

Mensile di elettronica applicata, attualità scientifica, novità tecnologiche.

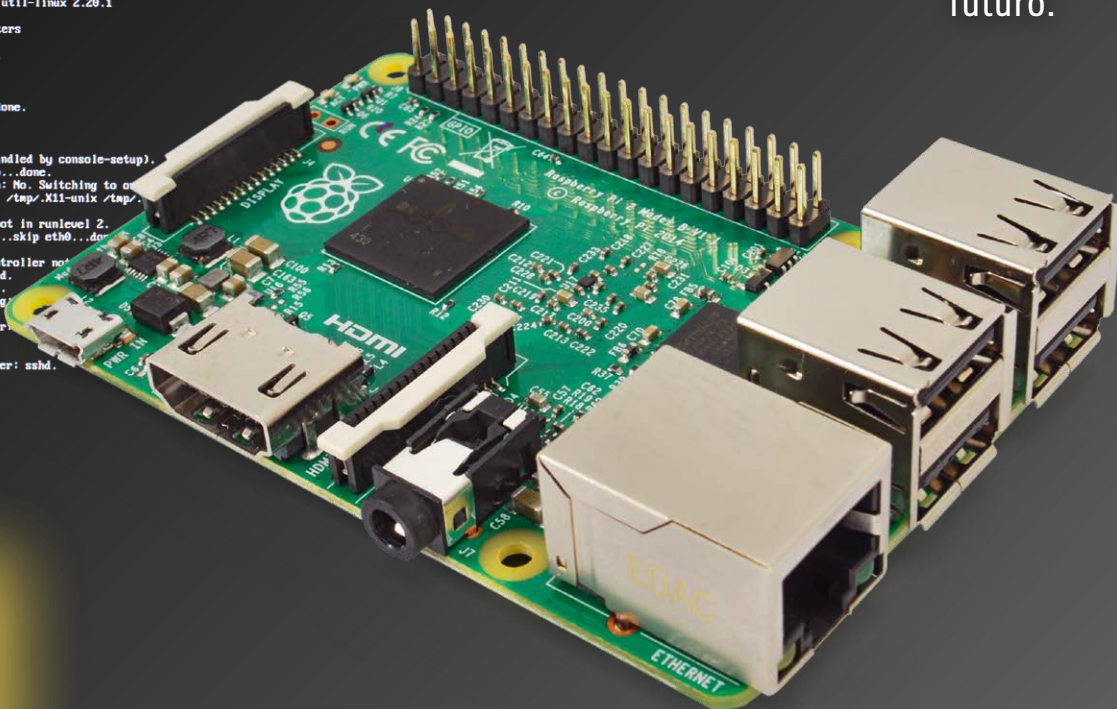
Elettronica In

www.elettronica.in.it

oltre l'elettronica



```
[ 12.517511] UFS: Mounted root (ext4 filesystem) readonly on device 179:2.
[ 12.531616] devtmpfs: mounted
[ 12.540749] Freeing unused kernel memory: 396K (c0767000 - c07ca000)
INIT: version 2.88 booting
[info] Using makefile-style concurrent boot in runlevel 3.
[....] Starting the hotplug events dispatcher: udevd[ 13.757360] udevd[1751]: starting version 175
[ ok ]
[ ok ] Synthesizing the initial hotplug events...done.
[ ok ] Waiting for /dev to be fully populated...done.
Starting fake hwclock: loading system time.
Sat Jan 31 21:25:30 UTC 2015
[ ok ] Setting preliminary keymap...done.
[ ok ] Activating swap...done.
[ 16.173495] EXT4-fs (mxcblkp2): re-mounted. Opts: (null)
[....] Checking root file system...fsck from util-linux 2.20.1
e2fsck 1.42.5 (29-Jul-2012)
/dev/mxcblkp2: clean, 65260/196224 files, 649586/784640 blocks
done.
[ 16.430026] EXT4-fs (mxcblkp2): re-mounted. Opts: (null)
[ ok ] Cleaning up temporary files.../tmp.
[info] Loading kernel module snd-bcm2835.
[ ok ] Activating iwl and md swap...done.
[....] Checking file systems...fsck from util-linux 2.20.1
dosfsck 3.0.13, 30 Jun 2012, FAT32, LFN
/dev/mxcblkp1: 32 files, 1837/7161 clusters
done.
[ ok ] Mounting local filesystems...done.
[ ok ] Activating swapfile swap...done.
[ ok ] Cleaning up temporary files....
[ ok ] Setting kernel variables...done.
[ ok ] Configuring network interfaces...done.
[ ok ] Cleaning up temporary files....
[ ok ] Setting up ALSA...done.
[info] Setting console screen modes.
[info] Skipping font and keymap setup (handled by console-setup).
[ ok ] Setting up console font and keymap...done.
[ ok ] Checking if shift key is held down: No. Switching to a
[ ok ] Setting up X socket directories.../tmp/.X11-unix /tmp.
INIT: Entering runlevel: 2
[info] Using makefile-style concurrent boot in runlevel 2.
[ ok ] Network Interface Plugging Daemon...skip eth0...done.
[info] Initializing cgroups.
[warn] Kernel lacks cgroups or memory controller not
[ ok ] Starting enhanced syslogd: rsyslogd.
Starting dhcpcd-swapfile swapfile setup...
want /var/swap-100MByte, checking existing
done.
[ ok ] Starting periodic command scheduler: cron.
[ ok ] Starting NTP server: ntpd.
[ ok ] Starting system message bus: dbus.
[ ok ] Starting OpenSSH Secure Shell server: sshd.
My IP address is 192.168.0.156
Raspbian GNU/Linux 7 raspberrypi tty1
raspberrypi login: _
```



RASPBERRY PI 2 O MEGLIO "QUATTRO"

di MARCO MAGAGNIN

Ecolo qui. Sulla scrivania, acceso e funzionante, che senza scaldarsi sta elaborando le nostre applicazioni scritte per i Raspberry Pi ... precedenti, con prestazioni da strabuzzare gli occhi. Stiamo parlando di Raspberry Pi 2. Stesso aspetto della versione B+ ma un cuore con quattro cpu ed un GB di memoria. Chi vive nel mondo dell'informatica da molti anni se l'aspettava, nonostante le

affermazioni della fondazione di qualche tempo fa che indicavano una certa "stabilità" di produzione sui modelli ormai consolidati. Ovviamente questa posizione appariva come un segno di continuità per un prodotto destinato al mondo "education" per vocazione ... di nascita e per statuto. Nella realtà molti segni percepibili nell'aria lasciavano abbastanza perplessi, ed invitavano a tenere dritte

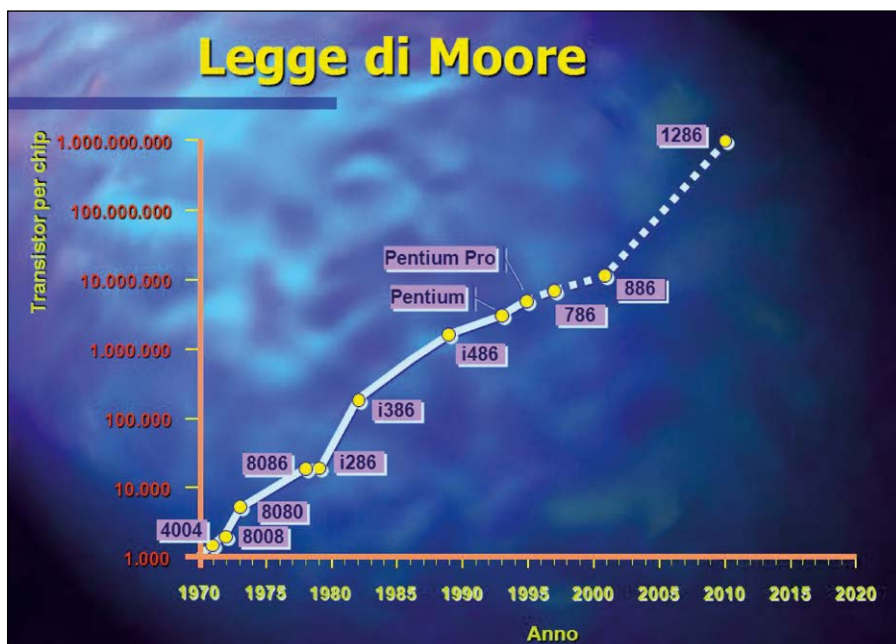


Fig. 1

le antenne. Di fondo, per chi ha la mia età, il ricordo delle affermazioni del Sig. Moore, cofondatore di Intel, che nel 1965 aveva constatato che dal 1959 ad allora circa ogni 18 mesi il numero di transistor contenuto nei microcontrollori raddoppiava e, di conseguenza, aumentava proporzionalmente la loro capacità di elaborazione, mentre i costi si riducevano in modo inversamente proporzionale. Come si può vedere in **Fig. 1** tale andamento si è verificato sino ad oggi, con scarsa varianza, tanto che le sue costatazioni sono state promosse al rango di legge, quella nota come "prima legge di Moore".

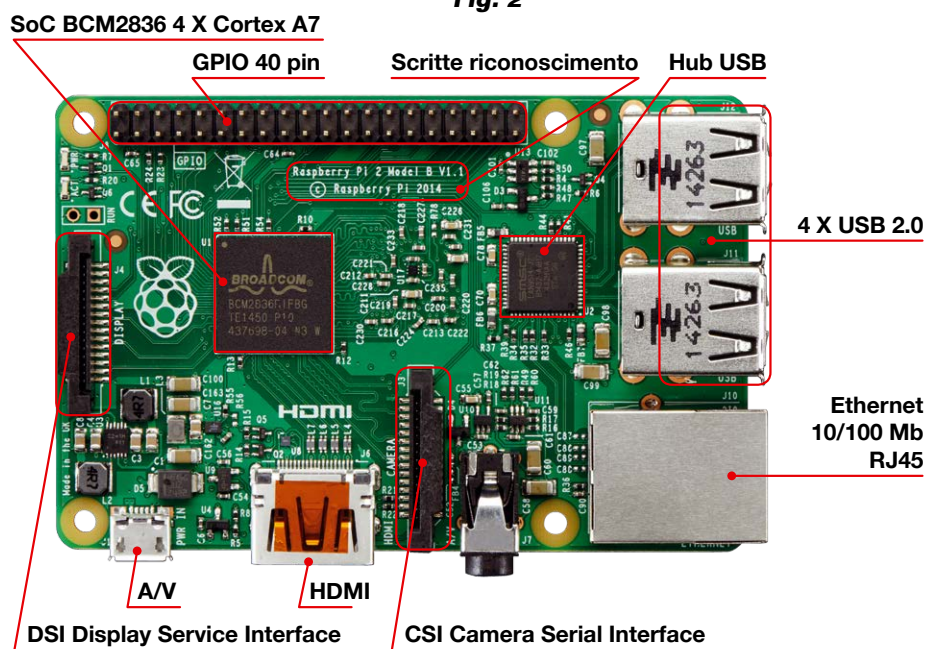
Quindi, a rigore di logica, anche Raspberry Pi avrebbe dovuto farci i conti. A rafforzare la convinzione anche il continuo apparire di microcomputer, "cloni" di "Raspberry Pi", con CPU più potenti e con sistemi operativi e applicazioni scarsamente compatibili tra di loro. Ultimamente, anche da parte della fondazione Raspberry Pi, i rilasci delle distribuzioni si sono fatti più frequenti, con modifiche alla partizione di "root" ... sospette, come l'apparire dei file di configurazione hardwa-

re .dtb, dei quali parleremo in seguito. Infine il momento storico nel quale stiamo vivendo, classificato come di profonda crisi. Ma da che esiste il mondo i periodi di crisi sono quelli nei quali maturano le idee e le soluzioni più rivoluzionarie, che segneranno il cammino per il prossimo periodo di crescita e benessere. Chiaramente i periodi di crisi sono accompagnati anche da disparità, da tecnologie che perdono

importanza e redditività economica, in contrapposizione ad altre che appaiono sulla scena in modo dirompente. Alcune avranno vita breve, altre sono destinate a durare e sostenere per lungo tempo la rinascita. Probabilmente una di quelle destinate a durare è la fondazione Raspberry Pi che, almeno a nostro parere, sta gestendo al meglio la spinta innovativa mantenendo la compatibilità applicativa tra le versioni di prodotti vecchi e nuovi, comportamento che nel passato ha costituito il motivo principale della sopravvivenza, anche sino ai giorni nostri, delle aziende ad alto contenuto tecnologico.

Il nostro modo di verificarlo è stato sfilare la SD Card con una nostra applicazione attualmente in sviluppo, per la verità che gira abbastanza al limite delle risorse sui Raspberry Pi attuali, ed inserirla nel nuovo ... mostro. Tutto è partito senza esitazioni con prestazioni decisamente non confrontabili. Ma andiamo con ordine. Innanzi tutto vediamo cosa è cambiato nella carrozzeria

Fig. 2



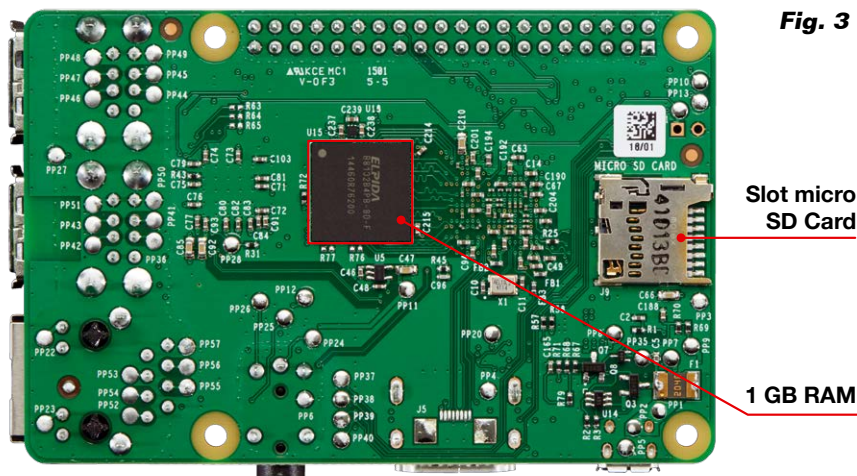


Fig. 3

e sotto il cofano del nuovo Raspberry Pi 2 e cosa non è cambiato: le dimensioni, i fori di montaggio, la disposizione dei connettori e dello slot per la SD Card. Così è possibile riutilizzare contenitori e alloggiamenti esistenti. Chiaramente è cambiata la disposizione dei componenti sul circuito stampato, viste le accresciute dimensione e numero dei principali chip. Ah, dimenticavo, non è cambiato il prezzo rispetto alla versione precedente. Questo, sempre per la “legge” di Moore. Vediamo invece le caratteristiche tecniche, dove i cambiamenti si vedono, e come.

CARATTERISTICHE TECNICHE

Per riconoscere se quella che avete in mano è un Raspberry Pi 2 od un “vecchio” Raspberry Pi B+ potete analizzare i seguenti elementi, visibili nelle Fig. 2 e Fig. 3 assieme alle indicazioni del posizionamento dei principali componenti. Per primo la scritta sulla parte superiore del circuito stampato “Raspberry Pi 2 – Model B”. Poi il processore con il logo Broadcom, sempre sulla parte superiore del circuito stampato (Fig. 2) ed il chip con la memoria RAM situato invece nella parte inferiore (Fig. 3). Le caratteristiche principali sono:

- Memoria: 1GB LPDDR2 SDRAM
- Il resto è equivalente a Raspberry Pi B+:
- Processore Video: VideoCore IV
- 4 porte USB standard su bus dedicato
- Un ingresso Ethernet 10/100 Mb
- Uscita video HDMI e A/V (i terminali del connettore sono visibili in Fig. 4)
- Connettore GPIO da 40 pin con 26 pin di I/O, bus UART, I2C, SPI
- Un connettore CSI (Camera serial Interface)
- Un connettore DSI (Display Serial Interface)
- Una porta micro USB per l'alimentazione per la quale si raccomanda un alimentatore da 2A, in quanto il consumo, senza periferiche collegate, può raggiungere i 700 mA, a

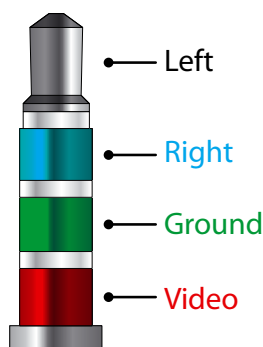


Fig. 4

5V, che vengono superati non appena si collega qualche periferica USB.

In Fig. 5 è visibile il pinout del GPIO di Raspberry Pi 2, identico a quello di Raspberry Pi B+

Fig. 5

Pi Model B/B+			
3V3 Power	1	2	5V Power
GPIO2 SDA1 I2C	3	4	5V Power
GPIO3 SCL1 I2C	5	6	Ground
GPIO4	7	8	GPIO14 UART0_TXD
Ground	9	10	GPIO15 UART0_RXD
GPIO17	11	12	GPIO18 PCM_CLK
GPIO27	13	14	Ground
GPIO22	15	16	GPIO23
3V3 Power	17	18	GPIO24
GPIO10 SPI0_MOSI	19	20	Ground
GPIO9 SPI0_MISO	21	22	GPIO25
GPIO11 SPI0_SCLK	23	24	GPIO8 SPI0_CE0_N
Ground	25	26	GPIO7 SPI0_CE1_N
ID_SD I2C ID EEPROM	27	28	ID_SC I2C ID EEPROM
GPIO5	29	30	Ground
GPIO6	31	32	GPIO12
GPIO13	33	34	Ground
GPIO19 PCM_FS	35	36	GPIO16
GPIO26	37	38	GPIO20 PCM_DIN
Ground	39	40	GPIO21 PCM_DOUT
Pi Model B+			


```
pi@raspberrypi ~ $ cat /proc/cpuinfo
processor       : 0
model name     : ARMv7 Processor rev 5 (v7l)
BogoMIPS      : 38.40
Features      : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part      : 0xc07
CPU revision   : 5

processor       : 1
model name     : ARMv7 Processor rev 5 (v7l)
BogoMIPS      : 38.40
Features      : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part      : 0xc07
CPU revision   : 5

processor       : 2
model name     : ARMv7 Processor rev 5 (v7l)
BogoMIPS      : 38.40
Features      : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part      : 0xc07
CPU revision   : 5

processor       : 3
model name     : ARMv7 Processor rev 5 (v7l)
BogoMIPS      : 38.40
Features      : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x0
CPU part      : 0xc07
CPU revision   : 5

Hardware      : BCM2709
Revision     : a01041
Serial       : 0000000070a5c4a6
pi@raspberrypi ~ $
```

Fig. 6

stesso microcomputer, passando con facilità da una all'altra, oppure il configuratore raspi-config o il pacchetto di aggiornamento della partizione di root rpi-update. Inoltre molti sviluppatori indipendenti hanno costruito applicazioni, tutorial, libri e documentazioni basandosi su Raspberry Pi. Passare ad un nuovo hardware significa spesso invalidare tutto questo patrimonio di investimenti, software e competenze per "ricominciare" da capo su nuove piattaforme dalle caratteristiche spesso non compatibili con le precedenti. Da questo problema non sono state e non sono immuni nemmeno grandi e grandissime organizzazioni di produzione di hardware e software, nel passato come tuttora. Nello sviluppo della nuova Raspberry Pi 2 tutto questo è stato salvaguardato. Broadcom ha messo a disposizione il nuovo SoC BCM2836 dove la "vecchia" CPU ARMv6 è stata "ritagliata" e sostituita con la nuova Cortex A7 quadricore. Il banco di memoria da 1 Gb è stato collocato nella parte inferiore del circuito stampato, con vantaggio sulle capacità di raffreddamento del processore principale. Il processore Videocore IV è lo stesso del BCM2835. A questo punto è stato realizzato il porting del kernel di Raspbian per aderire alle specifiche hardware del nuovo Soc. In questa fase è dove si è rivelata vincente l'architettura del sistema operativo GNU/Linux, peraltro progettata da sempre proprio per questo tipo di operazioni. GNU/Linux è un

In Fig. 6 possiamo verificare la presenza delle quattro CPU, ottenuta con il comando:

`cat /proc/cpuinfo`

già anticipata dai quattro lampori presenti nella schermata dei messaggi di console che accompagnano il processo di "boot" di Raspberry Pi.

In Fig. 7 è visibile la quantità di memoria disponibile ed utilizzata.

GRANDE ATTENZIONE ALLA COMPATIBILITÀ

Questo elenco di caratteristiche potrebbe far pensare ad un semplice upgrade delle caratteristiche di un microcomputer di successo, per adeguarlo alle possibilità offerte dalle nuove tecnologie. Invece, a nostro parere, c'è molto di più. In particolare la volontà della fondazione di salvaguardare gli investimenti dedicati allo sviluppo di software, sistemi operativi ed applicazioni utente, adattati e resi particolarmente efficienti per essere utilizzati sui microcomputer Raspberri Py della serie precedente, dotati di processore Broadcom BCM2835.

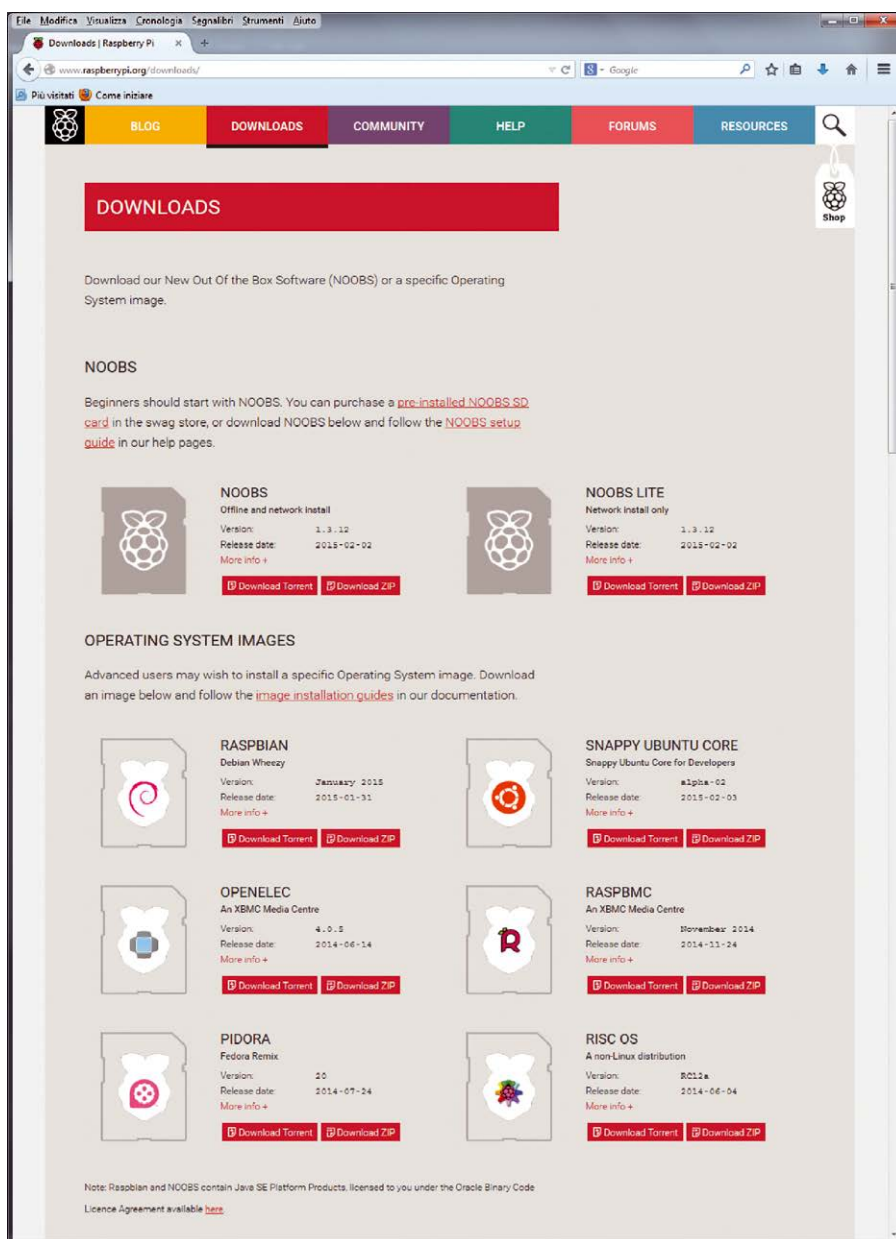
Stiamo parlando delle migliaia di ore investite, e dei relativi costi economici, destinate al "porting" della distribuzione GNU/Linux Debian per funzionare sull'hardware ARMv6, incluso il supporto hardware per le operazioni in floating point e per rendere stabile e sempre più performante la distribuzione stessa. Altrettanto lavoro ed impegno economico è stato speso nell'ottimizzare per Raspberry Pi, una grande quantità di librerie ed applicazioni Open Source come WebKit, LibreOffice, Scratch, Pixman, XBMC/Kodi, libav e PyPy. Infine lo sforzo profuso nello sviluppare e ottimizzare continuamente strumenti che potessero facilitare la vita agli utilizzatori alle prime armi (e non) come lo strumento NOOBS che permette di gestire diverse distribuzioni GNU/Linux sullo

```
root@raspberrypi:~# free -m
              total        used        free      shared    buffers     cached
Mem:          974         85         889          0         15         35
-/+ buffers/cache:        34         940
Swap:          99           0          99
root@raspberrypi:~#
```

Fig. 7

Fig. 8

sistema operativo appartenente alla famiglia dei sistemi basati su kernel "monolitici", con una netta separazione tra "spazio kernel" e "spazio utente". Lo scopo è di semplificare la vita all'utilizzatore del sistema, interfacciando e "normalizzando" la comunicazione tra le applicazioni e l'hardware e le periferiche fisiche del computer. In estrema sintesi, il kernel (nocciolo) del sistema operativo, tra l'altro, si occupa di interfacciare l'hardware (CPU, memorie, bus, periferiche fisiche) e di fornire un primo livello di driver "standardizzati". Lo strato successivo, si occupa di semplificare ulteriormente le modalità di comunicazione con le periferiche, normalizzando il tutto e rendendolo "visibile" all'utilizzatore finale attraverso il file system "virtuale", unica e sola "finestra" disponibile all'utente per comunicare con il sistema fisico. Di fatto il file system virtuale costituisce il confine tra quello che abbiamo definito "spazio kernel" e lo "spazio utente" dove scriviamo e/o elaboriamo le nostre applicazioni. L'architettura di GNU/Linux è stata descritta già diverse volte nella rivista e nel libro "Raspberry Pi – Il mio primo Linux embedded" (che, tra l'altro, mantiene, in gran parte, la sua validità anche per quanto riguarda Raspberry Pi 2). Per quanto riguarda i componenti al di fuori del kernel non sono ancora stati ottimizzati per la nuova CPU, in modo da mantenere la compatibilità con le schede precedenti. Questo avverrà, a detta della fondazione, nel tempo e selezionando accuratamente quali librerie aggiornare per ottenere il miglior compromesso di funzionalità, stabilità e prestazioni da entrambi gli ambienti. Già da subito, tra le altre novità degne di nota la disponibilità di nuove distribuzioni di sistemi operativi



finalizzati alla nuova versione, tra le quali Snappy Ubuntu Core for Developers (Fig. 8), già disponibile e scaricabile in versione alpha e ... udite, udite Windows 10 for Raspberry Pi, al momento allo stato di annuncio, ma che sembra sarà distribuito gratuitamente agli sviluppatori (Fig. 9). Entrambi queste distribuzioni sono orientate allo sviluppo di applicazioni per quello che viene definito IoT (Internet of Thing), un universo in fermento che, nel futuro, dovrebbe vedere un mondo sempre più interconnesso, ma i

contorni del quale sono ancora in corso di definizione, almeno questo è il mio parere personale. Ciò che è importante della presenza di queste due distribuzioni nell'elenco ufficiale è l'attenzione ed il conseguente supporto, in particolare finanziario, che i due competitor sul mercato Microsoft per quanto riguarda Windows 10 e Canonical per Ubuntu Snappy stanno ponendo nei confronti dei prodotti della fondazione Raspberry Pi, anche se non esclusiva. Il punto sostanziale è la presa d'atto da parte dei produttori mondiali

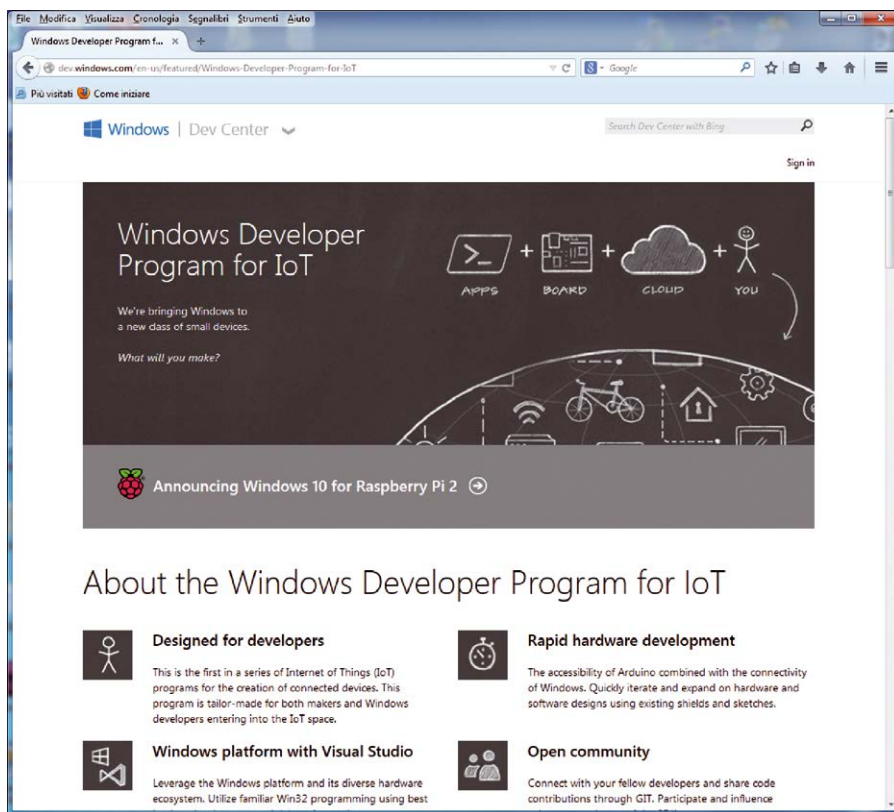


Fig. 9

dello spostamento dell'asse dello sviluppo dell'informatica sempre più verso applicazioni in campi quantomeno "non tradizionali", per il mondo dell'informatica, come auto motive, automazione industriale, casalinga e ambientale, mobile, in quantità e pervasività sempre più accentuate.

ACCENDIAMO RASPBERRY PI 2

Per "dare vita" al nostro nuovo Raspberry Pi abbiamo diverse possibilità, la prima è di scaricare la nuova immagine della distribuzione Raspbian, oppure lo strumento NOOBS, dal sito ufficiale di Raspberry Pi, trasferire l'immagine su una micro SD Card (attenzione, è necessario utilizzare una micro SD Card, come già richiesto per Raspberry Pi B+; le SD Card in formato standard non sono più utilizzabili, anche se da tempo, pressoché tutte le SD Card in commercio sono costituite da una micro SD Card ed un adattatore), inserirla nell'apposito slot sul microcomputer e collegare l'alimentatore per mezzo del

connettore micro USB. Questo appena descritto, è sicuramente il metodo da preferire, per chi, con l'occasione, si avvicina per la prima volta a Raspberry Pi. Per chi è un utilizzatore delle versioni precedenti di Raspberry Pi può verificare che le distribuzioni precedenti a quella datata 31-1-2015 non possono, ovviamente funzionare sul nuovo microcontrollore. In questa situazione una possibilità è fare un salvataggio dei propri file, partire da una distribuzione "fresca" ed eseguire il restore dei nostri file. In questo modo dovremo però anche reinstallare tutti i pacchetti che ci necessitano e riconfigurarli secondo le nostre necessità. Una mole di lavoro che nasconde notevoli insidie e quantomeno la necessità di rieseguire i test di tutte le funzionalità. Noi abbiamo preferito la terza via, che è probabilmente il metodo da preferire nel caso si utilizzi la distribuzione Raspbian di GNU/Linux. Anzi, abbiamo voluto sottoporre il nuovo gioiello alla prova della compatibilità nel modo più ...

rude possibile. Alcune SD Card della collezione che utilizziamo per lavoro risalgono alle prime distribuzioni di Raspbian apparse nel 2012. Ovviamente sono state sempre aggiornate con i comandi che vi raccomandiamo in ogni articolo ed in alcuni casi sono state clonate trasferendo l'immagine dalla SD Card originale ad una più nuova, e di tipo micro. Per inciso vi raccomandiamo di eseguire regolarmente la "manutenzione" delle SD Card in quanto il loro utilizzo sui microcomputer ne "accorcia" la vita. Le SD Card hanno un numero massimo di scritture possibili, numeri molto grandi ed in continua crescita, che garantiscono vita eterna nel caso vengano utilizzate in macchine fotografiche, smartphone, tablet e simili. Nel caso di microcomputer con applicazioni ad alta intensità di scrittura, in particolare con l'utilizzo di database, le scritture si concentrano in gran numero sulla stessa "zona" dell'area di memoria della SD Card, e prima o poi il numero massimo di scritture viene raggiunto con conseguente blocco dell'applicazione e perdita di dati. Quindi, se sviluppate applicazioni "professionali" con Raspberry Pi, cosa che diventa assolutamente possibile con la nuova versione, è fondamentale tenere conto e progettare accuratamente le funzionalità di backup e salvaguardia dei dati. Detto questo abbiamo preso una delle nostre SD Card storiche e, utilizzando un Raspberry Pi "tradizionale", dopo averlo collegato alla rete, abbiamo eseguito i comandi che permettono di aggiornare completamente la distribuzione per farla funzionare sul nuovo Raspberry Pi, kernel compreso. Se siete nella nostra situazione vi raccomandiamo questo metodo. Come prima operazione vi consigliamo di realizzare un'im-

immagine di backup della SD Card, utilizzando WinDiskImager. Poi aggiorniamo la distribuzione nel modo consueto, loggati con l'utente "root", con i comandi:

```
apt-get update
apt-get upgrade
apt-get dist-upgrade
```

Se la distribuzione non viene aggiornata da qualche tempo, il processo si rivelerà abbastanza lungo. Da qualche versione la distribuzione Raspbian adotta una nuova versione dell'interfaccia grafica Lxde (Lightweight X11 Desktop Environment), che prevede la barra delle applicazioni in alto nello schermo, un desktop vuoto da popolare con le applicazioni che preferiamo ed il menu "Start"

in alto a sinistra. Aggiorniamo la distribuzione alla nuova interfaccia grafica con il comando:

```
apt-get install raspberrypi-ui-mods
```

Dopo questa operazione è opportuno "ripulire" la distribuzione con i comandi:

```
apt-get autoremove
```

che elimina eventuali pacchetti non più utilizzati e:

```
apt-get purge
```

che elimina i file di configurazione non più utilizzati.

Infine aggiorniamo la partizione

di root con il comando:

```
rpi-update
```

se ottenete un errore del tipo "command not found" potete installare lo strumento di upgrade con il comando:

```
apt-get install rpi-update
```

Fatto tutto? Potete spegnere il "vecchio" Raspberry Pi trasferire la SD Card sul nuovo modello Raspberry Pi 2 e collegare l'alimentazione. Lanciate le vostre applicazioni. Misurate le prestazioni sia visivamente che con i comandi

top

per l'utilizzo di cpu (si esce con il tasto "q") e:

free -m

per l'utilizzo di memoria.

Ovviamente la stessa SD Card è in grado di funzionare su tutte le versioni di Raspberry Pi. Questo è quello che consideriamo il vero concetto di retro compatibilità. Per capire di più su come la distribuzione Raspian è stata configurata per ottenere questo risultato, andiamo ad analizzare la partizione di root, il cui contenuto è visibile nella Fig. 10. Se abbiamo ispezionato in passato la partizione di root troviamo la presenza di un certo numero di file aggiuntivi rispetto ai soliti: il file immagine del kernel Linux, i file start.elf, config.txt e cmdline.txt. Fondamentalmente troviamo l'immagine di un secondo kernel di nome "kernel7.img", un chiaro riferimento alla nuova cpu Cortex A7 ed alcuni file con suffisso .dtb (device tree blob).

I file .dtb sono sempre più utilizzati per comunicare la struttura

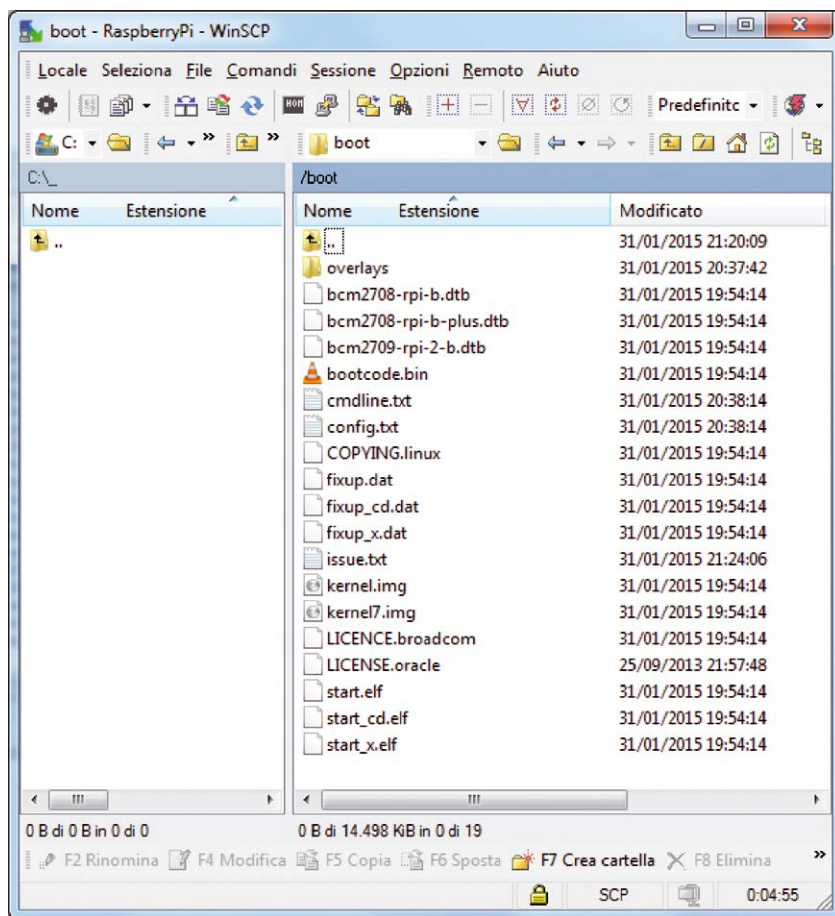


Fig. 10

Fig. 11

```

[ 0.000000] popu-alloc: s12864 r8192 d24000 u45056 alloc=11*4096
[ 0.000000] popu-alloc: [0] 0 [0] 1 [0] 2 [0] 3
[ 0.000000] Built 1 zonelists in Zone order, mobility grouping on. Total pages: 251968
[ 0.000000] Kernel command line: dma.dmachans=0x7f35 bcm2708_fb.fbwidth=1920 bcm2708_fb.fbheight=1080 bcm2709.boardrev=0xa01041
bcm2709.serial=0x70a5e4a6 smmc95xx.macaddr=B8:27:EB:A5:E4:A6 bcm2708_fb.fbwap=1 bcm2709.disk_led_gpio=47
bcm2709.disk_led_active_low=0 sdhci-bcm2708.emmc_clock_freq=250000000 vc_mem.mem_base=0x3ea00000 vc_mem.mem_size=0x3f000000
| dwc_otg.lpm_enable=0 console=tty1 root=/dev/sdal rootfstype=ext4 elevator=deadline boot_delay=32 rootdelay=10 rootwait
[ 0.000000] PID hash table entries: 4096 (order: 2, 16384 bytes)
[ 0.000000] Dentry cache hash table entries: 131072 (order: 7, 524288 bytes)
[ 0.000000] Inode-cache hash table entries: 65536 (order: 6, 262144 bytes)
[ 0.000000] Memory: 989508K/1015808K available (5785K kernel code, 377K rwdats, 1760K rodata, 396K init, 771K bss, 26300K reserved)

```

dell'hardware da parte del bootloader al kernel durante il processo di boot. Di fatto il file.dtb è un database compilato in formato binario che contiene la descrizione della struttura dell'hardware in uno standard comprensibile sia al bootloader che al kernel. Con questa configurazione, a seconda della versione di Raspberry Pi dove viene installata la SD Card, durante il processo di boot viene caricato il corretto bootloader di stage 1, al quale fa seguito il caricamento del corretto kernel con i rispettivi file di configurazione. I vecchi file di configurazione config.txt e cmdline.txt sono tuttora condivisi da entrambi i sistemi mantenendo ancora una

volta la compatibilità applicativa tra le due generazioni di prodotti. Nel nostro caso, per esempio, in una delle nostre applicazioni è previsto l'utilizzo per la partizione di root, di un hard disk esterno collegato via usb. Dopo il processo di aggiornamento che vi abbiamo indicato, la soluzione con hard disk esterno, che richiede di personalizzare i file cmdline.txt ed /etc/fstab (vedi libro), ha funzionato a dovere su entrambe le configurazioni, come visibile in Fig. 11 dal frammento dei messaggi di boot di Raspberry Pi 2. Lo strumento di configurazione "raspi-config" è stato arricchito con nuove funzionalità. Oltre alle solite funzioni che permettono di:

espandere la partizione di root, lanciare l'interfaccia grafica al boot, eseguire l'overclocking della CPU, abilitare la Camera PI ed altre, visibili in Fig. 12. scegliendo la voce "Advanced Options" si accede al menu di secondo livello, che permette di eseguire diverse configurazioni che prima dovevano essere eseguite da linea di comando. In Fig. 13 oltre alle funzionalità tradizionali come "Over-scan" che permette di gestire le "cornici" attorno allo schermo, se necessario, SSH che permette di abilitare e disabilitare il server per l'accesso da remoto, memory split per assegnare più o meno memoria alla GPU video e hostname per cambiare nome al mi-

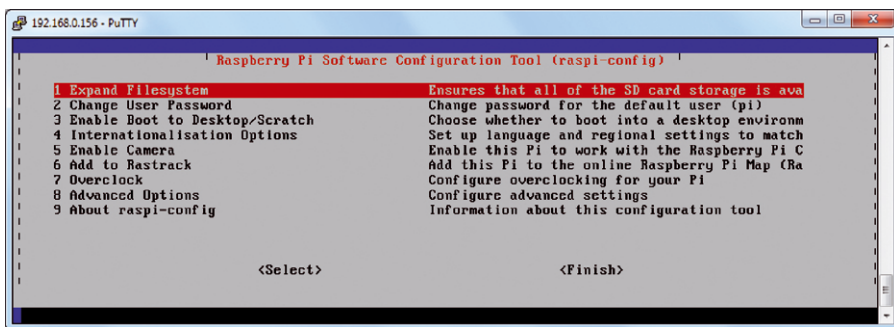


Fig. 12

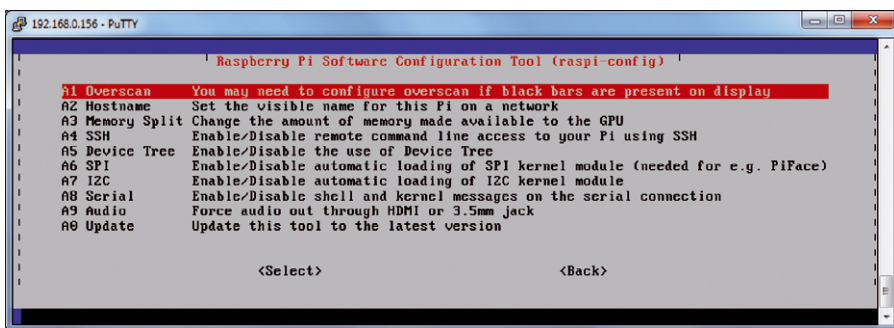


Fig. 13



Fig. 14

Fig. 15

```
top - 20:01:44 up 3 min, 2 users, load average: 1.04, 0.60, 0.26
Tasks: 77 total, 1 running, 76 sleeping, 0 stopped, 0 zombie
%Cpu(s): 24.2 us, 9.6 sy, 0.0 ni, 51.6 id, 9.6 wa, 0.0 hi, 5.0
KiB Mem: 494508 total, 168732 used, 325776 free, 15468 buf
KiB Swap: 102396 total, 0 used, 102396 free, 72992 cac
```

PID	USER	PR	NI	UIRT	RES	SHR	S	%CPU	%MEM	TIME+
2542	mysql	20	0	318m	42m	9376	S	13.4	8.8	0:08.45
2894	root	20	0	24032	9.9m	6700	S	10.1	2.1	0:05.04
2896	root	20	0	26420	12m	7072	S	10.1	2.6	0:06.24
2907	root	20	0	4660	2396	2084	R	1.3	0.5	0:00.69
7	root	20	0	0	0	0	S	0.3	0.0	0:00.70
1	root	20	0	2152	1332	1228	S	0.0	0.3	0:00.92

Fig. 16

```
top - 19:56:13 up 12 min, 2 users, load average: 1.55, 1.27, 0.76
Tasks: 96 total, 1 running, 95 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.2 us, 0.1 sy, 0.0 ni, 93.6 id, 5.1 wa, 0.0 hi, 1.1
KiB Mem: 998096 total, 178280 used, 819816 free, 17060 buf
KiB Swap: 102396 total, 0 used, 102396 free, 74252 cac
```

