componenti. Per un semplice montaggio si consiglia di fare riferimento al piano di montaggio riportato di seguito. Montare sulla scheda le 5 resistenze

Schema elettrico

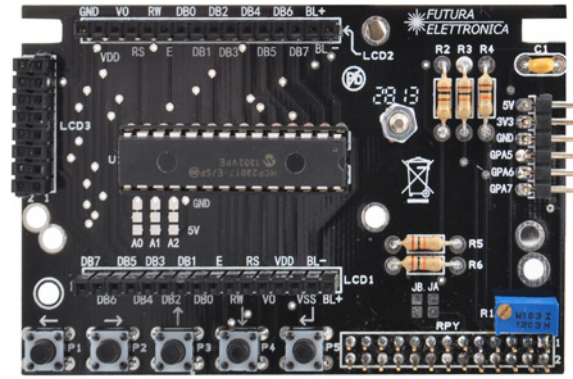


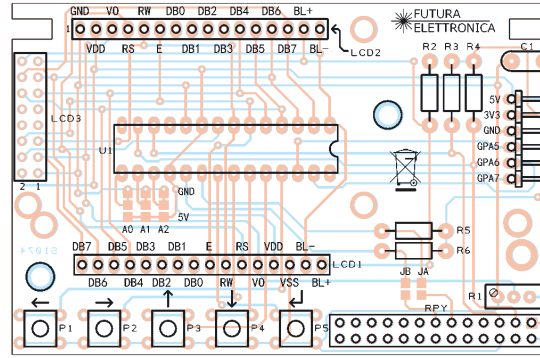
(R2÷R6), il condensatore multistrato C1 e i 5 micropulsanti (P1÷P5) quindi saldare i relativi terminali alle piazzole della scheda utilizzando un saldatore da non più di 30 watt. Inserire e saldare un pin-strip maschio a 6 poli, 90° in corrispondenza delle piazzole con serigrafia "5V, 3V3, GND, GPA5, GPA6 e GPA7" come visibile nell'immagine riportata nel piano di montaggio. Procedere col montaggio dello zoccolo per IC (14+14 pin), del trimmer multigiri e dei pin-strip femmina LCD1, LCD2 (16 pin) e LCD3 ricordando che quest'ultimo si ottiene affiancando 2 unità da 8

Piano di montaggio

Elenco Componenti:

C1: 100 nF multistrato
R1: Trimmer multigiri
10 kohm
R2: 100 ohm
R3: 10 kohm
R4: 10 kohm
R5: 10 kohm
R6: 10 kohm
P1: Microswitch
P2: Microswitch
P3: Microswitch
P4: Microswitch
P5: Microswitch
U1: MCP23017-E/SP
Varie:
- Display LCD (ACM1602B-FL-YBW)
- Torretta M/F 18 mm
- Vite 8 mm 3 MA
- Dado 3 MA
- Zoccolo 14+14
- Strip maschio 6 poli 90°
- Strip F 8 poli (2 pz.)
- Strip F 16 poli (2 pz.)
- Connettore F 26 poli per RaspberryPi
- Circuito stampato





pin. Inserire dal lato saldature, nel foro bordato di bianco presente sulla scheda (sopra al simbolo "Bidone"), la parte filettata del distanziale esagonale da 18mm quindi avvitare un dado M3 per fissarlo al cs. Ora inserite nelle piazzole identificate con la serigrafia "RPY" il connettore femmina a 26 poli (con pin lunghi) previsto per il collegamento alla scheda Raspberry, rivolgendo verso il basso il lato femmina.

Prima di saldare i relativi pin conviene posizionare correttamente il connettore a 26 pin procedendo come di seguito descritto:

- Applicare lo shield sulla scheda Raspberry, facendo appoggiare su quest'ultima il distanziale esagonale, quindi unire tra di loro i due connettori a 26 pin.