PID	USER	PR	NI	UIRT	RES	SHR	S	%CPU	%MEM	TIME+
2456	mysql	20	0	311m	41m	9116	S	5.9	4.3	0:38.71
2774	root	20	0	24036	9.8m	6664	S	4.3	1.0	0:25.28
2780	root	20	0	26424	12m	7092	S	3.9	1.3	0:25.78
2798	root	20	0	4676	2404	2080	R	1.0	0.2	0:00.12
3	root	20	0	0	0	0	S	0.3	0.0	0:00.27
7	root	20	0	0	0	0	S	0.3	0.0	0:02.41

crocomputer, assegnare o meno il bus seriale alla console di sistema e configurare l'uscita audio verso il connettore HDMI oppure verso il jack da 3,5mm. Particolare attenzione va dedicata alle funzioni per abilitare e disabilitare i

bus I2C e SPI. Il vecchio metodo della blacklist e del caricamento del driver con "modprobe" deve essere integrato con l'attivazione dei bus i2C e/o SPI utilizzando le funzioni messe a disposizione. Dal punto di vista delle presta-

zioni c'è poco da dire, niente di comparabile rispetto alle versioni precedenti, i benchmark reperibili sul web danno un incremento delle prestazioni di circa sei volte quelle attuali. Ovviamente dipende dal tipo di applicazioni e dal modo nel quale queste utilizzano le risorse. Per quanto riguarda uno dei nostri progetti, che presenteremo presto sulla rivista, la schermata principale del quale è visibile in Fig. 14. Le Fig. 15 e Fig. 16, che riportano il consumo di CPU rispettivamente su Raspberry Pi B+ e 2, parlano da sole. Di fatto, allo stato attuale, Raspberry Pi 2 può essere paragonato ad un personal computer di fascia bassa, dove è possibile eseguire le normali applicazioni d'ufficio e didattiche in modo più che accettabile. Ne sono un esempio LibreOffice e Geogebra, quest'ultimo non certo parco nell'utilizzo di risorse, che vi mostriamo nelle Fig. 17 e Fig. 18. Definitivamente uscita dall'essere confinata al campo hobbistico ed

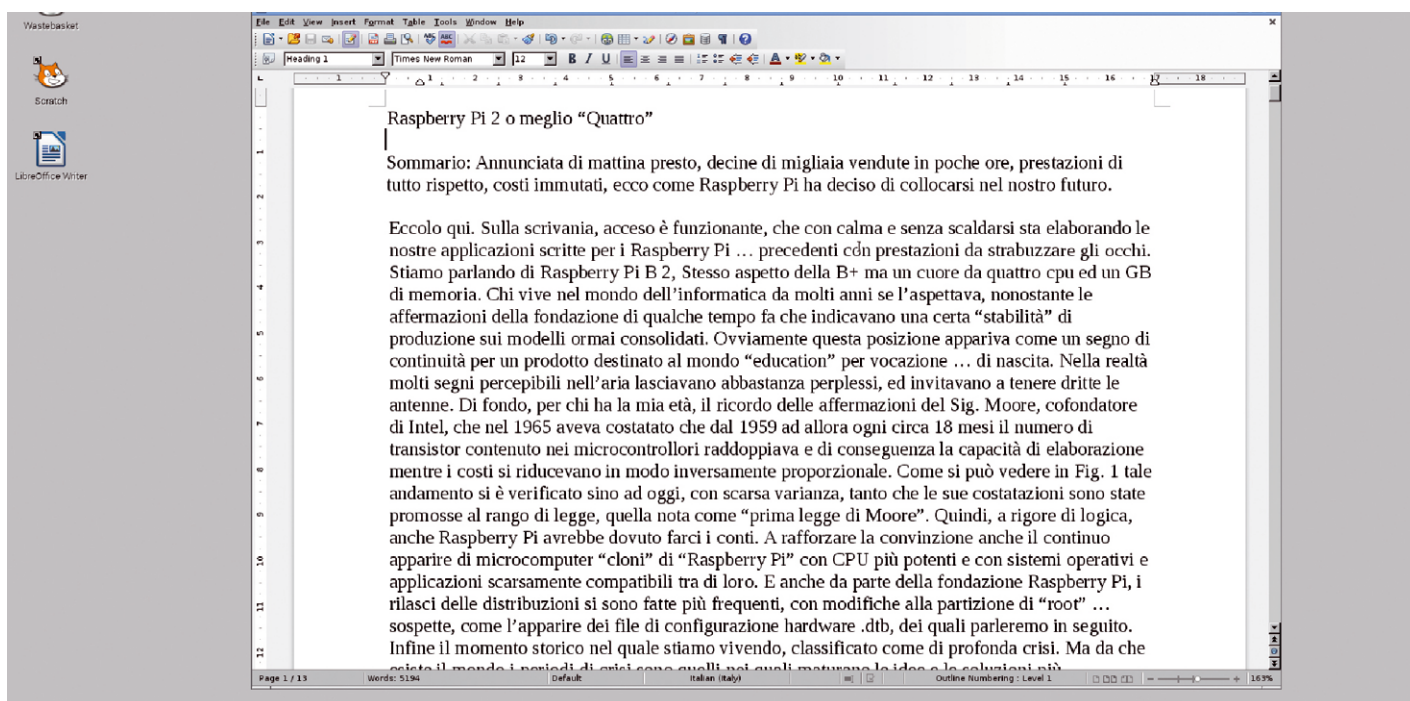


Fig. 17

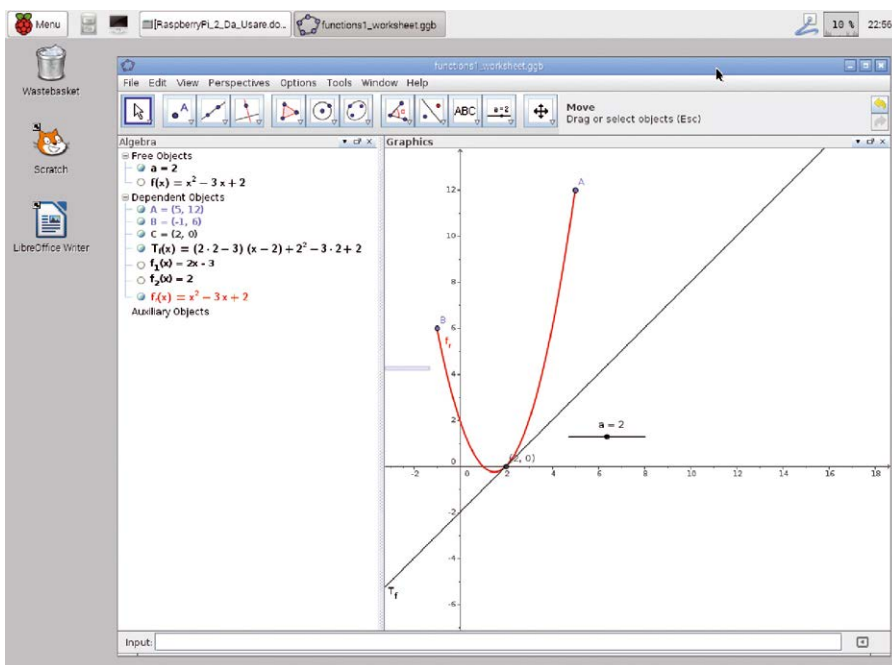


Fig. 18

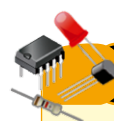
ed approfondire gli argomenti trattati. Diverse SD Card permettono di configurare il laboratorio con architetture diverse a seconda delle esigenze, sottoreti, server, client, ecc. Senza dimenticare, al riguardo, che una legge del 2012 richiede alle amministrazioni pubbliche di preferire l'adozione di soluzioni open, nel caso queste siano disponibili.

Per concludere, normalmente teniamo sulla scrivania le soluzioni migliori che appaiono sul mercato per utilizzarle nei nostri progetti. Questa volta il personal computer che utilizziamo normalmente ci sta guardando preoccupato, molto probabilmente ha intravisto qualcosa in grado di ... rottamarlo. Forse non è proprio arrivato il momento definitivo ma come apparirà Raspberry Pi 3 non vi saranno più dubbi. Già, la "legge" di Moore. A proposito, come avrete notato in Fig. 17, questo articolo lo abbiamo scritto con LibreOffice sul nuovo Raspberry Pi, utilizzando schermo, tastiera e mouse del vecchio PC. Ora lo riaccendo, altrimenti si offende.

alle applicazioni poco esose in termini di risorse, Raspberry Pi 2 approda nel mondo professionale a tutti gli effetti, oltre che apparire a pieno titolo la soluzione ad oggi più adatta nella maggior parte delle applicazioni didattiche, risolvendo anche questioni organizzative finora difficili da risolvere.

Pensiamo per esempio ad un laboratorio scolastico, con un certo numero di personal computer installati, sul quale ruotano gli studenti di numerose classi, magari con esigenze applicative differenti a secondo dell'indirizzo e del livello del corso seguito. Oggi, questa architettura, richiede un'amministrazione del parco macchine, degli utenti e dei permessi, abbastanza complesso, se vogliamo che ciascuno abbia a disposizione le risorse necessarie a svolgere i compiti assegnati senza interferire con il lavoro degli altri studenti. Con le dotazioni attuali basate su sistemi Microsoft alcune tipologie di esercitazioni sono abbastanza complesse da portare a termine. Pensiamo alla disciplina di progettazione e realizzazione di reti di comunicazioni, in particolare TCP/IP, dove

sia necessario configurare sistemi con diversi indirizzi IP assegnati alla stessa interfaccia di rete, configurare router, firewall, access point, VPN e realizzare applicazioni ponte e broker basate su socket TCP/IP. Con una "manciata" di Raspberry Pi ed un investimento economico equivalente all'acquisto di un paio di personal computer tradizionali è possibile allestire un intero laboratorio di reti e informatica in senso più generale, senza necessità di costi aggiuntivi per l'acquisto di licenze di pacchetti di applicazioni di ufficio, o di ambienti di sviluppo applicazioni. E infatti lo scopo principale che si prefigge la fondazione Raspberry Pi è proprio fornire strumenti didattici open source, economici ed al passo con la tecnologia emergente. Sempre dal punto di vista organizzativo, provate a pensare ciascun studente che entra in laboratorio con la propria SD Card con la situazione aggiornata all'ultima esercitazione svolta, che il laboratorio sia lo stesso od un altro, senza interferenze con altri studenti. La stessa SD Card potrà poi essere usata sul Raspberry di casa per continuare il lavoro iniziato a scuola



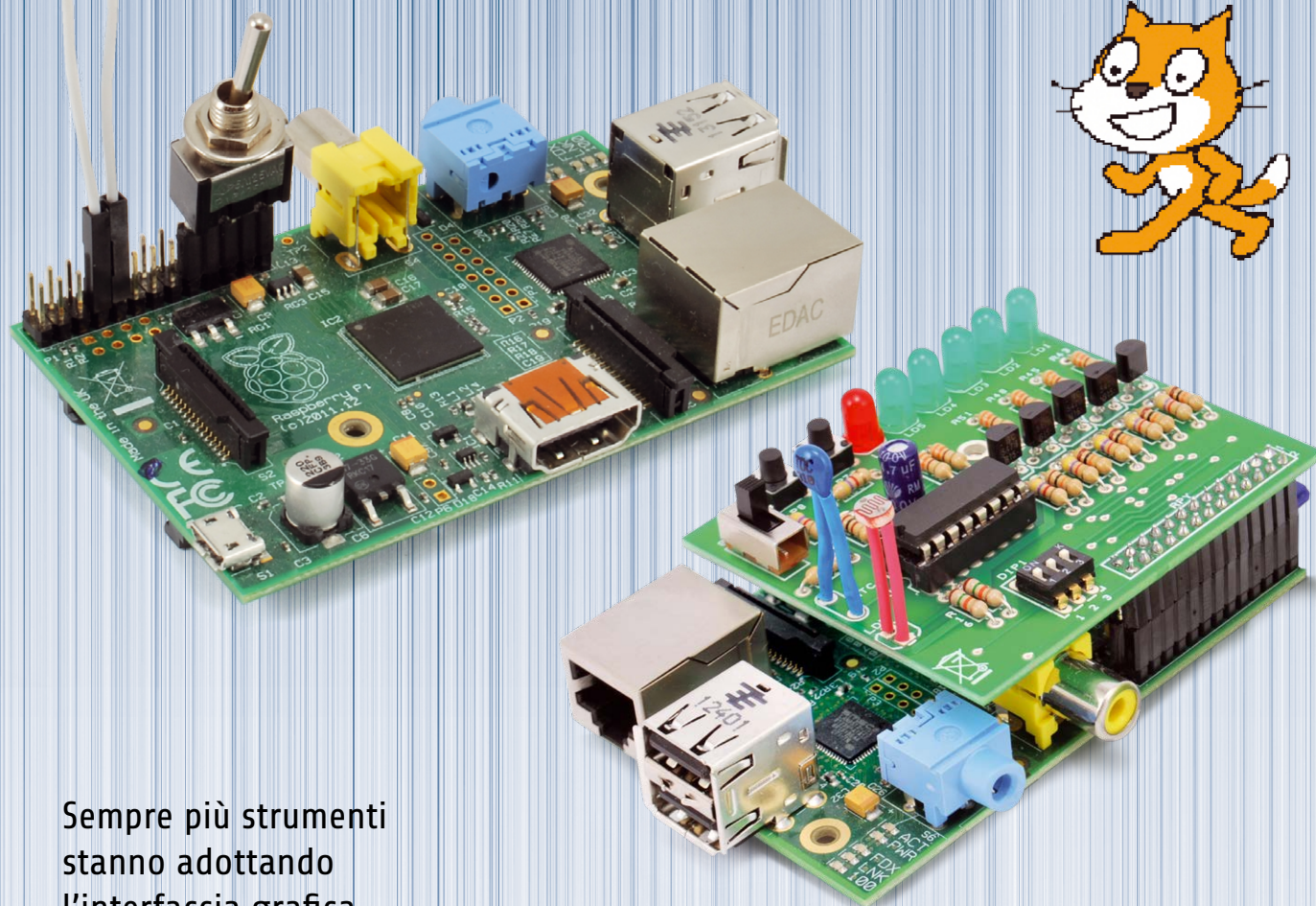
per il MATERIALE

La board Raspberry Pi 2 Model B è disponibile al prezzo di 46,00 Euro. È anche disponibile lo Starter Kit cod. RASPKITV4 che comprende la board, l'alimentatore da 3A, i cavi USB (0,7 m), FTP (1,8 m) e HDMI (1,5 m), nonché la Micro SD da 8GB con adattatore e NOOBS. Lo starter kit costa 74,00 Euro. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:
Futura Elettronica, Via Adige 11,
21013 Gallarate (VA)
Tel: 0331-799775 • Fax: 0331-792287
<http://www.futurashop.it>

SCRATCH E RASPBERRY PI

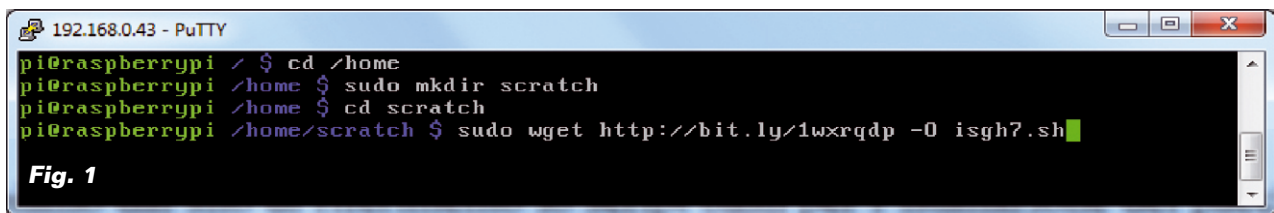
..... di MARCO MAGAGNIN



Sempre più strumenti stanno adottando l'interfaccia grafica di Scratch. Uno di questi, ScratchGPIO7, ci permette di utilizzare il linguaggio grafico di Scratch per gestire il GPIO di Raspberry Pi. Terza parte.

Quando abbiamo deciso di proporre questa serie di articoli, con l'obiettivo di facilitare l'apprendimento della programmazione dei calcolatori elettronici per mezzo di uno strumento didattico dall'aspetto ludico ma dal contenuto professionale, non pensavamo alla reale portata di

questo approccio. Pur essendo disponibile da diverso tempo, questa tipologia di strumenti ha conosciuto negli ultimi mesi una diffusione inaspettata. Oltre alla versione di Scratch tradizionale, quella che vi abbiamo presentato fino ad ora, sono apparse nuove soluzioni che ricalcano la stessa



```
192.168.0.43 - PuTTY
pi@raspberrypi ~ $ cd /home
pi@raspberrypi /home $ sudo mkdir scratch
pi@raspberrypi /home $ cd scratch
pi@raspberrypi /home/scratch $ sudo wget http://bit.ly/1wxrqdp -O isgh7.sh
```

Fig. 1

filosofia, soprattutto per quanto riguarda l'impostazione grafica delle applicazioni. Si va dagli ambienti di sviluppo totalmente utilizzabili all'interno di un browser web, come la versione 2.0 di Scratch o l'ambiente Snap (messo a disposizione dall'Università di Berkeley), a quelli dedicati a specifici hardware, fino alle soluzioni, per così dire, proprietarie.

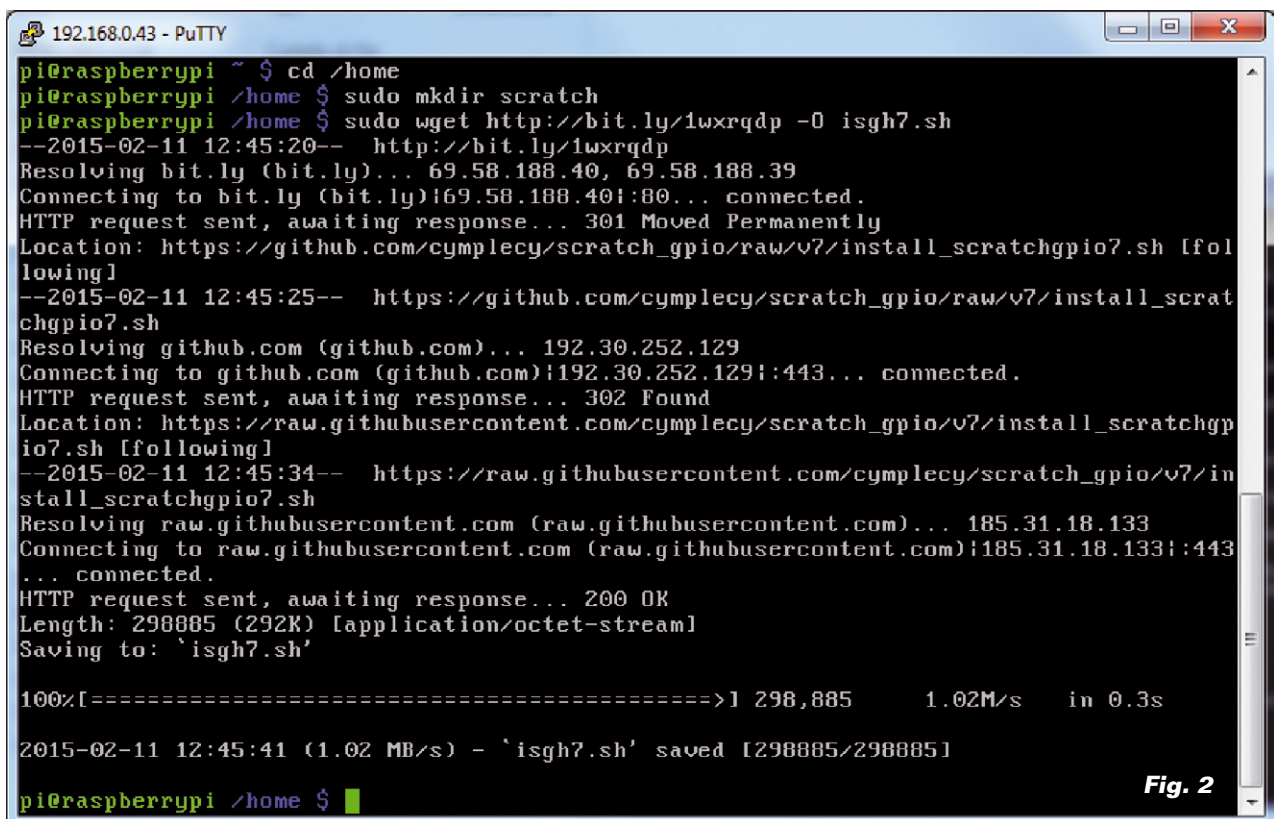
Nel caso delle applicazioni destinate ad uno specifico hardware citiamo ScratchGPIO7 che include le "istruzioni" che permettono di interfacciare il GPIO di Raspberry Pi e che, ovviamente, "gira" su Raspberry Pi; allo stesso modo

S4A che sta per "Scratch For Arduino", permette di scrivere gli sketch per Arduino utilizzando elementi simili a quelli utilizzati dal linguaggio grafico Scratch. Tra le soluzioni "proprietarie" citiamo Wyliodrin, un ambiente che permette di utilizzare i sensori di diversi produttori anche da remoto, in modo da poter realizzare una sorta di sistema di automazione distribuito e gestibile in remoto.

La soluzione prevede un utilizzo gratuito fino ad un numero limitato di schede e di applicazioni, per poi proporre diverse tipologie di abbonamento, in base alle esigenze degli utilizzatori. Ma torniamo alla implemen-

tazione di Scratch che permette di interfacciare il GPIO di Raspberry Pi, ovvero il pacchetto ScratchGPIO7. L'ultimo numero rappresenta la versione del software ed incrementa molto velocemente, probabilmente al tempo dell'uscita dell'articolo potrebbe già essere disponibile una nuova versione.

Procediamo con l'installazione. Partiamo da una immagine fresca del sistema operativo Raspbian su Raspberry Pi. ScratchGPIO7 richiede di essere lanciato dall'interfaccia grafica di Raspberry Pi, quindi dobbiamo "allestire" il nostro Raspberry Pi in "stile" desktop, dotato di monitor HDMI, tastiera e mou-



```
192.168.0.43 - PuTTY
pi@raspberrypi ~ $ cd /home
pi@raspberrypi /home $ sudo mkdir scratch
pi@raspberrypi /home $ sudo wget http://bit.ly/1wxrqdp -O isgh7.sh
--2015-02-11 12:45:20-- http://bit.ly/1wxrqdp
Resolving bit.ly (bit.ly)... 69.58.188.40, 69.58.188.39
Connecting to bit.ly (bit.ly):69.58.188.40:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://github.com/cymplecy/scratch_gpio/raw/v7/install_scratchgpio7.sh [following]
--2015-02-11 12:45:25-- https://github.com/cymplecy/scratch_gpio/raw/v7/install_scratchgpio7.sh
Resolving github.com (github.com)... 192.30.252.129
Connecting to github.com (github.com):192.30.252.129:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/cymplecy/scratch_gpio/v7/install_scratchgpio7.sh [following]
--2015-02-11 12:45:34-- https://raw.githubusercontent.com/cymplecy/scratch_gpio/v7/install_scratchgpio7.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.31.18.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com):185.31.18.133:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 298885 (292K) [application/octet-stream]
Saving to: 'isgh7.sh'

100%[=====>] 298,885      1.02M/s   in 0.3s

2015-02-11 12:45:41 (1.02 MB/s) - 'isgh7.sh' saved [298885/298885]

pi@raspberrypi /home $
```

Fig. 2

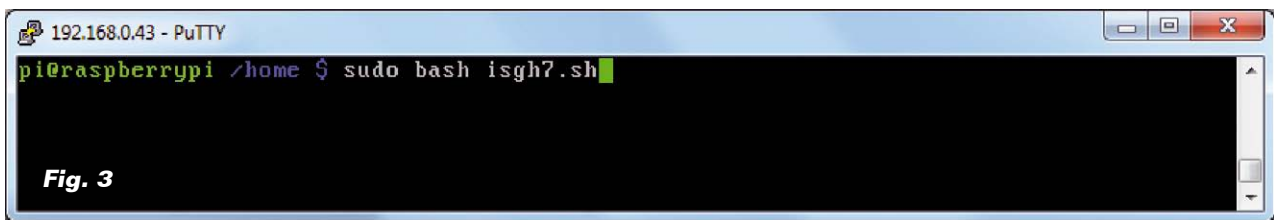


Fig. 3

se. Una possibile alternativa è utilizzare, in ambiente Windows, lo strumento di accesso remoto MobaXterm, come indicato nell'articolo dedicato alla scheda RandA pubblicato nel numero 190 della rivista. Ovviamente, come è di prassi nei sistemi GNU/Linux, Raspberry Pi deve essere collegato alla rete internet, per poter scaricare i pacchetti necessari all'installazione. Apriamo un terminale, in locale od in remoto, ci logghiamo con l'utente "pi" (password "raspberrypi") ed aggiorniamo il sistema operativo con i comandi consueti, preceduti dal comando "sudo" in quanto non siamo utente "root":

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get dist-upgrade
```

Per l'utilizzo di base di Raspberry Pi e del sistema operativo Raspbian, distribuzione di GNU/Linux basata su Debian, rimaniamo agli argomenti contenuti nel libro "Raspberry Pi – Il mio primo Linux embedded". Terminato l'aggiornamento, che può richiedere un riavvio del sistema con il comando:

```
sudo reboot
```

possiamo procedere all'installazione del pacchetto ScratchGPIO7. Il sito di riferimento, in inglese, per il pacchetto ScratchGPIO7 è "<http://simplepi.net/scratchgpio/scratch-raspberrypi-gpio/>". Per prepararci all'installazione entriamo nella cartella /home e creiamo al suo interno una cartella di nome "scratch", dove

scaricheremo lo script di installazione, come visibile in Fig. 1, con i comandi:

```
cd /home
sudo mkdir scratch
cd scratch
```

Scarichiamo dal sito di riferimento lo script bash per l'installazione del pacchetto con il comando (Fig. 2):

```
sudo wget http://bit.ly/1wxrqdp
-O isgh7.sh
```

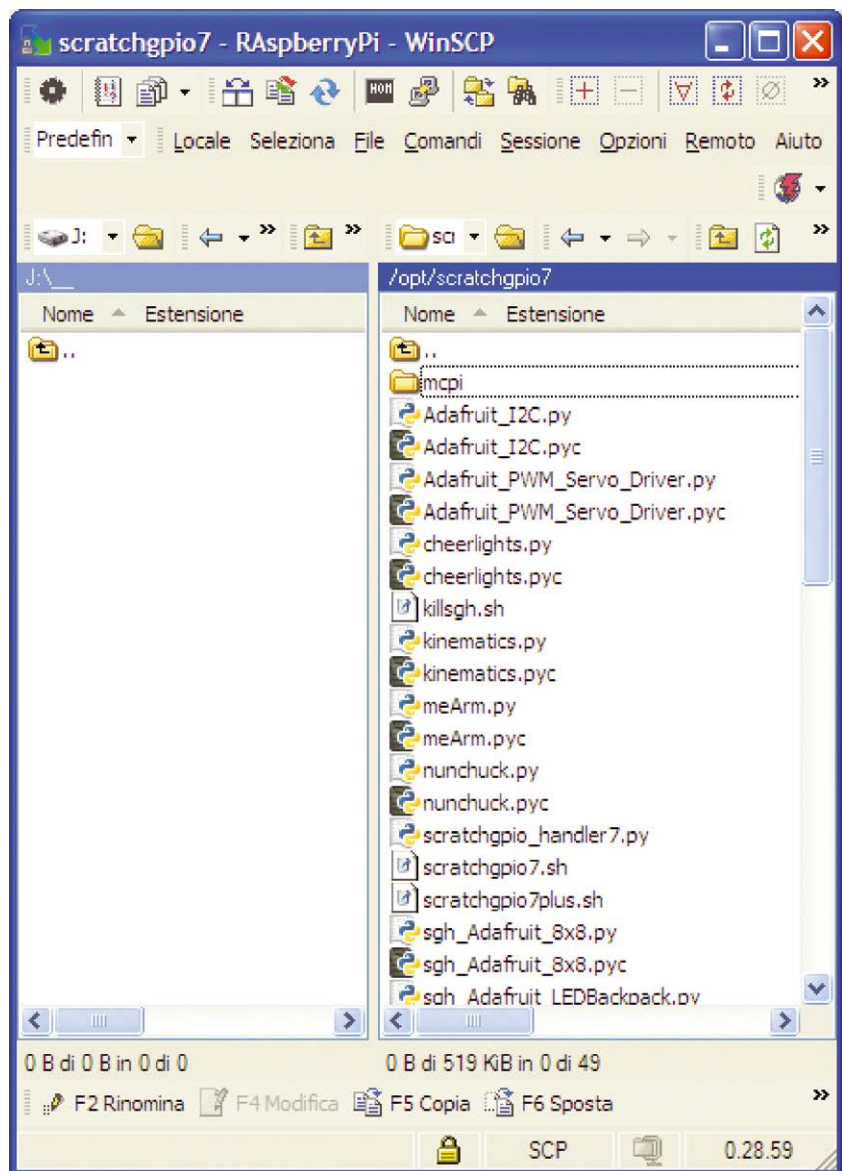
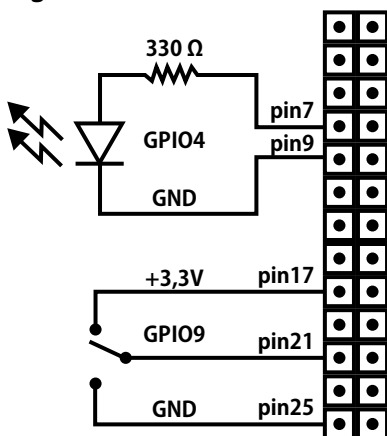


Fig. 4

Fig. 5



Una volta scaricato correttamente lo script di installazione lo possiamo eseguire con il comando (Fig. 3):

```
sudo bash isgh7.sh
```

Una volta terminato il processo di installazione troveremo tutti i file dell'applicazione installati nella cartella `/opt/scratchgpio7` come visibile in Fig. 4, dove possiamo notare che l'estensione di Scratch per Raspberry Pi è costituita da una gerarchia di programmi scritti in linguaggio Python. Sul desktop di Raspberry Pi troveremo due icone aggiuntive. Un'icona di nome ScratchGPIO7 che permette di fare esperimenti con semplici circuiti ed un'icona di nome ScratchGPIO7 Plus che contiene le configurazioni predefinite per poter interfacciare numerose schede add-on disponibili in commercio. Visto che parliamo di icone e di entrate nel menu di "start", ne approfittiamo per approfondire uno dei principali cambiamenti avvenuti nell'interfaccia grafica Lxde (Lightweight X11 Desktop Environment) di Raspberry Pi, con il rilascio della versione di Raspian datata 22-12-2014. Il rilascio successivo, 31-1-2015 è finalizzato al supporto della nuova versione 2 di Raspberry Pi, con il nuovo processore Cortex A7 quadricore ed un GB di memoria RAM, che

la rende del tutto equivalente ad un piccolo PC desktop. A questa nuova versione di Raspberry Pi abbiamo dedicato un articolo di presentazione in questo stesso numero della rivista. Per inciso, tenete presente che al momento ScratchGPIO7 non "gira" sul nuovo Raspberry Pi 2 a causa della necessità di aggiornare alcune librerie dal quale dipende. Al momento dell'uscita della rivista il problema potrebbe essere già risolto. Tornando all'interfaccia grafica, la nuova versione prevede la barra delle applicazioni in alto sullo schermo, con il menu start in alto a sinistra. Questa soluzione è adottata da numerose distribuzioni GNU/Linux e sembra essere ispirata dalla somiglianza con il modo di utilizzare i libri tradizionali, che si scorrono dall'alto in basso e da sinistra a destra, almeno nella cultura occidentale, ed anche per uniformarsi alle applicazioni mobili, che hanno la barra delle notifiche in alto. La "scrivania" viene proposta vuota, con il solo "cestino" visibile in alto a sinistra. Tutte le altre icone possono essere posizionate a piacere sulla scrivania a partire dal menu start. Una volta individuata un'applicazione, è possibile portarne l'icona sulla scrivania, semplicemente, cliccandovi sopra col tasto

destro del mouse e selezionando l'opzione "Add to desktop". Per spiegare il funzionamento delle nuove istruzioni presenti nel pacchetto ScratchGPIO7 usiamo il "solito" metodo del LED e dell'interruttore. Dunque realizziamo il piccolo circuito, lo schema del quale è visibile in Fig. 5.

Noi lo abbiamo realizzato semplicemente saldando la resistenza direttamente sul terminale positivo del LED e "allungando" i due reofori con due spezzoni di cavetto terminati con un connettore femmina ciascuno. Lo stesso per l'interruttore, nel nostro caso abbiamo trovato un modello con i terminali distanti 5 mm tra loro. Abbiamo saldato sui terminali uno spezzone di strip femmina a cinque poli, in modo da lasciare liberi i poli pari. Il componente così realizzato trova perfetta corrispondenza con i pin +3.3V, GPIO9, e GND di Raspberry Pi, corrispondenti rispettivamente ai pin fisici 17, 21 e 25. Collegheremo il terminale di massa del LED al pin fisico 9 ed il positivo al pin fisico 7, che corrisponde all'uscita GPIO4, anch'essi "vicini" tra di loro, come si può notare consultando lo schema del GPIO in Fig. 6. La foto dell'insieme è visibile in Fig. 7.

Per collegare i due componenti così realizzati è raccomanda-

			P1					
<50mA	3V3		1	2		5V		
BCM GPIO00/02	SDA0/1	8	3	4		5V		
BCM GPIO01/03	SCL0/1	9	5	6		GND		
BCM GPIO04		7	7	8	15	TX	BCM GPIO14	
	GND		9	10	16	RX	BCM GPIO15	
BCM GPIO17		0	11	12	1	PWM0	BCM GPIO18	
BCM GPIO21/27		2	13	14		GND		
BCM GPIO22		3	15	16	4		BCM GPIO23	
<50mA	3v3		17	18	5		BCM GPIO24	
BCM GPIO10	SPIMOSI	12	19	20		GND		
BCM GPIO9	SPIMOSO	13	21	22	6		BCM GPIO25	
BCM GPIO11	SPI SCLK	14	23	24	10	SPI CE0 N	BCM GPIO08	
	GND		25	26	11	SPI CE1 N	BCM GPIO07	

Fig. 6

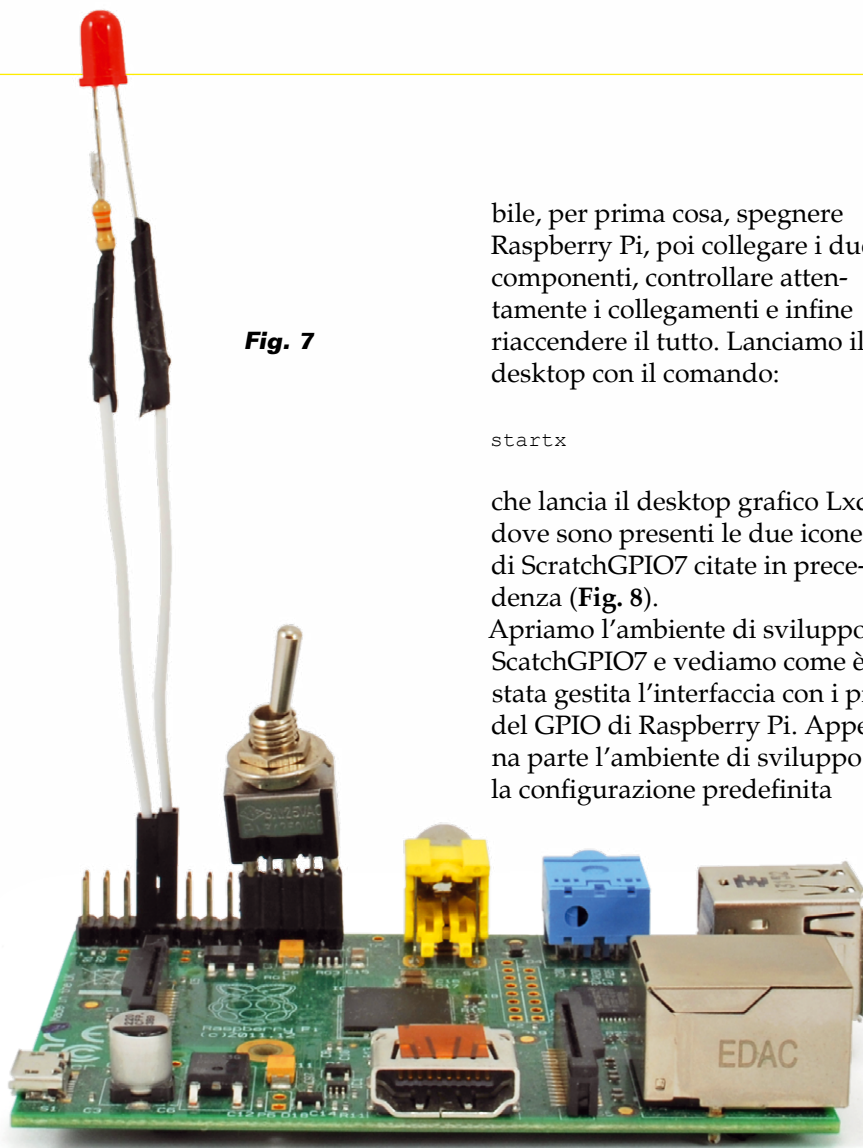


Fig. 7

bile, per prima cosa, spegnere Raspberry Pi, poi collegare i due componenti, controllare attentamente i collegamenti e infine riaccendere il tutto. Lanciamo il desktop con il comando:

```
startx
```

che lancia il desktop grafico Lxde dove sono presenti le due icone di ScratchGPIO7 citate in precedenza (**Fig. 8**).

Apriamo l'ambiente di sviluppo ScratchGPIO7 e vediamo come è stata gestita l'interfaccia con i pin del GPIO di Raspberry Pi. Appena parte l'ambiente di sviluppo la configurazione predefinita

prevede alcuni pin configurati come input ed altri come output. In particolare sono impostati come input i pin 22, 7, 3, 5, 24, 26, 19, 21, 23, 8 e 10 e come output i pin 11, 12, 13, 15, 16 e 18. Tenete presente che la numerazione utilizzata in ScratchGPIO7 fa corrispondere i pin alla loro posizione fisica sul connettore GPIO e non al nome di "sistema" del pin. Potete personalizzare la lingua nella quale sono presentate le istruzioni di ScratchGPIO7 cliccando sull'icona in alto a sinistra, a forma di piccolo "mondo" e scegliendo la lingua desiderata, come visibile in **Fig. 9**. Una delle istruzioni che permettono di interagire con i pin è la "invia a tutti" ("broadcast") visibile in **Fig. 10**. "Invia a tutti", in Scratch, è l'istruzione utilizzata per inviare messaggi ad altri script o, come in questo caso, a periferiche di sistema. Nello spazio "messaggio" dell'istruzione possiamo impostare lo stato dei pin inserendo "messaggi" costruiti con una sintassi particolare. Per portare

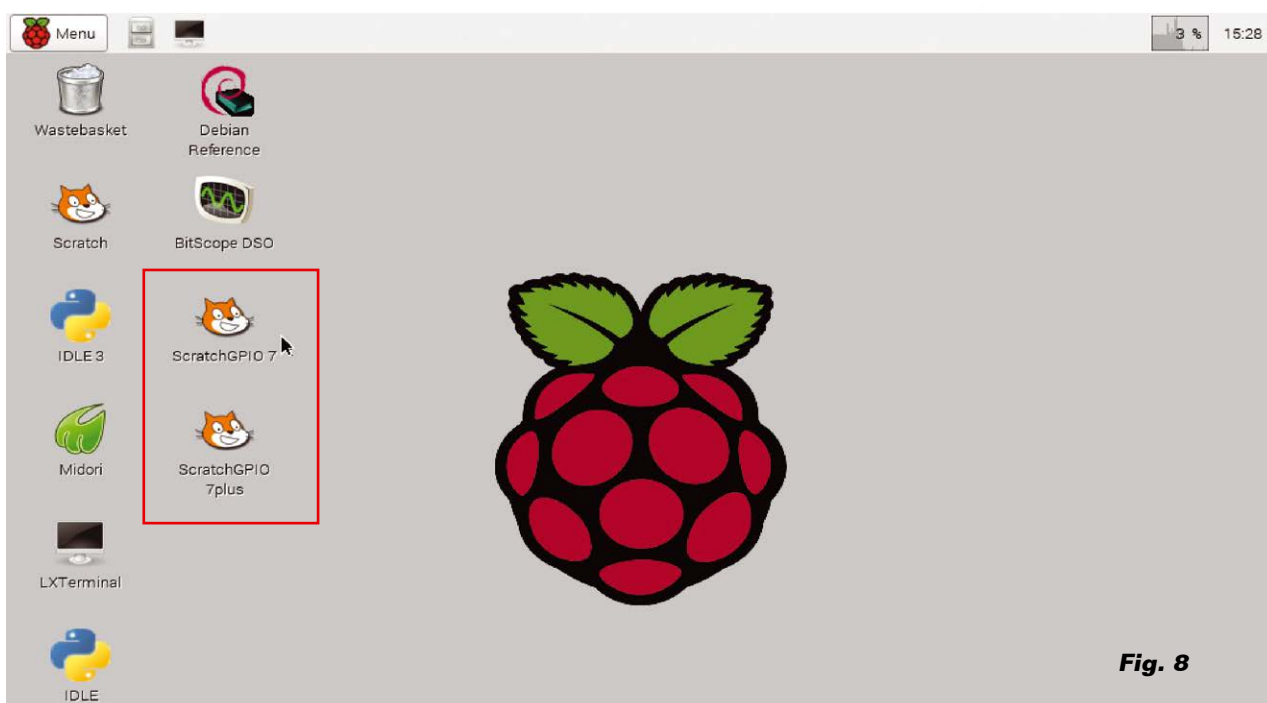


Fig. 8



Fig. 9

desiderata. È possibile anche alzare o abbassare i livelli di tutti i pin contemporaneamente con i messaggi "allon" e "alloff". Nello stesso messaggio possono essere concatenati più "ordini" come "pin7on pin5off".

In questo modo possiamo scrivere il nostro primo programma Scratch in grado di spegnere ed accendere un LED, il "listato" del quale è visibile in **Fig. 11**.

Sono validi anche i messaggi "pin7on" e "pin7off", come visibile nel "listato" di **Fig. 12**.

Con questo metodo potremmo riprodurre realmente il semaforo presentato nel primo articolo di questa serie, utilizzando tre LED, uno rosso, uno giallo ed uno verde. Piccola digressione. Un altro modo di interfacciare i pin è di creare delle variabili, nominarle in modo standard con i nomi dei pin come "pin26" o "pin19". In questo modo avremo a disposizione (**Fig. 13**):

- una variabile utilizzabile nel campo "variabile" delle istruzioni che lo prevedono;
- le istruzioni predefinite per l'impostazione dei valori da assegnare alle variabili;
- la possibilità di rendere visibile la variabile sul "palcoscenico".

Un piccolo accorgimento ma importante per evitare di perdere tempo a capire le cause di quello che sembra un malfunzionamento. Scratch, almeno fino alla versione attuale, "ricorda" lo stato di una variabile ed imposta un nuovo valore solo se questo è diverso da quello corrente. Quando si "clicca" sulla bandierina verde, Scratch invia solo i valori delle variabili che sono stati modificati rispetto alle esecuzioni precedenti, in pratica, le variabili che non hanno cambiato valore non vengono inviate e lo stato dei pin

può apparire incongruente con ciò che ci aspettiamo.

Un modo per "girare attorno" a questo problema è quello di impostare le variabili all'inizio del programma, prima del primo ciclo "loop" a valori simbolici, anche di tipo non corrispondente ai dati attesi dalla variabile,



Fig. 10



Fig. 11



Fig. 12



Fig. 13

il pin 7 a livello alto utilizziamo il messaggio "pin7high" mentre per portarlo a livello basso utilizziamo il messaggio "pin7low". Non esiste controllo formale per i messaggi, quindi è necessario prestare la massima attenzione alla loro scrittura. In caso di errore, durante la composizione dello script, appare un messaggio di errore generico e, una volta lanciata l'esecuzione, il contorno dello script appare rosso invece che bianco.

In questi casi è molto utile la funzione di "aiuto", che si attiva "cliccando" sull'istruzione

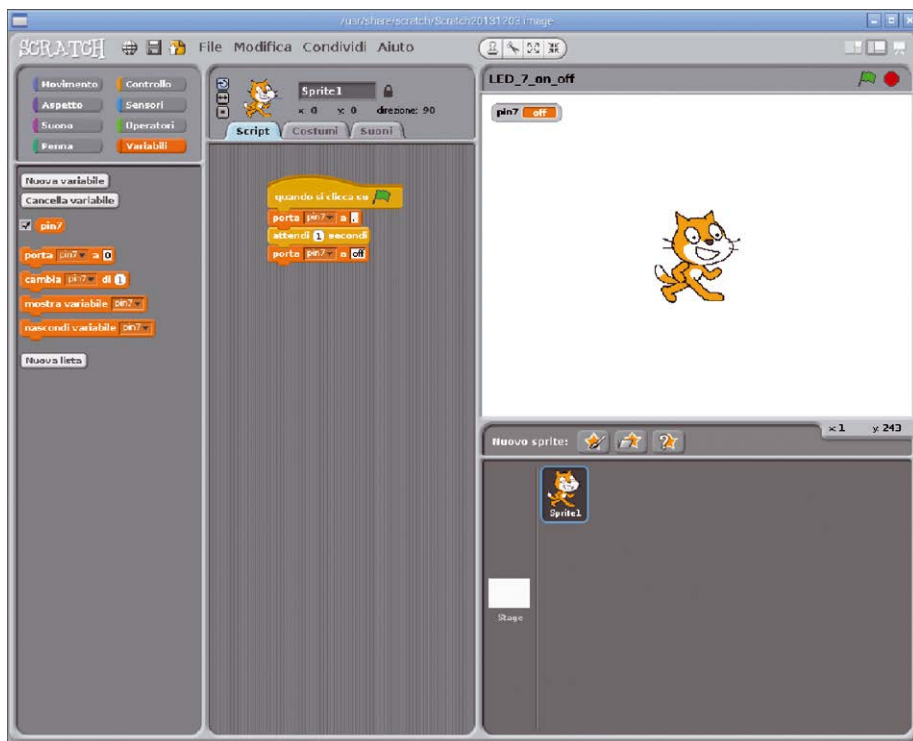


Fig. 14

dell'istruzione è di "intercettare" il cambiamento di stato, "0" quando il deviatore è collegato a massa e "1" quando collegato alla tensione di +3,3V.

Se per le vostre applicazioni avete bisogno di un numero di pin di output superiore a quello messo a disposizione nella configurazione predefinita, è sufficiente per esempio, eseguire un'istruzione "invia a tutti" ("broadcast") sul pin 10, impostato in modo predefinito come "input", per esempio "invia a tutti pin10on", per trasformarlo automaticamente in un pin di output. Il pacchetto ScratchGPIO7 comprende la possibilità di riconoscere e gestire un

per esempio con ".". Nel ciclo normale, poi, impostiamo i valori desiderati, come nell'esempio di **Fig. 14**.

Tornando al programma di accensione e spegnimento del LED, clicchiamo sulla bandierina per far partire l'esecuzione del programma. Inizialmente abbiamo collegato il terminale centrale del nostro deviatore al pin 21 e i terminali estremi rispettivamente a massa ed al pin +3,3V di Raspberry Pi (pin 25 e 17). Con questa configurazione possiamo realizzare un piccolo programma per verificare in pratica l'utilizzo delle istruzioni di gestione dei pin in ingresso. Andiamo nella sezione "Sensore" ed ispezioniamo le opzioni disponibili nell'istruzione "valore del sensore". Si può notare che sono indicati i pin 22, 7, 3, 5, 24, 26, 19, 21, 23, 8 e 10, ovvero quelli predefiniti come pin di ingresso, come mostrato in **Fig. 15**.

Creiamo un programma come quello visibile in **Fig. 17** dove l'istruzione centrale è quella indicata con la freccia. Scopo

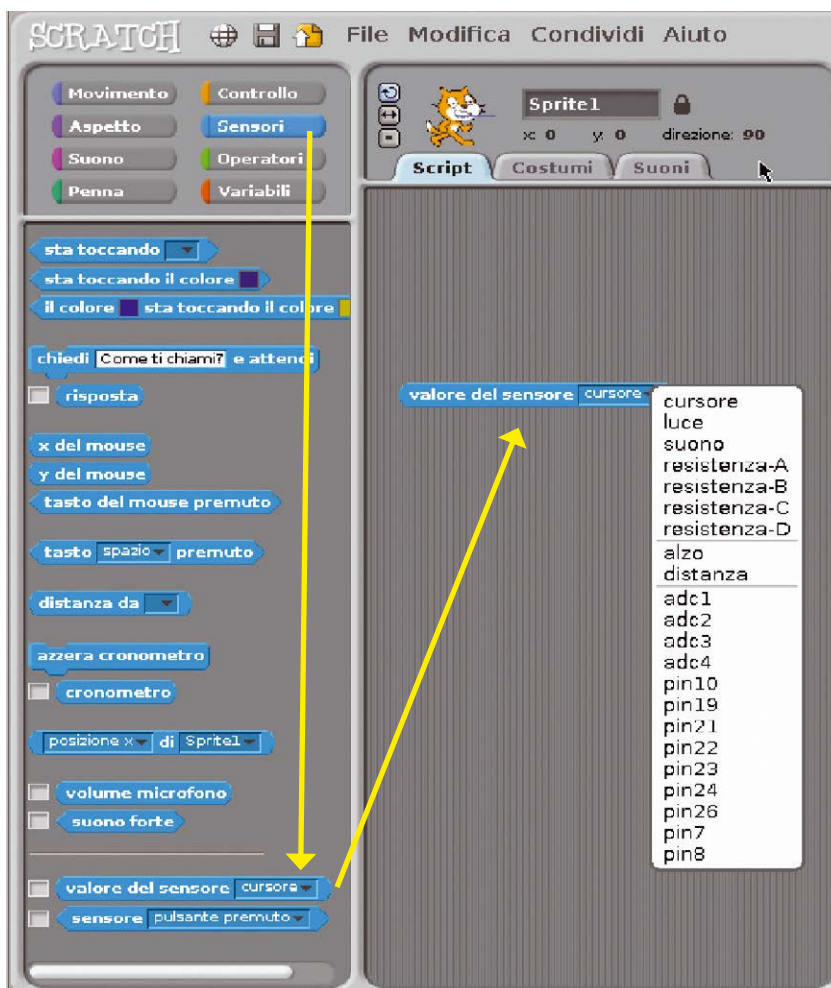
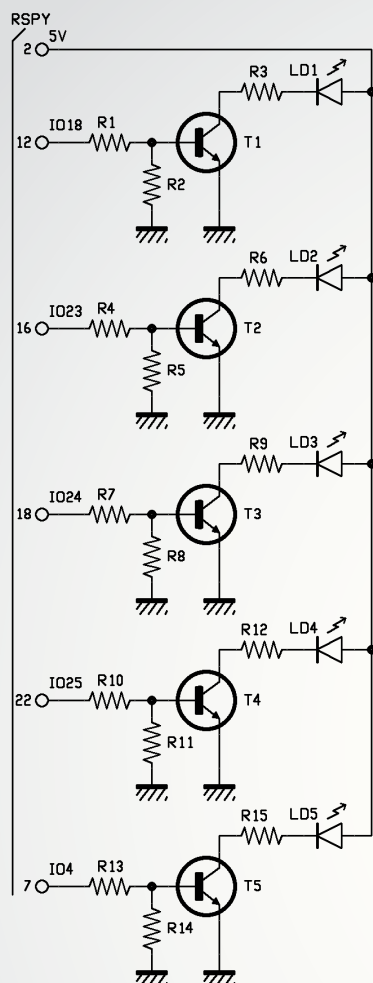
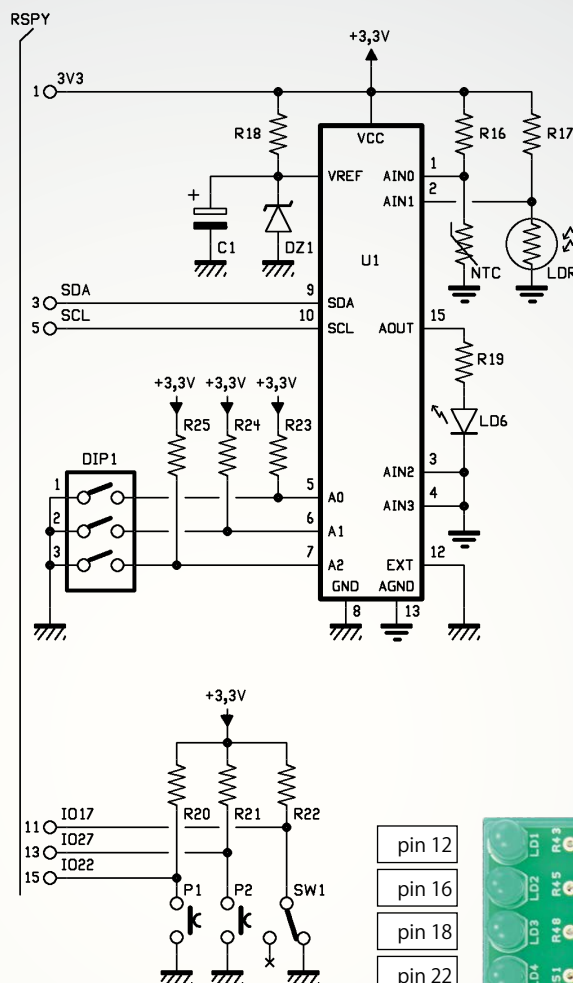


Fig. 15



Elenco Componenti:

R1, R4, R7, R10, R13, R20÷R25: 4,7 kohm
 R2, R5, R8, R11, R14: 10 kohm
 R3, R6, R9, R12, R15, R19: 330 ohm
 R16, R17: 15 kohm
 R18: 100 ohm
 C1: 4,7 μ F 100 VL elettrolitico
 DZ1: Zener 3,3V 400mW
 T1÷T5: BC547
 LD1÷LD5: LED 5 mm verde
 LD6: LED 5 mm rosso
 U1: PCF8591



P1, P2: Microswitch
 SW1: Deviatore slitta verticale
 DIP1: Dip-Switch 3 vie
 NTC: NTC 10 kohm
 LDR: Fotoresistenza 2-20 kohm

Varie:
 - Zoccolo 8+8
 - Strip Femmina 13x2 vie per Raspberry Pi
 - Distanziale M/F 16mm
 - Vite 8 mm 3 MA (2 pz.)
 - Circuito stampato

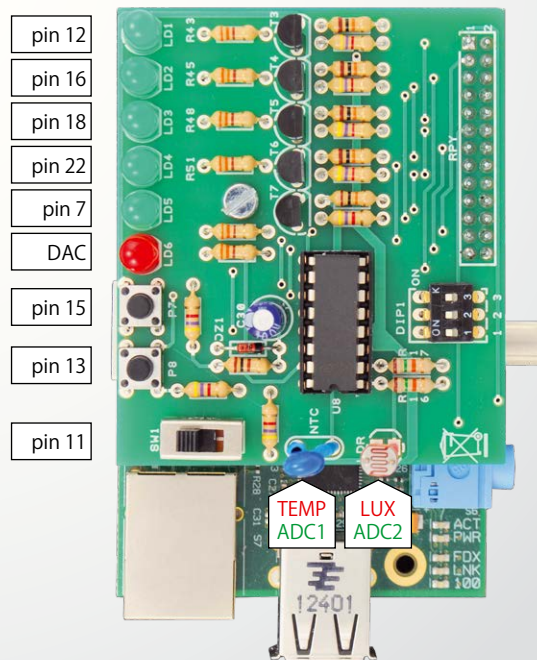


Fig. 16

certo numero di shield disponibili sul mercato. Una di queste permette a Raspberry Pi di acquisire dati analogici interfacciando l'integrato PCF8591 per mezzo del bus I2C, esattamente come avviene per la scheda FT1060, di

corredo al libro "Raspberry Pi – Il mio primo Linux embedded". L'integrato PCF8591, come visibile nello schema di Fig. 16 dispone di quattro ingressi analogico digitali a otto bit ed una uscita DAC, sempre ad otto bit.

All'ingresso ADC1 dello shield è collegato un termistore (NTC nello schema) che possiamo utilizzare per misurare la temperatura ambiente. All'ingresso ADC2 è collegata una fotoresistenza (LDR nello schema) che possiamo



Fig. 17

utilizzare per misurare la luminosità ambiente. Infine all'uscita DAC è collegato un LED rosso (LD6) che potremo far illuminare con un'intensità proporzionale al valore digitale impostato sull'uscita DAC stessa. Nel libro citato potete approfondire le caratteristiche dell'integrato PCF8591. Per provare questa funzionalità, spegniamo Raspberry Pi con il comando:

```
sudo shutdown -h now
```

Smontiamo il LED ed il deviatore e posizioniamo la scheda FT1060 sul connettore GPIO di Raspberry Pi, facendo bene

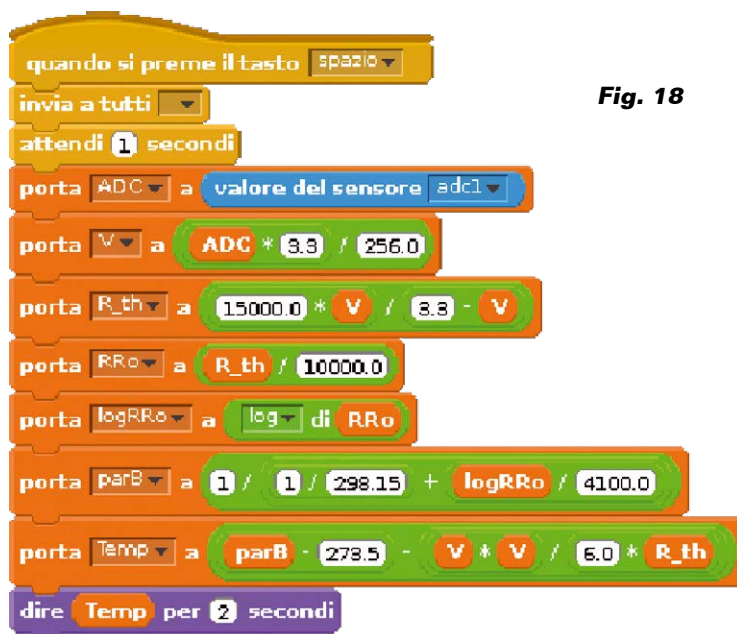


Fig. 18

attenzione a far corrispondere i pin dello shield con quelli del microcomputer. Dopo aver controllato tutto ancora una volta riaccendiamo Raspberry Pi ed attendiamo la

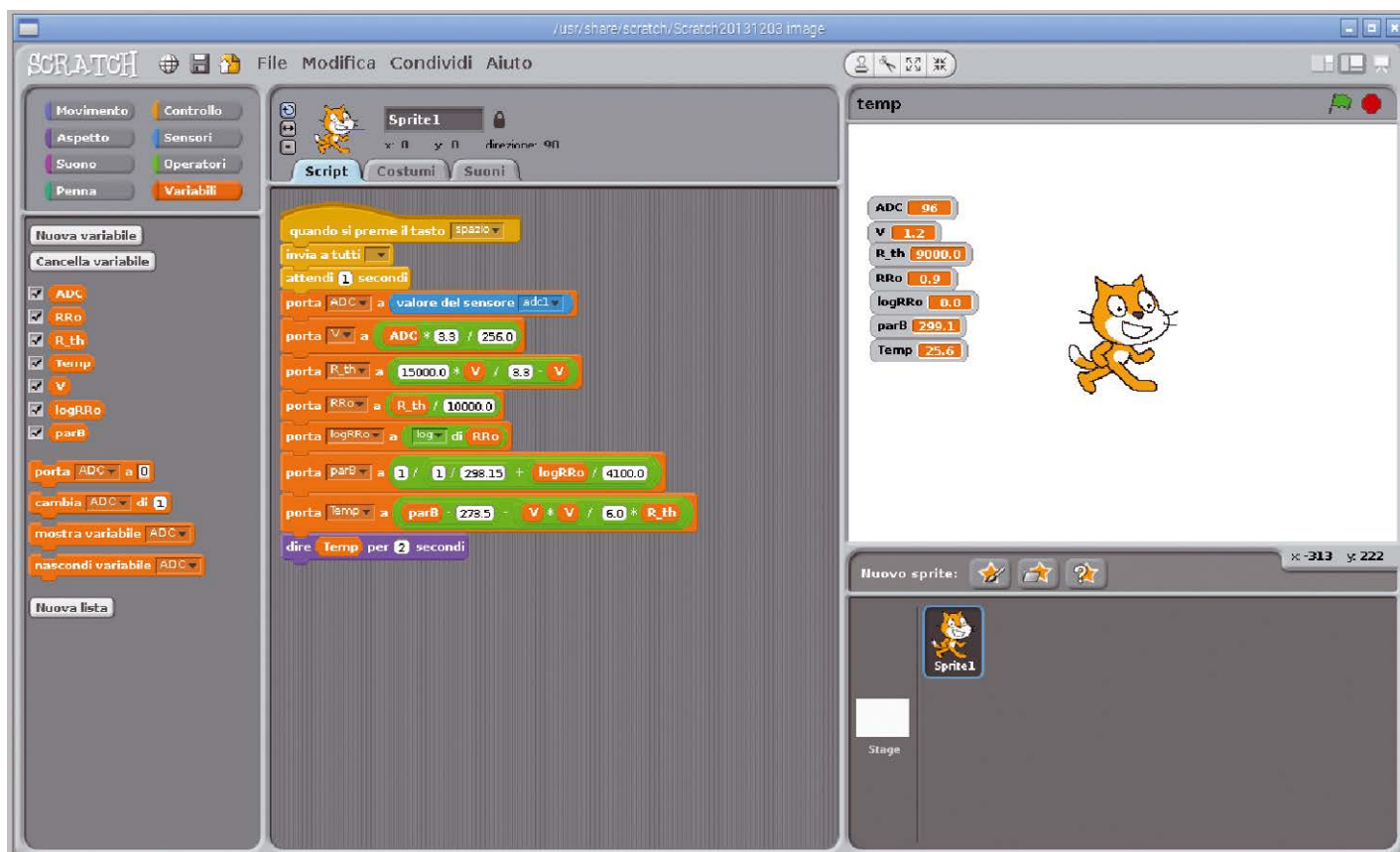
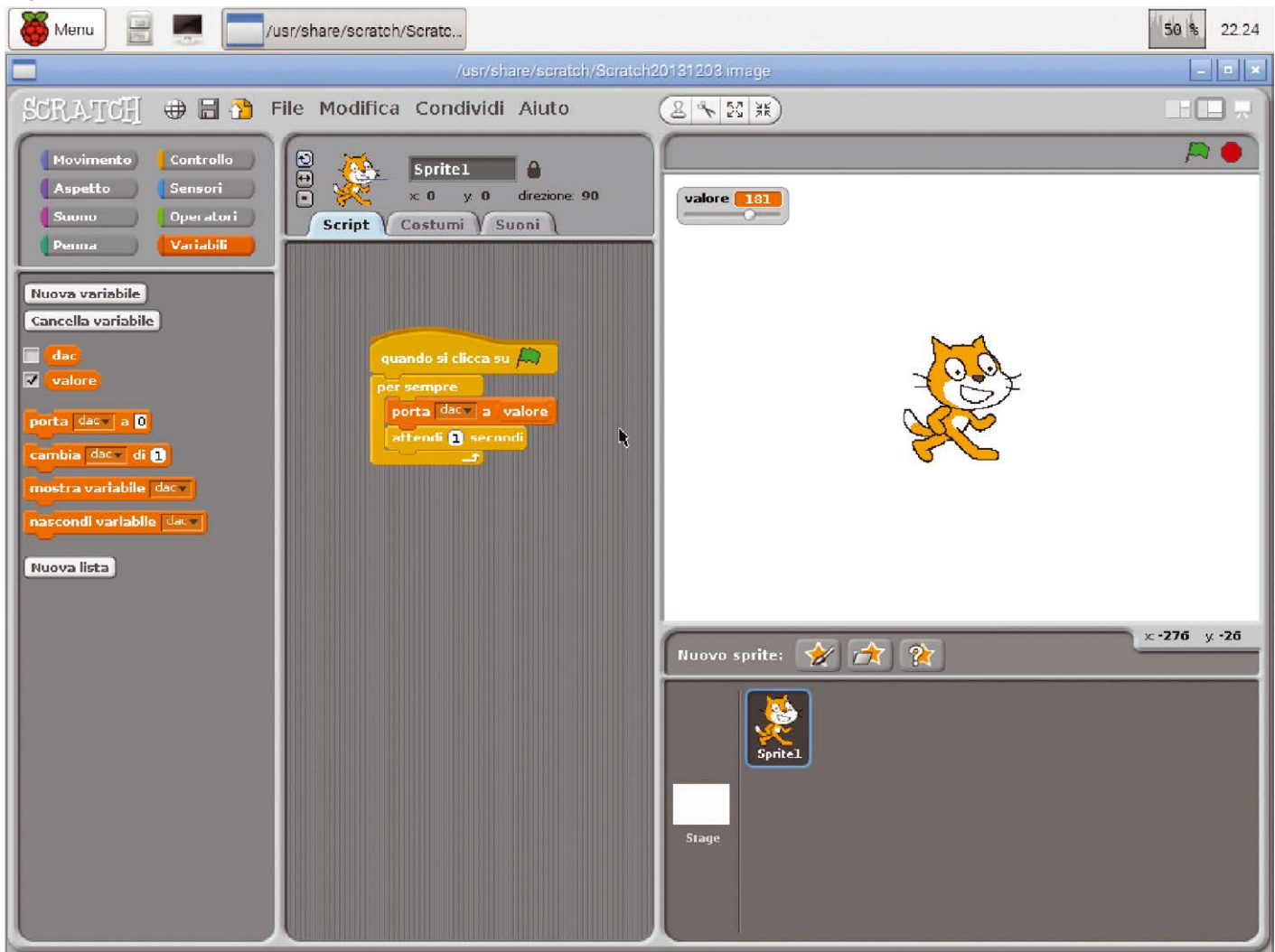


Fig. 19

Fig. 20

fine del processo di boot. Ci connettiamo con l'utente generico "pi" e password "raspberrypi", lanciamo l'interfaccia grafica con il comando "startx" ed il programma ScratchGPIO7. Prima però verifichiamo se abbiamo attivato il driver di gestione del bus I2C.

**Fig. 21**

Se così non fosse, e stiamo utilizzando le versioni di Raspbian più recenti, apriamo una finestra di terminale e lanciamo il configuratore raspi-config. Selezioniamo la voce di menu "I2C" e nelle pagine di configurazione successive rispondiamo "yes" a tutti i passaggi. Se ci viene chiesto di riavviare il microcomputer rispondiamo affermativamente e attendiamo il riavvio. Nel nostro caso è stato necessario attivare il bus I2C anche nel modo tradizionale, commentando l'entrata "# blacklist i2c-bcm2708" nel file /etc/modprobe.d/raspi-blacklist.conf, dare il comando modprobe i2c-dev e installare il pacchetto

to "i2c-tools". Altro riavvio per poi riportarci nell'applicazione ScratchGPIO7. Ora proviamo a leggere il valore del sensore di temperatura collegato all'ingresso ADC1 dell'integrato PCF8591. Per eseguire la lettura utilizziamo l'istruzione "invia a tutti ADC1". Per acquisire il risultato utilizziamo la "variabile" "valore del sensore adc1", come potete vedere nel programma di Fig. 18 e nella vista d'insieme di Fig. 19, dove abbiamo anche riprodotto in Scratch le operazioni matematiche per calcolare la temperatura a partire dal valore ADC, con il metodo del "parametro B" descritto numerose volte nella

rivista e nel libro citato all'inizio. Per impostare l'uscita DAC con un valore digitale da convertire in una tensione analogica proporzionale al valore impostato si utilizza l'istruzione "porta dac a <valore>" dove <valore> è un numero compreso tra "0" e "255". Il programma visibile in **Fig. 21** permette di impostare la luminosità del LED rosso proporzionalmente alla posizione del cursore (**Fig. 20**).

Per i nostri esperimenti, della scheda FT1060, con riferimento allo schema di Fig. 16, possiamo utilizzare, oltre ai convertori ADC e DAC, anche i seguenti pin:

- Come Uscite possiamo utilizzare i pin 12 (GPIO18), pin 16 (GPIO23), pin 18 (GPIO24), pin 22 (GPIO25) e pin 7 (GPIO4). Ciascuna di queste uscite è provvista di un LED pilotato da un transistor configurato ad emettitore comune per evitare di sovraccaricare i pin del GPIO.
- In ingresso abbiamo a disposizione due pulsanti collegati ai pin 13 (GPIO27) e pin 15 (GPIO22). Un deviatore è collegato al pin 11 (GPIO17). Prima di utilizzare questi pin ricordiamo di impostarli come pin di ingresso con istruzioni

del tipo "invia a tutti Config11In".

In ScratchGPIO7 l'istruzione "invert" permette di invertire la logica di comportamento di un pin, per esempio riconoscere il pin come On quando il livello di tensione è basso e viceversa. L'istruzione "invia a tutti Config11In" permette di impostare in modo esplicito il pin 11 da output ad input. L'istruzione "invia a tutti GetIP" permette di "leggere" l'indirizzo IP di Raspberry Pi, che può poi essere evidenziato con l'istruzione "dire" e la variabile "valore del sensore ipaddress" come visibile in **Fig. 22**. ■

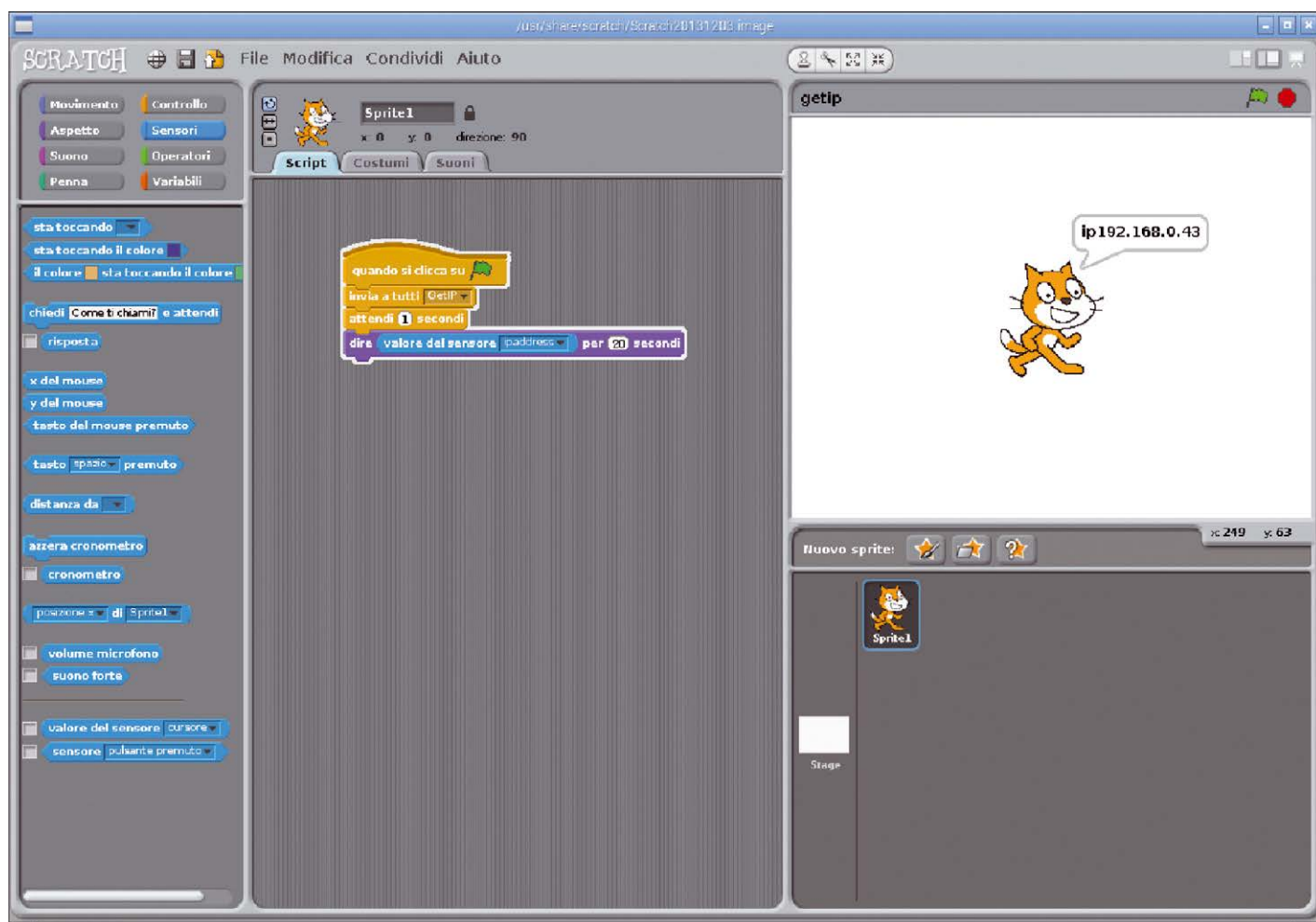


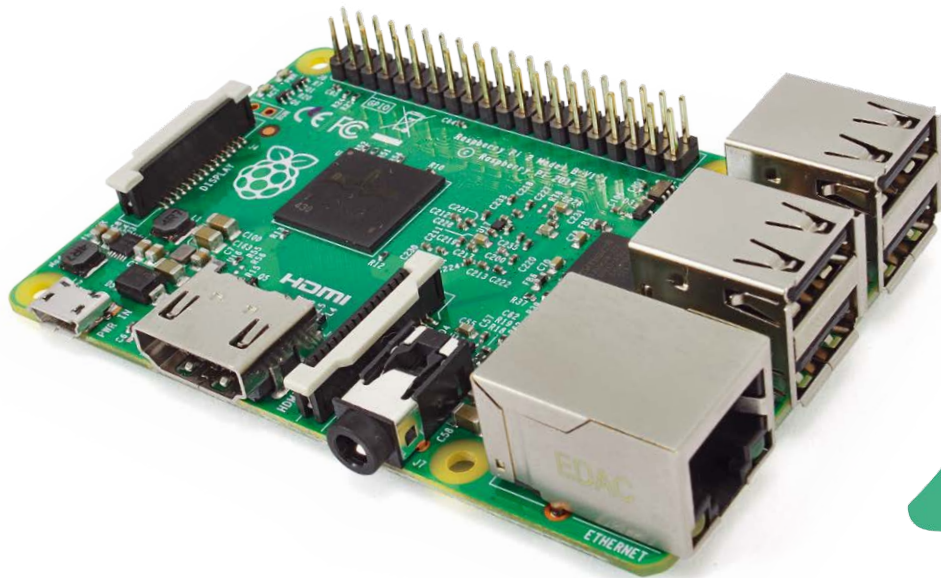
Fig. 22

Mensile di elettronica applicata, attualità scientifica, novità tecnologiche.

Elettronica In

www.elettronica.in.it

oltre l'elettronica



Utilizziamo Pushetta, un servizio web che permette di inviare notifiche di eventi a smartphone iOS e Android.



NOTIFICHE PUSH DA RASPBERRY PI

di FABRIZIO GUGLIELMINO

Sempre più spesso ci troviamo a realizzare progetti connessi ad Internet, che rientrano in quella categoria ormai nota con il nome di Internet of Things (IoT) alla quale appartengono elettrodomestici "intelligenti", sistemi di home automation, dispositivi medicali, sistemi di sicurezza e innumerevoli altri, tutti accomunati dalla capacità di connettersi alla Rete. L'IoT ha coinvolto anche il mondo dei maker, tanto che da tempo si trovano progetti basati su Arduino, o schede simili, che collegano ad Internet il frigorifero, la lavatrice o la macchinetta del caffè. Malgrado l'eterogeneità di questi progetti, una caratteristica li accomuna quasi tutti: la necessità di comunicare con l'utente. Se il frigorifero fosse in grado di inviarci una notifica quando no-

stro figlio ha consumato l'ultima bottiglia di latte, potremmo provvedere, rientrando dal lavoro ed evitando di uscire nuovamente. Essendo un tema di attualità, abbiamo deciso di farci coinvolgere e di coinvolgervi in un'applicazione basata sulla popolare e prestante Raspberry Pi; quello che faremo in questo articolo sarà utilizzare uno shield di sperimentazione per Raspberry Pi -l'FT1060M- per simulare condizioni legate alla variazione di grandezze fisiche come la temperatura, inviando degli allarmi su soglie predefinite. Questa applicazione vi permetterà di acquisire le competenze necessarie ad adattarla alle vostre necessità per sviluppare a vostra volta applicazioni dedicate. L'esigenza di allertare un utente da remoto non è nuova, anzi: i



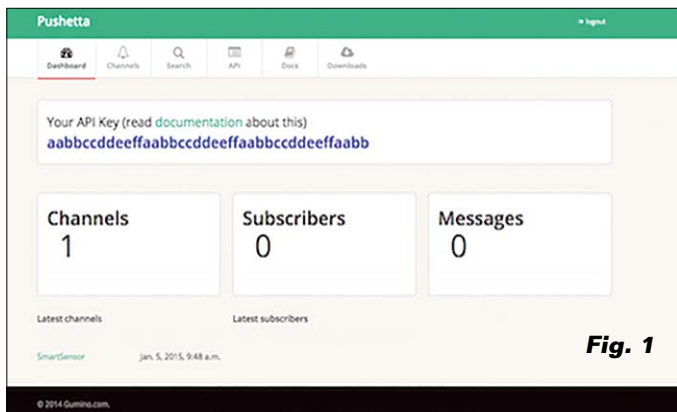


Fig. 1

sistemi più usati finora per rispondervi sono stati gli SMS e in casi minori l'e-mail. Entrambe le soluzioni sono a loro modo efficienti ma oggi, grazie in particolare alla diffusione degli smartphone, abbiamo una possibilità in più: le push notification.

NOTIFICHE PUSH: COSA SONO?

Con il termine push notification si identifica un preciso meccanismo che prevede la possibilità di inviare messaggi a dispositivi connessi alla Rete. La diffusione delle notifiche push si deve all'adozione da parte dei sistemi operativi per smartphone: la prima ad introdurle è stata Apple, con la versione 3.0 di iOS. Le push notification si distinguono dai metodi citati in precedenza per varie ragioni, ma la più significativa è probabilmente il modello funzionale. Per inviare un SMS ci basta conoscere il numero destinatario; altrettanto vale per l'e-mail, per spedire la quale basta conoscere l'indirizzo del destinatario. Nel caso delle notifiche push il meccanismo è invece quello che tecnicamente viene chiamato "publish/subscribe": è l'utente che "sottoscrive" un certo topic (o canale) per il quale vuole ricevere le notifiche, ed è in grado di annullare la sottoscrizione in qualsiasi momento. Il funzionamento delle push notification è abbastanza complesso, ma fortunatamente possiamo descriverci l'applicazione senza dover scendere nei dettagli tecnici. Quello che invece vedremo in modo approfondito è Pushetta, un servizio nato con il preciso compito di rendere le push notification utilizzabili in modo semplice da chiunque. Pushetta può essere considerato una sorta di gateway che intermedia la comunicazione tra chi invia le notifiche e chi le riceve. Il problema di fondo delle notifiche push è che non è prevista una specifica applicazione per riceverle, come è ad esempio per gli SMS o le e-mail. Sono più che altro uno strumento in mano allo sviluppatore di App che decide il formato dei dati da inviare e come interpretarli alla ricezione. Questo implica una grossa complicazione per lo scenario inizialmente descritto, infatti per fare in modo che il nostro frigorifero sia

in grado di mandarci una notifica, dovremmo anche realizzare un'App specifica per il nostro smartphone! Pushetta risolve questo problema: offre le applicazioni abilitate a ricevere le notifiche ed un meccanismo semplice per inviarle. Sia le App che il servizio sono totalmente gratuiti. Il primo step per lavorare con Pushetta è andare su <http://www.pushetta.com>, iscriversi e scaricare l'App per il proprio telefono dalla pagina "downloads".

Registrandosi a Pushetta si ottiene un token di autenticazione, definito API Key, che serve per autorizzare le chiamate alle API che useremo in seguito. L'API key è visualizzata accedendo alla propria Dashboard sul sito (Fig. 1).

Prendiamo quindi nota dell'API Key in modo da non dover tornare ogni volta alla Dashboard quando scriveremo il codice per l'invio delle notifiche. L'invio delle notifiche con Pushetta si basa sul concetto di "canale"; riprendiamo per un momento l'esempio del frigo smart al quale siamo ormai affezionati. Dobbiamo creare un canale che fungerà da identificativo per il meccanismo di "publish/subscribe"; per gli scopi dell'esempio lo chiameremo "Frigo" (in un impeto di originalità). Quando invieremo una notifica specificheremo di utilizzare il canale "Frigo"; dall'App sottoscriveremo lo stesso canale e ogni notifica inviata ad esso ci sarà automaticamente recapitata.

La creazione di un canale è un'operazione semplicissima: dal menu "Channels" si accede alla lista di quelli appartenenti all'utente (quando ci si registra per la prima volta è ovviamente vuota). Con il pulsante "Add a channel" accediamo alla pagina per

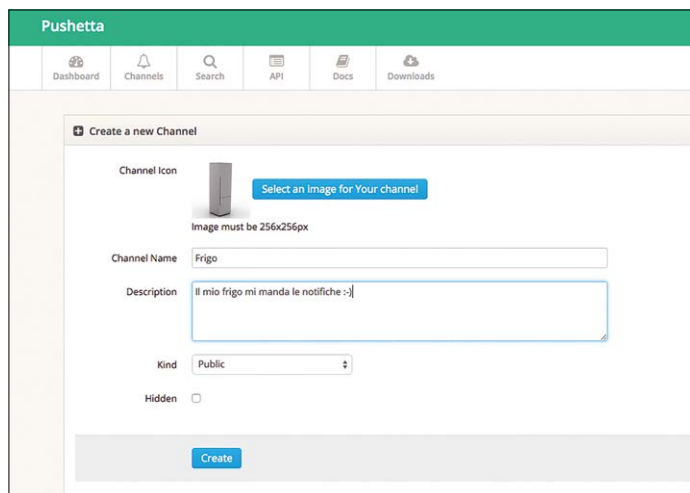


Fig. 2 - I passi per predisporci all'utilizzo di Pushetta sono tutti qui; una volta che ci siamo registrati ed abbiamo creato uno o più canali il sistema è pronto ad inviare notifiche.

crearne uno nuovo che richiede un'immagine per identificarlo, un nome non utilizzato da altri ed una descrizione. Possiamo creare canali pubblici o privati: per essere sottoscritti, quest'ultimi richiedono l'autorizzazione del proprietario.

Infine possiamo decidere di mantenere nascosto al sistema di ricerca il canale; in questo caso per sottoscriverlo bisognerà conoscere il nome a priori (o usare l'url univoco che viene automaticamente generato per ogni canale).

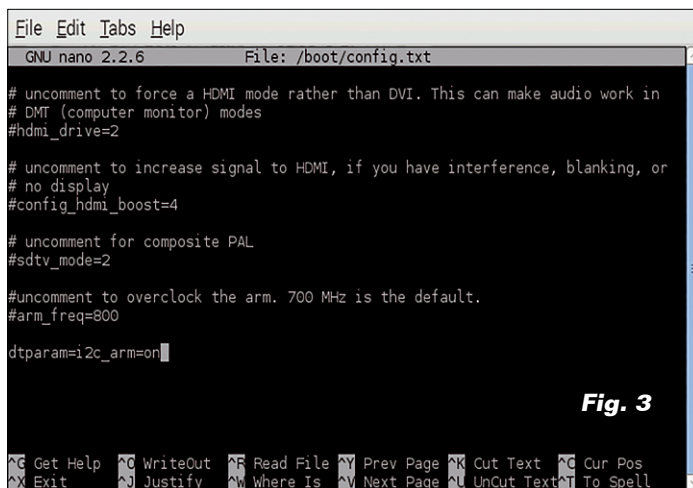
CONFIGURIAMO RASPBERRY PI

Passiamo ora alla configurazione di Raspberry Pi, che non spiegheremo dalla base: vista l'abbondanza di tutorial presenti sul web, ci limiteremo a fornire le sole indicazioni specifiche per il nostro progetto. Nel nostro esempio utilizziamo la distribuzione Raspbian del 31.01.2015; tenete conto che release differenti potrebbero richiedere piccoli adattamenti.

Partiamo quindi dall'assunto che la scheda abbia il sistema operativo installato, che risulti connessa ad Internet e che si sia in grado di utilizzare la shell di sistema direttamente da tastiera o con connessione remota via SSH. Lo shield di sperimentazione utilizza il bus I²C per la comunicazione con alcuni sensori (in particolare l'NTC per la temperatura e il fotoresistore per la luminosità). Dobbiamo quindi configurare la distribuzione in modo che carichi i driver per l'utilizzo di questo bus.

Nelle release precedenti di Raspbian, il bus I²C non richiedeva parametri di boot per il kernel ed era solo necessario agire sul caricamento dei moduli necessari. Nell'ultima versione la situazione è un po' cambiata, perché sebbene i moduli vadano ancora caricati esplicitamente, è necessario uno specifico parametro di boot per il kernel per attivare il bus; per questa ragione modifichiamo `/boot/config.txt` aggiungendo la riga `dtoverlay=i2c-arms` come visibile in Fig. 3. Per modificare i vari file useremo l'editor nano: senza dilungarci in un corso sul suo utilizzo, diciamo che quello che ci serve è sapere come aprire un file, specificando `nano nomefile`, e salvare le modifiche apportate, con la combinazione di tasti CTRL + X. L'editor offre molte altre funzioni ma non è il caso di spiegarle qui; lasciamo a voi la scelta di approfondire o meno la materia, attingendo al materiale disponibile in Internet. Un'ultima nota prima di procedere: alcuni file non sono modificabili da un utente che non sia amministratore del sistema; in questi casi useremo `sudo nano nomefile` che, a fronte della richiesta password, eseguirà l'editor come se fosse lanciato da root.

Come anticipato, dobbiamo anche attivare il carica-



```
File Edit Tabs Help
GNU nano 2.2.6 File: /boot/config.txt

# uncomment to force a HDMI mode rather than DVI. This can make audio work in
# DMT (computer monitor) modes
#hdmi_drive=2

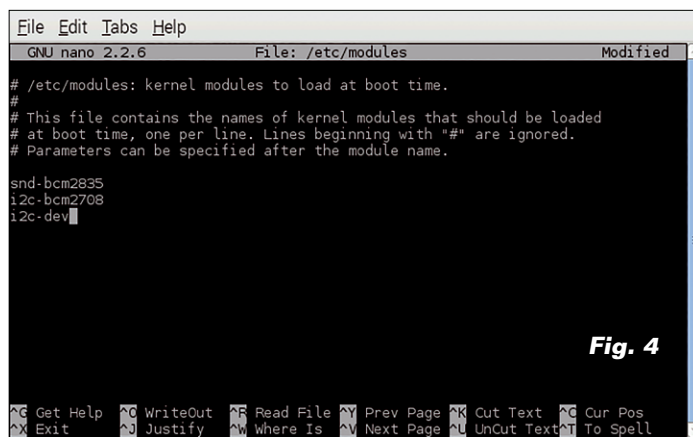
# uncomment to increase signal to HDMI, if you have interference, blanking, or
# no display
#config_hdmi_boost=4

# uncomment for composite PAL
#sdtv_mode=2

#uncomment to overclock the arm. 700 MHz is the default.
#arm_freq=800

dtoverlay=i2c-arms
```

Fig. 3



```
File Edit Tabs Help
GNU nano 2.2.6 File: /etc/modules Modified

# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.
# Parameters can be specified after the module name.