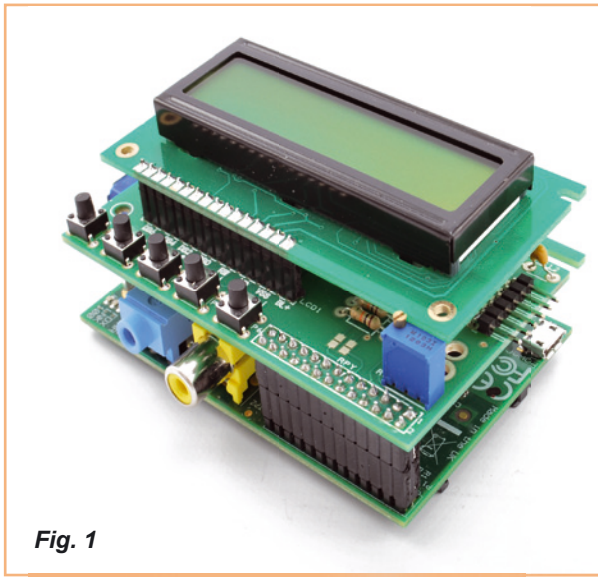


Fig. 1

- Fissare lo shield alla Raspberry avvitando nel distanziale esagonale la vite M3 fornita nel kit.
- Ora che le schede sono allineate e correttamente distanziate l'una dall'altra, è possibile saldare sullo shield i terminali maschio del connettore a 26 poli. Per ultimo inserire l'integrato U1 nel relativo zoccolo e montate il display LCD (***Fig. 1***). Terminate le saldature verificate che non vi siano falsi contatti o cortocircuiti.

Utilizzo pratico dello shield

Dopo aver montato lo shield LCD sulla scheda Raspberry Pi, collegare periferiche, rete e alimentazione a Raspberry Pi quindi fornire tensione. Per comunicare con l'integrato MCP23017 è necessario utilizzare il bus I²C, di conseguenza bisogna attivare il modulo di gestione del bus I²C che nell'installazione predefinita di Raspbian è disabilitato.
Per chi è nuovo all'ambiente Raspbian - il sistema operativo GNU/Linux di Raspberry Pi - si consiglia di consultare il numero 173 della rivista "***Elettronica In***" nonché il libro "***Raspberry Pi il mio primo Linux embedded***" (acquistabile presso Futura Elettronica cod. RASPBOOK1). Ora bisogna abilitare il driver per la gestione del bus I²C, installare la libreria in Python per la gestione dell'LCD e realizzare un primo programma di prova per vedere che tutto funzioni a dovere.
Di seguito è riportato brevemente il processo per aggiornare il sistema operativo ed abilitare il driver per il bus I²C.
Innanzitutto dare i comandi (come utente "root"):

apt-get update
apt-get upgrade

Per rendere utilizzabile il modulo di gestione del bus I²C è necessario toglierlo dalla blacklist e poi "aggiungerlo" all'insieme dei moduli conosciuti dal kernel. Aprire il file di configurazione che contiene l'elenco dei moduli blacklisted (oscurati), con il comando (***Fig. 2***):

nano /etc/modprobe.d/raspi-blacklist.conf

Nano è un editor di testo minimale che funziona in ambiente terminale.
Eliminare il modulo I²C dalla blacklist cancellando la riga o commentandola con un "**#**" (***Fig. 3***).

Premere Ctrl-X e poi Y alla richiesta di salvare il file dopo le modifiche.
Eseguire un reboot per rendere effettive le modifiche. Ora bisogna fare in modo che il modulo "liberato" venga caricato e diventi parte integrante del kernel. Per questa operazione sono disponibili due possibilità: la prima permette di caricare il modulo a comando, e ha validità per tutto il tempo nel quale Raspberry Pi rimane acceso. Al boot successivo il modulo dovrà essere ricaricato a comando. La seconda consente di caricare il modulo direttamente al boot del sistema operativo e renderlo disponibile alle applicazioni subito dopo il boot, condizione indispensabile in un sistema server unattended.
La prima possibilità richiede l'utilizzo del comando modprobe.
Scrivere (***Fig. 4***):

modprobe i2c-dev

Si può vedere il buon esito dell'attivazione dei driver con il comando che mostra la lista di tutti i moduli installati (***Fig. 5***):