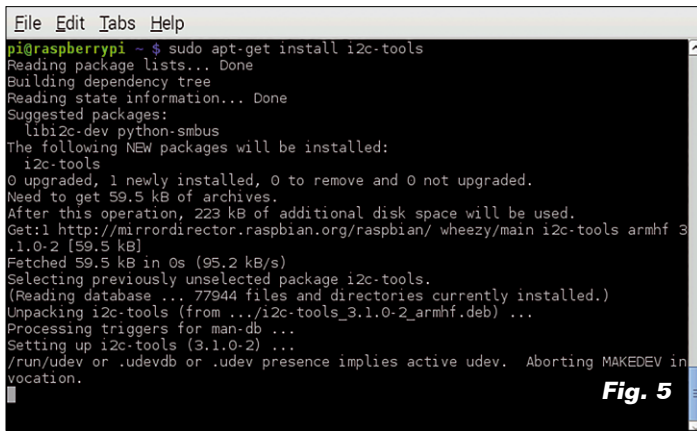
snd-bcm2835
i2c-bcm2708
i2c-dev
```

Fig. 4

mento dei moduli; in questo caso il file da modificare è `/etc/modules`, nel quale sono elencati quelli caricati al boot. I moduli in realtà possono essere caricati manualmente usando il comando `modprobe`, ma questo implicherebbe la necessità di ricordarsi di eseguirlo ogni volta che riavviamo la Raspberry Pi; pertanto è preferibile attivare il caricamento automatico. Eseguiamo quindi `sudo nano /etc/modules` e aggiungiamo `i2c-bcm2708` e `i2c-dev` (Fig. 4). Se facessimo un reboot ora, il sistema caricherebbe i moduli I²C, con relativa creazione in `/dev` di un nuovo device file. Questo device file, che utilizzeremo in modo indiretto per comunicare con il bus, per impostazione predefinita può essere acceduto solo da root o dagli utenti appartenenti al gruppo `i2c`. Per non dover usare `sudo` ogni volta che dobbiamo leggere o scrivere dal bus, aggiungiamo l'utente `pi` al gruppo specifico con il comando:

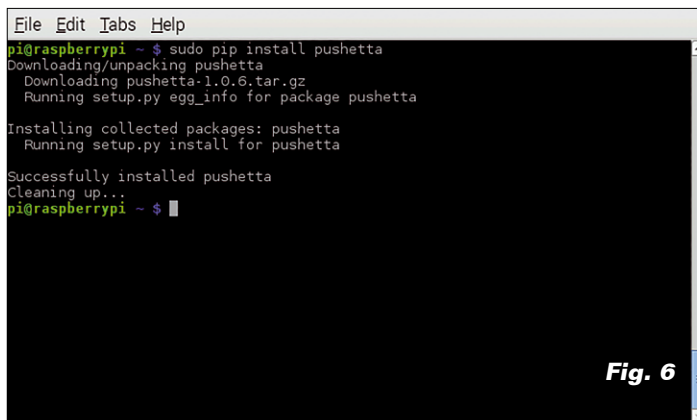
`sudo adduser pi i2c`

Ora la configurazione del bus è completa; installiamo i tool che ci serviranno per fare i primi test di comunicazione e verificare che tutto funzioni correttamente,



```
File Edit Tabs Help
pi@raspberrypi ~ $ sudo apt-get install i2c-tools
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  libi2c-dev python-smbus
The following NEW packages will be installed:
  i2c-tools
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 59.5 kB of archives.
After this operation, 223 kB of additional disk space will be used.
Get:1 http://mirrordirector.raspbian.org/raspbian/ wheezy/main i2c-tools armhf 3.1.0-2 [59.5 kB]
Fetched 59.5 kB in 0s (95.2 kB/s)
Selecting previously unselected package i2c-tools.
(Reading database ... 77944 files and directories currently installed.)
Unpacking i2c-tools (from .../i2c-tools_3.1.0-2_armhf.deb) ...
Processing triggers for man-db ...
Setting up i2c-tools (3.1.0-2) ...
/run/udev or .udevdb or .udev presence implies active udev. Aborting MAKEDEV invocation.
pi@raspberrypi ~ $
```

Fig. 5



```
File Edit Tabs Help
pi@raspberrypi ~ $ sudo pip install pushetta
Downloading/unpacking pushetta
  Downloading pushetta-1.0.6.tar.gz
    Running setup.py egg_info for package pushetta
Installing collected packages: pushetta
  Running setup.py install for pushetta
Successfully installed pushetta
Cleaning up...
pi@raspberrypi ~ $
```

Fig. 6

con il comando `sudo apt-get install i2c-tools` e la distribuzione è pronta per i nostri test (Fig. 5). Come già anticipato, nelle precedenti release di Raspbian gli step di configurazione I²C erano un po' diversi; in particolare, era necessario modificare il file `/etc/modprobe.d/raspi-blacklist.conf` rimuovendo i moduli `spi-bcm2708` e `i2c-bcm2708` per evitare che il loro caricamento venisse bloccato perché blacklisted. Se si dispone di queste release è bene fare riferimento alle indicazioni specifiche per la configurazione I²C che si trovano sul web.

UTILIZZO DELLO SHIELD

Passiamo ora al collegamento dello shield a Raspberry Pi, molto semplice perché lo shield è dotato di un connettore che va incastrato con il pettine dei GPIO disponibili sulla scheda. Nel caso si disponesse delle versioni B+, A+ o la nuova Raspberry Pi 2, che implementano un connettore più lungo, bisogna fare attenzione ad allineare il pin 1 con il medesimo della scheda. Lo shield dispone tra l'altro di un paio di pulsanti, del sensore di luminosità e di quello di temperatura: questo ultimo è quello che ci apprestiamo a utilizzare. Impostate i tre dip switch che definiscono l'indirizzo del bus I²C tutti su ON; pur non essendo obbligatoria, tale impostazione renderà più semplice utilizzare il codice senza dover apportare modifiche

per adattarlo alle vostre esigenze.

Scriveremo il codice degli esempi in Python avvalendoci di alcune librerie già pronte per interagire sia con Pushetta che con il bus dei sensori. Procediamo quindi installando questi prerequisiti.

La prima libreria è quella necessaria ad interagire con Pushetta: eseguiamo il comando `pip install pushetta`, come riportato nella Fig. 6. Se l'installazione va a buon fine, siamo già in grado di inviare la nostra prima notifica; l'operazione richiede solo tre righe di codice (di cui una per importare la libreria):

```
from pushetta import Pushetta
p = Pushetta("aabbccddeeff0011223344556677889900aabbcc")
p.pushMessage("Frigo", "Compra il latte!")
```

Spieghiamo cosa fa il codice: la prima riga importa la libreria Pushetta dal modulo omonimo, cosa abbastanza comune in Python. Il "vero" codice inizia alla riga successiva, che istanzia l'oggetto per la comunicazione con l'API di Pushetta. Qui dobbiamo sostituire l'API Key fittizia ("aabbccddeeff0011223344556677889900aabbcc") riportata nell'esempio, con la nostra letta dalla Dashboard del sito. L'ultima riga è quella che effettua realmente la chiamata e che scatena l'invio della notifica; anche qui dobbiamo sostituire il canale d'esempio "Frigo" con quello che abbiamo creato dopo esserci registrati su Pushetta. Se abbiamo seguito correttamente tutti gli step il nostro telefono emetterà un suono e visualizzeremo una notifica con scritto "Compra il latte!". Bene! Il primo obiettivo è raggiunto.

PASSIAMO AL CODICE

Ora che siamo in grado di inviare notifiche push possiamo implementare la lettura dei sensori per arrivare finalmente all'obiettivo finale. Per semplificare questa attività installiamo un'altra libreria per Python, in questo caso useremo `apt-get`, ovvero il gestore dei pacchetti usato da Raspbian, perché ciò che cerchiamo non è disponibile con `pip`. Eseguiamo quindi:

```
sudo apt-get python-smbus
```

Completata l'installazione possiamo passare all'implementazione del codice, ma prima è necessario identificare l'indirizzo I²C usato dal nostro shield. Quando abbiamo configurato la nostra distribuzione ci siamo preoccupati di installare anche il pacchetto `i2c-tools`, il quale fornisce una serie di comandi di diagnosi per il bus I²C che è arrivato il momento di usare; in particolare, utilizziamo `i2cdetect`, che effettua un probe del bus mostrando l'indirizzo di tutti i dispo-

sitivi rilevati. Otterremo un risultato simile a quello visibile in Fig. 7.

Nell'esempio riportato, `i2cdetect` è stato lanciato con i parametri `-y 1`: il numero (1) indica il bus I²C sul quale eseguire il probe. Nelle prime revisioni di Raspberry Pi, il bus usato era lo 0, se si dispone di una di queste schede il comando da eseguire diventa `i2cdetect -y 0`. Una volta identificato l'indirizzo usato dal nostro shield, abbiamo tutto il necessario per iniziare ad interagire con i sensori.

Allo scopo di fornire strumenti il più possibile riutilizzabili, scriveremo il codice seguendo le modalità dell'object oriented programming (OOP); ciò non implica una maggiore complessità del codice, ma di proposito non useremo quelle caratteristiche che possono impattare sulla comprensione.

In estrema sintesi e semplificando molto, possiamo considerare un sensore come un oggetto che permette di acquisire la misura di una grandezza fisica.

Listato 1

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

class TemperatureSensor(GenericSensor):

    adc_address = 0x48 # Indirizzo I2C letto con i2cdetect
    adc_channel = 0x40 # Channel of device
    V_REF = 3.3 # Tensione riferimento misura ADC
    R1 = 15000.0 # Resistenza R1
    V_IN = 3.3 # Tensione alimentazione partitori tensione
    R_th0 = 10000.0 # Resistenza riferimento termistore a 25 °C
    A = 0.00335402 # Steinhart-Hart Costante A
    B = 0.000256985 # Steinhart-Hart Costante B
    C = 2.62013e-6 # Steinhart-Hart Costante C
    D = 6.38309e-8 # Steinhart-Hart Costante D
    pB = 4100.0 # Costante parametro B
    K = 6.0 # Fattore dissipazione K 6mV C

    def readValue(self):
        bus = SMBus(1) # 0 per le rev.1
        bus.write_byte(self.adc_address, self.adc_channel)
        raw_val = bus.read_byte(self.adc_address)
        raw_val = bus.read_byte(self.adc_address)
        raw_val = bus.read_byte(self.adc_address)

        V = raw_val * self.V_REF / 256.0
        R_th = self.R1 * V / (self.V_IN - V)

        # Calcolo dei gradi Kelvin con la formula di Steinhart-Hart

        logR = math.log(R_th / self.R_th0)
        logR2 = logR ** 2
        logR3 = logR ** 3
        Stein = 1.0 / (self.A + self.B * logR + self.C * logR2 + self.D * logR3)

        # Conversione in Celsius
        return round(Stein - 273.15 - V ** 2 / (self.K * R_th), 2)

    def measureUnit(self):
        return 'C'
```



Fig. 7

Creiamo quindi l'interfaccia che formalizza il comportamento del sensore:

```
class GenericSensor:

    def readValue(self):
        raise NotImplementedError('Metodo readValue da implementare')

    def measureUnit(self):
        raise NotImplementedError('Metodo measureUnit da implementare')
```

La classe `GenericSensor` rappresenta esattamente quanto abbiamo descritto: ogni sensore ha un metodo `readValue` che espone la grandezza fisica e un metodo `measureUnit` che ci dice l'unità di misura. Iniziamo quindi implementando la classe per il sensore di temperatura, con una premessa: il dettaglio funzionale dello shield usato qui è descritto in maniera esaustiva nel libro "Raspberry Pi - Il mio primo Linux Embedded" perciò ci limitiamo a riportare nel Listato 1 il codice che esegue la lettura dei sensori, rimandando al libro per la descrizione del suo funzionamento.

`TemperatureSensor` è quel che si dice un'implementazione concreta dell'interfaccia `GenericSensor`; questa classe è di fatto un piccolo componente che possiamo riusare nei nostri progetti. Il suo utilizzo è triviale, come visibile in questo esempio:

```
sensor = TemperatureSensor()
print "La temperatura letta è " + str(sensor.readValue())
```

Ora che abbiamo tutti gli elementi per raggiungere l'obiettivo iniziale, "arriviamo al dunque" e scriviamo le righe di codice occorrenti:

```
pushetta = Pushetta("0011223344556677aabbccddeeff001122334455")
sensor = TemperatureSensor()
while True:
    temp = sensor.readValue()
    if temp < 18.0:
        pushetta.pushMessage("Camera", "Brrrr...qui fa freddo")

    time.sleep(5)
```

Il codice completo è quello mostrato nel Listato 2, adeguatamente commentato. Creiamo i due oggetti

Listato 2

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

from smbus import SMBus
from pushetta import Pushetta

import math
import time

# Interfaccia generica per i sensori

class GenericSensor:

    def readValue(self):
        raise NotImplementedError('Metodo readValue da implementare')

    def measureUnit(self):
        raise NotImplementedError('Metodo measureUnit da implementare')

# Implementazione concreta per il sensore di temperatura

class TemperatureSensor(GenericSensor):

    adc_address = 0x48
    adc_channel = 0x40

    V_REF = 3.3 # Tensione riferimento misura ADC
    R1 = 15000.0 # Resistenza R1
    V_IN = 3.3 # Tensione alimentazione partitori tensione
    R_th0 = 10000.0 # Resistenza riferimento termistore a 25 °C
    A = 0.00335402 # Steinhart-Hart Costante A
    B = 0.000256985 # Steinhart-Hart Costante B
    C = 2.62013e-6 # Steinhart-Hart Costante C
    D = 6.38309e-8 # Steinhart-Hart Costante D
    pB = 4100.0 # Costante parametro B
    K = 6.0 # Fattore dissipazione K 6mV C

    def readValue(self):
        bus = SMBus(1) # 0 per le rev.1
        bus.write_byte(self.adc_address, self.adc_channel)
        raw_val = bus.read_byte(self.adc_address)
        raw_val = bus.read_byte(self.adc_address)
        raw_val = bus.read_byte(self.adc_address)

        V = raw_val * self.V_REF / 256.0
        R_th = self.R1 * V / (self.V_IN - V)

        # Calcolo dei gradi Kelvin con la formula di Steinhart-Hart

        logR = math.log(R_th / self.R_th0)
        logR2 = logR ** 2
        logR3 = logR ** 3
        Stein = 1.0 / (self.A + self.B * logR + self.C * logR2 + self.D * logR3)

        # Conversione in Celsius

        Celsius = round(Stein - 273.15 - V ** 2 / (self.K * R_th), 2)

        return Celsius

    def measureUnit(self):
        return 'C'

pushetta = Pushetta('0011223344556677aabbccddeeff001122334455')
sensor = TemperatureSensor()
while True:
    temp = sensor.readValue()
    if temp < 18.0:
        pushetta.pushMessage('Camera', 'Brrrr...qui fa freddo')

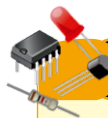
    time.sleep(1)
```

per interagire rispettivamente con Pushetta e con il sensore di temperatura, poi avviamo un ciclo infinito che legge la temperatura e se la trova sotto i 18 °C invia una notifica al canale "Camera". Come ultima cosa attendiamo 5 secondi prima di riprendere l'esecuzione del ciclo successivo e così all'infinito (o finché non interrompiamo manualmente il programma con CTRL + C). Abbiamo ora dei componenti riutilizzabili per poter adattare facilmente quanto fatto ai casi più disparati. Come per il primo esempio d'uso di Pushetta, ricordate che l'Api Key ("0011223344556677aabbccddeeff001122334455") e il nome del canale ("Camera") vanno sostituiti con quelli che desiderate.

CONCLUSIONI

Siamo riusciti a inviare notifiche push da Raspberry Pi ad uno smartphone iOS o Android; non solo: abbiamo capito come strutturando il codice a oggetti, secondo il paradigma OOP, si possa costruire con il tempo un set di componenti che risultino semplici da riutilizzare in più progetti, esattamente come avviene per i componenti elettronici. Un esercizio che vi proponiamo consiste nell'implementare la classe LightSensor che, sulla falsa riga di TemperatureSensor, legga il valore del sensore di luminosità. Risulterà evidente come l'uso di sensori diversi non comporti alcuna difficoltà ulteriore rispetto a quanto fatto negli esempi precedenti.

Ora che avete appreso come inviare notifiche push, le casistiche d'uso sono limitate solo dalla vostra fantasia; probabilmente molti progetti già realizzati, in particolare quelli in ambito domotico, sono già candidati ad essere associati alla generazione di notifiche push, con interventi minimi. ■



per il MATERIALE

La scheda di sperimentazione analogico/digitale utilizzata in questo articolo (cod. FT1060M) è disponibile già montata a 15,00 Euro.

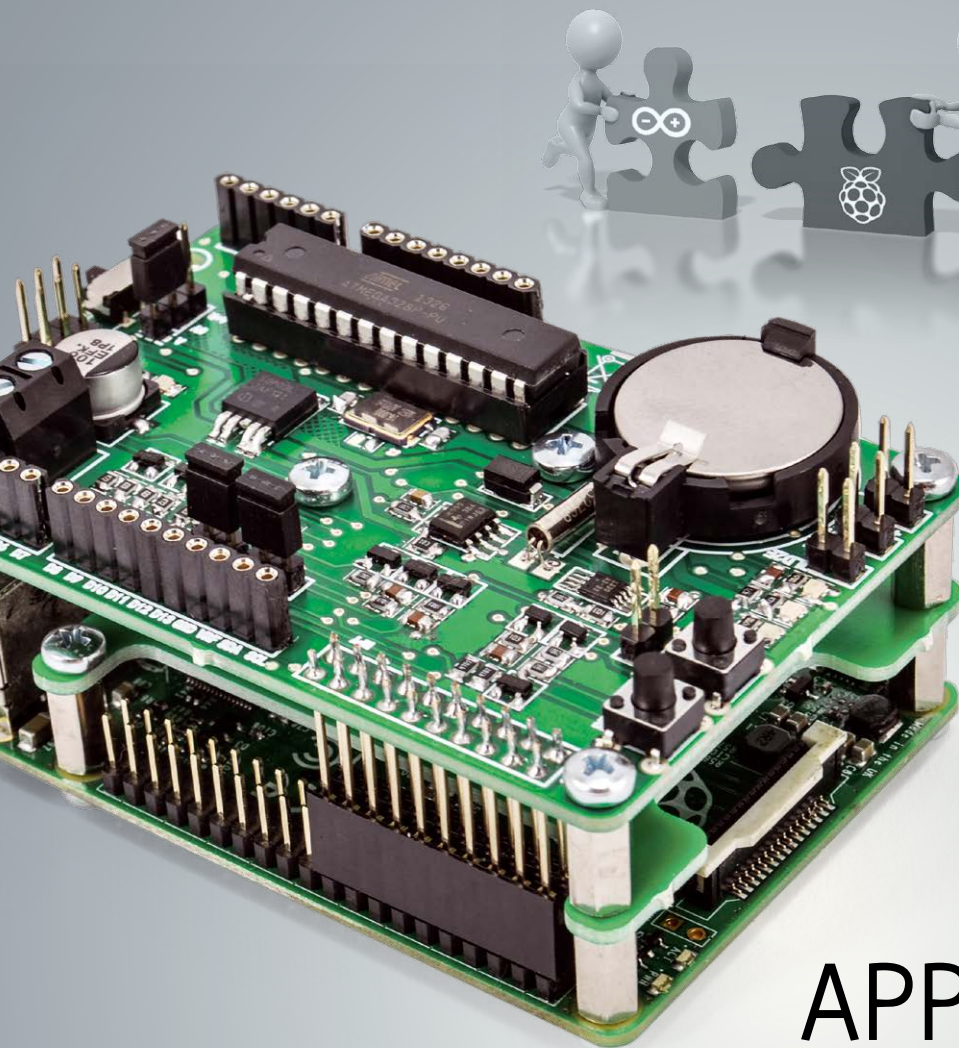
La board Raspberry Pi Tipo B+ (cod. 8111284RS) è acquistabile al prezzo di 36,00 Euro IVA compresa, mentre la board Raspberry Pi 2 Tipo B (cod. 8326274RS) costa 46,00 Euro.

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)

Tel: 0331-799775 • Fax: 0331-792287

<http://www.futurashop.it>



Facciamo un po' di pratica con la scheda che fa da ponte tra i mondi Arduino e Raspberry Pi, realizzando un sofisticato sistema antintrusione.

RANDA: PRIMA APPLICAZIONE

dell' Ing. DANIELE DENARO

Randa è la scheda che abbiamo realizzato per far cooperare Arduino e Raspberry Pi: si monta su quest'ultima board e apre un mondo di possibilità che non esisterebbe con le due unità prese distintamente. Per immaginare i possibili utilizzi basta un po' di fantasia, perché in essa ci sono gli elementi per utilizzarla dove si impiegano sia Arduino che Raspberry Pi. Prima di sottoporvi qualche applicazione complessa e completa cui stiamo pensando, per farvi fare un po' di pratica vi proponiamo qualche spunto concreto, partendo da una prima applicazione in cui si fa uso di un dispositivo molto diffuso e ormai economicissimo: la Webcam.

CONTROLLO AMBIENTALE VIDEO

Utilizzando una Webcam standard a interfaccia USB e un servo di tipo micro (come ad esempio il servo 7300-SERVO206 della Futura Elettronica,

www.futurashop.it), possiamo costruire una piattaforma con Webcam brandeggiante di cui controllare il movimento entro un'escursione di 180 gradi (Fig. 1a e 1b). Bisogna quindi collegare la Webcam a Raspberry Pi, tramite una presa USB, e i tre fili del servocomando a Randa, badando che il rosso (centrale) vada sul pin corrispondente al 5V, il marrone vada sul pin GND ed infine il giallo termini su un pin di output per il pilotaggio: per esempio il D10. Per visualizzare l'immagine fornita dalla telecamera tramite web possiamo utilizzare il software "motion" installabile su Raspberry Pi tramite il comando:

```
sudo apt-get install motion
```

Ma prima è meglio aggiornare il sistema tramite il comando:

Fig. 1a - Elementi per controllo ambientale.

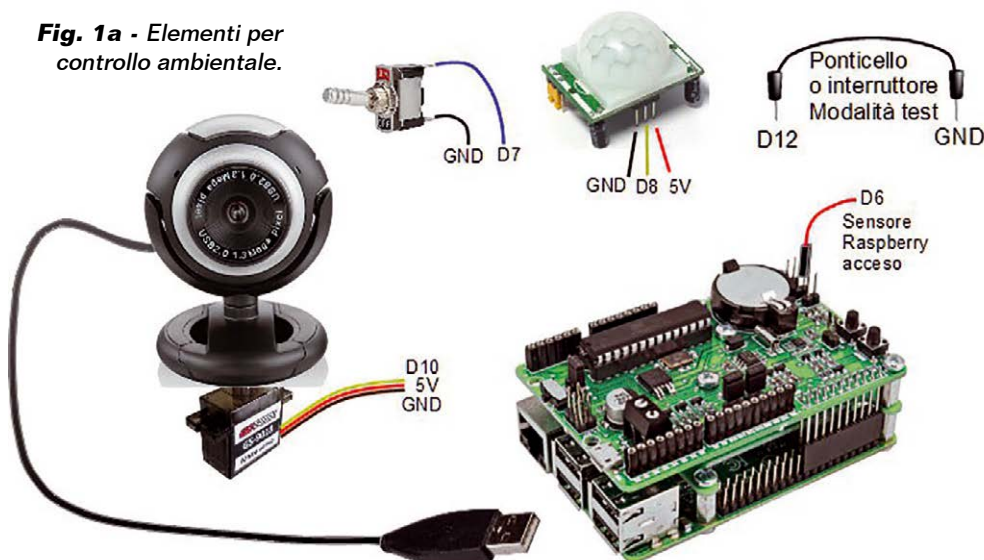


Fig. 1b - Webcam con brandeggio e PIR per rilevazione intrusione.

`sudo apt-get update.`

Tale software ha anche una gestione integrata del rilevamento del movimento, tuttavia per questa volta utilizzeremo, come rilevatore di intrusione, un sensore PIR (per esempio 2846-PIRMOD della Futura Elettronica): il motivo è che sfrutteremo la possibilità offerta da RandA di accendere Raspberry Pi, così da mantenere acceso solo Arduino, che ha un consumo molto basso. Per darvi un'idea dello scenario che si configura, nella **Tabella 1** sono riassunti alcuni dati di consumo;

da essa vediamo che con Arduino in power-down, il consumo si riduce a soli 7 mA (o 12 mA in configurazione operativa per il controllo ambientale). Potremo allora collegare il sensore PIR al pin D3, che corrisponde all'interrupt esterno numero 1, per far svegliare Arduino, che a sua volta attiva Raspberry Pi. Ma, per non complicare troppo lo sketch, in questa applicazione ci accontenteremo di consumare "ben" 28mA, in attesa dell'evento "intrusione".

Riassumendo, solo in presenza di segnalazione da parte del

sensore PIR, Arduino accenderà Raspberry Pi, la quale ordinerà di scattare una foto e poi la acquisirà e la spedisce via e-mail; allo scopo Raspberry Pi rimarrà accesa per un eventuale collegamento via web, permettendo una visione continua e panoramica dell'ambiente.

Il controllo ambientale sarà attivato o disattivato dalla commutazione di un interruttore collegato ad Arduino (al pin D7) che permette l'uscita o il rientro nel locale sotto controllo. Inoltre un ulteriore pin (il pin D6) sarà usato per verificare se Raspberry Pi è accesa o meno. Questo pin è infatti collegato al polo positivo del connettore LD3, che è in parallelo all'alimentazione; notare che se il pin LD3 venisse usato per pilotare un LED esterno, la tensione al polo positivo si ridurrebbe a 1,2V e quindi bisognerebbe leggerne lo stato con un ingresso analogico (ad esempio A0). Risolta la parte meccanica ed effettuati i collegamenti, si tratta ora di passare al software. In particolare dovremo:

- configurare opportunamente il programma "motion";
- creare una pagina HTML in cui visualizzare lo "stream" della Webcam;

Tabella 1 - Consumi.

Arduino Uno (alimentato da USB (con clock a 16MHz))	Funzionamento normale. (Consumo dovuto ad ATmega328P + circuiti di interfaccia USB).	Circa 48mA (senza collegamenti sulle porte).
	Funzionamento in Sleep (Power Down Mode) risvegliabile con interrupt esterno (pin 2 o 3). I circuiti di interfaccia USB rimangono attivi; l'ATmega in realtà consuma pochi microampere.	Circa 34mA (dovuti quasi completamente alla scheda).
RandA (con Raspberry Pi Plus e clock dell'ATmega328P di 16MHz)	Funzionamento normale Arduino+RaspberryPi Plus (Arduino comprende solo l'ATmega328P, perché mancano i circuiti di interfaccia USB).	Circa 330mA .
	Funzionamento con Raspberry Pi spenta. (Solo ATmega328P ma con i collegamenti a PIR e servo attivi).	Circa 28mA . (23mA senza servo)
	Funzionamento con Raspberry Pi spenta e Arduino in Sleep (Power Down Mode) risvegliabile con interrupt esterno. Questa volta l'ATmega consuma qualche milliampere anche in power down a causa delle porte collegate a Raspberry Pi.	< 7mA 3,2mA(ATmega) 3,7mA(scheda senza altro) 12mA con PIR e servo collegati

- inserire nella pagina i comandi per il brandeggio;
- creare gli script CGI per comandare Arduino;
- creare uno sketch Arduino per rilevare la segnalazione del PIR, accendere Raspberry Pi e pilotare il servo secondo i comandi ricevuti da Raspberry Pi sulla seriale; lo sketch deve inoltre rilevare la commutazione dell'interruttore di attivazione/disattivazione.

L'attività principale è quella di rilevare il movimento sulla base della modifica nel tempo di un certo numero di pixel. Tutti i parametri, ossia numero di pixel necessari a far scattare l'evento, margine di errore, possibili zone dell'inquadratura da controllare ecc., sono modificabili agendo su un lungo file di configurazione di nome "motion.conf".

Nel caso si realizzi l'evento, può scattare una (o più) foto, o registrare un filmato di lunghe-

vo dell'applicazione, le decine di secondi necessari allo startup di Raspberry Pi non hanno alcuna influenza sulla cattura dell'evento, che certamente non si esaurisce in pochi secondi.

A questo punto Raspberry Pi rimane accesa ed è pronta a ricevere una eventuale connessione http sulla pagina HTML predisposta allo scopo; la pagina permette la visualizzazione dello stream instradato dalla porta del server di "motion" (porta 8081,

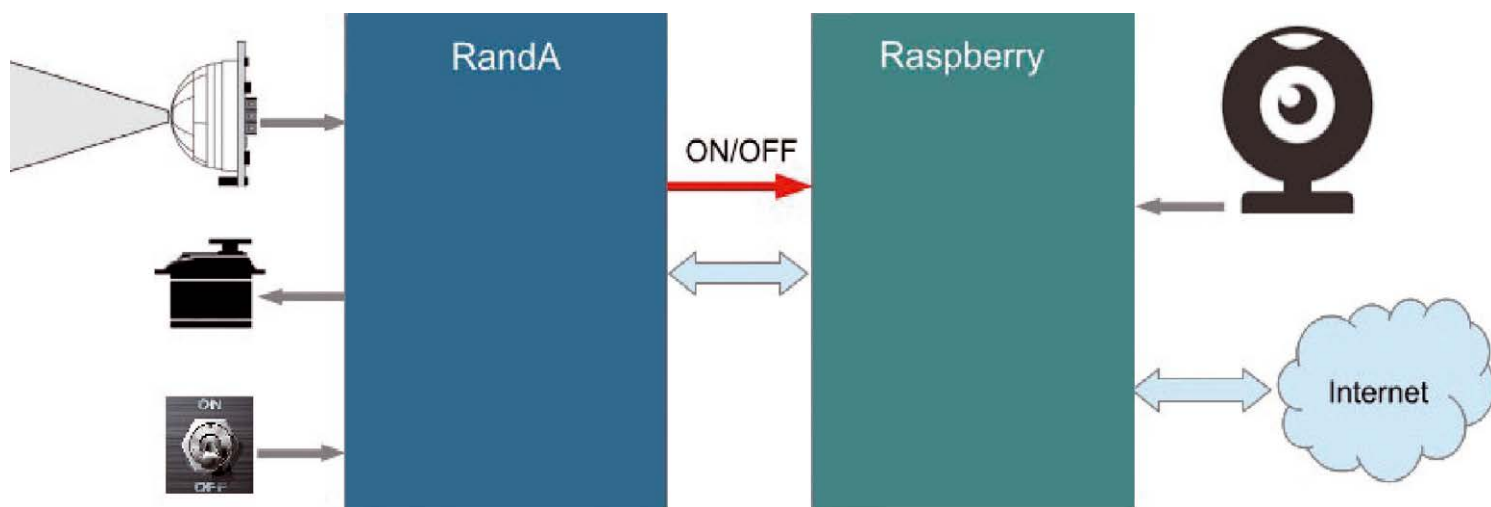


Fig. 2 - Schema generale.

IL PROGRAMMA "MOTION"

Il software "motion" è un potente strumento (gratuito) di gestione video finalizzato al rilevamento del movimento che avviene di fronte ad una Webcam. Nel sito www.lavrsen.dk/foswiki/bin/view/Motion/WebHome è presente una completa documentazione.

In realtà, "motion" ha molte altre funzionalità: innanzi tutto comprende un server http dedicato allo streaming della Webcam; poi può essere pilotato nella gestione delle caratteristiche video anche da remoto, mediante una ulteriore porta http (funzionalità che noi non useremo).

za prestabilita. Inoltre, sempre in caso di evento, può lanciare uno script; nell'applicazione qui descritta useremo questa funzione per inviare una e-mail con foto allegata.

In sostanza, quando il nostro sensore primario PIR rileverà un movimento, dirà a RandA di accendere Raspberry Pi, la quale avvierà "motion". Dopo un certo ed inevitabile ritardo dovuto al tempo di startup di Raspberry Pi, "motion" sarà pronto a rilevare in modo autonomo l'attività nel locale sotto osservazione e quindi potrà scattare la foto e spedire l'e-mail. È chiaro che, visto l'obietti-

modificabile). La pagina permette anche di inviare un comando di reset che ripristina la situazione di partenza, spegnendo Raspberry Pi.

Per consentire il controllo della visione della Webcam, è stata predisposta una modalità "test Webcam", in cui Raspberry Pi viene accesa immediatamente quando RandA rileva la commutazione dell'interruttore, e rimane accesa senza rilevare il movimento e senza scattare foto. In questo modo il server web è sempre disponibile per la visione ed il movimento della Webcam. La modalità "test" viene attivata,

Tabella 2

Variabile	Valore originale	Configurazione per rilevazione movimento	Configurazione per visione da browser	Descrizione
daemon	off	on	on	Parte in background.
process_id_file	/var/run/motion/motion.pid	/tmp/motion.pid	/tmp/motion.pid	File che contiene il PID del processo. Utile per chiuderlo.
framerate	2	24	24	Fotogrammi al secondo per sequenza di foto e filmati.
output_normal	on	first	off	Scatta e salva foto quando rileva movimento: <i>off</i> disabilita, <i>first</i> salva la prima foto scattata.
gap	60	10	60	Secondi per dichiarare chiuso l'evento.
ffmpeg_cap_new	on	off	off	Filmati mpeg (disabilitati).
Webcam_maxrate	1	24	24	Fotogrammi al secondo come Webcam.
Webcam_localhost	on	off	off	Visione solo locale (no).
control_port	8080	0	0	Controllo remoto del programma (disabilitato).
on_picture_save	non attivo (commentato)	/home/pi/sendPicture.sh %f	non attivo	Script attivato quando viene scattata una foto. %f contiene il nome del file.

allo startup di RandA, dal collegamento a GND del pin D12. Il file di configurazione standard di "motion" viene copiato, dall'installazione, nella directory "/etc/motion". Questo file va adattato sia per l'attività di controllo di movimento che per la visione della Webcam, ragion per cui ne faremo due copie: una che posizioneremo nella cartella "/home/pi" (cartella dell'utente pi) ed una che posizioneremo nella stessa cartella degli script CGI. La prima copia sarà utilizzata nel caso di accensione di Raspberry Pi per allarme ed avrà lo scopo di configurare "motion" per scattare la foto ed attivare lo script

che la spedisce. La seconda sarà utilizzata per l'attivazione della Webcam senza scattare foto. Il file di configurazione sarà modificato per poche voci, così come descritto nella **Tabella 2**. Secondo quanto descritto nel file di configurazione base, in caso di rilevamento di movimento è scattata una sola foto ed è posizionata nella cartella "/tmp/motion". Il nome del file conterrà un riferimento al timestamp ed è contenuto nella variabile "%f". Tra le diverse tipologie di eventi utilizzabili, noi useremo solo quella attivata dallo scatto della foto passando allo script anche il nome del file.

Notate che prima di spedire e-mail con il programma "SendMail" bisogna configurare il file "Mail.properties". Lo script attivato dallo scatto della foto si chiama "/home/pi/sendPicture.sh" ed è visibile in **Listato 1**. Il programma "motion" verrà lanciato con il comando:

motion -c nomefileconfigurazione

Come detto, il programma "motion" sarà lanciato all'accensione di Raspberry Pi, che, normalmente spenta, sarà alimentata per scattare la foto del movimento e spedirla per e-mail. Il sistema operativo Linux lancia, alla partenza, lo script "/etc/rc.local", ma invece di inserire il lancio di "motion" alla fine di questo file, consigliamo di modificare rc.local aggiungendo un collegamento ad un eventuale script posizionato nella cartella dell'utente; in tal modo potremo modificare il lancio di programmi o script senza dover andare a mettere le mani su file di sistema e senza aver bisogno di autorizzazioni di "root".

Per esempio, come mostrato nel **Listato 2**.

Nel **Listato 3** è invece descritto lo script lanciato alla partenza.

Listato 1 - sendPicture.sh

```
#!/bin/bash
#Nome della foto da spedire in $1 (parametro passato da motion)
/home/pi/bin/SendMail mailto=pippo.pluto@gmail.com subject="Alarm!" attach=$1
#Fatto il lavoro chiude motion
sudo pkill motion
```

Listato 2 - da aggiungere alla fine di /etc/rc.local

```
# se il file di startup personalizzato esiste lo lancia come utente pi
FSTARTUP="/home/pi/pistartup.sh"
if [ -x $FSTARTUP ];
then
    sudo -u pi /home/pi/pistartup.sh
fi
exit 0
```

PAGINA HTML DEL WEB SERVER

In questa applicazione il Web Server è attivato solo in conseguenza di un allarme (eccetto che in modalità test); l'accesso dovrà essere abilitato solo tramite password, per impedire ad estranei la possibilità di sbirciare con la Webcam nel nostro locale.

In Tomcat, per abilitare la protezione bisogna intervenire sul file `"/WEB-INF/web.xml"` presente nella cartella che contiene l'applicazione. Per isolare l'applicazione è il caso di non usare la cartella base `"ROOT"` del Web Server e creare, invece, una nuova cartella in `"webapps"` che è la cartella generale delle applicazioni web. Infatti, oltre all'applicazione di base presente in `ROOT`, in `webapps`, si trovano altre cartelle corrispondenti ognuna ad un'applicazione (che sia statica, con servlet o con uso di CGI). Ma ogni cartella deve contenere le sotto-cartelle `META-INF` e soprattutto `WEB-INF` con un file `web.xml`.

Creiamo, quindi una cartella `"ControlloAmbientale"` in cui creiamo le due sottocartelle `"META-INF"` e `"WEB-INF"`. In `"ControlloAmbientale"` metteremo la nostra pagina HTML. Invece in `"WEB-INF"` creeremo un file `web.xml`, come si vede nel **Listato 4**. A questo punto dovremo creare l'autorizzazione per l'applicazione intervenendo sul file `"/home/apache-tomcat-7.0.47/conf/tomcat-users.xml"` aggiungendo la riga:

```
<user username="pippo"
password="pluto" roles="Controllo" />
```

ovviamente username e password sono stati scelti casualmente e dovete sostituirli con quelli che desiderate. Infine creeremo la cartella `"cgi"` in `"WEB-INF"` che conterrà gli

Listato 3 - pistartup.sh

```
#!/bin/bash
#exit 0 #per disabilitare lo script togliere il commento
ser=/dev/ttyS0
stty -F $ser 9600
sleep 5
echo "M" > $ser
echo "Comando M" > /home/pi/start.log
# ritardo per dare tempo ad Arduino di rispondere
sleep 1
# legge la risposta di Arduino
read -r -t 2 replay < $ser
if [ ${#replay} -lt 4 ]; then
    echo "Nessuna risposta!" >> /home/pi/start.log
    exit 1
fi
replay=${replay:0:4}
echo $replay >> /home/pi/start.log
if [ $replay = "TEST" ]; then
    echo "No motion" >> /home/pi/start.log
    exit 0 #se in modalità test esce
fi
if [ $replay = "CTRL" ]; then
    echo "Start motion!" >> /home/pi/start.log
    # altrimenti chiude eventuale processo motion attivo
    sudo pkill motion
    # e attiva motion con rilevatore di movimento ed invio foto
    motion -c /home/pi/motion-detect.conf
fi
```

script bash dell'applicazione. La pagina HTML (per esempio di nome `"WebCam.html"`) va posta nella cartella dell'applicazione

(`ControlloAmbientale`) e quindi sarà referenziata come:

<http://...../ControlloAmbientale/WebCam.html>

Listato 4 - web.xml

```
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
version="3.0"
metadata-complete="true">

<description> Controllo ambientale con password</description>
<display-name>ControlloAmbientale</display-name>

<!-- Definisce il nome di una regola di sicurezza -->
<security-role>
  <description>ControlloAmbientale</description>
  <role-name>Controllo</role-name>
</security-role>

<!-- La regola di sicurezza si intende applicata alla seguente URL -->
<!-- Il metodo di sicurezza applicato è quello username-password basico -->
<security-constraint>
  <web-resource-collection>
    <web-resource-name>ControlloAmbientale</web-resource-name>
    <url-pattern>/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>Controllo</role-name>
  </auth-constraint>
</security-constraint>

<login-config>
  <auth-method>BASIC</auth-method>
  <realm-name>ControlloAmbientale</realm-name>
</login-config>
<!-- -->
</web-app>
```

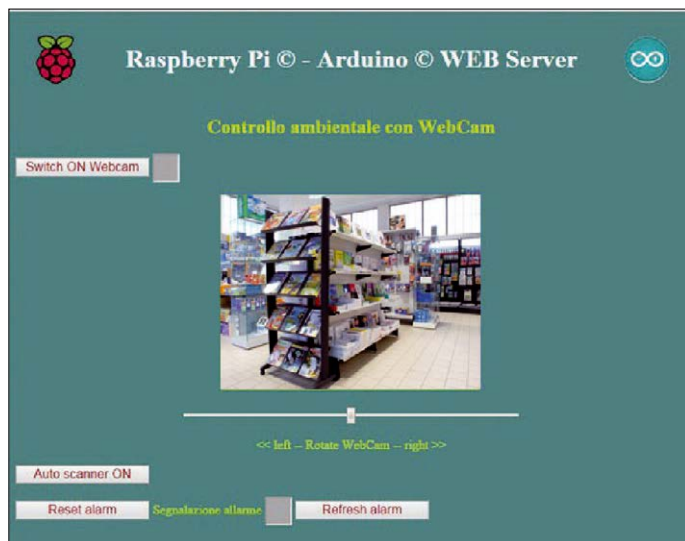



Fig. 3
Pagina HTML
Webcam.

Per l'accesso sarà richiesta l'autorizzazione tramite lo username e la password che abbiamo specificato. La pagina prevede l'utilizzo di un browser abilitato ad

HTML5 a causa del cursore (slide) e della modalità semplificata per visualizzare lo stream della Webcam; vi suggeriamo Chrome, che presenta una migliore grafica

per questi nuovi elementi di HTML5 rispetto a Internet Explorer. La pagina sostanzialmente contiene:

- il riquadro dove è visualizzato lo stream della Webcam;
- un cursore per muovere la Webcam;
- tre pulsanti, che sono uno per attivare la ricezione della Webcam, uno per attivare la modalità auto-scan (brandeggio continuo, tipo movimento del tergicristallo...) più un pulsante per fare un eventuale reset dell'allarme;
- un pulsante di refresh del segnale di allarme utile nella modalità test.

Il riquadro per lo stream è un semplice campo immagine il cui "source" fa riferimento ad un altro web server con porta 8081; infatti, il software "motion" può creare anche un suo web server per l'instradamento dello stream:

```
<div align="center">
&nbsp;
</div>
```

Naturalmente i pulsanti sono controllati da funzioni Javascript. La pagina HTML insieme a tutto il software di questa applicazione è contenuta nell'archivio compresso "ControlloAmbientale.zip" scaricabile dal sito <https://github.com/open-electronics/RandA/>.

FUNZIONI JAVASCRIPT PER COMANDARE LA WEBCAM

Le funzioni agiscono in modalità AJAX e quindi dialogano con il web server in background; in sostanza, lanciano script bash passandogli dei parametri (modalità CGI). Per l'attivazione della Webcam basta uno script bash su Raspberry Pi, ma per il movimento di essa lo script bash

Listato 5 - StartWCam.sh

```
#!/bin/bash

# Rilevazione dei parametri passati dalla chiamata http
# Sono presenti nella variabile di ambiente QUERY_STRING
# nel formato nome1=val1&nome2=val2...

qstring=$QUERY_STRING
# Sostituisce & con spazio per isolare le coppie par=val
qstring=${qstring//&/ }
# Rileva le coppie par=val e le mette in un array chiamato par
read -r -a par <<< "$qstring"
ang=-1

# Ciclo per ogni elemento p del array par
for p in ${par[@]}; do
pn=${p%*=*}          # pn contiene il nome
pv=${p#*=*}          # pv contiene il valore

# parsing del comando
if [ $pn = "wcam" ]; then
fstart=$pv
fi

done          # fine ciclo

# azione

mconfig="/home/apache-tomcat-7.0.47/webapps/ControlloAmbientale/WEB-INF/cgi/motion.conf"
if [ $fstart = "OK" ]; then
motion -c $mconfig          # lancia motion e risponde alla chiamata http
echo "Content-type: text/html"
echo ""
echo "OK"
else
pid=$(< /tmp/motion.pid)    # altrimenti chiude motion e risponde
sudo kill $pid
echo "Content-type: text/html"
echo ""
echo "NOK"
fi
```

di Raspberry Pi avrà il compito di spedire un comando a RandA tramite la porta seriale. Il cursore del brandeggio chiama la funzione corrispondente solo quando si rilascia il pulsante del mouse, quindi la posizione della Webcam viene aggiornata in modo non continuo.

SCRIPT CGI ATTIVATI DALLE FUNZIONI JAVASCRIPT

Per questa applicazione si è deciso di utilizzare la modalità CGI per non essere costretti a descrivere un'applicazione con elementi di Java e Servlet che appartengono al campo delle Web Application e non a tutti familiari.

Chi ha pratica di Java Web Application può passare facilmente ad implementare il dialogo AJAX tramite Servlet o JSP. Gli script sono concettualmente divisi in due parti: nella prima c'è la rilevazione dei parametri passati dalla funzione Javascript, mentre nella seconda parte c'è l'azione vera e propria. I parametri sono compresi nella variabile di ambiente "QUERY_STRING" che il sistema operativo passa allo script (è una attività espletata da Tomcat). In pratica si tratta di scandire la stringa per rilevare le coppie *nome=valore* separate dal carattere '&'. Una volta espletata la funzione, si può anche rispondere per esempio tramite un "echo". Infatti lo stream di output è intercettato da Tomcat e spedito via http. Gli script vanno collocati nella cartella "/WEB-INF/cgi/" dell'applicazione. Il primo script, "StartWCam.sh", lancia il programma "motion". Il secondo, "RandAcmd.sh", esegue l'interazione con RandA tramite la porta seriale "/dev/ttyS0" che corrisponde alla porta a cui RandA è collegato.

SKETCH PER IL CONTROLLO AMBIENTALE

Come detto, lo sketch prenderà il controllo del sistema, accendendo e spegnendo Raspberry quando necessario. Per prima cosa si tratta, quindi, di spostare il ponticello di SW2 sulla posizione "Arduino sempre alimentato" (fatelo con il sistema spento). Gli altri due ponticelli JP1 e JP2 sono chiusi come da configurazione standard. All'avvio, lo sketch deve abilitare il pin D7 in modalità "INPUT_PULLUP" per verificare la posizione dell'interruttore, il pin D8 in modalità "INPUT" per ricevere il segnale del PIR ed il pin D10 in modalità "OUTPUT" per comandare il servo tramite la libreria "Servo.h" che va quindi inclusa nello sketch. Inoltre deve abilitare il pin D6 in modalità "INPUT" per verificare se Raspberry Pi è accesa o spenta ed il pin D12 in modalità "INPUT_PULLUP" per verificare se l'attuale modalità è TEST o NORMALE. Infine deve predisporre la porta seriale alla velocità

Raspberry

All'accensione viene attivato lo script pistartup.sh
Lo script chiede ad Arduino se si è:
in modalità test o normale
Se in modalità test finisce
Se in modalità normale (allarme) lancia "action"
Che dopo aver operato si chiude

Fig. 4 - Attività su Raspberry Pi.

Raspberry

Il server Web è attivo
Se viene chiamata la pagina WebCam.html
Può lanciare "action" in modalità webcam
E muoverla
O controllare/resettare il flag di allarme
(tramite script
presenti nella cartella ControlloAmbientale)

predefinita (9.600) e inizializzare il servo con il comando "attach". Dalla porta seriale arriveranno i comandi degli script CGI per modificare la posizione del servo. Nella modalità normale, nel "loop" continuo, lo sketch dovrà:

1. leggere lo stato dell'interruttore per decidere se attivare le altre funzioni o mantenersi in standby (controllo disattivato); nel caso di passaggio da stato inattivo a stato operativo, do-

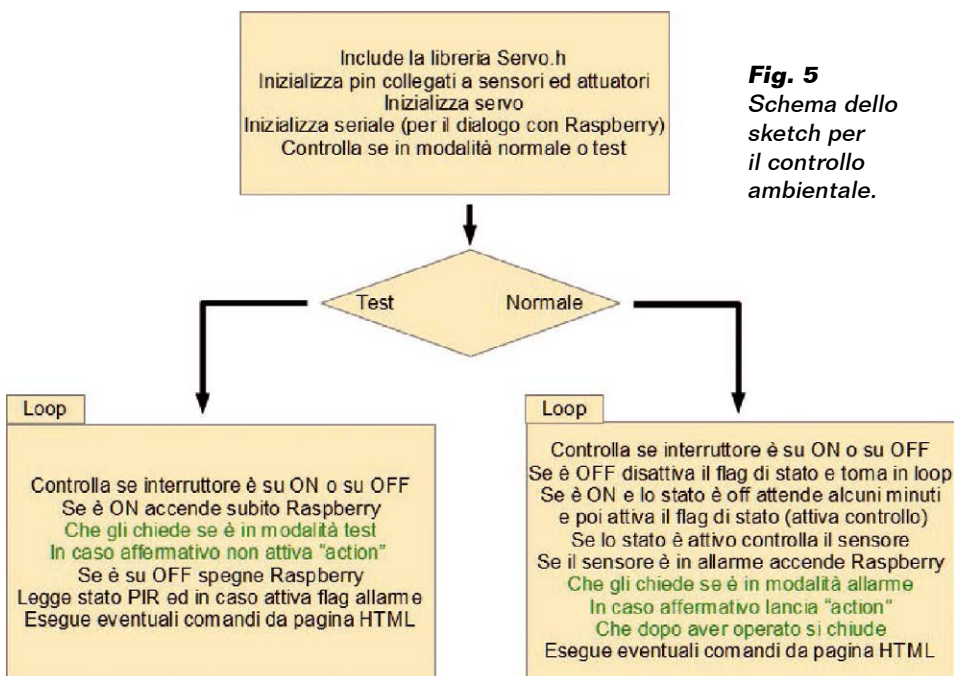


Fig. 5
Schema dello sketch per il controllo ambientale.

Listato 6 - RandAcmd.sh

```
#!/bin/bash
ser=/dev/ttyS0

# Rilevazione dei parametri passati dalla chiamata http
# Sono presenti nella variabile di ambiente QUERY_STRING
# nel formato nome1=val1&nome2=val2...

qstring=$QUERY_STRING
# Sostituisce & con spazio per isolare le coppie par=val
qstring=${qstring//&/ }
# Rileva le coppie par=val e le mette in un array chiamato par
read -r -a par <<< "$qstring"

# Ciclo per ogni elemento p del array par
for p in ${par[@]}; do
  pn=${p%*=*}          # pn contiene il nome
  pv=${p#*=}           # pv contiene il valore