lsmod

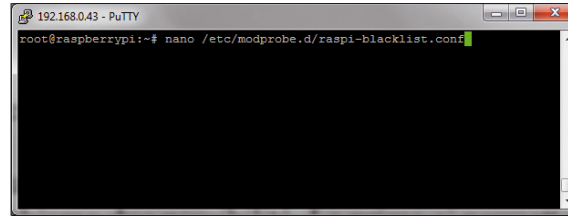


Fig. 2

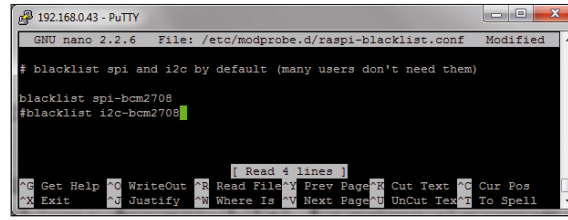


Fig. 3

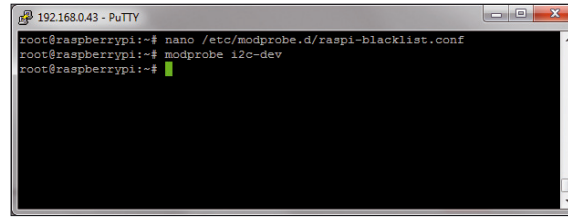


Fig. 4

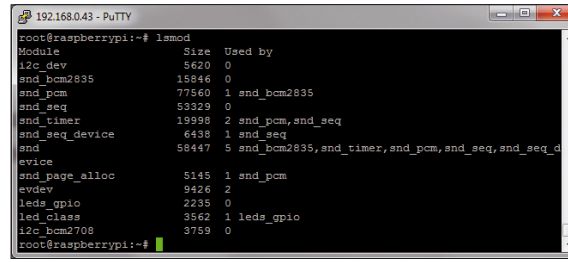


Fig. 5

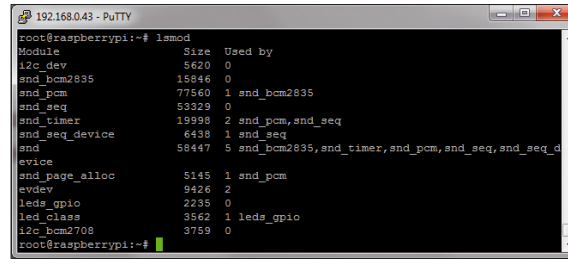


Fig. 6

Dato che in GNU/Linux tutto (o quasi) è un file, se si entra nella cartella /dev si vedranno apparire i file di collegamento ai device i2c-0 ed i2c-1 (***Fig. 6***).
Il comando modprobe permette di caricare e scaricare i moduli a runtime e mantiene i suoi effetti fintanto che Raspberry Pi resta acceso. In caso di spegnimento, o anche solo di reboot, il proprio modulo dovrà essere ricaricato manualmente.
Questa condizione non è adatta a funzionare con un'applicazione stand alone, che deve funzionare in modo automatico. Il comando modprobe, con l'opzione remove, può essere utilizzato anche per disattivare un modulo caricato in precedenza.

modprobe -r i2c-dev

Se si desidera che il modulo venga caricato all'accensione di Raspberry Pi è necessario abilitare il caricamento permanente del driver I²C che si realizza modificando opportunamente il file di configurazione /etc/modules, contenente la lista dei driver da caricare al momento del boot.

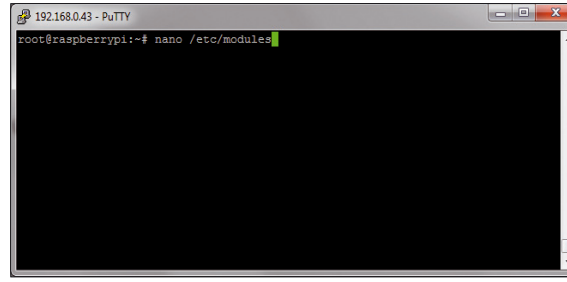


Fig. 7

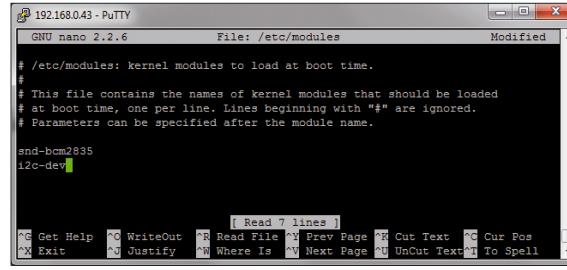


Fig. 8

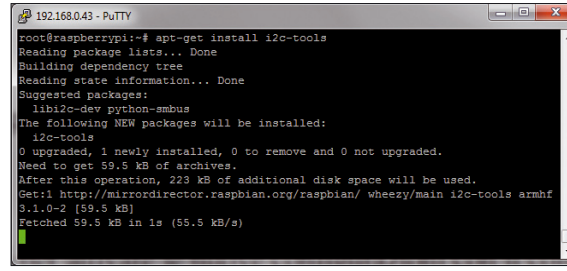


Fig. 9

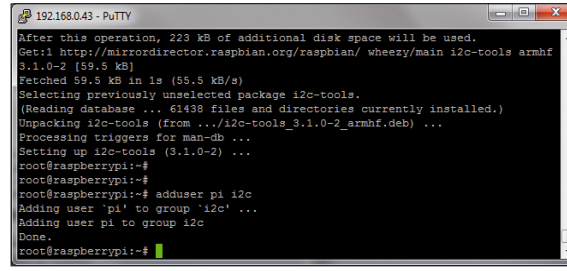


Fig. 10

In caso contrario bisogna ricordarsi di caricare il modulo ad ogni accensione con modprobe. Per modificare il file è possibile usare il comando:

nano /etc/modules

ed aggiungere una nuova linea al file di configurazione che contiene (***Fig. 7***)

i2c-dev

Premete CTRL X e poi y per salvare le modifiche al file ed uscire (***Fig. 8***).
Ora installare il pacchetto i2c-tools che fornisce una serie di funzioni utilizzabili a linea di comando per verificare il funzionamento del bus I²C (***Fig. 9***):

apt-get install i2c-tools

Aggiungere il proprio utente pi al gruppo I²C (***Fig. 10***):

adduser pi i2c

Eseguire il reboot di Raspberry Pi per attivare le nuove configurazioni con il comando

reboot

Dopo che Raspberry Pi si è riavviato e ci si è riconnessi con Putty o Kitty, verificare se sul bus I²C è visibile il convertitore ADC con il comando:

i2cdetect -y 0 per Raspberry Pi rev. 1
oppure
i2cdetect -y 1 per Raspberry Pi rev. 2

Si dovrà ottenere un risultato simile a quello visibile in ***Fig. 11*** dove l'indirizzo 0x20 identifica l'integrato MCP23017.
Ora è possibile installare la libreria di supporto all'LCD. Verrà utilizzata la libreria messa a disposizione con licenza BSD da Adafruit Industries nel repository GitHub. Il procedimento migliore per scaricarla è utilizzare lo strumento di gestione delle versioni di software "git". Per installare "git" utilizzare il comando:

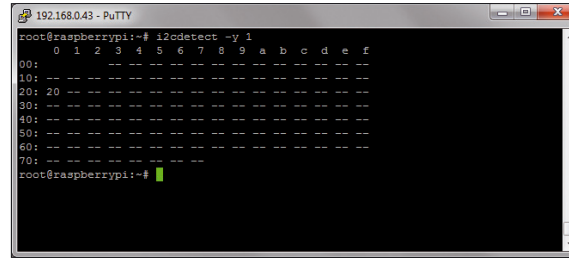


Fig. 11

apt-get install git

poi andare alla cartella home con il comando:

cd /home

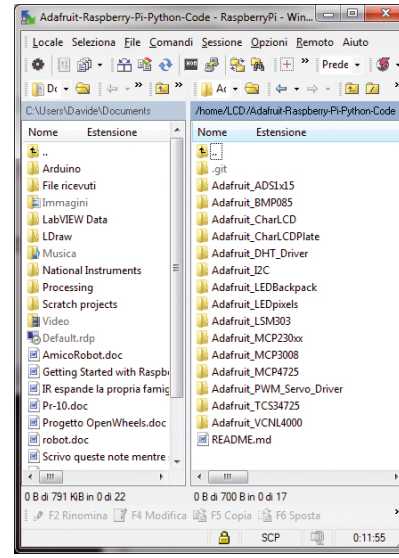


Fig. 12

creare una cartella per il proprio progetto, per esempio:

mkdir LCD

posizionarsi nella cartella con il comando:

CD LCD

quindi scaricare le librerie Python realizzate da Adafruit con il comando:

git clone https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git

infine posizionarsi nella cartella che contiene la libreria per la gestione dell'LCD con i comandi:

cd Adafruit-Raspberry-Pi-Python-Code

È possibile vedere il contenuto della cartella posizionandosi sulla stessa con WinSCP (***Fig. 12***).
Posizionarsi nella cartella che contiene la libreria Adafruit_CharLCDPlate con il comando:

cd Adafruit_CharLCDPlate

Installare la libreria per la gestione del protocollo I²C con il linguaggio Python:

apt-get install python-smbus

A questo punto si è quasi pronti a verificare il funzionamento dello shield LCD.
Nella libreria Adafruit_CharLCDPlate, i pin GPA6 e GPA7 sono utilizzati per la gestione dei colori di un LCD a colori RGB. In questo caso, utilizzando LCD monocromatici, verranno lasciati disponibili i suddetti pin. Per poterli utilizzare ad esempio, come ingressi è necessario modificare la libreria.
La libreria contiene il codice che esprime la classe Adafruit_CharLCDPlate per la gestione dei display LCD. Nel costruttore della classe è stata modificata l'impostazione iniziale dei registri per rendere i pin GPA5, GPA6 e GPA7 dei pin di ingresso, con resistenza di pull up interna e funzionanti a logica invertita (valore "1" quando il pin è a massa).
È possibile scaricare la libreria modificata direttamente dalla scheda del prodotto FT1074K disponibile su *www.futurashop.it*.
In particolare sono state modificate le seguenti linee:

riga 99 [0b11111111, # IODIRA R+G LEDs=outputs, buttons=inputs orig. 0b00111111
riga 101 0b11111111, # IPOLA Invert polarity on button inputs orig. 0b00111111
riga 112 0b11111111, # GPPUA Enable pull-ups on buttons orig. 0b00111111

Alla riga 425 è stato modificato il metodo backlight in modo da eliminare l'utilizzo dei pin del banco GPA per la gestione del colore.

```
def backlight(self, color):
    c = ~color
#   self.porta = (self.porta & 0b00111111) | ((c & 0b011) << 6)
    self.portb = (self.portb & 0b11111110) | ((c & 0b100) >> 2)
    # Has to be done as two writes because sequential operation is off.
#   self.i2c.bus.write_byte_data(
#     self.i2c.address, self.MCP23017_GPIOA, self.porta)
    self.i2c.bus.write_byte_data(
      self.i2c.address, self.MCP23017_GPIOB, self.portb)
```

Infine (riga 436) sono stati aggiunti tre metodi per la lettura dei singoli pin "avanzati".

```
# Read and return bitmask of pin A5
def buttonpin5(self):
    return (self.i2c.readU8(self.MCP23017_GPIOA) >> 5) & 1

# Read and return bitmask of pin A6
def buttonpin6(self):
    return (self.i2c.readU8(self.MCP23017_GPIOA) >> 6) & 1

# Read and return bitmask of pin A7
def buttonpin7(self):
 return (self.i2c.readU8(self.MCP23017_GPIOA) >> 7) & 1
```