  # parsing del comando ed invio ad Arduino
  case $pn in
    QM)
      echo "M" > $ser
      ;;
    QA)
      echo "Q" > $ser
      ;;
    AN)
      echo "A"$pv > $ser
      ;;
    SC)
      echo "S"$pv > $ser
      ;;
    RA)
      echo "R" > $ser
      ;;
  esac
done # fine del ciclo

sleep 0.2 # Ritardo per dare tempo ad Arduino di rispondere
# Legge la risposta da Arduino
read -r -t 2 replay < $ser

# La ripete indietro come risposta alla chiamata http
# ma prima deve chiudere l'istestazione della risposta http
# con una linea vuota.
# Lo standard output è automaticamente letto dal server Tomcat.
echo "Content-type: text/html"
echo ""
echo $replay # risposta http (alla funzione Javascript)
```

vrà attendere qualche minuto prima di operare, per permettere l'uscita dal locale;

2. leggere lo stato del PIR per verificare l'avvenuto allarme; in caso affermativo dovrà procedere con l'accensione di Raspberry Pi e, appena questa sarà pronta, farle scattare una foto da allegare alla mail che sarà spedita, quindi lasciarla accesa fino all'eventuale comando di stand-by inviato tramite la pagina HTML.

Essendo questo un esempio

dimostrativo, abbiamo utilizzato un semplice interruttore per abilitare o disabilitare il controllo ambientale. In un'applicazione reale dovremo mimetizzare l'interruttore o utilizzare una tastiera di sicurezza con uscita a relé, nella quale digitare un codice di attivazione e disattivazione. Potremo anche gestire l'attivazione/disattivazione da telecomando, così da disabilitare il sistema prima di accedere al locale e attivarlo dopo esserne usciti. In Fig. 5 è rappresentato lo schema a blocchi dello sketch; non

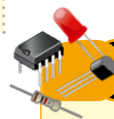
pubblichiamo il sorgente completo per ragioni di spazio, ma lo trovate nell'archivio scaricabile dal nostro sito.

Per rendere più flessibile il sistema, abbiamo inserito anche una modalità "test Webcam" che viene attivata ponendo a 0 (GND) il pin D12 con un ponticello o un interruttore.

Tale modalità è attivata allo startup (o al reset) di RandA e fa sì che Raspberry Pi venga accesa subito senza considerare eventuali allarmi; così il sistema finisce per operare come un server per la visualizzazione e il movimento della Webcam.

CONCLUSIONI

Avete visto come è facile realizzare un valido controllo ambientale che ci permette anche di osservare in streaming gli effetti dell'intrusione. Questo sistema potrebbe essere completato da una chiavetta modem GSM/HSDPA per aggiungere l'invio di SMS o per lavorare in "mobile". Sul nostro sito è disponibile il software completo, che dovete personalizzare con password e indirizzo di posta elettronica vostri. ■



per il MATERIALE

La scheda RandA è disponibile già montata e collaudata al prezzo di 39,00 Euro IVA compresa.

La board Raspberry Pi Tipo B+ (cod. 8111284RS) è acquistabile al prezzo di 36,00 Euro IVA compresa, mentre la board Raspberry Pi 2 Tipo B (cod. 8326274RS) costa 46,00 Euro.

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11,
21013 Gallarate (VA)

Tel: 0331-799775 • Fax: 0331-792287
<http://www.futurashop.it>

RANDA PhotoSharing



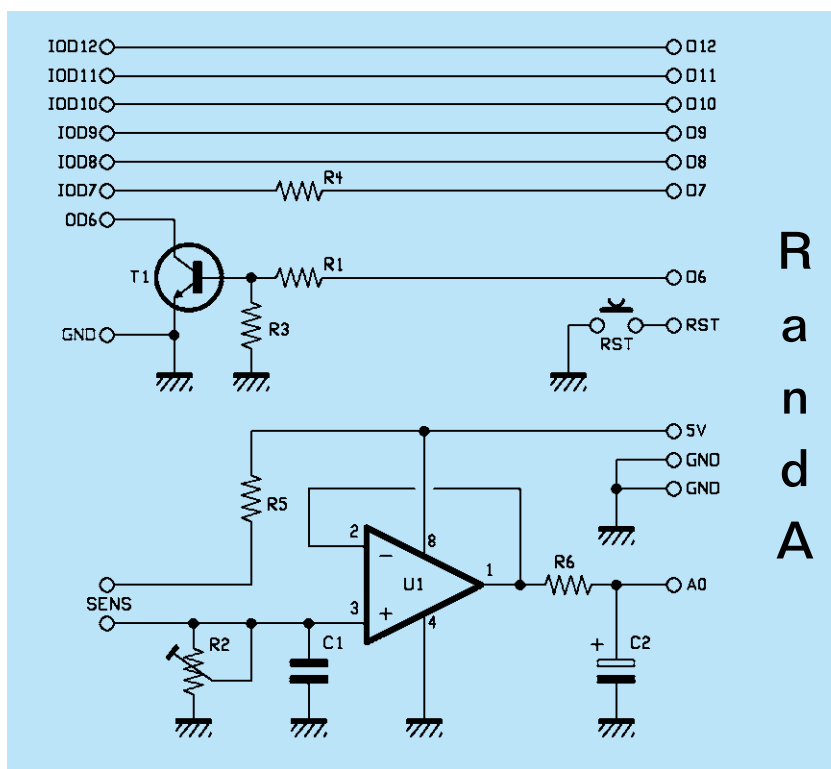
Scatta la foto, scegli l'effetto e condividila! Raspberry Pi 2 e RandA si uniscono per dare vita a un'applicazione interattiva che abbiamo presentato alla recente Maker Faire di San Mateo.



di LUCA e MARCO BELLAN

Alla Maker Faire tenutosi lo scorso maggio a San Mateo (California – U.S.A.), abbiamo presentato in anteprima un progetto nato dalla necessità di mostrare ai visitatori le potenzialità di RandA unita a Raspberry Pi, in modo suggestivo e divertente. Sono state più di cinquecento le foto scattate dai visitatori tramite RandA PhotoSharing -questo è il nome del

progetto- alcune delle quali condivise automaticamente sui social network di Open Electronics (Twitter e Facebook). In questo articolo vi spiegheremo passo dopo passo come assemblare e programmare il sistema, soffermandoci sulla configurazione di Raspberry Pi, per poi passare all'installazione e all'utilizzo del software RandA PhotoSharing. Il progetto che andre-

R
a
n
d
A

mo a realizzare sarà un misuratore multi-tematico nel quale potremo impostare, con un pizzico di empirica amenità, il tema della misurazione: amore, amicizia, intelligenza...

L'utente, attraverso diverse schermate, potrà scattarsi una foto (scegliendo eventualmente un effetto da applicarci), misurare la propria conducibilità elettrica (afferrando due poli conduttori) che sarà poi utilizzata per determinare un punteggio relativo al tema e visualizzare il risultato della misurazione posto in sovrapposizione nella foto precedentemente scattata (Fig. 1); potrà poi scegliere se condividere la foto sui social network oppure riceverla via e-mail. Questa applicazione può rivelarsi utile anche in termini di marketing, perché gli indirizzi e-mail inseriti dagli utenti per ricevere la propria foto vengono salvati in un database per poi essere esportati in un file Excel; allo stesso tempo la condivisione sui social network delle foto scattate contribuisce ad incrementare la propria visibilità sul web: infatti su ogni foto saran-

no presenti degli hashtag specifici e un logo personalizzabile. L'utente interagirà con la nostra applicazione attraverso un monitor, una tastiera e tre pulsanti che serviranno a procedere nelle diverse schermate: inizio, scelta effetto foto, scatto foto e misurazione, condivisione sui social network, invio della foto appena scattata al proprio indirizzo e-mail e chiusura della sessione.

COMPONENTI E ASSEMBLAGGIO

Per realizzare questo progetto avremo bisogno dei seguenti componenti (reperibili su www.futurashop.it):

- Raspberry Pi 2, per eseguire l'applicazione con l'interfaccia grafica tramite un monitor, elaborare foto e inviarle;
- Raspberry Pi Camera, per scattare le foto agli utenti;
- SD-Card (consigliata da 8 GB o più), per il sistema operativo Raspbian e salvare le foto
- RandA;
- shield dedicato a questo progetto, per collegare i pulsanti e i poli conduttori a RandA;

- monitor con ingresso HDMI;
- mouse e tastiera USB;
- tre pulsanti, che avranno la funzione di Prev/Sì, OK, Next/No;
- due poli conduttori che i partecipanti al "gioco" dovranno impugnare per effettuare la misurazione;
- una connessione ad Internet con cavo ethernet oppure chiavetta WiFi.

Inoltre necessiteremo di un account Gmail e alcuni account di social network come Facebook e Twitter (opzionali) su cui la nostra applicazione posterà le foto. Per condividere le foto sui social network e salvarle in Google Drive utilizzeremo il sito www.ifttt.com; IFTTT è un servizio gratuito e molto comodo in quanto ci permette di impostare delle operazioni che verranno eseguite solo alla ricezione di una e-mail con determinati requisiti (recipes); per esempio, dopo aver impostato un "recipe" che carica una foto su Facebook prelevandola dall'allegato di una e-mail, possiamo inviare all'indirizzo trigger@recipe.ifttt.com una e-mail con un oggetto contenente "#photofb" e con allegata una foto: IFTTT preleverà in automatico l'allegato dall'e-mail e caricherà la foto sull'account Facebook precedentemente associato.

LO SHIELD ARDUINO

Per interconnettere a RandA e tramite essa a Raspberry Pi i tre pulsanti, abbiamo previsto un semplicissimo shield che realizza anche l'interfaccia verso il "sensore di emozioni"; con riferimento al relativo schema elettrico, illustrato in queste pagine, vediamo che i pulsanti si collegano ai pin IOD8, IOD9, IOD10, che portano alle linee rispettivamente D8, D9, D10 di RandA. I due elettrodi cilindrici che si impugnano per misurare le

emozioni vanno collegati ai contatti SENS, senza un preciso ordine: nello shield, uno porta all'uscita 5V di RandA e l'altro all'ingresso di un buffer realizzato dall'amplificatore operazionale U1, l'uscita del quale termina sulla linea A0 di RandA. Notate che gli elettrodi permettono di inserire il corpo dell'utente in un circuito partitore di tensione che pilota il buffer: tramite la resistenza R5, la linea 5V alimenta una mano attraverso l'apposito elettrodo cilindrico, quindi la debole e -teniamo a precisarlo a scanso di equivoci- innocua corrente che fluisce nel corpo, termina sul secondo elettrodo cilindrico e quindi trimmer R2, determinando all'ingresso non-invertente dell'operazionale un potenziale che è funzione sia del valore resistivo inserito con R2 (che permette di tarare la sensibilità del sistema), sia della conducibilità elettrica del corpo di chi tocca gli elettrodi. Tale conducibilità varia, nello stesso soggetto, in base alla sudorazione, alle emozioni ecc.

Per questo la tensione applicata all'operazionale e restituita di pari valore (ben filtrata dalla cella passa-basso R6/C2 per trattenere i disturbi impulsivi ed evitare

brusche variazioni dovute all'assestamento delle mani sui contatti) alla linea A0 di RandA, rispecchia lo stato d'animo dell'utente collegato. Sullo shield prende posto un pulsante che permette di resettare RandA e si trova anche un transistor montato a emettitore comune in connessione open-collector, pilotato in base dalla linea D6 di RandA e il cui collettore è connesso al contatto OD6; tale NPN può essere usato per attivare eventuali LED, avvisatori acustici ecc., ma al momento è "spare", vale a dire di scorta e perciò non utilizzato.

REALIZZAZIONE PRATICA

Per assemblare il tutto, colleghiamo la Raspberry Pi Camera facendo attenzione che la parte

blu del flat sia rivolta verso la porta ethernet di Raspberry Pi, montiamo RandA con il relativo supporto e successivamente montiamo la shield di RandA PhotoSharing sopra a RandA. Colleghiamo ai seguenti pin dello shield i tre pulsanti: pin 8 per il pulsante che avrà la funzione "Next/No", pin 9 per il pulsante che avrà la funzione "OK" e pin 10 per il pulsante che avrà la funzione "Prev/Yes"; i tre pulsanti avranno un estremo in comune, connesso a GND di RandA. Cabliamo un polo conduttore all'uscita 5V e l'altro all'ingresso A0 di RandA; successivamente colleghiamo mouse, tastiera, cavo ethernet o chiavetta WiFi (necessari per la parte di configurazione). Dopo aver preparato la SD-Card e configurato Raspbian (spiegheremo nel paragrafo successivo come fare), alimentiamo con 5V/2A RandA, che a sua volta alimenterà Raspberry Pi.

CONFIGURAZIONE

Possiamo scaricare l'immagine per Raspberry Pi già pronta per questo progetto dal sito di Elettronica In e passare direttamente alla fase di caricamento dello sketch su RandA, oppure possiamo scarica-

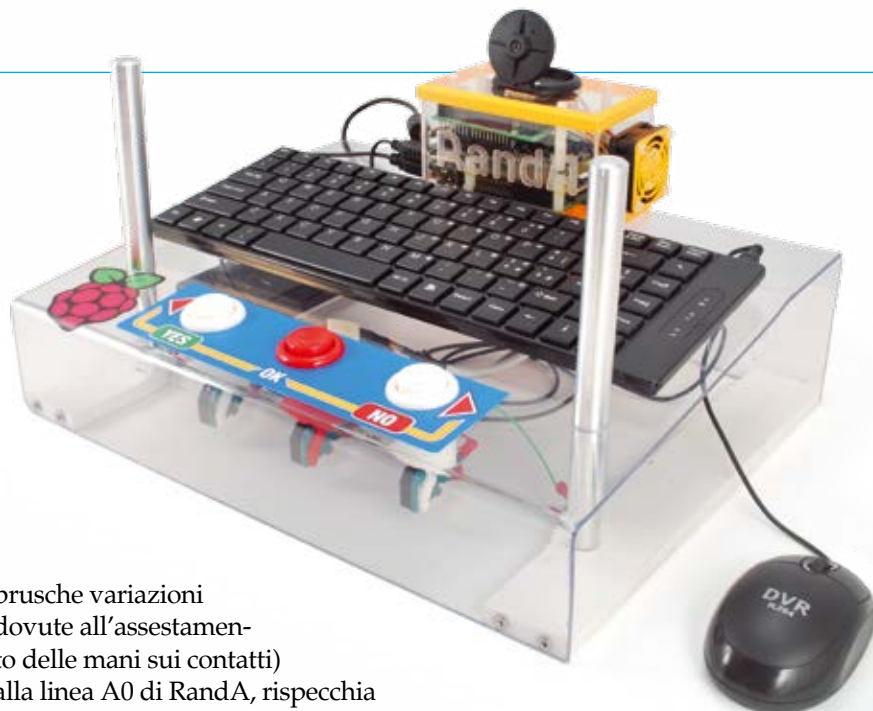
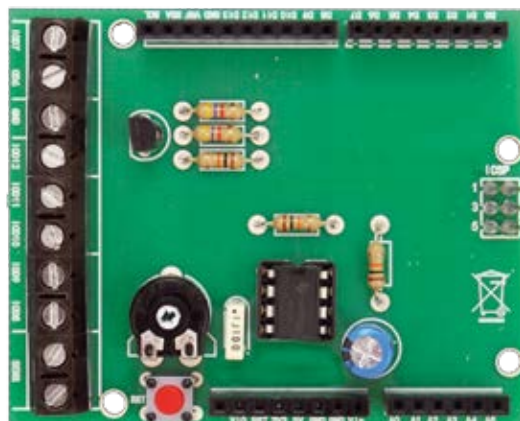
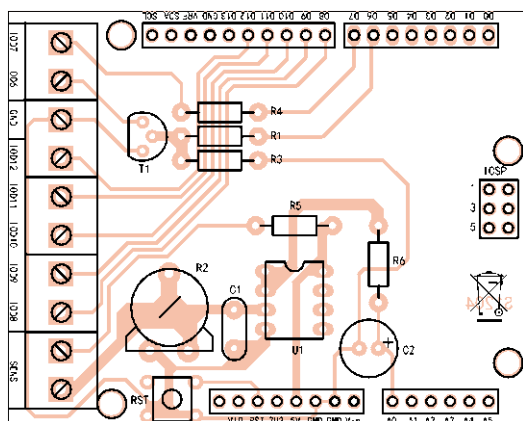


Fig. 1 - Il prototipo di RandA PhotoSharing.



Elenco Componenti:

R1: 4,7 kohm
R2: Trimmer 220 kohm MO
R3, R5: 10 kohm
R4: 470 ohm
T1: BC547
C1: 100 nF ceramico

C2: 100 μ F 16VL elettrolitico
U1: LM358
RST: Microswitch

Varie:
- Zoccolo 4+4

- Strip Maschio/Femmina 6 vie
- Strip Maschio/Femmina 8 vie (2 pz.)
- Strip Maschio/Femmina 10 vie
- Strip Maschio/Femmina 2x3 vie
- Circuito stampato S1204

re dal sito ufficiale di Raspberry Pi l'immagine dell'ultima versione di Raspbian e seguire tutta la procedura di configurazione.

Nel caso scegliessimo di partire con l'immagine già pronta, dovremo ricordarci di modificare il file `/etc/ssh/ssmtp/ssmtp.conf` come indicato nella sezione di configurazione delle librerie per l'invio dell'e-mail. Scriviamo l'immagine di Raspbian su SD Card con il software Win32DiskImager reperibile gratuitamente a questo indirizzo: sourceforge.net/projects/win32diskimager/. Inseriamo poi l'SD-Card nella slot di Raspberry Pi e alimentiamo RandA per accendere il tutto. I successivi passaggi della configurazione potranno essere eseguiti sia tramite l'ambiente grafico di Raspberry Pi, sia da remoto in SSH con client come PuTTY.

Durante la prima accensione sarà visualizzato il tool di configurazione di Raspberry Pi, con cui dovremo compiere diverse operazioni:

1. Expand Filesystem;
2. Enable Boot to Desktop/Scratch:

selezioniamo la seconda voce "Desktop Log in...":

3. Internalisation Options:

- a. Change Locale: con la barra spaziatrice deseleggiamo `en_GB.UTF-8` e selezioniamo invece `it_IT ISO-8859-1` (settandola successivamente come *Default locale for the system environment*);
- b. Change Timezone: selezioniamo *Europe* e successivamente *Rome*;
- c. Change Keyboard Layout: scegliamo il modello di tastiera che stiamo utilizzando (solitamente *Generica 105 tasti (Intl) PC*), successivamente selezioniamo *Altro* e poi *Italiana* come lingua della tastiera, scegliamo di nuovo *Italiana* come disposizione della tastiera e infine confermiamo cliccando su *Ok* due volte;

4. Enable Camera: *Enable*;

5. Advanced Options:

- a. SSH: *Enable*;
- b. PC: rispondiamo *Yes* a entrambe le domande.

Spostiamoci con il tasto TAB sulla

scritta *<Finish>* e confermiamo con *INVIO* per riavviare il sistema. A questo punto dovremo disporre di una connessione ad Internet per procedere: se disponiamo di un cavo ethernet colleghiamolo a Raspberry Pi, altrimenti seguiamo la procedura contenuta nel riquadro "Installazione e configurazione chiavetta WiFi per Raspberry Pi" che trovate nelle prossime pagine. Dopo aver connesso Raspberry Pi ad internet, lanciamo il seguente comando per aggiornare il sistema e, infine, riavviarlo:

```
sudo apt-get update && sudo
apt-get upgrade && sudo rpi-
update && sudo reboot
```

La procedura di aggiornamento potrà richiedere molto tempo e a volte dovremo confermare l'installazione di alcuni pacchetti premendo S e poi INVIO. Scarichiamo ora l'ultima versione del setup di RandA dal sito <https://github.com/open-electronics/RandA/releases>. Una volta scaricato il pac-

chetto, al suo interno troveremo il manuale da seguire per installare RandA (*InstallationREADME.TXT*) e i relativi file di installazione. Dopo aver installato RandA, dovremo cambiare la porta di ascolto del servizio Tomcat per evitare conflitti con il servizio Apache che installeremo successivamente. Infatti Tomcat è utilizzato per "hostare" il sito del pannello di controllo in Java di RandA, che a noi non servirà; dovremo invece utilizzare Apache per "hostare" il sito del pannello di controllo della nostra applicazione, in modo che, digitando l'indirizzo IP del nostro Raspberry Pi su qualsiasi browser, verremo rediretti al pannello di controllo di RandA PhotoSharing. Spostiamoci quindi nella directory dove risiede la configurazione di Tomcat:

```
cd /home/apache-tomcat-7.0.47/
conf
```

Apriamo il file `server.xml` con il comando:

```
nano server.xml
```

Editiamo la riga "`<Connector port = "80" ...`" sostituendo 80 con 8080. Adesso chiudiamo e salviamo il file con CTRL+X, Y e INVIO; successivamente riavviamo Raspberry Pi:

```
sudo reboot
```

Disattiviamo ora il risparmio energetico di Raspberry Pi editando questo file:

```
sudo nano /etc/lightdm/lightdm.conf
```

Aggiungiamo la seguente riga sotto alla sezione [SeatDefaults]:

```
xserver-command=X -s 0 dpms
```

Chiudiamo e salviamo il file con

CTRL+X, Y e INVIO: in questo modo Raspberry Pi non disattiverà l'output dell'HDMI dopo alcuni minuti di inattività.

Lanciamo i seguenti comandi in successione per installare Apache, PHP e il server MySQL che ci serviranno per far funzionare il pannello di controllo Web di RandA PhotoSharing:

```
sudo apt-get install apache2
apache2-doc apache2-utils
sudo apt-get install libapache2-
mod-php5 php5 php-pear php5-
xcache
sudo apt-get install php5-mysql
sudo apt-get install mysql-ser-
ver mysql-client
```

Durante l'installazione di alcuni pacchetti verrà richiesta la conferma, premere S e INVIO per confermare. Quando ci sarà chiesta una password per il database inseriamola e teniamola a portata di mano poiché ci sarà richiesta in seguito. Ora installiamo PHPMyAdmin che

ci aiuterà a gestire il database di RandA PhotoSharing:

```
sudo apt-get install phpmyadmin
```

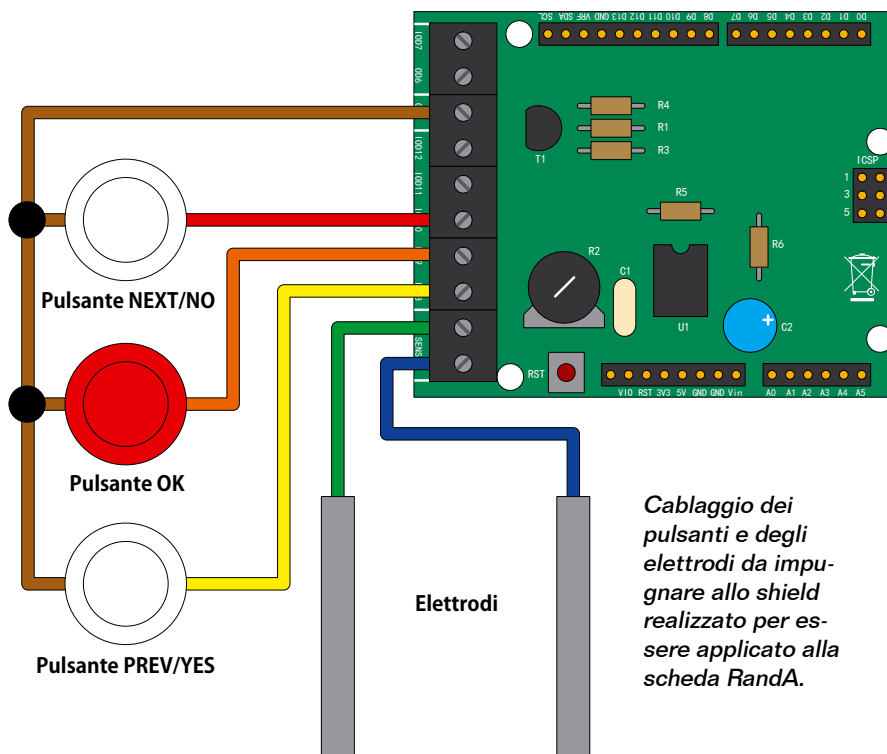
Quando ci verrà chiesto quale Web Server configurare automaticamente, con lo SPAZIO selezioniamo "apache2", poi confermiamo premendo TAB e INVIO. Alla domanda "Configurare il database di PHPMyAdmin con dbconfig-common?" rispondiamo "Sì". Terminata l'installazione, apriamo il file di configurazione per apache2:

```
sudo nano /etc/apache2/apache2.conf
```

e aggiungiamo al termine di esso la riga:

```
Include /etc/phpmyadmin/apache.conf
```

Chiudiamo e salviamo il file con CTRL+X, Y e INVIO; riavviamo



apache2 per rendere effettive le modifiche:

```
sudo /etc/init.d/apache2 restart
```

Lanciamo il comando:

```
sudo visudo
```

e aggiungiamo alla fine del file la seguente stringa:

```
www-data ALL=(ALL) NOPASSWD: ALL
```

Chiudiamo e salviamo il file con CTRL+X, Y e INVIO: in questo modo daremo i permessi ai file PHP per agire sulle foto scattate

(lettura, modifica e cancellazione). Passiamo ora ad installare le librerie che serviranno al nostro script Python (il programma vero e proprio) per interagire con il database MySQL e modificare le immagini. Lanciamo quindi i seguenti comandi in successione:

```
sudo apt-get install python-dev
sudo apt-get install python-imaging-tk
sudo apt-get install python-mysqldb
sudo apt-get install imagemagick
```

Installiamo il mail server e le utility che ci serviranno per inviare e-mail da Python lanciando i seguenti comandi in successione:

```
sudo apt-get install ssmtp
sudo apt-get install mailutils
```

Modifichiamo ora il file *ssmtp.conf* con il comando:

```
sudo nano /etc/ssmtp/ssmtp.conf
```

in modo che risulti come segue:

```
root=postmaster
mailhub=smtp.gmail.com:587
hostname=Raspberry Pi Pipi
AuthUser=UtenteGmail@gmail.com
AuthPass=PasswordGmail
UseSTARTTLS=YES
```

Ricordiamoci di sostituire "UtenteGmail" e "PasswordGmail" con i dati del nostro account, chiudiamo e salviamo il file con CTRL+X, Y e INVIO.

Modifichiamo il file *revalias*:

```
sudo nano /etc/ssmtp/revalias
```

Aggiungendo ora la riga seguente:

```
root:root@gmail:smtp.gmail.com:587
```

Chiudiamo e salviamo il file con

CTRL+X, Y e INVIO. Cambiamo il permesso del file *ssmtp.conf* con il seguente comando:

```
sudo chmod 774 /etc/ssmtp/ssmtp.conf
```

Riavviamo il sistema per rendere effettive tutte le modifiche:

```
sudo reboot
```

Abbiamo completato la configurazione di Raspberry Pi installando e configurando tutto il necessario, ora dobbiamo scaricare il pacchetto di RandA PhotoSharing; posizioniamoci quindi sotto alla cartella corretta:

```
cd /var/www
```

e scarichiamo il pacchetto di RandA PhotoSharing con il comando:

```
sudo git clone https://github.com/open-electronics/randaps.git
```

La cartella che verrà creata (riferitevi alla Fig. 2) conterrà:

- **admin/**: cartella che contiene i file del pannello di controllo Web;
- **data/**: cartella che contiene tutte le grafiche necessarie al funzionamento (temi, overlay, logo, ...);
- **photos/**: cartella che ospiterà tutte le foto che verranno scattate;
- **randaps.py**: eseguibile di RandA PhotoSharing;
- **randaps_sketch.hex**: sorgente compilato per RandA pronto per essere caricato;
- **randaps_sketch.ino**: sorgente per RandA PhotoSharing, nel caso in cui volessimo effettuare delle modifiche;
- **RandA-PhotoSharing.sql**: installazione del database MySQL;
- **README.pdf**: guida completa per il setup e l'utilizzo di RandA PhotoSharing (traduzione di



Installazione e configurazione chiavetta WiFi per Raspberry Pi

questo articolo in inglese).

Accediamo, con un browser qualunque, all'indirizzo `http://IP_RASPBERRY_PI/phpmyadmin` avendo cura di sostituire la dicitura "IP_RASPBERRY_PI" con l'indirizzo IP effettivo. Facciamo il login utilizzando come utente root e come password quella che abbiamo impostato durante l'installazione di MySQL e PHPMyAdmin.

Spostiamoci sulla tab *Privilegi*, facciamo clic su *Aggiungi un nuovo utente* e compiliamo i campi come indicato qui di seguito:

- Utente: [Utilizza campo text:] *randaps*;
- Host: [Locale] *localhost*;
- Password: [Utilizza campo text:] *randaps*;
- Reinserisci: *randaps*;
- Database per l'utente: *creare un database con lo stesso nome e concedi tutti i privilegi*;
- Privilegi globali: *selezionare tutti*.

Adesso facciamo clic sul pulsante *Crea utente*: sul lato sinistro della pagina dovremmo vedere comparire il database *randaps* (se così non fosse aggiorniamo la pagina); successivamente facciamo clic sul nome del database appena creato e selezioniamo la tab *Importa*. Dobbiamo importare il file SQL contenuto del pacchetto di installazione di RandA PhotoSharing; scegliamo quindi il file e facciamo click su *Esegui*. Editiamo il file eseguibile di RandA PhotoSharing:

```
sudo nano /var/www/randaps/randaps.py
```

Nella sezione "CUSTOMIZABLE VARIABLES" inseriamo la password del nostro account di Gmail al posto della scritta "INSERT_YOUR_E-MAIL_PASSWORD", facendo attenzione a mantenere le virgolette iniziali e finali; salviamo e chiudiamo il file con CTRL+X, Y

In alternativa alla connessione ethernet, è possibile impiegare una soluzione wireless affacciando a Internet il sistema mediante WiFi. Allo scopo bisogna procurarsi un dongle WiFi USB e inserirlo nella presa USB di Raspberry Pi, apriamo il terminale e digitiamo il comando:

```
dmesg | more
```

Controlliamo che Raspberry Pi abbia riconosciuto la periferica: dobbiamo premere SPAZIO fino a quando non troviamo una riga simile a "Product: 802.11n WLAN Adapter ..." e per uscire premiamo il tasto Q. Editiamo il file delle interfacce di rete con il comando:

```
sudo nano /etc/network/interfaces
```

Inseriamo le righe per la connessione alla rete WiFi; alla fine il file dovrebbe risultare così:

```
auto lo
iface lo inet loopback
iface eth0 inet dhcp

allow-hotplug wlan0
auto wlan0

iface wlan0 inet dhcp
    wpa-ssid "Nome della rete WiFi"
    wpa-psk "Password della rete WiFi"
```

Chiudiamo e salviamo il file con la combinazione di tasti CTRL+X, Y e INVIO; successivamente lanciamo il seguente comando per ricaricare le impostazioni di rete:

```
sudo service networking reload
```

Con il comando `ifconfig` possiamo verificare che il dongle WiFi sia connesso e abbia ottenuto un indirizzo IP. La configurazione è terminata.

e INVIO.

Carichiamo ora lo sketch su RandA con il seguente comando:

```
ArduLoad /var/www/randaps/randaps_sketch.hex
```

Arrivati a questo punto, passiamo a configurare gli account che serviranno a RandA PhotoSharing per inviare le foto:

- **Gmail:** RandA PhotoSharing utilizzerà questo account per spedire le foto a IFTTT e all'utente (se richiesto);
- **IFTTT:** ogni volta che riceverà una mail (dall'account di Gmail) provvederà a caricarla su Google Drive (se impostato) o condividerla su Facebook o Twitter (se impostato).

Accediamo con il nostro account a Gmail, visitiamo il seguente indirizzo: `https://www.google.com/settings/security/lesssecureapps` e abilitiamo l'accesso per le app meno

sicure: in questo modo RandA PhotoSharing potrà inviare e-mail a nome di questo account. Creiamo un account IFTTT (`www.ifttt.com`) e registriamo i seguenti recipes:

- Salva le foto su Google Drive: `https://ifttt.com/recipes/192360-send-gmail-attachments-to-google-drive`;
- Carica le foto su Twitter: `https://ifttt.com/recipes/129743-twitter-`

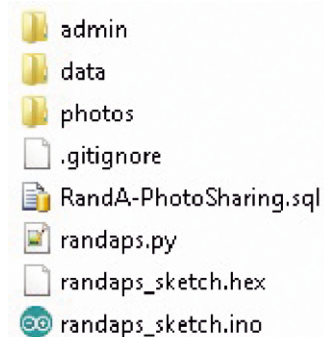


Fig. 2 - Disposizione dei file nella cartella *randaps*.

Control Panel

PHOTOS
SETTINGS

Theme

Maker

Add new theme

Value	Value	Description	Description	
				+

Standby

Start screen

Logo

X	750	Y	655	
	—		—	

Overlay

X	10	Y	520	
	—		—	

Result

X	110	Y	655	
	—		—	

Photos

Off	☒	On	☐	Save to SD card
Off	☐	On	☐	Save to cloud
Off	☐	On	☐	Send to social
Off	☐	On	☐	Send via eMail

Fig. 3
Il pannello di controllo.

subject-w-tv-body-tweet-attachment-photo;

- Carica le foto su Facebook: <https://ifttt.com/recipes/13714-upload-photo-to-facebook-from-e-mail>.

PANNELLO DI CONTROLLO

Per accedere al pannello di controllo da qualsiasi dispositivo connesso alla stessa rete di RandA PhotoSharing occorre digitare: http://IP_RASPBERRY_PI/randaps/admin. Avuto l'accesso, inseriamo come utente "admin" e come password "randaps". La pagina si divide in due sezioni: la prima elenca tutte le foto scattate (se facciamo clic sulla data/ora le possiamo visualizzare), se l'utente ha dato o meno il consenso per i social, e l'eventuale e-mail inserita; possiamo anche generare un file Excel con tutti i dati raccolti, op-

pure cancellare tutti i dati (sia dal database che nella cartella "photos") tramite i due pulsanti nella parte inferiore della tabella. La seconda sezione, come mostrato in Fig. 3, contiene tutte le impostazioni di RandA PhotoSharing: dopo averle modificate occorre riavviare il programma, oppure attendere che venga caricata la Start screen (schermata iniziale). Di seguito analizziamo le impostazioni:

- **Theme:** il tema della misurazione (grado di maker, amore, intelligenza, amicizia ecc.);
- **Add new theme:** consente di aggiungere un nuovo tema; nel campo Value dovremo scrivere il nome fisico del tema (ad esempio "love") mentre nel campo Description potremo scrivere la descrizione del tema (esempio: Love Meter); dopo aver aggiun-

to un nuovo tema comparirà un messaggio che ci ricorderà di posizionare i file immagine del tema sotto alla cartella data;

- **Standby:** la prima schermata mostra l'immagine iniziale del tema oppure uno slideshow di tutte le foto scattate fino al momento in cui è visualizzata;
- **Coordinate elementi:** coordinate X e Y in pixel degli elementi che verranno posizionati sulla foto (logo, overlay, result); potremo agire a piacimento su questi valori per cambiare la disposizione dei tre elementi nella foto, tenendo presente che la foto sarà sempre scattata nel formato 1280x720 (a meno che non si modifichi la relativa variabile di configurazione contenuta in randaps.py) e l'origine degli elemento (X=0 e Y=0) è l'angolo superiore sinistro della foto;
- **Save to SD card:** se attivato, lascia le foto nella cartella "photos", altrimenti le cancella;
- **Save to cloud:** se attivato, salva una copia delle foto in Google Drive;
- **Send to social:** se attivato, chiede all'utente se può caricare la foto su Facebook/Twitter (in base a configurazione IFTTT);
- **Send via e-mail:** se attivato, chiede all'utente se vuole ricevere la sua foto via e-mail;
- **Sender e-mail:** l'e-mail del nostro account Gmail che si occuperà di inviare le foto scattate all'utente e a IFTTT;
- **e-mail object:** oggetto dell'e-mail che verrà inviata;
- **IFTTT object:** in questo campo possiamo inserire i tag che serviranno a IFTTT per caricare le foto sui social network;
- **e-mail body:** editor HTML che permette di creare il corpo dell'e-mail che verrà inviata;
- **text_start:** testo che verrà mostrato nella Start screen;
- **text_preview:** testo che verrà

- mostrato durante la scelta dell'effetto da applicare alla foto;
- **text_measurement**: testo che verrà mostrato durante la misurazione dell'emozione;
 - **text_photo**: testo che viene mostrato quando viene scattata la foto;
 - **text_wait**: testo che invita ad attendere dopo che viene scattata la foto;
 - **text_social**: testo con cui possiamo chiedere all'utente se vuole caricare la foto appena scattata sui nostri social network;
 - **text_email**: testo con cui possiamo chiedere all'utente se vuole ricevere la foto appena scattata al proprio indirizzo e-mail;
 - **text_yes**: traduzione della parola "Sì";
 - **test_no**: traduzione della parola "No";
 - **text_end**: verrà mostrato nella schermata Foto risultato (se sono disattivate entrambi le schermate Social ed e-mail) per invitare l'utente a premere OK per concludere.

PERSONALIZZAZIONE

Le immagini che verranno applicate in sovraimpressione ad ogni foto scattata sono contenute nella cartella "data"/; quindi ogni volta che creiamo un nuovo tema dal pannello di controllo dovremo ricordarci di creare le seguenti immagini:

- **[tema]_screen.gif**: immagine GIF (1920x1080) che verrà ridimensionata in base alla risoluzione corrente dello schermo e caricata nello Start screen secondo il tema impostato (ad esempio *love_screen.gif*);
- **[tema]_overlay.png**: immagine PNG (dimensioni libere) che rappresenterà il tema impostato in modo stilizzato (ad esempio *love_end.gif*);
- **[tema]_end.gif**: immagine GIF (1920x1080) che verrà ridimensionata in base alla risoluzione dello schermo e mostrata alla fine dell'esecuzione (ad esempio *love_overlay.png*).

Il file **logo.png** è un'immagine PNG (dimensioni libere) che viene applicata alla foto una volta scattata e non cambia rispetto al tema. Potremo personalizzare ulteriormente l'applicazione agendo sul file *randaps.py* nella sezione "CUSTOMIZABLE VARIABLES". Per farlo, apriamo il file con il comando:

```
sudo nano /var/www/randaps/randaps.py
```

Una volta completate le modifiche è sufficiente chiuderlo e salvarlo con CTRL+X, Y e INVIO.

Per aggiungere o modificare un utente del pannello di con-

trollo web, dobbiamo utilizzare PHPMyAdmin: accediamo all'indirizzo http://IP_RASPBERRY_PI/phpmyadmin (sostituendo "IP_RASPBERRY_PI" con l'indirizzo IP effettivo), selezioniamo il database "randaps" e, successivamente, la tabella "users". Facciamo clic sulla tab "Inserisci", compiliamo il campo "user" e il campo "password"; il campo "password" è codificato in MD5. Per codificare qualsiasi stringa di testo, basta utilizzare uno dei numerosi servizi di codifica MD5 disponibili sul web.

FUNZIONAMENTO

Per lanciare RandA PhotoSharing basta aprire il terminale dal desktop grafico di Raspberry Pi (non in SSH) e scrivere il comando:

```
sudo python /var/www/randaps/randaps.py
```



Listato 1

```
//Inizializzo pin pulsanti
byte pinNext=8;
byte pinOk=9;
byte pinPrev=10;

bool dx=false;
bool sx=false;
bool ok=false;

bool dxTX=false;
bool sxTX=false;
bool okTX=false;

unsigned long value=0;
unsigned int measurements=0;

void setup() {
  //Apro comunicazione seriale con RandA
  Serial.begin(9600);
  pinMode(pinNext, INPUT);
  pinMode(pinPrev, INPUT);
  pinMode(pinOk, INPUT);
  pinMode(A0, INPUT);
  //Abilito resistenze pull-up
  digitalWrite(pinNext, HIGH);
  digitalWrite(pinPrev, HIGH);
  digitalWrite(pinOk, HIGH);
}

void loop(){
  char input = Serial.read();
  if(input=='1'){
    //Se ricevo 1 da RandA: restituisco pulsante premuto
    if (dx && !dxTX){
      dx=false;
      dxTX=true;
      Serial.println("1");
    }else if (sx && !sxTX) {
      sx=false;
      sxTX=true;
      Serial.println("0");
    }else if (ok && !okTX) {
      ok=false;
      okTX=true;
      Serial.println("2");
    }else{
      Serial.println("9");
    }
  }else if(input=='2'){
    //Se ricevo 2 da RandA: restituisco misurazione dai poli
    measureIN();
    int average= value/measurements;
    Serial.println(average);
    value=0;
    measurements=0;
  }
  buttons();
  delay(10);
}

void buttons(){
  if(digitalRead(pinNext)==LOW) dxTX=false;
  if(digitalRead(pinNext)==HIGH && !dxTX) dx=true;

  if(digitalRead(pinPrev)==LOW) sxTX=false;
  if(digitalRead(pinPrev)==HIGH && !sxTX) sx=true;

  if(digitalRead(pinOk)==LOW) okTX=false;
  if(digitalRead(pinOk)==HIGH && !okTX) ok=true;
}