È stato preparato un programma di prova modificando il programma di test incluso nella cartella della libreria.
Il programma, premendo i pulsanti presenti sullo shield, permette di ottenere informazioni sulle performance e lo stato del sistema, sintetizzate sul display LCD.
Copiare il programma in un file di nome LCDProva.py, oppure scaricarlo dalla scheda del prodotto FT1074K disponibile su *www.futurashop.it*.
Ricordarsi di lanciare il comando:

modprobe i2c-dev

Posizionarsi nella cartella Adafruit_CharLCDPlate e lanciare il programma con il comando:

python LCDProva.py

Si dovrebbe vedere il messaggio di apertura sul display LCD e poi il menu dei pulsanti.
Se non si vede nulla o lo schermo ha contrasto eccessivo, è possibile regolarlo mediante il potenziometro R1 fino ad ottenere un risultato soddisfacente.
Come è possibile vedere, il listato è molto semplice: la maggior parte del lavoro viene svolto dai metodi forniti dalla libreria.
All'apertura del programma si importa la classe sleep, per gestire i ritardi di esecuzione, la libreria Adafruit_CharLCDPlate e la libreria subprocess. La prima istruzione del programma richiama un metodo che permette di determinare a quale revisione appartiene il Raspberry Pi in proprio possesso e imposta di conseguenza il bus I^2C, 0 o 1. Poi viene inviato il messaggio di apertura sul display, e, successivamente con un ritardo di un secondo, il messaggio di menu.
Infine, nel ciclo while vengono letti i valori dei tre pin di ingresso e poi viene intercettato quale pulsante viene premuto per eseguire la funzione corrispondente con la relativa presentazione dei risultati sul display.
I valori dei pin sono presentati in basso a destra nel display di ciascun comando.
A scopo didattico, in questo programma di esempio, si è voluto presentare il risultato di alcuni comandi che permettono di raccogliere informazioni sullo

Fig. 13

Fig. 14

Fig. 15

Fig. 16

Fig. 17

stato e/o sulle performance del sistema. Inutile ricordare che gli stessi comandi possono essere digitati nella finestra terminale come comandi normali, ottenendo i medesimi risultati.
I comandi sono eseguiti lanciando processi esterni al programma in esecuzione e recuperando il risultato alla fine dell'elaborazione.
In particolare la pressione del pulsante P1 richiama il menu; il pulsante P2 mostra l'indirizzo IP di Raspberry Pi e lo stato dei tre pin d'ingresso aggiuntivi, P3 l'utilizzo di CPU, P4 lo spazio occupato su SD Card o su disco - sia in GB, sia in percentuale - e P5 il numero di processi in esecuzione su Raspberry Pi.
Per modificare lo stato dei pin di ingresso è possibile utilizzare sperimentalmente e con molta attenzione un cavetto con i terminali femmina/femmina alle estremità.
Collegare un'estremità del cavetto al terminale a massa e l'altro terminale a uno dei pin di ingresso.
Premendo uno dei pulsanti da P2 a P5 si dovrebbe vedere lo stato dei pin in basso a destra (***Fig 13, 14, 15, 16, 17***).

L'articolo completo del progetto è stato pubblicato su: Elettronica In n. 178

A tutti i residenti nell'Unione Europea. Importanti informazioni ambientali relative a questo prodotto

Questo simbolo riportato sul prodotto o sull'imballaggio, indica che è vietato smaltire il prodotto nell'ambiente al termine del suo ciclo vitale in quanto può essere nocivo per l'ambiente stesso. Non smaltire il prodotto (o le pile, se utilizzate) come rifiuto urbano indifferenziato; dovrebbe essere smaltito da un'impresa specializzata nel riciclaggio.
Per informazioni più dettagliate circa il riciclaggio di questo prodotto, contattare l'ufficio comunale, il servizio locale di smaltimento rifiuti oppure il negozio presso il quale è stato effettuato l'acquisto.

Prodotto e distribuito da:
FUTURA GROUP SRL
Via Adige, 11 - 21013
Gallarate (VA)
Tel. 0331-799775
Fax. 0331-778112
Web site:
www.futurashop.it
Info tecniche:
supporto@futurel.com