void measureIN(){
  //Leggo per 10 volte input analogico per misurare il voltaggio
  for ( measurements=0; measurements<10; measurements++){
    int val;
    val=analogRead(A0);
    value+=val;
    delay(20);
  }
}
```

Il primo avvio potrebbe richiedere diversi secondi perchè il software sta ridimensionando tutte le immagini contenute nella cartella "data" in base alla risoluzione corrente; per chiudere RandA PhotoSharing basta premere F11 (abilita/disabilita modalità tutto schermo) e fare clic sulla X in alto a destra.

Le schermate che appariranno (in sequenza) sono le seguenti:

1. **Start screen:** visualizza l'immagine iniziale del tema oppure uno slideshow delle foto fatte (in base alle impostazioni) [Attende OK per andare avanti];
2. **Preview:** visualizza immagine della Raspberry Pi Camera, con i bottoni Prev e Next si scorrono gli effetti [Attende OK per andare avanti];
3. **Misurazione:** invita a toccare i due poli (5V e A0) e intanto mostra un conto alla rovescia alla fine del quale scatta la foto, infine mostra un invito ad attendere mentre la foto viene elaborata;
4. **Social:** se attivata, questa schermata mostra la foto risultato e chiede se la si vuole caricare sui social network (collegati a IFTTT) [Attende Prev o Next, che in questo caso valgono come Sì e No];
5. **e-mail:** se attivata, questa schermata mostra la foto risultato e chiede se si vuole ricevere la versione in alta definizione via e-mail offrendo una textbox dove inserirla [Attende la pressione di uno dei tre pulsanti per andare avanti];
6. **Foto risultato:** se sia Social che e-mail (punti 4 e 5) non sono attivati, questa schermata mostra semplicemente la foto risultato [Attende OK per andare avanti];
7. **End screen:** ringrazia l'utente tramite un'immagine personalizzabile (intanto in background

si occupa di inviare l'e-mail e condividere la foto sui social network).

A questo punto il programma ritorna allo Start screen (punto 1) caricando i settaggi del pannello di controllo e applicando le eventuali modifiche alle impostazioni. Per "foto risultato" si intende la foto che RandA PhotoSharing ha scattato durante la misurazione, con l'effetto che abbiamo selezionato, un eventuale logo, e l'immagine di overlay relativa al tema impostato, con all'interno il risultato della misurazione effettuata: per esempio, se il tema

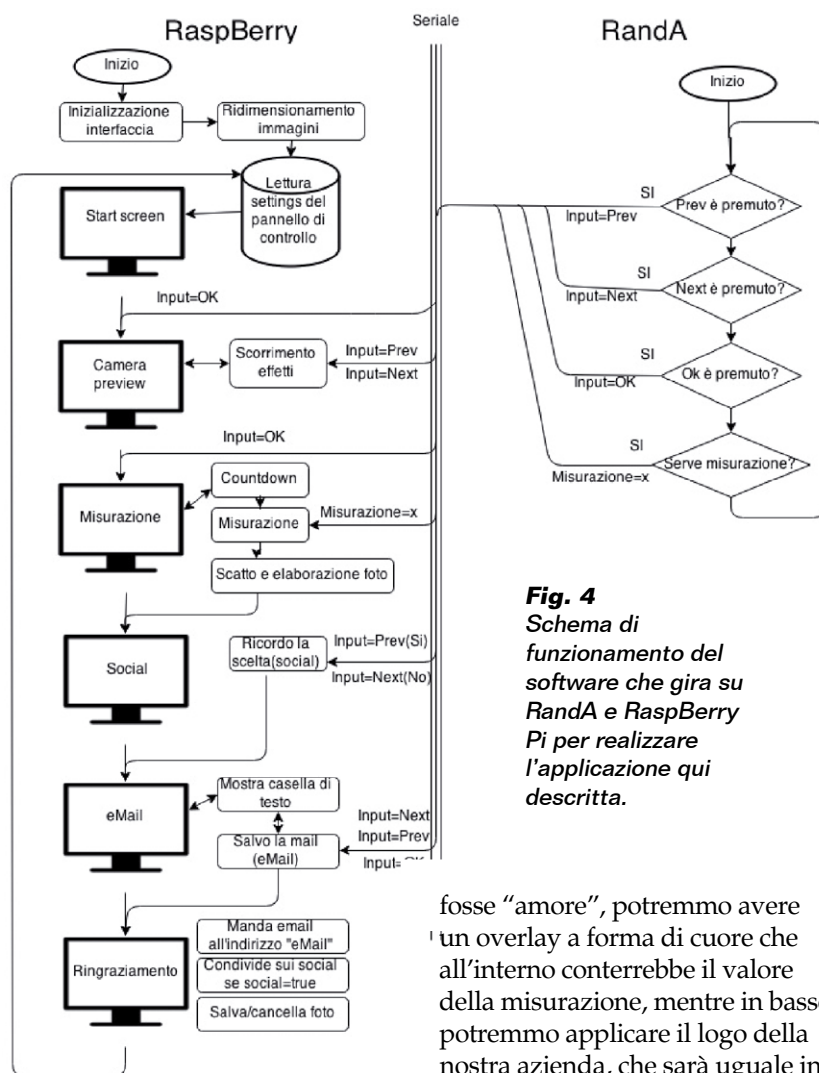
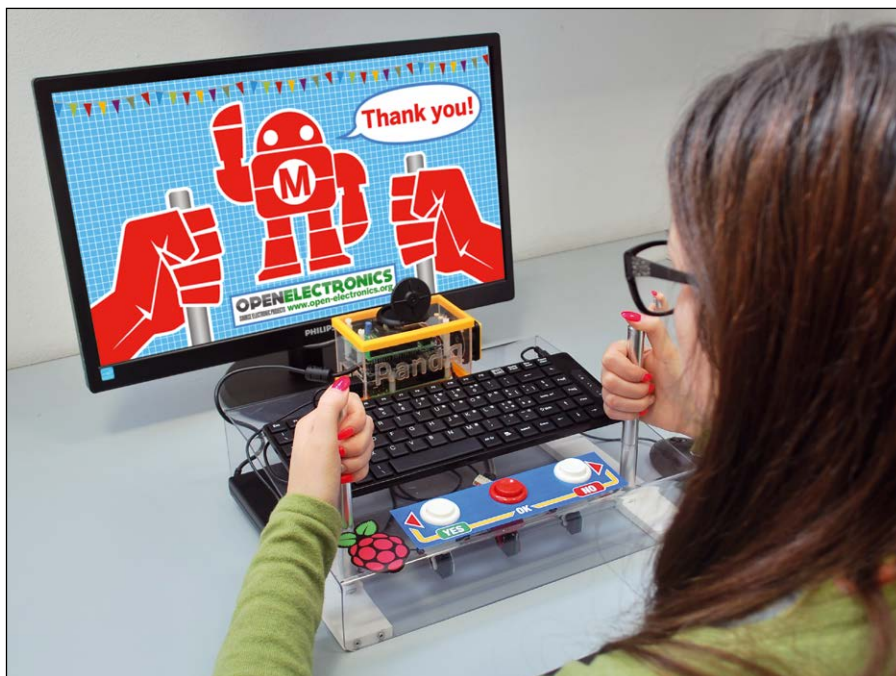
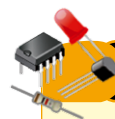


Fig. 4
Schema di funzionamento del software che gira su RandA e RaspBerry Pi per realizzare l'applicazione qui descritta.

fosse "amore", potremmo avere un overlay a forma di cuore che all'interno conterrebbe il valore della misurazione, mentre in basso potremmo applicare il logo della nostra azienda, che sarà uguale in

tutte le foto.

Possiamo vedere lo schema di funzionamento del software in Fig. 4. Nel Listato1 trovate lo sketch da caricare in RandA, che si occuperà di ritornare a RaspBerry Pi i pulsanti premuti e la tensione misurata. Buon divertimento!



per il MATERIALE

Tutti i componenti sono di facile reperibilità e posso essere acquistati presso Futura Elettronica. Il master del circuito stampato, così come l'immagine di Raspbian e lo sketch di RandA, sono scaricabili dal sito della rivista.

Raspberry Pi 2 Tipo B (cod. 8326274RS) è reperibile a Euro 46,00 Euro; il camera module per Raspberry Pi (cod. 7757731RS) costa 31,00 Euro; la scheda RandA, (cod. RANDA) è disponibile al prezzo di 39,00 Euro.

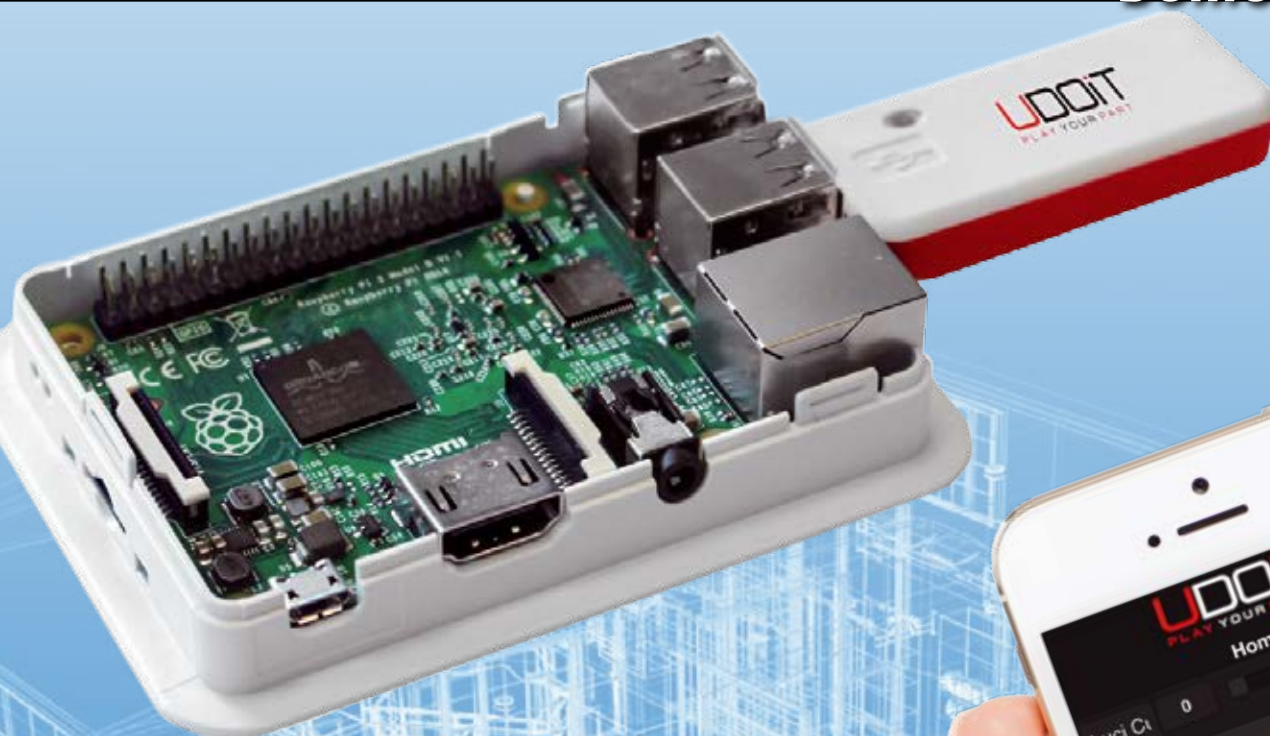
Il materiale va richiesto a:
Futura Elettronica, Via Adige 11,
21013 Gallarate (VA)
Tel: 0331-799775 • Fax: 0331-792287
<http://www.futurashop.it>

Mensile di elettronica applicata, attualità scientifica, novità tecnologiche.

Elettronica In

www.elettronica.in.it

oltre l'elettronica



DOMOTICA OPEN SOURCE con i moduli UDOiT

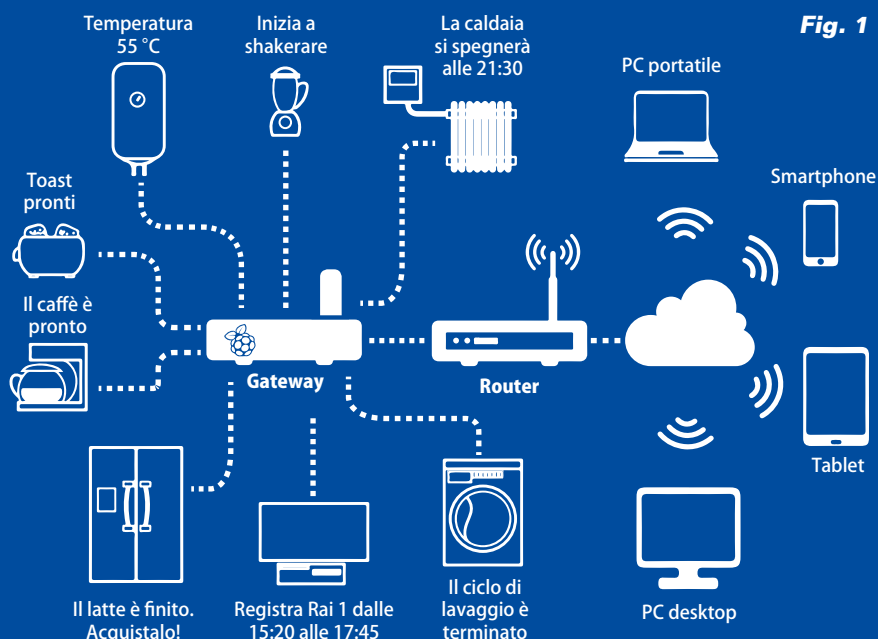
Si parla sempre più di Internet delle cose e delle sue applicazioni. Presentiamo una serie di articoli su una soluzione Open Source di UDOiT basata sui moduli RFTide.

..... di MARCO MAGAGNIN

Da un po' di tempo a questa parte si stanno diffondendo sempre più termini come Iot (Internet of Things), domotica, building automation, e simili. Di conseguenza, come sempre accade, molte aziende stanno orientando le loro proposte commerciali in questa direzione. Esempi si possono trovare nell'impiantistica elettrica, negli elettrodomestici, nei sistemi di supervisione e controllo di ambienti e di persone, soprattutto se bisognose di assistenza, senza trascurare il mondo dell'automobile, dell'intrattenimento, delle comunicazioni,

per finire all'agricoltura, alla meteorologia ed alla gestione degli animali domestici e non. Data la nostra natura di "curiosi" tecnologici ci stiamo interessando, da un po' di tempo, all'evoluzione di questo settore, sia dal punto di vista degli "oggetti" che si sono resi via via disponibili sul mercato, che degli aspetti "filosofici" che caratterizzano le diverse proposte. Parallelamente a questo lavoro di analisi delle soluzioni possibili abbiamo "provato" a costruirci una "mappa" di riferimento basata sulla nostra percezione personale del fenomeno in

Fig. 1

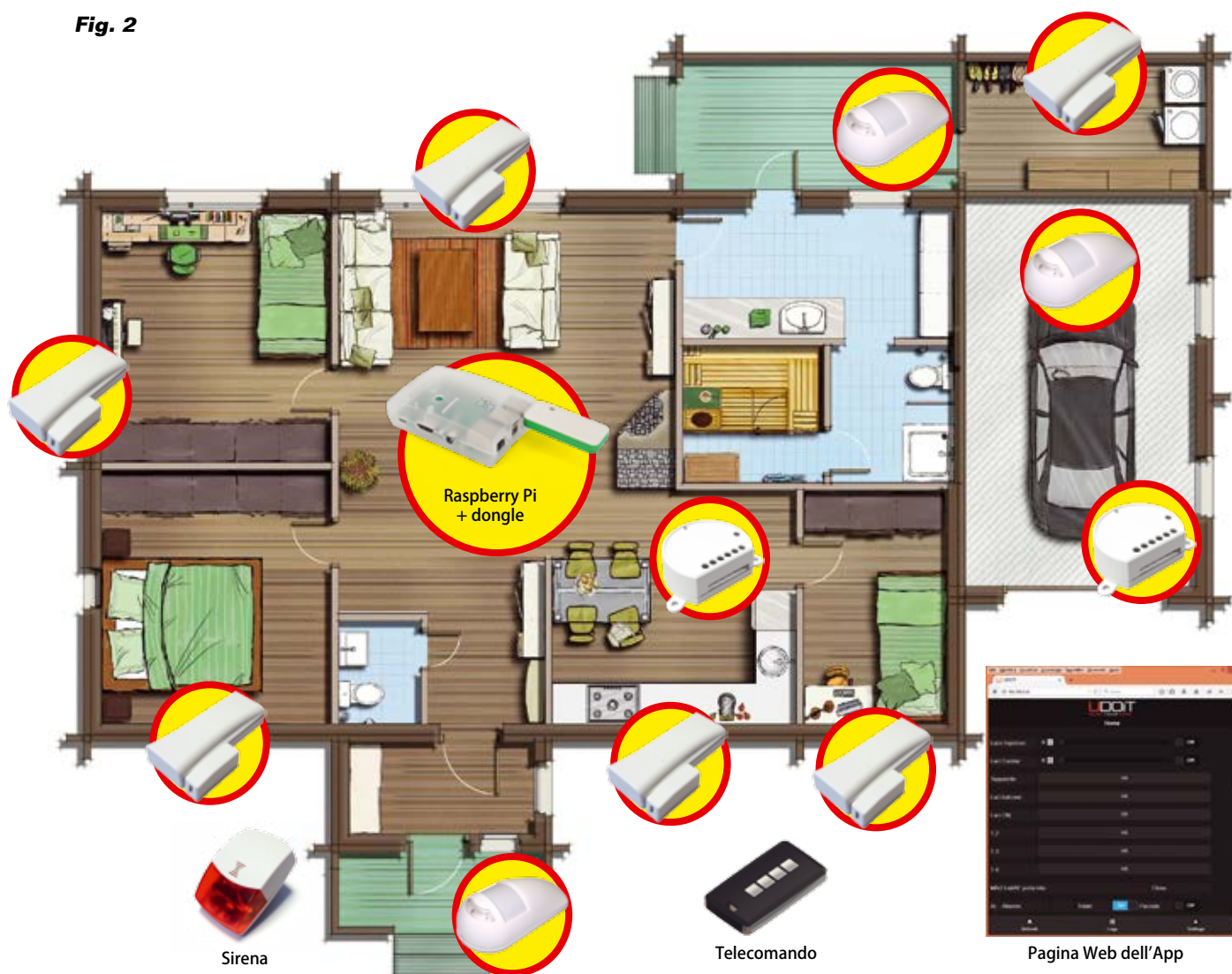


questione. In particolare pensiamo che il motore principale di questo “movimento” sia il desiderio di ridurre sempre di più l’intermediazione umana nella gestione di “contesti” dove la tecnologia è già entrata o si sta apprestando ad entrare. Prendiamo per esempio una casa riscaldata da un impianto a termosifoni, dove i termosifoni hanno delle valvole tradizionali, da operare manualmente. Gli abitanti della casa, quando hanno freddo, aprono le valvole dei termosifoni, che in questo modo trasferiscono calore all’ambiente, mentre quando hanno caldo chiudono le valvole riducendo il trasferimento di calore. Per automatizzare il processo di regolazione della temperatura ambiente si potrebbe utilizzare un “termostato”. Ciò richiede un intervento sull’impianto per sostituire gli “attuatori” manuali, con altri in grado di essere operati automaticamente. Si può intervenire a livello centrale, sulla caldaia, oppure sostituendo tutte le valvole dei termosifoni con altre, operabili automaticamente, in modo da poter regolare singolarmente la temperatura di ogni stanza. Poi, per poter “automatizzare” il processo è necessario “concre-

tizzare” il concetto di caldo e di freddo in termini di grandezze numeriche, per esempio in “gradi centigradi”, per poter regolare, per esempio, un termostato. A questo punto il controllo umano si riduce a “regolare” il termostato centrale o quelli delle singole stanze. Ma perché riscaldare un’abitazione allo stesso modo sia quando è abitata sia quando tutti gli abitanti sono fuori e la casa è disabitata? Ogni volta che si esce di casa si dovrebbe regolare il o i termostati, e lo stesso al ritorno. È vero, esistono termostati con regolazioni multiple basate su timer. Ma ancora, perché riscaldare a tutta potenza quando apriamo le finestre? Una soluzione potrebbe essere quella di collegare un sistema di gestione di presenza, come i controlli di sicurezza periferici dell’antifurto, con il sistema di controllo termico, in modo da farli interagire in base ad un insieme di regole di comportamento finalizzate a gestire correttamente le diverse condizioni che si possono presentare. Bene, avete mai provato a collegare un antifurto ad un termostato? O la centralina meteo all’impianto di irrigazione del giardino? O ancora il citofono al televisore ed al

suo telecomando? E tutto quanto insieme gestibile dallo smartphone quando siete in vacanza alle Maldive? Magari per aprire la porta di casa a qualcuno di fiducia dopo averlo riconosciuto dalle telecamere integrate nello spioncino? Per concludere, secondo noi la chiave di successo di tutto quello che abbiamo descritto nella nostra specie di sogno ad occhi aperti si chiama “interoperabilità” dei sistemi. Che significa? Significa che al momento stiamo assistendo alla proliferazione sul mercato di sistemi, diciamo così, di gestione ambientale, di tipo proprietario che operano in base a specifiche ed a protocolli propri e, per lo più, non documentati, e quindi chiusi rispetto alla possibilità di interfacciarsi ed interagire con sistemi di produttori diversi. Ciò accade, per ironia, anche quando i sistemi proprietari di produttori diversi adottano gli stessi standard di comunicazione tra centralina di controllo, sensori ed attuatori. Questo non giova certo a realizzare la visione di un mondo interconnesso e dotato della capacità, entro certi limiti, di autogestirsi. Per “interoperabilità” intendiamo l’esatto contrario di quanto abbiamo appena descritto. Devo poter scegliere il sistema di gestione dell’impianto termico che preferisco, quello per il condizionatore, quello per la gestione energetica, quello per le comunicazioni ed uno o più per gli elettrodomestici senza dovermi ritrovare con la le pareti tappezzate di centraline che non si parlano tra di loro. Il sistema di gestione di tutti gli impianti deve essere unico ed in grado di gestire tutti gli altri sistemi “periferici”. La soluzione? A nostro parere non è così a portata di mano come potrebbe

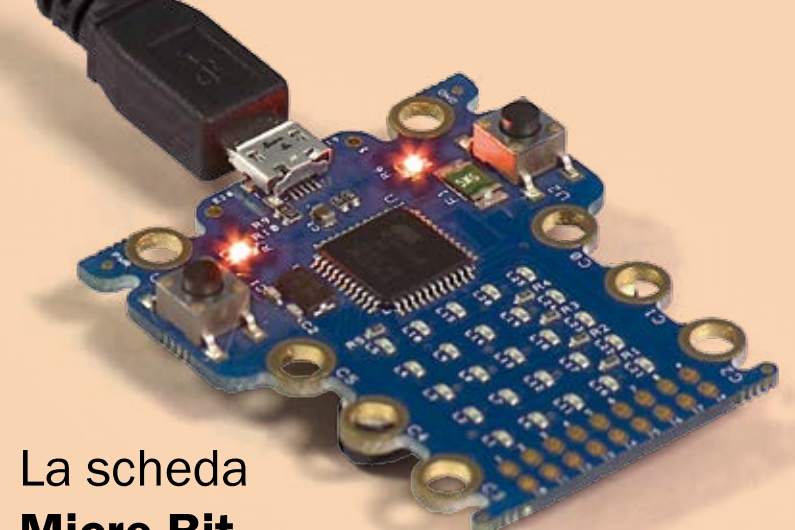
Fig. 2



apparire ad un'analisi superficiale, anche se qualcosa si sta muovendo. I produttori dovrebbero "aprire" i loro sistemi, o quantomeno dotarli di interfacce di comunicazione (API, librerie, web-service e simili) per mettere in grado altri produttori di realizzare sistemi in grado di interconnettere i singoli dispositivi e gestirli in base a regole unificate. Una prima risposta, anche se probabilmente non quella definitiva, sta arrivando dal mondo "open source", con soluzioni realizzate da comunità di sviluppatori finanziati dai produttori di hardware (centraline, sensori ed attuatori), in modo da garantire la possibilità

di integrare tra di loro in modo uniforme e normalizzato le soluzioni proposte dai diversi produttori. Utopia? Non sappiamo, di fatto è la modalità con la quale si sono imposti nel mondo le reti internet (TCP/IP), il web e GNU/Linux, dopo essere sopravvissuti alle infinite schiere di "competitori" con soluzioni proprietarie, divenendo i dominatori incontrastati dei rispettivi mercati di riferimento. In fondo stiamo parlando di Internet of Things, vero? Per coerenza sarebbe bene accettarne le regole. Che poi si parli di "internet" per reti di sensori ed attuatori basate su protocolli di comunicazione totalmente diversi da quelli di

internet, potrebbe essere un buon argomento di discussione. Quanto meno ci limitiamo a prendere atto che al momento si tratta più di uno slogan di marketing che la definizione di una precisa tecnologia, ancora lontana da essere disciplinata in termini di standard e metodologie di progettazione e sviluppo. In quest'ottica, affrontiamo l'argomento presentandovi una soluzione open source, tra le molte esistenti, per la gestione di un ambiente domestico, basata sui sensori, gli attuatori e le periferiche commercializzate da UDOIT, una start-up italiana che realizza prodotti elettronici per il mondo dei makers. Oltre



La scheda **Micro Bit** della BBC

Così come nel passato per aumentare e diffondere la vendita dei libri è stato necessario che i potenziali clienti imparassero a “leggere”, oggi è altrettanto necessario che, gradualmente, gli utilizzatori di sistemi di controllo evoluti, imparino a conoscere, configurare e personalizzare i loro sistemi, utilizzando strumenti di codifica di regole, questo sì, i più semplici possibili. Negli stati più attenti a questi fenomeni in divenire non solo per quanto riguarda l'IoT, ma pressoché in tutti gli aspetti della vita umana, si stanno introducen-

do nuove discipline nelle scuole di tutti i gradi, fin dalle primarie, come i corsi di “coding”, mentre le discipline informatiche stanno cessando di essere considerate materie a sé e stanno divenendo competenze “trasversali” in tutte le discipline oggetto di studio.

Citiamo solo due fatti che hanno caratterizzato il mondo della scuola inglese. Il “regalo” di 15'000 Raspberry Pi da parte di Google alle scuole inglesi e quello della BBC di un milione di schede Micro Bit agli studenti delle scuole secondarie.

alla piattaforma Raspberry Pi (utilizzata nel progetto della domotica qui descritto), sviluppa shield Arduino dedicate al mondo dell'IoT e del M2M con moduli radio in tecnologia LoRa e dispositivi wireless long range per la terminazione volo dei droni, come richiesto dalla recente normativa ENAC. Maggiori informazioni sono disponibili sul sito web www.udoit.eu il tutto 100% made in Italy. Nella **Fig. 1** è rappresentato lo schema di riferimento generale di quello che noi crediamo essere il dominio definibile come Iot, all'interno del quale si può collocare la soluzione hardware e software che vi presentiamo.

Prima di descrivere lo schema di riferimento ci permettiamo alcune doverose precisazioni riguardo l'applicazione software che vi

presentiamo, valide del resto per tutte le applicazioni open source in genere ed anche, a leggere bene le clausole di licenza, anche per le applicazioni con licenza a pagamento. Le applicazioni rilasciate con licenze open source come la Creative Common, ed altre, senza oneri economici da parte degli utilizzatori, in altre parole gratuita, è bene sottolineare che vengono rilasciate “AS IS” ovvero nello stato in cui si trovano, e non è prevista né può essere pretesa alcuna forma di garanzia o di assistenza al loro utilizzo. L'installazione, la personalizzazione e la gestione dell'applicazione è a totale carico dell'utilizzatore e non è previsto alcun supporto in caso di malfunzionamento, richieste di configurazioni, personalizzazioni, modifiche o nuove funzio-

nalità. La disponibilità dei codici sorgenti permette, a chi ha le conoscenze adatte, di risolvere in autonomia eventuali situazioni anomale o di apportare le modifiche desiderate per personalizzare la propria installazione. A nostro avviso è necessario che gli utilizzatori di queste applicazioni abbiano già acquisito un certo grado di conoscenza e dimestichezza con la gestione, amministrazione e programmazione di sistemi GNU/Linux. Al di là della disponibilità di soluzioni Open Source l'argomento IoT introduce una complessità più elevata rispetto a quella tipica associata alle soluzioni di controllo di impianti singoli. Un sistema di riscaldamento gestito da un controllo a termostato richiede funzionalità di gestione limitate ed adatte a soddisfare pressoché ogni ambito dove il controllo viene installato. Un sistema nella filosofia IoT richiede di integrare la gestione di diversi impianti in base ad un insieme di “regole” che devono essere adattate, in base alla tipologia degli impianti ed alle esigenze degli utilizzatori che, tra l'altro sono soggette a continue variazioni nel tempo. È anche vero che le tendenze commerciali spingono a far passare per “semplice” tutto ciò che si vuole abbia ampia diffusione sul mercato. Nel nostro caso, oltre che di una rivoluzione tecnologica, siamo in presenza di una rivoluzione culturale, per la quale viene richiesto agli utilizzatori di acquisire nuove competenze per usufruire al meglio dei nuovi servizi. La soluzione di questo “problema” sta nel formare gradualmente gli utilizzatori ad usufruire delle nuove tecnologie. Vedi al riguardo il box “La scheda MicroBit della BBC”. Tornando al tema più specifico dell'IoT vediamo di descriverne gli

aspetti essenziali, facendo riferimento sempre allo schema di **Fig. 1** Possiamo vedere l'universo IoT come un insieme di discipline e tecnologie che collaborano per interconnettere gli aspetti più svariati del mondo reale con applicazioni in grado di gestirli per conto degli "umani", sollevandoli dal doversene occupare di continuo. Come contropartita, agli "umani" spetta il compito di "insegnare" ai sistemi di comunicazione, coordinamento e controllo, le regole in base alle quali si desidera che i sistemi stessi si comportino.

Per prima cosa vediamo l'esistenza di "insiemi" di sensori che fanno capo ad uno o più "concentratori" e "gateway". Consideriamo un concentratore come un dispositivo che coordina un insieme di sensori ed attuatori della stessa famiglia, finalizzati a presidiare un luogo circoscritto, come un appartamento od una fabbrica. In genere queste "famiglie" di sensori ed attuatori utilizzano per comunicare diverse tipologie di "mezzi": bus su connessioni cablate, sistemi wireless su frequenze dedicate, tecnologie e protocolli di varia natura. Raramente vengono utilizzate tecnologie

e protocolli propri della rete internet. Un gateway, invece, si occupa delle "relazioni esterne", ovvero di far comunicare i sensori con entità di controllo distribuite, in genere messe a disposizione sfruttando la tecnologia cloud e dislocate ovunque nel mondo, dalla stessa stanza agli antipodi del pianeta. Nel caso che vi presentiamo concentratore e gateway "alloggiano" nello stesso dispositivo. Nello stesso dispositivo, che d'ora in poi chiameremo "gateway", alloggiano anche i motori delle regole di gestione "locali". Non necessariamente per accendere una lampada quando entriamo in una stanza dobbiamo coinvolgere un servizio su un server che sta in Australia. Il servizio che sta in Australia ci può tornare utile per gestire il nostro "mondo" quando siamo in vacanza oppure per unificare la gestione di regole da applicare a diversi ambienti in modo coordinato. Per esempio quando torno verso la casa di città ad una certa distanza di chilometri alzo il termostato, oppure faccio accendere il forno quando esco dall'ufficio. Per la parte di gestione assumono estrema importanza le tecnologie di interazione tra "umani" e sistemi. In pratica applicazioni interattive sviluppate in base a molteplici tecnologie in grado di integrare dispositivi di diversa natura come telecomandi, applicazioni web, tablet, smartphone, video-lavagne, smart-watch, smart-car ecc. L'applicazione Open Source che vi presentiamo, si configura come un ponte tra una o più reti di moduli di acquisizione e controllo e la rete internet che permette la remotizzazione della visibilità e dell'archiviazione dei dati acquisiti oltre che il controllo, sempre da remoto delle peri-

Fig. 4



feriche gestite. Di fatto lo scopo dell'applicazione nel suo insieme è di garantire la trasmissione e la gestione delle informazioni dal mondo fisico dei sensori, degli attuatori e delle eventuali periferiche ad essi collegati. Da un lato il gateway gestisce la trasmissione delle informazioni mediante la rete mesh costituita dai moduli RF presenti su ciascun sensore ed attuttore e dall'altro, contemporaneamente, sta la capacità di elaborare i dati ricevuti in base

Fig. 3



Fig. 5

Fig. 6

a regole prestabilite, permettere la gestione in remoto da parte dell'utente fornendo un'apposita interfaccia grafica in tecnologia web. In **Fig. 2** è visibile lo schema d'insieme della soluzione proposta da UDOiT.

I moduli RF di cui sono dotati i sensori, gli attuatori e gli altri componenti del sistema sono i moduli RFTide di Aurel, lavorano in trasmissione e ricezione sulla banda degli 868 Mhz ed i dati scambiati sono codificati sulla base di un protocollo unificato del quale sono rilasciate le specifiche. Un dongle USB dotato anch'esso di un modulo RFTide, permette ai vari sensori ed attuatori di comunicare con l'unità centrale di elaborazione, il gateway vero e proprio, che utilizza un Raspberry Pi B 2. L'unità centrale di elaborazione raggruppa in sé diverse funzionalità che vengono svolte in multiprogrammazione, grazie al sistema operativo GNU/Linux ed alla nuova CPU a quattro processori che forniscono funzionalità e capacità di elaborazione ottimali per l'applicazione di controllo. I sensori e gli attuatori

messi a disposizione da UDOiT permettono di realizzare una rete "mista" di controllo ambientale. Il modulo "indispensabile" per la gestione della rete, che deve essere fisicamente installato su Raspberry Pi è il "dongle" USB, visibile nella **Fig. 3**

Nel protocollo adottato dai moduli UDOiT, che descriveremo nel dettaglio nei prossimi articoli, ciascuna "rete" di sensori ed attuatori è riconosciuta da un codice "network" univoco, in modo che non possano verificarsi "intrusioni" tra moduli di reti adiacenti. Il dongle USB nasce codificato con un identificativo di "network" univoco e non modificabile. Sulla base di questo codice, utilizzando le funzionalità di "learning" previste nella applicazione di controllo, vengono "personalizzati" ed "associati" alla rete tutti i successivi moduli aggiuntivi. L'applicazione di controllo presenta un'interfaccia utente grafica fruibile da un browser web sia utilizzando un PC che un qualsiasi dispositivo mobile, smartphone, tablet, notebook, ecc. Una volta creata la rete, installati ed associati tutti i moduli desiderati, i comandi possono essere impartiti dall'interfaccia di "comando" dell'applicazione, che si configura "dinamicamente" in base ai moduli associati. Un'alternativa è di "associare" alla rete uno o più telecomandi, del tipo visibile in **Fig. 4**.

Ciascun telecomando è dotato di quattro tasti, a ciascuno dei quali, come vedremo in dettaglio nei prossimi articoli, può essere assegnato il compito di comandare uno o più attuatori ma anche di eseguire azioni (programmi) sul gateway, anche predisposti dall'utilizzatore. I diversi telecomandi possono svolgere tutti le stesse funzioni oppure, ciascuno,

**Fig. 7**

funzioni diverse, a scelta dell'utilizzatore. Le funzioni associate ai "tasti" dei telecomandi, tutte o in parte, possono essere rese disponibili anche sull'interfaccia grafica "web" di comando. Il sensore "PIR", visibile in **Fig. 5**, permette di rilevare una presenza o, più precisamente, una sorgente di calore, nell'ambiente antistante il suo angolo di visione. A fronte della rilevazione di una presenza può essere attivata un procedura di segnalazione dell'evento, per

Fig. 8

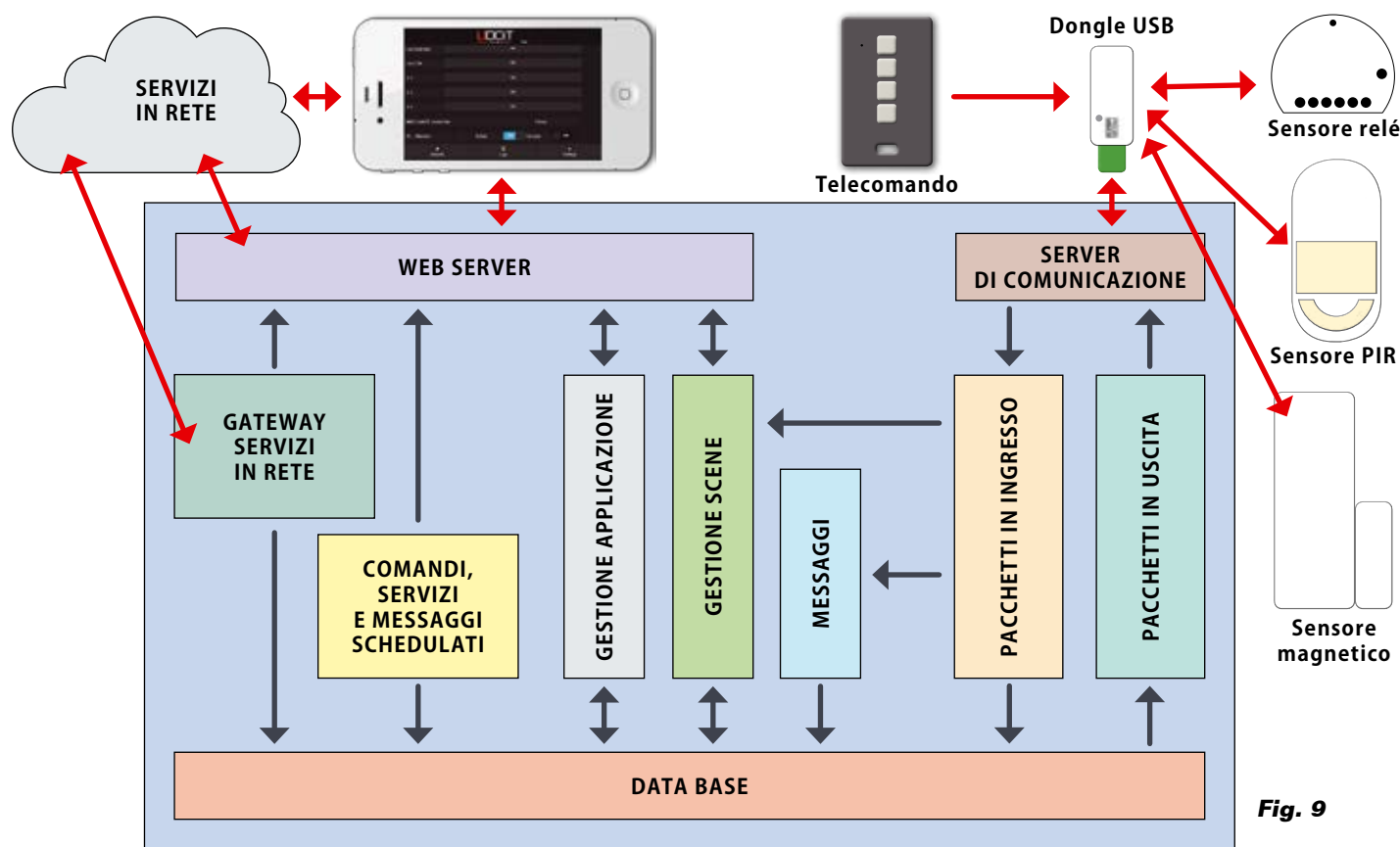
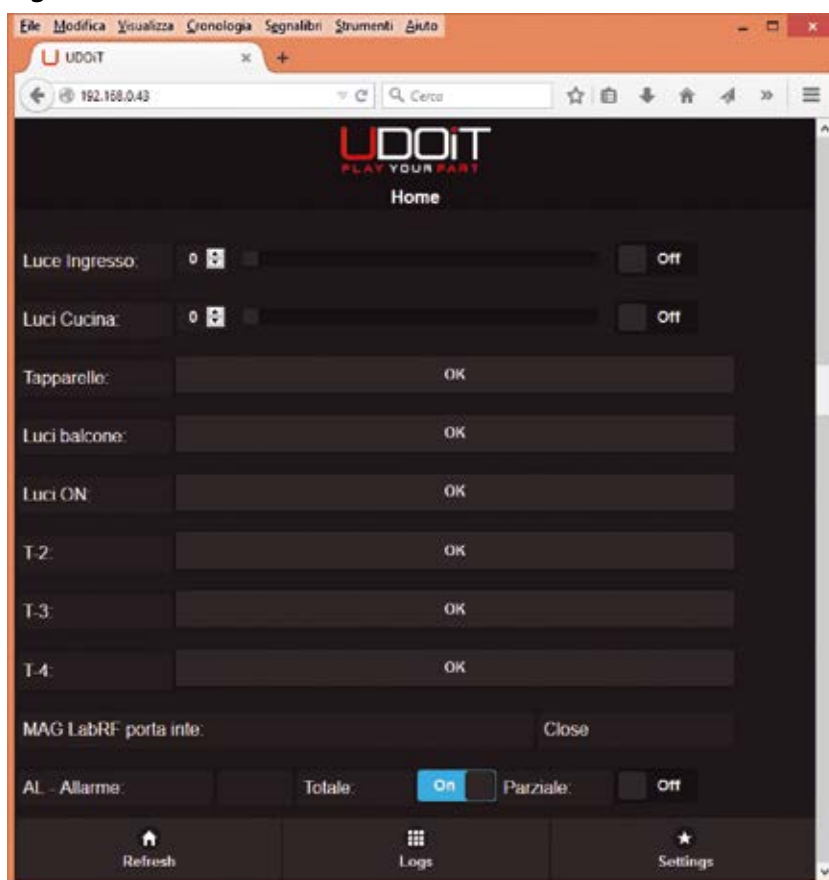


Fig. 9

esempio attivando una sirena per un certo tempo ed inviando un messaggio e-mail o su un altro sistema di messaggistica, oppure attivare un'utenza, come per esempio accendere una o più lampade. Ovviamente le diverse funzioni possono essere combinate. Il sensore PIR segnala anche i tentativi di manomissione e lo stato di carica della batteria. Queste segnalazioni vengono visualizzate sull'interfaccia grafica dell'applicazione di controllo. In **Fig. 6** è visibile il sensore magnetico, sempre collegato in radio frequenza, che si utilizza per rilevare l'apertura e la chiusura di porte e finestre, ma anche di qualsiasi oggetto dotato di meccanismi di apertura come cassetti o bauli. Può essere utilizzato anche per rilevare se viene asportato un oggetto semplicemente appoggiato ad un piano. Anche questo sensore può far parte di una "sottorete" di sensori dedicati al controllo

Fig. 10



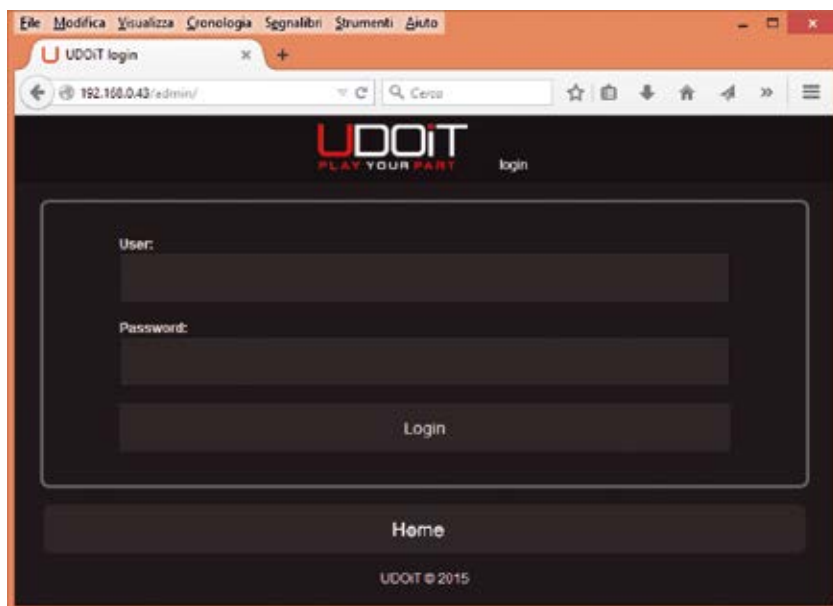


Fig. 11

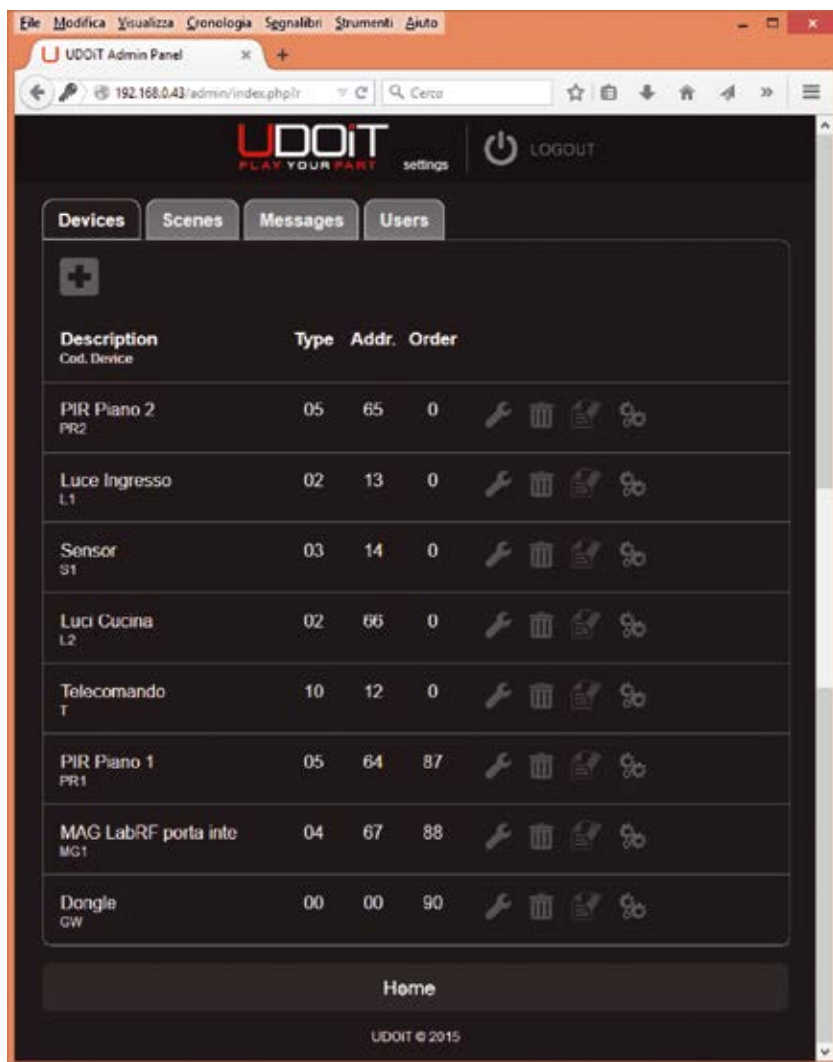


Fig. 12

ambientale, oppure utilizzato per innescare un comando verso uno o più attuatori od anche inviare messaggi o scrivere loghi di accessi, per esempio correlati da fotografie di chi o cosa ha aperto o chiuso un accesso. Anche questo sensore segnala i tentativi di manomissione e lo stato della batteria. In Fig. 7 è possibile vedere il modulo relè. In realtà il modulo relè funziona contemporaneamente sia come attuttore che come sensore. Il modulo relè si utilizza per alimentare, sia da remoto che con intervento manuale, un carico a 220V fino ad un massimo di 10 Ampere e può essere operato sia da remoto, tramite la rete senza fili e l'applicazione di gestione, sia manualmente come un comune interruttore. La funzione sensore permette al modulo di trasmettere il suo stato, in modo da informare il sistema di gestione, anche quando viene operato manualmente. Il modulo relè è stato progettato per essere alloggiato nelle comuni cassette dei frutti degli impianti elettrici tradizionali. Il modulo dimmer è in fase di sviluppo. La sua funzione è di accendere una fonte luminosa permettendone di regolare la luminosità, sia utilizzando un modulo telecomando che mediante l'interfaccia grafica dell'applicazione di controllo. Il modulo sirena è anch'esso in fase di sviluppo e permetterà di realizzare un sistema di controllo ambientale. In Fig. 8 è visibile un'ipotesi del prototipo, ma non è detto che corrisponda alla versione definitiva. Saremo più precisi nei prossimi articoli. In caso di segnalazione di intrusione il sistema può essere programmato, oltre che a segnalare l'evento sull'interfaccia grafica dell'applicazione di controllo o con un messaggio e-mail o

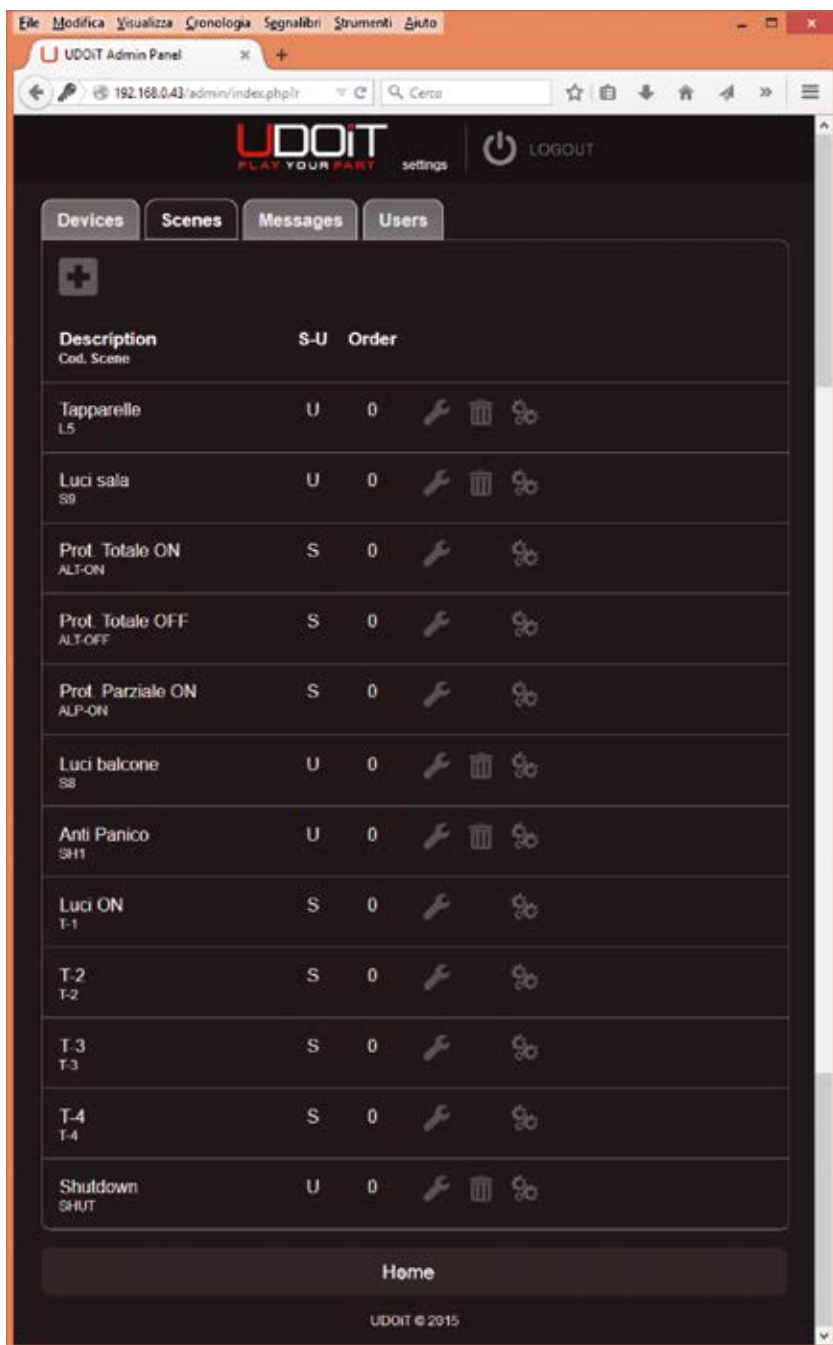


Fig. 13

cimali). I pacchetti in uscita dal sistema possono essere generati da azioni sul telecomando, sull'interfaccia grafica, da eventi del sistema, da "regole" di comportamento in base allo stato generale del sistema o da altre fonti. I pacchetti di comando in uscita vengono instradati verso il "dongle" USB, compatibilmente con la disponibilità di quest'ultimo a riceverli. Il "servizio" che gestisce i pacchetti in ingresso, analizza la provenienza ed il contenuto di informazioni di ciascun pacchetto. In base ai dati contenuti aggiorna i valori dei sensori nelle tabelle di stato ed in base alle regole di comportamento codificate nel sistema, esegue i comandi ed attiva i processi previsti. Il database di supporto all'applicazione è costituito da numerose tabelle, che approfondiremo nei prossimi articoli, che contengono le caratteristiche dei moduli fisici (sensori, attuatori e periferiche in genere), lo stato degli stessi e le regole di comportamento a fronte delle diverse situazioni che si vengono a creare. Altre tabelle raccolgono i comandi, provenienti dalle diverse origini, che devono essere immessi nella rete verso i moduli esterni e gli stati di ritorno dai sensori e dai moduli in genere.

Il comportamento del sistema può essere personalizzato con programmi e script che possono essere scritti o personalizzati direttamente dall'utilizzatore. Il "collante" del sistema è costituito dall'applicazione grafica per configurare ed operare l'intero sistema (la cui videata di "comando" è visibile in Fig. 10). Le videate di "comando" sono ad accesso libero, mentre le funzionalità di gestione dell'applicazione sono protette da user id e password (Fig. 11) e permettono di:

simile, anche con l'azionamento del modulo sirena per un certo periodo. Lo stesso modulo può essere utilizzato come segnalatore in un sistema antipanico. Nello schema di Fig. 9 è visibile la struttura logica dell'applicazione che realizza il gateway. Il modulo "server di comunicazione" è realizzato in logica "event driven", e gestisce i pacchetti in ingresso ed uscita dal dongle

USB. Il modulo di gestione dei pacchetti è composto da due sezioni principali, in pratica due server all'interno dello stesso modulo software, che gestiscono i due "rami" della comunicazione seriale, uno per la gestione dei "pacchetti" in uscita ed uno per quelli in ingresso dai moduli. Il protocollo di comunicazione è composto da pacchetti di undici byte (ventidue caratteri esade-

- Aggiungere e far riconoscere al sistema sensori, attuatori e moduli in genere, come i telecomandi, assegnando loro un nome di riconoscimento, una descrizione, un indirizzo e le modalità di comportamento (Fig. 12);
- Eseguire il "learning" di ciascun modulo periferico, in modo da "associarlo" al network di riferimento;
- Aggiungere al sistema le "scene", ovvero sequenze di operazioni preconfigurate che possono essere innescate ed eseguite a fronte di un evento

"automatico" o di un comando intenzionale dell'utilizzatore, via telecomando od interfaccia web (Fig. 13);

- Ciascun modulo associato al sistema che produce e trasmette "dati" ed informazioni, oltre ad innescare l'aggiornamento della tabella che contiene gli stati del sistema genera messaggi automatici che sono evidenziati nella relativa pagina web. È possibile personalizzare la descrizione dei messaggi e la loro posizione all'interno della pagina dedicata. È anche possibile

inserire messaggi personalizzati che vengono alimentati da programmi scritti dall'utilizzatore (Fig. 14);

- Le pagine web di controllo e di configurazione vengono composte automaticamente in base ai sensori alle scene ed ai messaggi configurati. È possibile personalizzare ulteriormente l'interfaccia web nascondendo eventuali elementi indesiderati ed ordinando i rimanenti nella sequenza desiderata (Fig. 10 e Fig. 15);
- In caso di "attivazione" di un sensore o di esecuzione di una scena è possibile comandare l'esecuzione di un processo (script) preparato dall'utilizzatore. Questa possibilità rende il sistema estremamente flessibile ed adattabile ad un gran numero di esigenze pratiche. È possibile inviare mail o messaggi su ciascun tipo di social network, alimentare database e sistemi remoti in genere, inviare sms gestire messaggi vocali, alimentare file di log, scattare foto ed archivarle od inviarle ad uno o più destinatari. Il limite all'espandibilità ed alla personalizzazione è dato dalla fantasia e dalle funzionalità disponibili nel sistema operativo di riferimento;
- Un sottosistema già preconfigurato, permette di allestire un sottoinsieme di sensori (o anche tutti) come facenti parte di un sistema di controllo ambientale che, anche se non è possibile definirlo come un vero e proprio antifurto, una volta attivato permette di comandare attuatori o processi di avviso a fronte di intrusioni non desiderate nella zona "sorvegliata" dai sensori. È possibile distinguere ed attivare in alternativa un

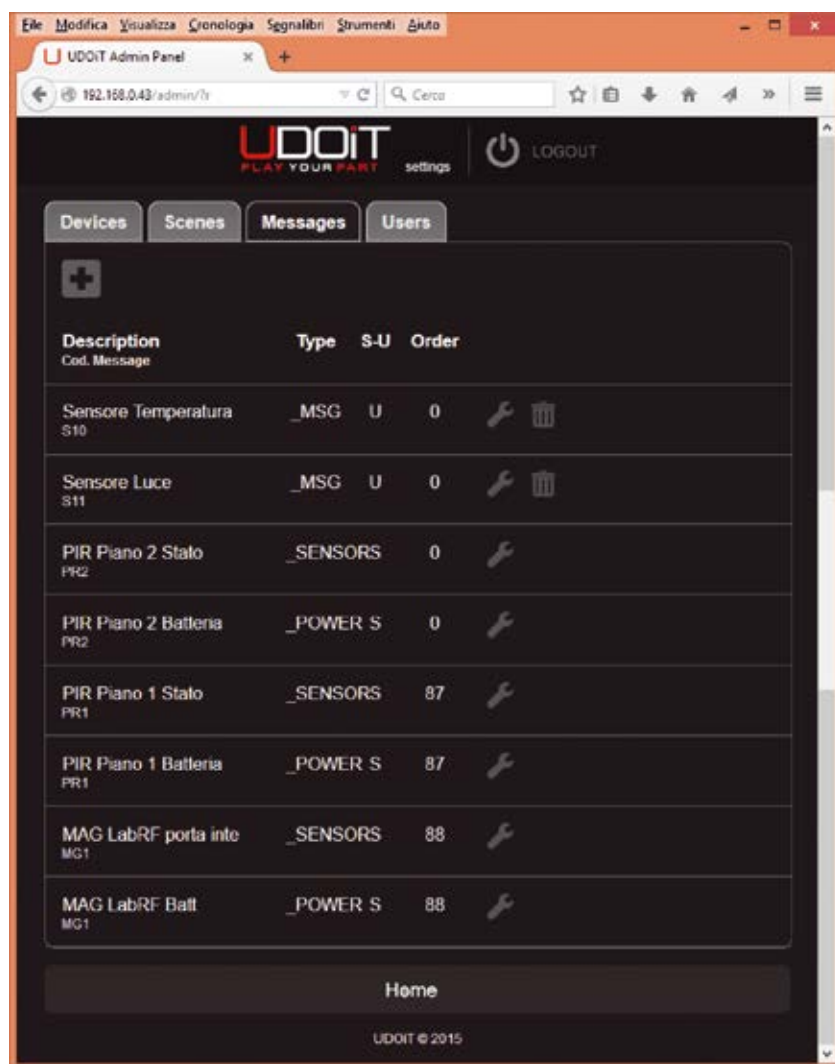


Fig. 14

Fig. 15



The screenshot shows a web browser window with the URL 192.168.0.43/message.php. The page title is 'UDOiT' with the tagline 'PLAY YOUR PART'. Below the title is a section titled 'Logs' containing a table of sensor data. At the bottom of the page are three buttons: 'Refresh', 'Home', and 'Settings'.

Sensore	Valore	Data e Ora
Sensore Luce:	220 Lux	11-05-2015 23:50:
Sensore Temperatura:	26,5 C	11-05-2015 23:50:
PIR Piano 1 Batteria:	Low	11-05-2015 23:54:
PIR Piano 1 Stato:	OK	11-05-2015 23:53:
MAG LabRF Batt:	OK	10-05-2015 15:16:
PIR Piano 2 Stato:	OK	11-05-2015 23:52:
PIR Piano 2 Batteria:	OK	11-05-2015 23:53:
MAG LabRF porta inte:	Tamper Mano.	10-05-2015 15:15:

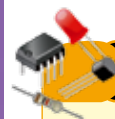
sistema di supervision parziale o periferico oppure globale;

- Esiste ovviamente la possibilità di integrare l'applicazione con programmi da eseguire a tempo od a scadenza, lanciati dallo schedatore "cron" di GNU/Linux. In questo modo è possibile, tra le mille altre possibilità, realizzare un impianto di irrigazione in grado di modificare il suo intervento in base ai dati acquisiti dai sensori di temperatura, umidità e pioggia se non addirittura dopo aver consultato le previsioni meteo;
- La possibilità di integrare e di adattare il comportamento dell'applicazione anche grazie

ad informazioni provenienti da fonti remote, come sensori controllati da sistemi long range, sms o via rete internet, cablata o wifi ne rendono possibile l'utilizzo per il controllo e la gestione anche di ambienti diversi dalle mura domestiche, pensiamo a campi coltivati, parcheggi, centraline meteo collocate in luoghi poco accessibili, ecc.

Nei prossimi articoli descriveremo i singoli moduli, sensori, attuatori, dongle e telecomandi che compongono la famiglia UDOiT, ciascuno con le funzionalità che lo caratterizzano ed il corrispondente protocollo di comunicazione.

ne. Descriveremo le funzionalità dell'applicazione e le modalità di installazione e configurazione su Raspberry Pi o altri microcomputer. Infine descriveremo l'utilizzo dell'interfaccia web dell'applicazione che permette di personalizzare la rete di sensori e degli altri moduli ed il loro utilizzo pratico. Sempre mediante l'interfaccia web è possibile definire scene di comportamento con le diverse possibilità di attivazione. Così come è possibile espandere le funzionalità dell'applicazione stessa mediante il meccanismo degli script e come integrare le funzionalità dell'applicazione con altre applicazioni o sistemi già esistenti, come per esempio l'invio di mail o comunicazioni su svariati sistemi come i social network o l'integrazione di sensori diversi da quelli della famiglia UDOiT.



per il MATERIALE

I prodotti presentati in questo progetto possono essere acquistati presso Futura Elettronica. Il dongle USB (cod. RFT868USB) costa 39,00 Euro, il sensore magnetico (cod. RFT868MAG) è disponibile a 49,00 Euro, il sensore di movimento (cod. RFT868PIR) è in vendita a 55,00 Euro, il modulo a relé (cod. RFT868REL) costa 55,00 Euro e il modulo a dimmer (cod. RFT868DIM) è disponibile a Euro 55,00.

Il radiocomando a 4 canali (cod. RFT8684CH) è in vendita a Euro 39,00 Euro, e la Raspberry Pi 2 Tipo B (cod. 8326274RS) a Euro 44,00.

Il materiale va richiesto a:
Futura Elettronica, Via Adige 11,
21013 Gallarate (VA)
Tel: 0331-799775 • Fax: 0331-792287
<http://www.futurashop.it>

Mensile di elettronica applicata, attualità scientifica, novità tecnologiche.

Elettronica In

www.elettronica.in.it

oltre l'elettronica



Abbiniamo l'elettrodo per il riconoscimento di gesture 3D a Raspberry Pi, per realizzare un'applicazione in cui con i gesti facciamo scorrere su uno schermo HDMI delle immagini.

di MATTEO DESTRO

GESTIC: PROVIAMOLA CON RASPBERRY PI

Nel fascicolo di giugno scorso abbiamo descritto un nuovo elettrodo GestIC pensato e realizzato per l'interfacciamento con le board Arduino Uno Rev. 3, Arduino Leonardo Rev. 3 e Raspberry Pi B+/2.0, e vi abbiamo proposto un'applicazione con

Arduino, promettendovi di proporre a breve l'abbinamento a Raspberry Pi; ebbene, è giunto il momento di vedere anche l'interfacciamento con questa scheda e analizzare l'implementazione della libreria necessaria a gestire l'integrato MGC3130. La libreria è



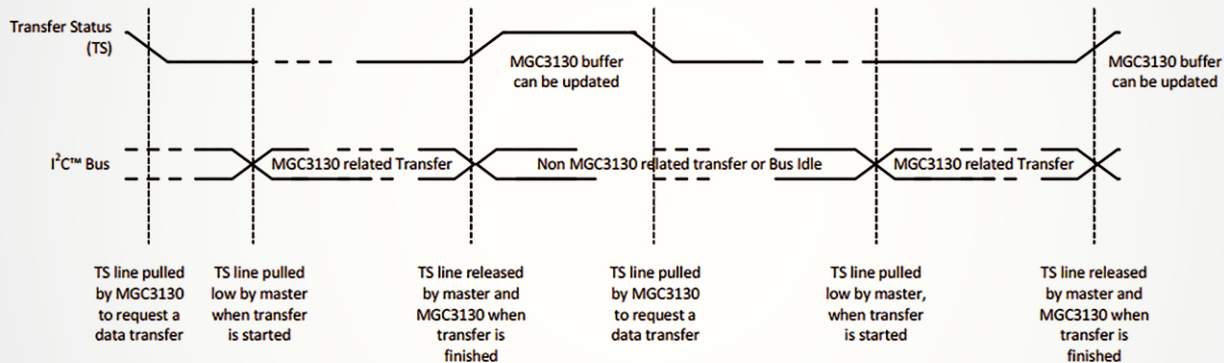


Fig. 1

realizzata in Python, ovvero un linguaggio di programmazione dinamico orientato agli oggetti utilizzabile per lo sviluppo software di qualsiasi tipologia. Inoltre permette lo sviluppo di codice di qualità e di facile manutenibilità, offre un forte supporto all'integrazione con altri linguaggi e programmi, è fornito di un'estesa libreria standard e può essere appreso in pochi giorni nelle sue funzioni di base.

BREVE PANORAMICA SULL'HARDWARE

Abbiamo già affrontato nella precedente puntata la descrizione dello schema elettrico del nostro nuovo elettrodo, quindi non ci ripetiamo (rinvandovi per gli approfondimenti del caso al fascicolo di giugno) e ci soffermeremo solo su alcuni brevi punti senza andare nei dettagli.

Come si può osservare dallo schema elettrico, della scheda Raspberry Pi vengono usati pochi pin tra cui i segnali SDA e SCL collegati direttamente all'integrato MGC3130, il quale lavora con una logica che ha lo stesso livello di tensione di quello della Raspberry Pi e quindi non necessita di adattatori di livello, diversamente da quanto invece è richiesto da Arduino. Stesso discorso per le linee di I/O, che vengono portate direttamente all'integrato e in particolare le ormai note linee TS e RESET. Brevemente ricordiamo che la linea TS serve per indicare alla Raspberry Pi che sono pronti dei dati nel buffer di MGC3130 come conseguenza del riconoscimento di una gesture. Non appena Raspberry Pi si "accorge" che la linea TS è andata a livello logico basso, provvede ad occupare tale linea configurandola come uscita e portando anch'esso la linea a livello logico basso; a questo punto può iniziare la lettura dei dati presenti nel buffer dell'MGC3130 sfruttando l'interfaccia di comunicazione I²C. Conclusa la lettura dei dati, la linea viene riportata a livello logico alto e riconfigurata come ingresso in attesa di un prossimo evento. La Fig. 1 mette in evidenza l'andamento della linea TS rispetto al bus I²C.

La linea RESET, come suggerisce il nome, serve per resettare l'integrato MGC3130 in caso di necessità. Solitamente si tende a mantenere la linea di RESET a livello logico basso durante il power-up dell'integrato e a rilasciare tale linea solo ad alimentazione stabilizzata (linea RESET riportata a livello logico alto). Infine alla scheda Raspberry Pi sono stati collegati tre pulsanti P1, P2 e P3 più un jumper J1. Solitamente per questa tipologia di componenti si prevede sempre una resistenza di pull-up da collegare sull'ingresso del microcontrollore che ne legge lo stato. Nel nostro caso si è preferito sfruttare la resistenza di pull-up interna messa a disposizione dal SOC (System On chip) presente su Raspberry Pi (Broadcom BCM2835 per Raspberry B+ e Broadcom BCM2836 per Raspberry 2). Il SOC permette di configurare gli ingressi sia con una resistenza di pull-up che di pull-down oppure nessuna delle due (non consigliato). Nella nostra applicazione, diversamente da quanto fatto per Arduino, abbiamo sfruttato solo i tre pulsanti P1, P2 e P3, tralasciando il jumper J1.

```
pi@RaspTest ~ $ find /usr | grep -i gpio
/usr/lib/pyshared/python2.7/RPi/GPIO.so
/usr/lib/pyshared/python2.6/RPi/GPIO.so
/usr/lib/python3/dist-packages/RPi/GPIO.cpython-32mu.so
/usr/lib/python3/dist-packages/RPi.GPIO-0.5.11.egg-info
/usr/lib/python2.7/dist-packages/RPi.GPIO-0.5.11.egg-info
/usr/lib/python2.7/dist-packages/RPi.GPIO-0.5.11.egg-info
/usr/lib/python2.6/dist-packages/RPi.GPIO-0.5.11.egg-info
/usr/lib/python2.6/dist-packages/RPi.GPIO-0.5.11.egg-info
/usr/share/pyshared/RPi.GPIO-0.5.11.egg-info
/usr/share/doc/python3-rpi.gpio
/usr/share/doc/python3-rpi.gpio/changelog.Debian.gz
/usr/share/doc/python3-rpi.gpio/changelog.gz
/usr/share/doc/python3-rpi.gpio/copyright
/usr/share/doc/python-rpi.gpio
/usr/share/doc/python-rpi.gpio/changelog.Debian.gz
/usr/share/doc/python-rpi.gpio/changelog.gz
/usr/share/doc/python-rpi.gpio/copyright
pi@RaspTest ~ $
```

Fig. 2

Il led LD8, anch'esso collegato a un pin di I/O, serve a indicare l'avvenuto riconoscimento di una gesture; la gestione del LED LD8 non dipende dalla libreria ma dal codice della demo.

Ricordiamo infine che con il nuovo elettrodo si è deciso di sfruttare gli I/O estesi EIO2, EIO3, EIO6 e EIO7, messi a disposizione da MGC3130, ai quali abbiamo collegato dei LED per segnalare alcune delle gesture riconosciute. Durante la parametrizzazione è quindi possibile decidere quale gesture deve essere monitorata e riportata su una delle possibili uscite o una combinazione di esse. Di seguito l'elenco:

- *Flick West to East;*
- *Flick East to West;*
- *Flick North to South;*
- *Flick South to North;*
- *Single Tap North;*
- *Single Tap South;*
- *Single Tap West;*
- *Single Tap East;*
- *Single Tap Centre;*
- *Clock Wise;*
- *Counter Clock wise.*

CONFIGURARE RASPBERRY PI

Prima di addentrarci nella descrizione della libreria Python è necessario eseguire dei passi intermedi di configurazione di Raspberry Pi. Vediamo passo passo come procedere: per prima cosa è necessario installare la libreria Python per la gestione degli I/O di Raspberry Pi e, se già installata, provvedere ad aggiornarla all'ultima release disponibile che attualmente si attesta alla 0.5.11.

Per verificare l'attuale versione della libreria GPIO eseguire il seguente comando:

```
find /usr | grep -i gpio
```

La Fig. 2 evidenzia il feedback del comando inviato; in questo caso la libreria risulta aggiornata all'ultima release. Se la release della libreria dovesse essere inferiore alla 0.5.11 provvedere ad un suo aggiornamento usando i seguenti comandi:

```
sudo apt-get update
sudo apt-get upgrade
```

oppure utilizzare i seguenti:

```
sudo apt-get install python-rpi.gpio
sudo apt-get install python3-rpi.gpio
```

Passiamo ora alla configurazione della periferica I²C messa a disposizione da Raspberry Pi; procediamo

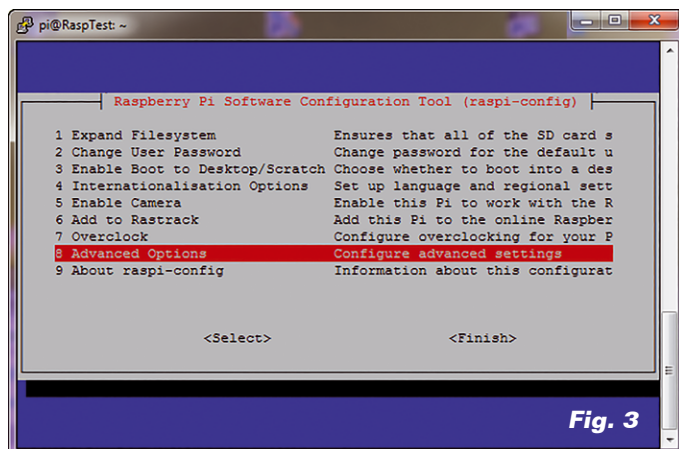


Fig. 3

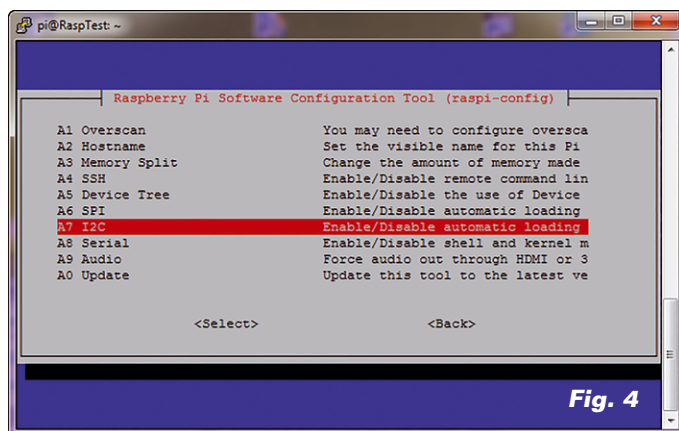


Fig. 4

con l'installazione della libreria Python "smbus" per la gestione del bus I²C:

```
sudo apt-get install python-smbus
```

seguita dai tools I²C:

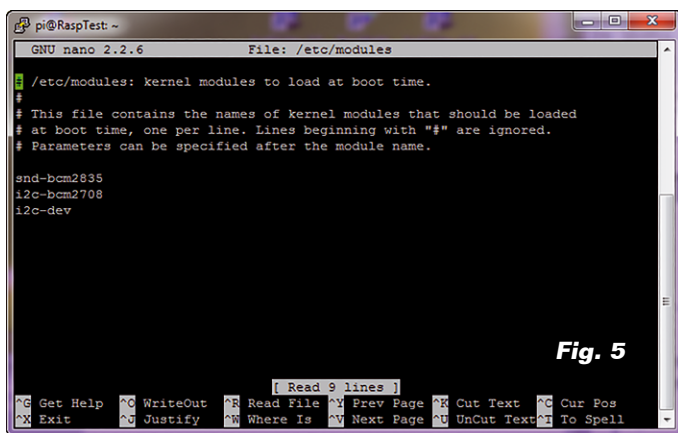
```
sudo apt-get install i2c-tools
```

Dobbiamo poi abilitare il supporto del bus I²C da parte del kernel; per fare ciò è necessario richiamare la finestra di configurazione, digitando il seguente comando:

```
sudo raspi-config
```

Come visibile in Fig. 3, selezioniamo la voce "Advanced Options", quindi:

- nella nuova schermata selezioniamo la voce "A7 I²C Enable/Disable automatic loading", come in Fig. 4;
- nelle seguenti schermate dobbiamo rispondere in sequenza "Yes", "Ok", "Yes" e infine ancora "Ok";
- usciamo dalla finestra di configurazione selezionando la voce "Finish".

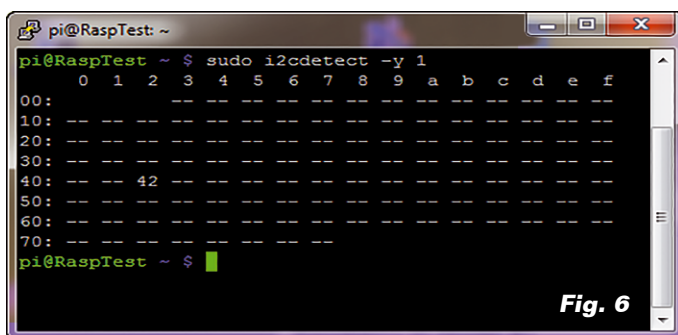


```
pi@RaspTest: ~
GNU nano 2.2.6 File: /etc/modules

/etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with '#' are ignored.
# Parameters can be specified after the module name.

snd-bcm2835
i2c-bcm2708
i2c-dev
```

Fig. 5



```
pi@RaspTest: ~
pi@RaspTest ~$ sudo i2cdetect -y 1
 0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- 42 -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
pi@RaspTest ~$
```

Fig. 6

Adesso eseguiamo il reboot del sistema eseguendo il comando:

```
sudo reboot
```

Dopo avere riavviato Raspberry Pi apriamo con un editor di testo generico il file “modules” mediante il comando:

```
sudo nano /etc/modules
```

A questo punto dobbiamo aggiungere, se mancanti, le seguenti due righe (Fig. 5):

```
i2c-bcm2708
i2c-dev
```

Per ultimo bisogna verificare se nella vostra distribuzione è presente il seguente file:

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

In caso affermativo apritelo con un editor di testo usando il comando:

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

e, se presenti, commentate (utilizzando il carattere “#”) le seguenti istruzioni:

```
blacklist spi-bcm2708
```

```
blacklist i2c-bcm2708
```

A questo punto non rimane che riavviare nuovamente la Raspberry Pi e verificare che la comunicazione I²C sia attiva; per fare ciò eseguite il comando seguente:

```
sudo i2cdetect -y 1
```

Se tutto è andato a buon fine, il comando inviato ritorna una tabella come quella mostrata in Fig. 6. In pratica il comando analizza il bus I²C in cerca di periferiche e nel nostro caso ha individuato l’integrato MGC3130 con indirizzo 0x42.

La configurazione dell’hardware I²C, e relativa libreria di gestione, è conclusa; ora si rende necessario scaricare e installare un modulo Python che simula la pressione dei tasti della tastiera, cosicché si possa associare una gesture a un tasto, ad esempio i tasti funzione F1, F2 ecc.

Il modulo da scaricare è reperibile al seguente link:

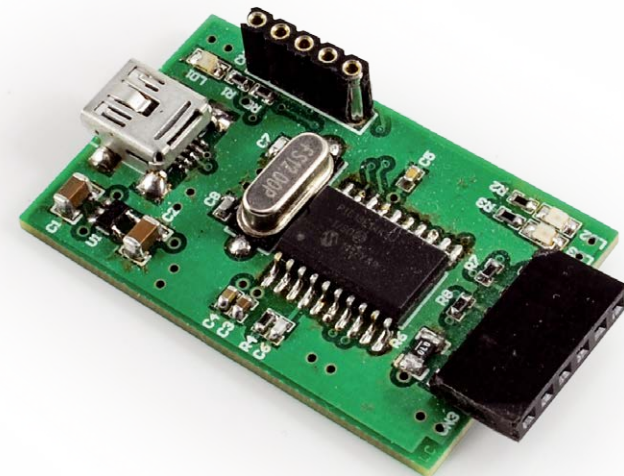
<https://github.com/SavinaRoja/PyUserInput>

Scaricate il file .zip e scompattatelo in una directory: ad esempio la “home” o, se preferite, in una directory da voi creata per l’occasione.

Tra i file scompattati ne troverete uno con nome “Setup.py” che vi servirà per installare il modulo nella vostra distribuzione e renderla disponibile nel path Python, cosicché la si possa includere nei propri progetti.

Digitate quindi il comando:

```
python setup.py install
```



Se tutte le dipendenze sono verificate, il modulo viene installato e reso subito disponibile per essere integrato nei vostri progetti. Tuttavia potrebbe succedere che non siano rispettate tutte le dipendenze e che si renda necessario installare il modulo aggiuntivo “pi3d” scaricabile dal seguente link:

<https://github.com/tipam/pi3d>

Come per il precedente scaricare, scompattare e infine installare il seguente comando:

```
python setup.py install
```

Infine potrebbe essere necessaria la libreria “xlib” per Python la quale può essere installata con il seguente comando:

```
sudo apt-get install python-xlib
```

Bene, arrivati a questo punto abbiamo tutto quello che ci serve per introdurre e studiare la libreria di gestione dell’integrato MGC3130 sviluppata in Python.

MGC3130 E LA LIBRERIA PER RASPBERRY PI

Parliamo ora della libreria scritta in Python per gestire l’integrato MGC3130, la quale si rifà sulla falsa riga di quella già presentata per Arduino. Cominciamo con una puntualizzazione sulla gestione delle strutture dati in Python necessarie a gestire il flusso dati proveniente dall’integrato MGC3130. In Python le strutture dati vengono costruite in modo leggermente differente rispetto al linguaggio di programmazione C, quindi di seguito riportiamo un confronto tra la struttura dati scritta in C per Arduino rispetto alla nuova libreria in Python per Raspberry Pi. Come vi ricorderete dalla precedente puntata, la struttura dati usata con Arduino è composta come mostrato nel **Listato 1**.

Ora mostriamo l’equivalente scritta in Python (**Listato 2**): notate che l’impostazione è differente ma la funzione svolta è la medesima.

Anche se sembrano diverse in realtà sono uguali e svolgono lo stesso compito. Nel nostro caso per accedere a questa struttura dati è necessario dichia-

Listato 1

```
union GestureOutput {
    uint32_t Gesture;
    uint8_t GestArray[4];
    struct {
        uint8_t Byte_0;
        uint8_t Byte_1;
        uint8_t Byte_2;
        uint8_t Byte_3;
    } GestureByte;
    struct {
        uint8_t TouchSouth      :1; // GESTURE_TOUCH_SOUTH      0x00000001
        uint8_t TouchWest       :1; // GESTURE_TOUCH_WEST       0x00000002
        uint8_t TouchNorth      :1; // GESTURE_TOUCH_NORTH      0x00000004
        uint8_t TouchEast       :1; // GESTURE_TOUCH_EAST       0x00000008
        uint8_t TouchCentre     :1; // GESTURE_TOUCH_CENTRE     0x00000010
        uint8_t TapSouth        :1; // GESTURE_TAP_SOUTH        0x00000020
        uint8_t TapWest         :1; // GESTURE_TAP_WEST         0x00000040
        uint8_t TapNorth        :1; // GESTURE_TAP_NORTH        0x00000080
        uint8_t TapEast         :1; // GESTURE_TAP_EAST         0x00000100
        uint8_t TapCentre       :1; // GESTURE_TAP_CENTRE       0x00000200
        uint8_t DoubleTapSouth   :1; // GESTURE_DOUBLE_TAP_SOUTH 0x00000400
        uint8_t DoubleTapWest   :1; // GESTURE_DOUBLE_TAP_WEST  0x00000800
        uint8_t DoubleTapNorth   :1; // GESTURE_DOUBLE_TAP_NORTH 0x00001000
        uint8_t DoubleTapEast   :1; // GESTURE_DOUBLE_TAP_EAST  0x00002000
        uint8_t DoubleTapCentre :1; // GESTURE_DOUBLE_TAP_CENTRE 0x00004000
        uint8_t GestWestEast     :1; // GESTURE_WEST_EAST        0x00008000
        uint8_t GestEastWest     :1; // GESTURE_EAST_WEST        0x00010000
        uint8_t GestSouthNorth   :1; // GESTURE_SOUTH_NORTH     0x00020000
        uint8_t GestNorthSouth   :1; // GESTURE_NORTH_SOUTH     0x00040000
        uint8_t EdgeGestWestEast :1; // GESTURE_EDGE_WEST_EAST   0x00080000
        uint8_t EdgeGestEastWest :1; // GESTURE_EDGE_EAST_WEST   0x00100000
        uint8_t EdgeGestSouthNorth :1; // GESTURE_EDGE_SOUTH_NORTH 0x00200000
        uint8_t EdgeGestNorthSouth :1; // GESTURE_EDGE_NORTH_SOUTH 0x00400000
        uint8_t GestClockWise    :1; // GESTURE_CLOCK_WISE       0x00800000
        uint8_t GestCounterClockWise :1; // GESTURE_COUNTER_CLOCK_WISE 0x01000000
        uint8_t FreeBit          :7;

    } Bit;
} GestureOutput;
```

Listato 2

```
class GestureBit (Structure):
    _fields_ = [("TouchSouth", c_uint32, 1),
                ("TouchWest", c_uint32, 1),
                ("TouchNorth", c_uint32, 1),
                ("TouchEast", c_uint32, 1),
                ("TouchCentre", c_uint32, 1),
                ("TapSouth", c_uint32, 1),
                ("TapWest", c_uint32, 1),
                ("TapNorth", c_uint32, 1),
                ("TapEast", c_uint32, 1),
                ("TapCentre", c_uint32, 1),
                ("DoubleTapSouth", c_uint32, 1),
                ("DoubleTapWest", c_uint32, 1),
                ("DoubleTapNorth", c_uint32, 1),
                ("DoubleTapEast", c_uint32, 1),
                ("DoubleTapCentre", c_uint32, 1),
                ("GestWestEast", c_uint32, 1),
                ("GestEastWest", c_uint32, 1),
                ("GestSouthNorth", c_uint32, 1),
                ("GestNorthSouth", c_uint32, 1),
                ("EdgeGestWestEast", c_uint32, 1),
                ("EdgeGestEastWest", c_uint32, 1),
                ("EdgeGestSouthNorth", c_uint32, 1),
                ("EdgeGestNorthSouth", c_uint32, 1),
                ("GestClockWise", c_uint32, 1),
                ("GestCounterClockWise", c_uint32, 1),
                ("Free", c_uint32, 7)]

class GestureByte (Structure):
    _fields_ = [("Byte0", c_uint8),
                ("Byte1", c_uint8),
                ("Byte2", c_uint8),
                ("Byte3", c_uint8)]

class Gesture (Union):
    _fields_ = [("Gesture32Bit", GestureBit),
                ("GestureByte", GestureByte),
                ("GestureLong", c_uint32),
                ("GestArray", c_ubyte * 4)]
```

rare una variabile nel file Python che richiama tale struttura, ad esempio:

```
_GestureOutput = Gesture()
```

Tale dichiarazione può essere fatta come tipo globale e quindi visibile in ogni funzione dichiarata nel file di libreria, a patto che in ogni funzione si dichiari la seguente:

```
global _GestureOutput
```

così facendo si indica all'interprete Python che la variabile è di tipo globale. Per accedere al valore, ad esempio, della gesture Tap su elettrodo Sud si deve scrivere la seguente riga di codice:

```
if (_GestureOutput.Gesture32Bit.TapSouth):
```

Se invece si vuole impostare il valore di tale bit si deve usare questa sintassi:

```
_GestureOutput.Gesture32Bit.TapSouth = 0
```

Oltre a livello di bit è possibile accedere ai dati a livello di singolo byte per un totale di quattro byte (32 bit totali di cui sette liberi per sviluppi futuri) oppure direttamente come double word agendo in un solo colpo su tutti i 32 bit della struttura. Quindi l'accesso diventerebbe:

```
_GestureOutput.GestureByte.Byte0 = 0x00
_GestureOutput.GestureLong = 0x00000000
```

Dopo questa premessa analizziamo le funzioni messe a disposizione dalla nostra libreria; in particolare, abbiamo le seguenti funzioni pubbliche:

```
def MGC3130_SetAdd(Addr)
def MGC3130_ResetDevice(GPIO, Rst):
def MGC3130_ExitResetDevice(GPIO, Rst):
def MGC3130_Begin(GPIO, Ts, Rst):
def MGC3130_GetTsLineStatus(GPIO, Ts):
def MGC3130_ReleaseTsLine(GPIO, Ts):
def MGC3130_GetEvent():
def MGC3130_DecodeGesture():
```

La funzione "MGC3130_SetAdd" serve per impostare l'indirizzo hardware occupato sul bus I²C da MGC3130, nel nostro caso è sempre e solo 0x42. L'unico parametro da passare a questa funzione è appunto l'indirizzo hardware.

La funzione "MGC3130_ResetDevice", come suggerisce il nome, serve per mantenere resettato l'integrato MGC3130 fintantoché non viene rimossa tale condizione. I parametri da passarle sono due: il primo è la libreria GPIO necessaria per gestire le linee di I/O generiche, mentre il secondo è il pin associato alla linea di RESET.

La funzione opposta è "MGC3130_ExitResetDevice" la quale rimuove la condizione di RESET; i parametri da passarle sono i medesimi della funzione precedente.

La funzione "MGC3130_Begin" serve per inizializzare le linee di comunicazione tra la Raspberry Pi e l'MGC3130. I parametri da passare alla funzione sono la libreria GPIO, il numero di pin per la linea TS e il numero di pin per la linea RESET.

Il pin TS viene impostato come ingresso con resistenza di pull-up interna abilitata, mentre la linea RESET

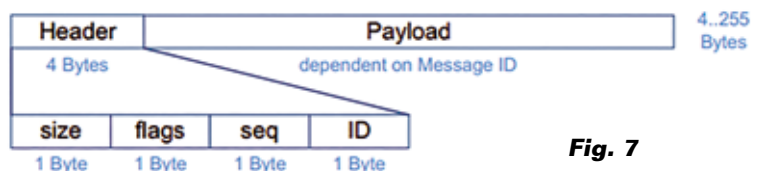


Fig. 7

Listato 3

```

GPIO.setmode(GPIO.BCM) # set board mode to Broadcom
GPIO.setup(JUMPER_J1, GPIO.IN, pull_up_down=GPIO.PUD_UP) # Jumper
GPIO.setup(PULSE_P1, GPIO.IN, pull_up_down=GPIO.PUD_UP) # Pulsante P1
GPIO.setup(PULSE_P2, GPIO.IN, pull_up_down=GPIO.PUD_UP) # Pulsante P2
GPIO.setup(PULSE_P3, GPIO.IN, pull_up_down=GPIO.PUD_UP) # Pulsante P3
GPIO.setup(LED_DIAGNOSTIC, GPIO.OUT, initial=GPIO.HIGH) # Led

GPIO.add_event_detect(PULSE_P1, GPIO.FALLING, bouncetime=100)
GPIO.add_event_detect(PULSE_P2, GPIO.FALLING, bouncetime=100)
GPIO.add_event_detect(PULSE_P3, GPIO.FALLING, bouncetime=100)

```

viene impostata come uscita con valore logico iniziale alto. L'inizializzazione prevede di mantenere in RESET forzato l'integrato per un periodo di 250mSec portandolo così alle condizioni iniziali come se fosse avvenuto un power-up.

La funzione "MGC3130_GetTsLineStatus" serve per monitorare la linea TS e intercettare quando l'integrato MGC3130 la impegna per avvisare che è stata riconosciuta una gesture. Non appena il sistema si accorge che la linea TS è stata impegnata, cambia lo stato del pin andando lui stesso ad occupare la linea TS, potendo così iniziare la procedura di lettura dei dati contenuti nel buffer dell'integrato MGC3130. Conclusa la lettura dei dati si può richiamare la funzione "MGC3130_ReleaseTsLine" rilasciando così la linea TS.

Per entrambe le funzioni appena esposte bisogna passare due parametri: il primo è la libreria GPIO, mentre il secondo è il pin cui è collegata la linea TS. La funzione "MGC3130_GetEvent" serve per leggere i dati memorizzati nel buffer dell'integrato MGC3130 e salvarli in apposite strutture dati private utilizzate dalla libreria. Sarà poi una successiva funzione a provvedere alla decodifica dei dati ricevuti e a renderli disponibili all'utente finale per la propria applicazione. Vi ricordiamo, come già ampiamente detto nelle precedenti puntate, che il pacchetto dati utilizzato per la lettura dei dati caricati nel buffer dell'integrato MGC3130 è del tipo raffigurato in Fig. 7. L'indirizzo contenuto nel pacchetto è quello hardware dell'integrato dove il bit meno significativo indica se si sta eseguendo una scrittura ("0") oppure una lettura ("1"). La sezione "MGC3130 message"

può essere espansa come indicato in Fig. 8 identificando due sezioni "Header" e "Payload". Espandendo ancora la sezione "Header", come in Fig. 9, si nota che nei primi quattro byte ritroviamo la lunghezza del pacchetto dati ricevuto e, cosa molto importante, l'ID che identifica il pacchetto dati. Nel caso di una lettura dati dopo il riconoscimento di una gesture l'ID del pacchetto sarà 0x91. Per l'elenco dei codici ID disponibili vi invitiamo a rileggere la precedente puntata.

La libreria gestisce anche l'ID 0x83 che identifica la revisione di libreria caricata nell'integrato. Questa informazione viene letta allo start-up dopo avere tolto la condizione di RESET. Attualmente c'è una limitazione nel numero di byte che si possono leggere dal bus I²C sfruttando la libreria "smbus", ovvero la lunghezza massima del buffer di ricezione della libreria è impostato a 32 byte il che è un attimino limitante. Questo comporta che il pacchetto dati con ID 0x83 letto dal buffer di MGC3130 è troncato. Per chi volesse cimentarsi è possibile scaricare il sorgente della libreria "smbus", cambiarne la lunghezza del buffer, ricompilare e re-installare la libreria nella propria distribuzione.

La funzione "MGC3130_GetEvent" applica dei filtri ai dati letti dal buffer di MGC3130 per eliminare le parti non necessarie ai nostri scopi. I dati fondamentali sono le informazioni inerenti le "GestureInfo" e le "TouchInfo" alle quali si applicano delle maschere di filtro per eliminare i bit indesiderati (la "MASK_GESTURE_RAW" viene applicata alle "GestureInfo" mentre la "MASK_TOUCH_RAW" viene applicata a "TouchInfo"). Questi valori, se non per validi motivi, non devono mai essere modificati dall'utente.

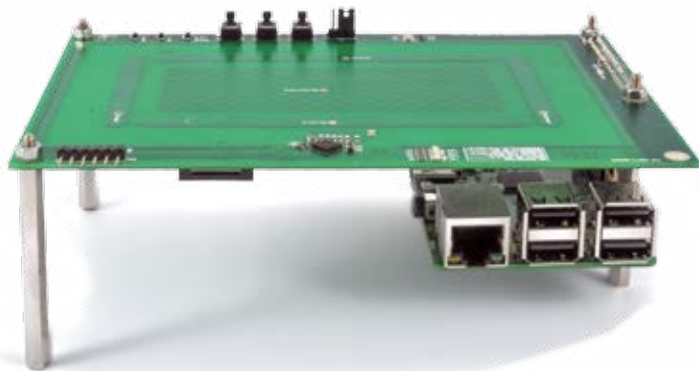
L'ultima funzione "MGC3130_DecodeGesture" decodifica i dati letti dalla precedente funzione e li ren-



Fig. 8



Fig. 9



de disponibili attraverso una struttura dati pubblica che l'utente può utilizzare nella propria applicazione (vedere la struttura dati descritta precedentemente). La libreria si accorge delle variazioni intercettate dall'integrato MGC3130 e le riporta sui rispettivi bit della struttura dati pubblica. La gesture è riconosciuta quando il bit corrispondente va a livello logico "1". Volendo, ma non è necessario, è possibile filtrare le gesture che non si vogliono utilizzare nel proprio progetto; infatti si può configurare una costante di filtro, presente nel file `MGC3130_DefVar.py`, che permette di filtrare le gesture non volute. La costante si chiama `"MASK_FILTER_GESTURE"` ed è un valore a 32 bit configurabile bit a bit. Se il bit è a "1" logico significa che si desidera filtrare la gesture viceversa la si vuole mantenere.

Per favorire lo studio della libreria ci sono delle parti di codice che possono essere attivate/disattivate attraverso delle variabili booleane appositamente dichiarate. Se la variabile è `"True"` attiva la parte di codice interessata, viceversa la disattiva. Le variabili sono le seguenti:

```
1) EnablePrintMGC3130RawFirmwareInfo = False
2) EnablePrintMGC3130RawData          = False
3) EnablePrintMGC3130Gesture          = False
4) EnablePrintMGC3130xyz               = False
```

Per impostazione predefinita, le variabili sono tutte settate al valore `"False"`; se attivate, vedrete a schermo la stampa dei dati letti dall'integrato ed eventualmente le relative decodifiche, in particolare in ordine abbiamo:

```
1) Stampa in formato RAW dei dati inerenti la revisione
   FW caricata nell'integrato
2) Stampa in formato RAW dei dati inerenti la gesture
   riconosciuta
3) Stampa della gesture riconosciuta dopo la decodifica
4) Stampa delle coordinate x, y e z
```

Ad esempio, se la variabile booleana `"EnablePrintMGC3130RawData"` è settata a `True` l'intercettazione della gesture `"Flick East to West"` genera

una tabella dati così formattata:

```
#####
Row data from MGC3130
Header: 1A081191
Payload: 1F01 | 86 | 80 | 0073 | 03100000 | 00000000
| 0000 | 00000000000000
#####
```

Per aumentarne leggibilità e interpretazione, il pacchetto dati viene diviso tra `"Header"` e `"Payload"`; in più il `"Payload"` viene suddiviso in sotto-pacchetti (vedere il datasheet della libreria GestIC).

MGC3130 DEMO

Descriviamo ora il nostro programma demo, sempre scritto in Python, che ci dà la possibilità di gestire un programma di visualizzazione delle immagini direttamente con le gesture riconosciute da MGC3130. Il programma si chiama `"Eye Of Gnome"` e può essere installato sulla propria Raspberry Pi semplicemente eseguendo il seguente comando:

```
sudo apt-get install eog
```

Analizziamo ora la demo realizzata per Raspberry Pi, il file si chiama `"DemoGestic.py"` e come per tutti i file scritti in Python in testa trova le dichiarazioni di importazione dei moduli necessari al suo corretto funzionamento, tra questi ritroviamo la nostra libreria per la gestione dell'integrato MGC3130 (`import MGC3130`), la libreria per la gestione degli I/O (`import Rpi.GPIO`), una libreria per la gestione dei Thread (`import threading`), una libreria per la gestione dei tempi (`import time`) e infine la libreria di emulazione dei pulsanti della tastiera (`from pykeyboard import PyKeyboard`).

Per semplicità di gestione del codice si dichiarano delle variabili a valore costante che identificano i pin di I/O della Raspberry Pi; in particolare, abbiamo le linee TS e RESET nonché le linee SDA e SCL, i pulsanti, il jumper e il LED di diagnostica. Tutte queste definizioni vengono utili nel momento in cui si vanno a configurare i pin di I/O da utilizzare con Raspberry Pi e tutte le volte che se ne vuole testare il valore o modificarne lo stato.

Come si può osservare nel **Listato 3**, con la funzione `"setup"` si definisce se il pin di interesse è un ingresso o una uscita e, nel caso dei pin di ingresso, si attivano anche le resistenze di pull-up interne. Inoltre per i pulsanti di ingresso P1, P2 e P3 si definisce un ulteriore parametro che introduce la gestione dell'evento pressione pulsante `"add_event_detect"` il

quale interviene sui fronti di discesa con un tempo di debouncing di 100mSec. Nel ciclo infinito del main ritroviamo quindi la gestione dei tre pulsanti che sono in attesa che si scateni l'evento sopra configurato, ad esempio:

```
if GPIO.event_detected(PULSE_P1):
```

Questa istruzione condizionale è in attesa che venga intercettato l'evento di pressione del pulsante P1 per eseguire la propria funzionalità. In particolare abbiamo definito che il pulsante P1 attiva la demo e quindi permette l'associazione di una gesture con un tasto della tastiera, il pulsante P2 disabilita quanto appena detto, mentre il pulsante P3 chiude la demo liberando le porte di I/O dalle configurazioni fatte. Prima di entrare in un ciclo "while" infinito il codice esegue una serie di configurazioni tra cui l'indirizzo hardware che occupa l'integrato MGC3130 sul bus I²C, nonché l'inizializzazione delle linee di I/O necessarie a gestire lo stesso. In questo secondo caso, alla funzione di configurazione viene passata come parametro anche la libreria GPIO; infatti questa libreria è bene che sia dichiarata in un solo punto e poi eventualmente passata alle funzioni che ne necessitano. La sezione di codice che si occupa di rilevare la variazione sulla linea TS e quindi provvedere alla lettura del buffer dell'integrato MGC3130 è la seguente:

```
if (MGC3130.MGC3130_GetTsLineStatus(GPIO, MGC3130_TS_LINE) == True):  
    MGC3130.MGC3130_GetEvent()  
    MGC3130.MGC3130_DecodeGesture()  
    MGC3130.MGC3130_ReleaseTsLine(GPIO, MGC3130_TS_LINE)
```

Utilizzando la funzione di libreria:

```
def MGC3130_GetTsLineStatus(GPIO, Ts):
```

si testa la linea TS in attesa che vada a livello logico basso per poi impegnarla e iniziare la procedura di lettura dei dati tramite la funzione di libreria:

```
def MGC3130_GetEvent():
```

Seguono le funzioni di decodifica dei dati letti per estrapolare le gesture riconosciute e la funzione per il rilascio della linea TS:

```
def MGC3130_DecodeGesture():  
def MGC3130_ReleaseTsLine(GPIO, Ts):
```

Descriviamo ora l'interazione tra le gesture riconosciute e il visualizzatore di immagini "Eye Of Gnome". Una volta avviato il software e selezionato

il direttorio con le immagini da guardare è possibile eseguire le seguenti operazioni:

- 1) sfogliare le immagini verso destra, gesture Flick West to East; simula il tasto "freccia destra";
- 2) sfogliare le immagini verso sinistra, gesture Flick East to West; simula il tasto "freccia sinistra";
- 3) ingrandire l'immagine (Zoom in), gesture Touch North Electrode; simula il tasto "+";
- 4) rimpicciolire l'immagine (Zoom out), gesture Touch South Electrode; simula il tasto "-";
- 5) visualizzare a schermo intero l'immagine, gesture Touch Centre; simula il tasto "F5";
- 6) ruotare l'immagine in senso orario, gesture Clockwise; simula la combinazione "CTRL + R";
- 7) ruotare l'immagine in senso antiorario, gesture Counter Clockwise; simula tasto "SHIFT + CTRL + R";
- 8) quando l'immagine è ingrandita è possibile spostare la stessa sullo schermo usando le gesture Flick West to East, Flick East to West, Flick North to South e Flick South to North.

Ma come fa il nostro codice a emulare i pulsanti della tastiera? Lo fa grazie alla libreria "PyUserInput". Una volta importata questa, con il comando:

```
import PyKeyboard
```

si deve creare l'oggetto:

```
KeyPressed = PyKeyboard()
```

L'oggetto creato può gestire i seguenti eventi:

- `press_key('h')`; l'esempio emula la pressione del tasto "h";
- `release_key('h')`; l'esempio emula il rilascio del tasto "h". Ad un "press_key" segue sempre obbligatoriamente un "release_key" (a meno che la situazione non richieda di mantenere premuto un tasto per un determinato lasso di tempo);
- `tap_key('h')`; l'esempio emula il tocco di un tasto della tastiera;
- `type_string('hello')`; l'esempio emula l'invio di una stringa.

Grazie a questi oggetti si possono anche simulare le combinazioni di tasti; ad esempio, per simulare "shift" + "ctrl" + "M" bisogna scrivere il codice seguente:

```
KeyPressed.press_key(KeyPressed.shift_key)  
KeyPressed.press_key(KeyPressed.control_key)  
KeyPressed.tap_key('M')  
KeyPressed.release_key(KeyPressed.control_key)  
KeyPressed.release_key(KeyPressed.shift_key)
```


Dopo questa premessa possiamo descrivere come abbiamo usato questa libreria per i nostri scopi. Ad esempio, il riconoscimento della gesture "West to East" viene associato al tasto freccia verso destra:

```
if (MGC3130._GestOutput.Gesture32Bit.GestWestEast == 1):
    MGC3130._GestOutput.Gesture32Bit.GestWestEast = 0
    GPIO.output(LED_DIAGNOSTIC, GPIO.LOW)
    BaseTimeTimer = threading.Timer(1, SlowBaseTime)
    BaseTimeTimer.start()
    KeyPressed.tap_key(KeyPressed.right_key)
```

Simulando il tasto freccia destro le immagini vengono scrollate verso destra. L'equivalente per lo scrolling delle immagini verso sinistra sarà:

```
if (MGC3130._GestOutput.Gesture32Bit.GestEastWest == 1):
    MGC3130._GestOutput.Gesture32Bit.GestEastWest = 0
    GPIO.output(LED_DIAGNOSTIC, GPIO.LOW)
    BaseTimeTimer = threading.Timer(1, SlowBaseTime)
    BaseTimeTimer.start()
    KeyPressed.tap_key(KeyPressed.left_key)
```

Quando una gesture viene riconosciuta viene fatto lampeggiare per un secondo il led LD8. Si è quindi creato un Thread che viene attivato e richiamato dopo un periodo di un secondo dopo ogni riconoscimento di una gesture. Più in dettaglio quando la gesture viene riconosciuta si accende il led LD8 sfruttando la linea di codice:

```
GPIO.output(LED_DIAGNOSTIC, GPIO.LOW)
```

Subito dopo viene attivato il Thread che si occuperà di spegnere il led LD8 dopo un secondo. Più precisamente l'attivazione del Thread viene fatta eseguendo il codice:

```
BaseTimeTimer = threading.Timer(1, SlowBaseTime)
BaseTimeTimer.start()
```

La funzione richiamata dal Thread è la seguente:

```
def SlowBaseTime():
    global BaseTimeTimer

    GPIO.output(LED_DIAGNOSTIC, GPIO.HIGH)
    BaseTimeTimer.cancel()
```

Questa funzione spegne il led LD8 e cancella il Thread richiamato cosicché sia possibile richiamarlo al prossimo evento.

Con questo abbiamo terminato la descrizione della nostra demo di gestione dell'elettrodo per MGC3130 abbinato a una scheda Raspberry Pi B+ o 2.0. Ricordatevi che per attivare l'interazione tra le gesture e la libreria che emula i tasti della tastiera dovete premere il pulsante P1, viceversa per disattivarla premere P2.

DUE PAROLE SU "EOG" E LA NOSTRA DEMO

Per avviare il visualizzatore di immagini "EOG" precedentemente installato bisogna, prima di tutto, avviare il server "X" con il comando "startx".

Dopodiché cliccare sulla voce "Menu → Graphics → Image Viewer" per aprire il programma "EOG". Selezionare la cartella contenente le foto su cui si vuole lavorare cliccando sulla voce di menu "Image → Open". Conclusa questa fase possiamo avviare la nostra demo Python utilizzando "LXTerminal" e al prompt digitare "python DemoGestic.py" per avviare la demo.

Ancora una volta vi ricordiamo che per attivare l'interazione tra la demo e il visualizzatore di immagini dovete premere sul pulsante P1, per disattivare l'interazione premere su P2 e per uscire dalla demo e liberare le porte di I/O premere su P3.

CONCLUSIONI

In questo quarto ed ultimo articolo dedicato alla tecnologia GestIC della Microchip abbiamo imparato ad utilizzare il nuovo elettrodo già descritto -sul piano hardware- nel fascicolo scorso (giugno 2015) con la scheda Raspberry Pi B+/2.0, abbiamo studiato la libreria Python di gestione dell'integrato MGC3130 e mostrato un esempio pratico di utilizzo dell'elettrodo.

Ora dopo questa rassegna di progetti sul riconoscimento delle gesture, possiamo dire di avervi dato le basi per poter sviluppare i vostri progetti integrando così il riconoscimento gestuale in una moltitudine di diverse applicazioni, come la visualizzazione di pagine di menu, di immagini, ma anche l'apertura e la chiusura di tapparelle e tende motorizzate, l'accensione e lo spegnimento di luci e via di seguito.



per il MATERIALE

Il software Aurea di Microchip può essere scaricato dal sito www.microchip.com/gestic.

Il sensore di prossimità (cod. MGC3130) è reperibile presso Futura Elettronica a Euro 9,80, così come la scheda Gesture (cod. FT1214M) che viene fornita assemblata al prezzo di Euro 58,00. Tutti i prezzi sono IVA compresa.

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)

Tel: 0331-799775 • Fax: 0331-792287

<http://www.futurashop.it>



Windows 10



Un tempo antagonista e icona del software chiuso, la casa di Redmond si è avveduta dell'inarrestabile avanzata di Arduino e Raspberry Pi, rendendo compatibile con esse il proprio sistema operativo.

MICROSOFT & OPEN SOURCE: LA STRANA COPPIA

di FABRIZIO GUGLIELMINO

Fino a qualche tempo fa accostare Microsoft all'open source era impensabile, perché Microsoft ha sempre rappresentato l'approccio commerciale al software e spesso ha intrapreso battaglie legali (a dire il vero mai troppo efficaci) contro il movimento open source. Con la fine dell'era Ballmer (ex amministratore delegato di Microsoft) ha però iniziato a "cambiare pelle", segno evidente che il modello di business che l'ha fatta tanto crescere, basato sulla posizione monopolistica di Windows e Office, inizia a perdere colpi. L'attuale CEO, Satya Nadella, sembra aver ben chiara questa situazione e, seppur da poco alla guida della compagnia, ha intrapreso evidenti cambi di direzione. La strategia di base sembra orientata all'apertura per migrare da un'azienda focalizzata sui prodotti verso una rivolta ai servizi. Azure, la piattaforma cloud targata MS, dimostra chiaramente questa apertura supportando ufficialmente Linux e tecnologie non particolarmente comuni nell'ambito Microsoft, come Python, Node.js e molti altri.

La nuova strategia sta gradualmente sconfinando anche verso il mondo dei Maker, che Microsoft sembra voler corteggiare con particolare attenzione, come spiegheremo in questo articolo valutando le opportunità che si prospettano.

WINDOWS 10: LA PIATTAFORMA UNICA

La nuova strategia non abbandona il sistema operativo, che da sempre identifica l'azienda, ma gli cambia pelle. Windows 10 non è più un semplice sistema operativo per PC ma una piattaforma unica per vari dispositivi.

La release ufficiale è attesa per questi mesi con un grosso carico di novità: l'unificazione renderà Windows 10 accessibile dal PC allo smartphone passando per tablet e console, fino a giungere ai sistemi embedded. Proprio quest'ultimi sono quelli che più ci interessano, ed in particolare Arduino e Raspberry Pi.

L'offerta di Microsoft non si limita in realtà ai soli dispositivi citati, perché già da qualche tempo Microsoft ha manifestato il proprio interesse per

l'IoT, sia supportando diverse piattaforme embedded, sia iniziando ad offrire servizi specifici nella propria piattaforma di Cloud: Azure. Entrando un po' nel dettaglio, Windows 10 sarà il primo sistema operativo certificato per Arduino.

Ma all'atto pratico cosa significa? Semplificando, Windows 10 introduce quella che viene definita la Universal Windows Platform (UWP), ovvero un set di API (Application Program Interface) che poggiano su Windows Core per rendere una serie di funzionalità del sistema operativo accessibili da diversi contesti. Riprendendo uno degli esempi presentati dall'azienda, se un Maker realizza una videocamera di sicurezza con motori per il brandeggio, controllata da Arduino, può usufruire delle funzionalità della UWP, per esempio per mandare le immagini sul cloud, implementare il riconoscimento dei volti o realizzare un'interfaccia di controllo. Tutto questo, avvalendosi di funzionalità nativamente presenti nel sistema.

LE NOVITÀ PER ARDUINO

I progetti che si basano su Arduino sono innumerevoli e buona parte di essi permette di interagire con smartphone e computer. Analizzandoli, però, salta subito all'occhio che vengono utilizzati tanti metodi di interfacciamento quanti sono i progetti stessi. In alcuni casi vengono scritti protocolli proprietari, così una funzionalità banale, come eccitare un relé collegato ad Arduino da uno smartphone, richiede l'implementazione del protocollo anche per il telefono.

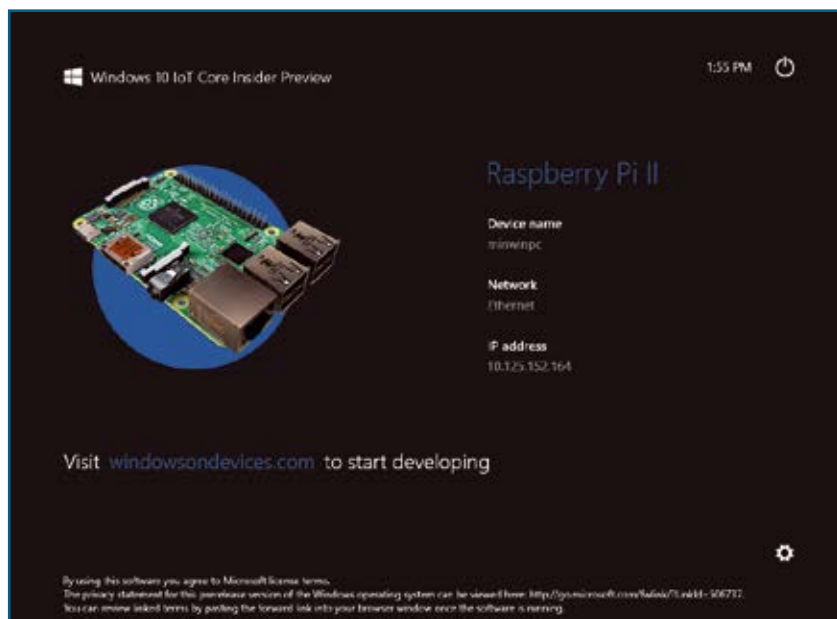
In altri casi vengono usate librerie più o meno standard che implementano meccanismi di comunicazione tra Arduino e il computer; questa condizione non agevola la condivisione dei progetti tra i membri della community. Integrare progetti diversi tra loro implica un costo, spesso non trascurabile, per capire e adattare i meccanismi di comunicazione implementati.

Microsoft ha compreso perfettamente questo scenario ed ha intuito che offrendo una soluzione standard coprirebbe quest'area

grigia con un ritorno di immagine e diffusione non trascurabile. Ma all'atto pratico, in cosa consiste la proposta Microsoft? Al momento in due progetti chiamati **Windows Remote Arduino** e **Windows Virtual Shield**. Vediamo di cosa si tratta, iniziando proprio da Windows Remote Arduino (WRA): si tratta di un'implementazione del protocollo "Firmata" per Windows; tale protocollo permette di comunicare direttamente con il processore della scheda da uno dei device su cui gira Windows 10.

In pratica risulta estremamente semplice creare applicazioni che girano, per esempio, su uno smartphone e comunicano con Arduino. Per vedere quanto può essere semplice effettuare questo tipo di operazioni, nel **Listato 1** proponiamo un pezzetto di codice (uno snippet, come si suol dire). Il codice è scritto in C# e viene eseguito sul dispositivo Windows 10 (per esempio, il cellulare). Non è un esempio completo, ma analizzandolo ci accorgiamo subito di una certa familiarità. Dopo le prime righe, che inizializzano gli oggetti necessari alla comunicazione, saltano all'occhio le similitudini con il codice che tipicamente si scrive nell'IDE di Arduino. Il setup dei due pin in output e input rispettivamente, il set allo stato alto del primo e la lettura di un valore analogico risultano comprensibili anche a chi non ha mai scritto codice per Windows, ma l'ha fatto su Arduino. La potenza di WRA è evidente, con meno di 10 righe di codice si è in grado di interagire con gli input/output della scheda in modo semplice ed immediato.

Non è da meno Windows Virtual Shield (WVS), anzi, se possibile, è ancora più interessante. In questo caso il tipo di interazione



Listato 1

```
using namespace Microsoft.Maker.RemoteWiring;
using namespace Microsoft.Maker.Serial;

// Connessione bluetooth e dell'interfaccia di gestione di Arduino
IStream bt = new BluetoothSerial();
RemoteDevice arduino = new RemoteDevice( bt );

// Avvio della connessione
bt.begin(115200, 0);

// Setup di due pin
arduino.pinMode(7, PinMode.OUTPUT);
arduino.pinMode(9, PinMode.INPUT);
// Set del pin 7
arduino.digitalWrite(7, PinState.HIGH);
// Lettura di un valore analogico dal pin 9
UInt16 val = arduino.analogRead(9);
```

è inverso rispetto a quello visto con WRA, in quanto il sistema trasforma il dispositivo Windows 10 in uno shield virtuale utilizzato da Arduino.

Uno dei punti di forza di Arduino sono proprio gli shield, che offrono la possibilità di estendere le funzionalità della scheda con dei componenti hardware standard; per questo attorno ad Arduino è sorto un ecosistema di accessori che copre praticamente ogni possibilità: esiste lo shield GPS per progetti che necessitano di localizzazione, lo shield WiFi per avere la connettività wireless, lo shield con il display per avere un dispositivo di output grafico o testuale ecc.

Con WVS il nostro device Windows 10 viene visto da Arduino come uno shield che rende disponibili tutti i sensori di cui dispone. A livello pratico la cosa è gestita da alcune librerie da usare negli sketch che eseguiamo su Arduino e da un'App da installare in Windows 10. Con queste due componenti, collegando la scheda a Windows via Bluetooth o direttamente via cavo, Arduino sarà in grado di accedere ai sensori, usare il telefono cellulare come gateway verso Internet o mostrare informazioni sul display.

Come per la gran parte degli shield ad oggi disponibili, l'utilizzo di WVS è mediato da una serie di funzionalità rese accessibili dalle librerie fornite da Microsoft; queste offrono le funzioni specifiche per interagire con i sensori del dispositivo Windows 10, quindi, per esempio, è disponibile una classe Text che permette di visualizzare del testo, per esempio sul telefono. Così come la classe Camera per interagire con la fotocamera, Speech per usufruire del sistema di sintesi vocale, o ancora Sms

per inviare messaggi su rete mobile e così via, praticamente per ogni componente che l'hardware su cui è installato Windows può offrire.

UTILIZZI PRATICI

Abbiamo visto, con un rapido excursus, alcune delle componenti di Windows 10 che lo rendono il primo sistema operativo certificato per Arduino. Con esso possiamo fare un'infinità di cose: con WRA per esempio possiamo creare l'interfaccia di controllo di un sistema domotico direttamente sul cellulare Windows Phone. Con WVS invece potremmo creare un apricancello collocato in auto, che leggendo le coordinate GPS dal cellulare invia il segnale di apertura solo quando siamo in sua prossimità.

Questi due semplici esempi dimostrano il potenziale di questi oggetti; va detto che le funzionalità di WRA e WVS non sono novità assolute; già oggi esistono tool e librerie che permettono di fare le stesse cose anche su sistemi operativi non Windows. La vera novità apportata da Windows 10 è la possibilità di rendere queste funzioni parti integranti del sistema, così da creare una sorta di ecosistema sul quale è altamente probabile che nascerà una miriade di applicazioni pratiche.

CONCLUSIONI

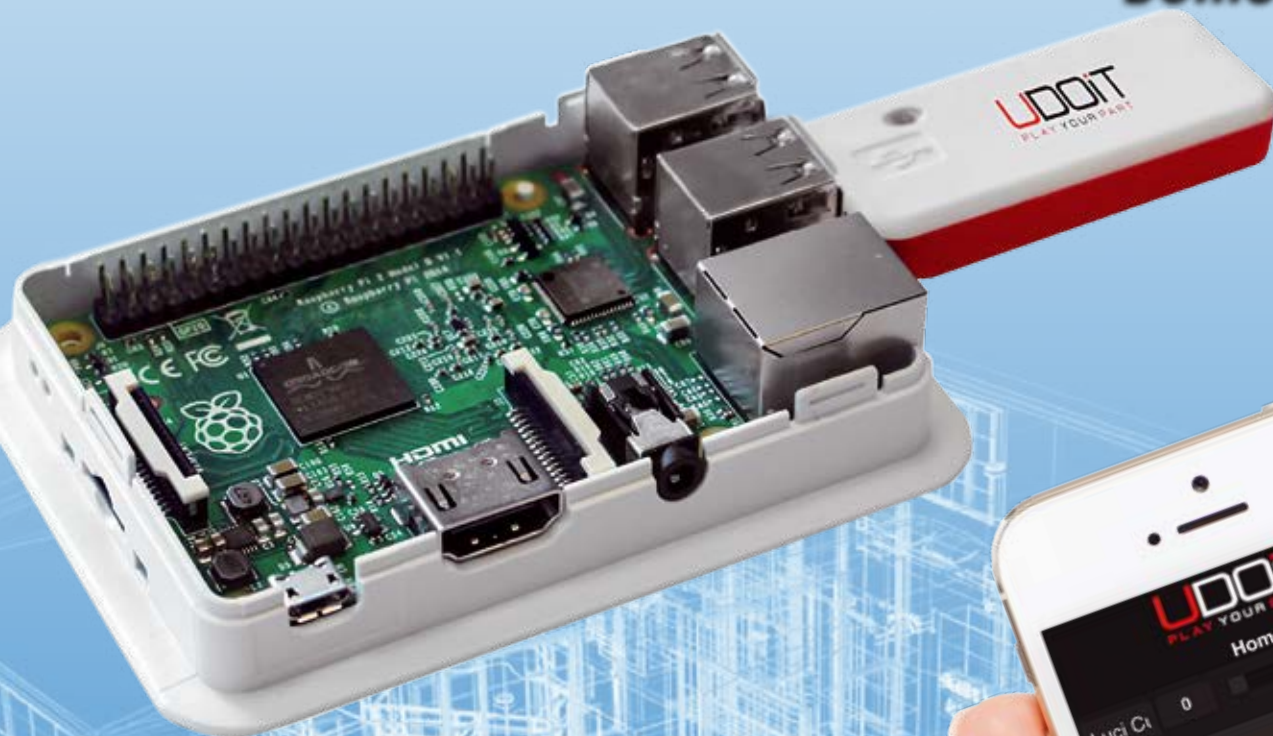
Abbiamo citato ad inizio articolo anche Raspberry Pi, ma la questione meriterebbe intere pagine. Per darvi un'anteprima diciamo che, a differenza di quanto visto per Arduino, è stata realizzata una versione di Windows 10 da installare su Raspberry Pi (solo sulla versione 2). Si tratta di una versione "ridotta" di Windows 10, definita Windows 10 IoT Core Insider, priva dell'interfaccia grafica ma con le funzionalità basilari (il core, appunto); va vista come un'alternativa alla Raspbian, che è ormai la distribuzione Linux di riferimento per Raspberry. Inoltre va detto che Windows 10 IoT Core per Raspberry Pi è gratuito il che rende la cosa ancora più interessante. Quello che abbiamo visto rappresenta solo la punta dell'iceberg, perché il grosso delle novità deve ancora palesarsi e i prossimi mesi saranno decisivi. Il fatto che un'azienda come Microsoft punti i riflettori su realtà come Arduino e Raspberry Pi dimostra come il movimento dei Maker stia facendosi sempre più strada in quello che viene definito il mondo "Enterprise". Non passerà molto tempo prima che le competenze su questi sistemi inizieranno ad essere richieste in ambito professionale, dando una spinta ancor maggiore all'innovazione. ■

Mensile di elettronica applicata, attualità scientifica, novità tecnologiche.

Elettronica In

www.elettronica.in.it

oltre l'elettronica



DOMOTICA OPEN SOURCE con i moduli UDOiT

Cominciamo ad entrare nel dettaglio della soluzione di domotica Open Source di UDOiT basata sui moduli RFTide.

..... di MARCO MAGAGNIN

Dopo avere presentato le linee generali e i diversi componenti della soluzione UDOiT per le applicazioni di domotica o, come si usa dire oggi, di Internet delle Cose, cominciamo ad addentrarci negli aspetti tecnici dell'hardware e del software di supporto alla soluzione. Come abbiamo già visto e schematizzato in Fig. 1, il "sistema" complessivo è strutturato in "livelli". Il primo livello è dato dal supporto di comunicazione senza fili che si basa sui moduli RFTide a 868,3 MHz, e che permettono di

realizzare una rete PAN (Personal Area Network), di sensori ed attuatori, cioè una rete ottimizzata per la comunicazione tra dispositivi a breve distanza, dove per breve distanza si intende qualche decina di metri.

Il secondo livello è costituito dal gateway, ovvero dal dispositivo in grado di far interagire la rete di sensori ed attuatori con l'applicazione di gestione della rete stessa. Il gateway è costituito da un dispositivo hardware, nel nostro caso un microcomputer



Raspberry Pi 2 B e da un'applicazione di gestione, che "anima" la rete di sensori ed attuatori e che deve essere configurata per far sì che il sistema, nel suo complesso, si comporti nel modo che desideriamo. Il terzo livello è costituito dalla rete internet alla quale è delegata la "remotizzazione" della gestione del sistema. Infine l'ultimo livello è costituito dalle interfacce utente di gestione del sistema. Qui è necessaria una distinzione tra i dispositivi hardware, come i telecomandi, che appartengono alla rete PAN, ma necessitano del gateway e dell'applicazione, per poter interagire con gli altri dispositivi della rete e l'interfaccia grafica di gestione, che utilizza l'interfaccia web di un qualunque dispositivo che ne disponga, PC desktop, portatile, tablet o smartphone.

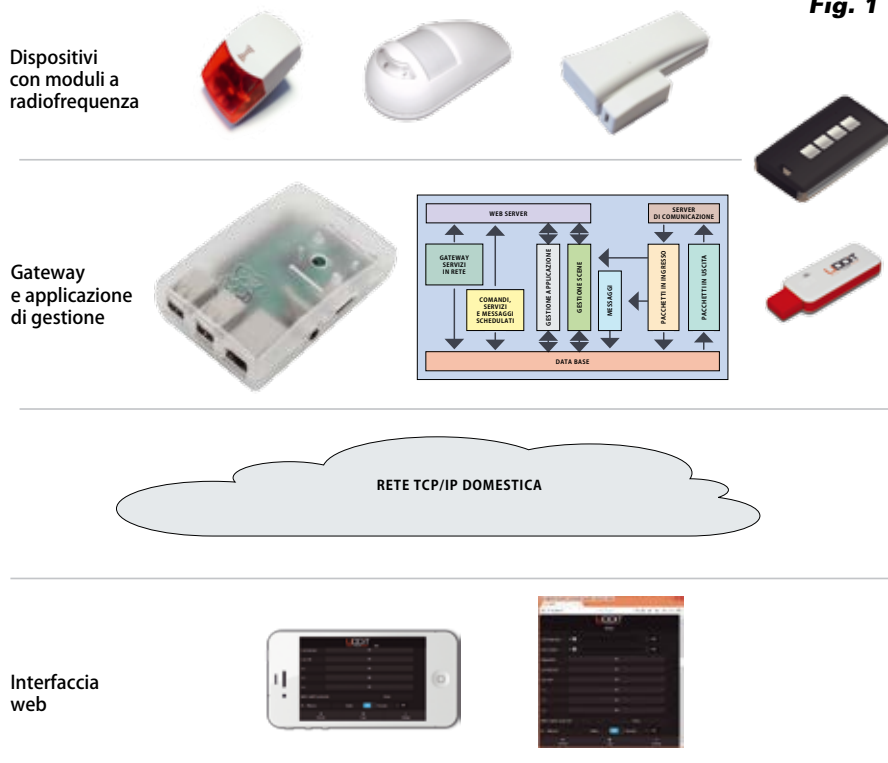
LA RETE RFTide

Il "cuore" della rete PAN RFTide

è costituito dal modulo TX/RX a radiofrequenza a 868,3 MHz visibile in Fig. 2, nato per lo sviluppo di reti PAN facili da installare e da mantenere, destinate principalmente alla domotica all'interno di abitazioni, ma anche alla telemetria e alla smart-energy. I moduli a radiofrequenza della rete RFTide possono essere configurati sia in modalità "Master" nel qual caso il modulo stesso diventa il "centro stella" della rete oppure in modalità "Slave" per andare ad equipaggiare i nodi "periferici" della rete, attuatori, sensori e telecomandi. Ogni rete RFTide è caratterizzata da un NetworkID univoco, in modo che i moduli appartenenti a due reti adiacenti non possano interferire tra di loro. In quest'ottica il nodo "Master" ricopre una particolare importanza, in quanto è quello che "propaga" il suo "NetworkID", univoco e codificato in fabbrica, agli altri nodi

della rete, mediante il processo di "learning", come vedremo in dettaglio in seguito. La comunicazione tra i moduli, tipicamente, avviene secondo due modalità: una modalità "comando", in genere dal modulo master verso gli slave, quando si desidera eseguire un'azione verso un attuatore, verificare lo stato di una periferica oppure eseguire il processo di learning, per assegnare il NetworkID ed un indirizzo univoco ad un modulo (nodo) periferico per associarlo alla rete. La seconda modalità, che possiamo definire di telemetria o più in generale di messaggistica, avviene in direzione opposta, dai moduli "slave" verso il "master", e viene utilizzata per un'ampia varietà di motivi e situazioni. Ciò avviene, per esempio quando viene attivato o manomesso un sensore PIR o magnetico, oppure un dispositivo remoto invia periodicamente le letture fatte da sensori specifici, oppure ancora per segnalare lo stato di carica insufficiente della batteria di un dispositivo periferico. Le unità Master e Slave sono basate sullo stesso modulo base RFTide; la distinzione tra Master e Slave si fa per come esso viene impiegato, di fatto il master è quello che propaga il suo NetworkID agli altri nodi della rete. Il modulo base RFTide, visibile in Fig. 2 è formato da un ricetrasmittitore RF ad 868,3 MHz interfacciato con un microcontrollore che, oltre a gestire la comunicazione tra i nodi, rende disponibili all'esterno due linee di uscita GPOUT (General Purpose Out) utilizzabili per comandare carichi mediante transistor o line-driver, e due linee digitali GPIN (General Purpose In) per acquisire livelli logici. Lo stadio RF è un ricetrasmittitore a 868,3 Mhz, con una sensibilità in ricezione di

Fig. 1



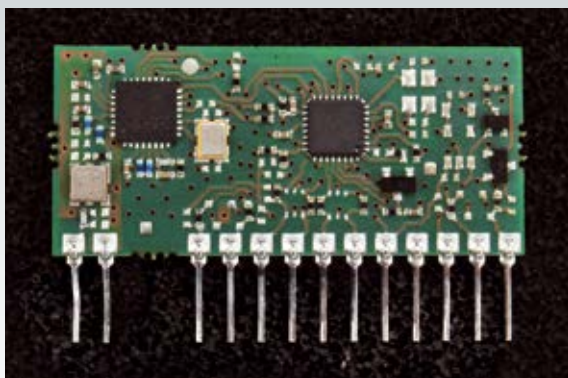
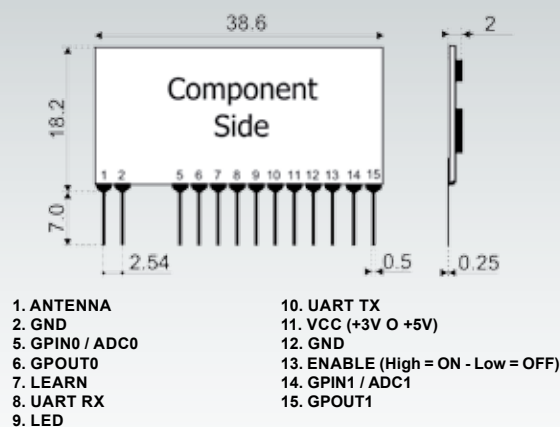


Fig. 2



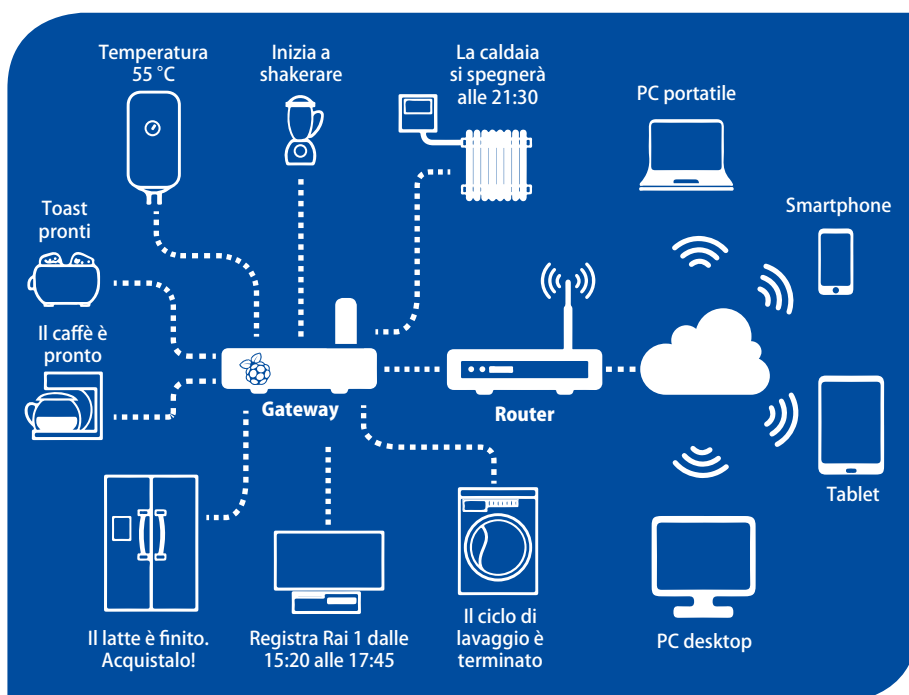
-100 dB e una potenza in trasmissione di +7 dB. Aurel produce i moduli RFTide in due versioni, distinte per tensione di alimentazione: quelli a 5V e quelli a 3V. Gli assorbimenti sono molto contenuti: 33 mA in trasmissione e 10 mA in ricezione; è prevista una modalità di Standby, nella quale l'assorbimento scende a 4 μ A. Nella versione a 3V, le due linee di ingresso possono essere configurate per leggere tensioni analogiche.

Per realizzare un Master è necessario utilizzare un modulo base RFTide provvisto di un numero seriale identificatore univoco (NetworkID di 4byte) che viene assegnato durante la fabbricazione e non può essere modificato; in questo caso si parla di "moduli serializzati". Ciascuno Slave ha invece come indirizzo predefinito 0x0000 e, durante il processo di associazione, questo assume come NetworkID lo stesso identificativo del Master, che viene copiato nella memoria non volatile interna. Solo moduli con lo stesso NetworkID possono comunicare tra loro. Per quanto riguarda la sicurezza, questa è garantita dal fatto che non è possibile clonare reti esistenti o comunque intromettersi in reti esistenti, se non copiando il numero seriale del Master che ha configurato la rete, in un altro Master, operazione eseguibile solo disponendo fisicamente del

Master da clonare. Una funzionalità interessante della rete RFTide, anche se non (ancora) sfruttata dal sistema che vi presentiamo, è data dal fatto che ciascun nodo può comunicare direttamente con qualsiasi altro; infatti ciascun modulo della rete ha un indirizzo univoco assegnato durante la fase di apprendimento. Questo indirizzo (unitamente al Network-ID) consente di distinguere i vari moduli connessi a una rete e di dirigere le comunicazioni da modulo a modulo (Point-To-Point) secondo necessità; pertanto gli Slave potranno essere programmati per comunicare tra

loro senza interessare il Master. È prevista anche una modalità di comunicazione "Broadcast" mediante la quale il Master (o un nodo) può comunicare con tutti gli altri contemporaneamente inviando loro un comando od un messaggio utilizzando l'indirizzo "0xFF".

Quando sulla rete viaggia un comando, solo il nodo Slave destinatario risponderà, eseguendo l'operazione richiesta, ed eventualmente trasmettendo, nel pacchetto di risposta, i dati richiesti; ciascun dispositivo Slave, infatti, dispone di un indirizzo univoco assegnato nel processo



di associazione alla rete.

I tempi di trasmissione e ricezione nelle reti RFTide sono indipendenti dal numero di nodi presenti: il tempo di latenza è di 200 ms tra l'immissione di un pacchetto nella rete ed il successivo. Il numero massimo di nodi ammessi in una rete è 255, con indirizzi da "0x00" a "0xFE".

Qualsiasi modulo RFTide non serializzato può svolgere la funzione di Slave ed il suo compito diventa quello di ricevere i comandi dal Master e di eseguirli; può altresì essere interrogato sullo stato dei suoi ingressi o venire usato per leggere il valore dei sensori ad esso collegati. In questo caso lo Slave è costantemente nello stato di standby (TX spento) e si risveglia solo quando deve trasmettere dei dati verso la rete. Il risveglio può essere programmato ad intervalli regolari, ma può anche avvenire alla comunicazione dell'ingresso GPIN0. Queste caratteristiche consentono di alimentare a batteria gli Slave, garantendo una lunghissima autonomia d'esercizio.

IL PROTOCOLLO RFTide

Nell'applicazione che vi presentiamo, i moduli RFTide a radiofrequenza, sono gestiti in modo "trasparente", sia dal "firmware" dei singoli moduli predisposti da UDOiT che dalla applicazione "embedded" che gira sul gateway Raspberry Pi. Tuttavia, vista la natura complessivamente open source della soluzione e la conseguente possibilità, come vedremo in un prossimo futuro, di introdurre componenti "personalizzati" nel sistema, è opportuno dare uno sguardo, anche superficiale,

Tabella 1

Funzione	Comando	Payload	Note
Associazione e Learning	0x0003	I byte da 0 a 6 a 0x00. Il byte 7 contiene l'indirizzo che si vuole assegnare al dispositivo.	Il comando non prevede un pacchetto di risposta
Pacchetto di risposta a pronte di un comando	0x0004	Il contenuto del payload dipende dal tipo di dispositivo	
Pacchetto inviato di iniziativa da un dispositivo di tipo sensore	0x0000	Il contenuto del payload dipende dal tipo di dispositivo	

agli "internals" del sistema, a partire dal protocollo di comunicazione, o almeno a quella sua parte disponibile all'utente, sulla porta seriale (piedini 8 e 10) dei moduli a radiofrequenza. Il protocollo è composto da pacchetti di struttura e lunghezza fissi, 11 byte esadecimale, come visibile nella **Fig. 3**. La comunicazione UART presenta le seguenti caratteristiche:

Baud Rate: 19200

Bit: 8

Parità: None

Stop bit: 1

La struttura di ciascun pacchetto è composta nel modo seguente:

- I primi due byte esprimono il codice del comando, o del tipo di messaggio contenuto nel resto del "pacchetto". Notate che i due byte sono ordinati secondo la convenzione Big-Endian, dove il byte meno significativo (LSB – Least Significant Byte) viene prima del byte più significativo (MSB – Most Significant Byte). La convenzione Big-Endian è molto utilizzata nei protocolli di comunicazione ed è conosciuta anche come "Network Byte Order". Ciascun dispositivo è caratterizzato da un proprio elenco di "Comandi" ai quali corri-

spondono differenti significati dei byte che costituiscono il "payload" (Byte da 0 a 7). Alcuni Comandi come i comandi di Learning e di richiesta di stato sono comuni a tutti i componenti del network;

- Il byte Address, contiene l'indirizzo del modulo, mediante il quale il modulo stesso e, di conseguenza il dispositivo sul quale è installato, vengono riconosciuti univocamente nel sistema. Gli indirizzi vanno da 0x00 a 0xFF dove 0xFF è l'indirizzo di broadcast e non può essere assegnato ad alcun modulo. Restano quindi 255 indirizzi effettivi che possono essere assegnati ai singoli moduli. Nel sistema impostato da UDOiT l'indirizzo 0x00 è assegnato in modo predefinito al Dongle USB, che deve essere installato sul gateway e costituisce il modulo Master dell'intero sistema. Tutti gli altri indirizzi sono utilizzabili liberamente;
- Gli ultimi otto byte del "pacchetto", utilizzabili singolarmente od in combinazione tra di loro contengono le informazioni da trasferire tra i moduli, come i parametri da associare al comando nel caso, appunto, di pacchetti di comando, oppure i dati provenienti dai dispositivi periferici in caso di pacchetti inviati su "iniziativa" di questi ultimi o a fronte di specifiche richieste da parte del "Master".

Fig. 3

Comando		Indirizzo	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
LSB	MSB									

Nella **Tabella 1** sono descritti i comandi più importanti, di uso generico, per la configurazione e la gestione del sistema. Ricordiamo che ogni pacchetto deve essere composto rispettando la struttura dei pacchetti, visibile in **Fig. 3**, che qui riassumiamo:

- Comando: due byte nel formato Big-Endian;
- Indirizzo del dispositivo al quale è destinato il messaggio o, in caso di messaggio di risposta, dal quale proviene il messaggio;
- Payload: otto byte dipendenti dal tipo di comando, o di risposta, e dal tipo di dispositivo.

DONGLE USB

Il dongle USB, come abbiamo già anticipato, è il dispositivo Master di tutta la rete. Viene fornito con un Network ID univoco e non richiede alcuna personalizzazione od operazione di configurazione. Nella pratica si tratta di un modulo RFTide al quale è stato aggiunto un convertitore USB-seriale in modo da renderlo compatibile con qualunque dispositivo dove sia presente una porta USB, come un PC, un microcomputer od un microcontrollore. Nel sistema UDOiT deve essere semplicemente inserito in una porta USB del gateway costituito da Raspberry Pi, dove, se è l'unico dispositivo USB-seriale inserito, verrà riconosciuto automaticamente e gli verrà assegnato il nome di device `/dev/ttyUSB0`. Contrariamente a quanto avviene con Windows, utilizzando Raspian non è necessario installare alcun driver per il riconoscimento del dongle. Potete sincerarvene con i comandi:

```
lsusb
e
dmesg
```

Per chi fosse alle prime armi con l'utilizzo di Raspberry Pi e del sistema operativo Raspian rimaniamo al libro "Raspberry Pi – Il mio primo Linux embedded". Sul dongle USB è visibile un LED multicolore (**Fig. 4**) che evidenzia lo stato di funzionamento del dongle. In presenza di tensione il LED è di colore BLU. Quando viene richiesto l'invio di un pacchetto il LED lampeggia in ROSSO. Il dongle restituisce propri pacchetti solo quando viene interrogato circa il suo stato, oppure quando si verifica una collisione, ovvero viene richiesto al dongle di trasmettere un pacchetto quando la rete è già impegnata da un pacchetto in arrivo oppure è sottoposta ad interferenze che impediscono la comunicazione corretta. In questo caso il don-

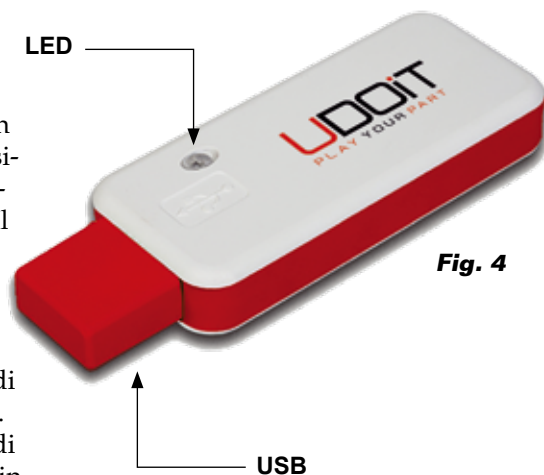


Fig. 4

gle restituisce un pacchetto che indica lo stato "busy" della rete e quindi la sua impossibilità a trasmettere. Il pacchetto "busy" è riconoscibile dal valore "0x000c" nei byte "Comando".

SENSORE PIR

Il sensore "PIR", il circuito stampato del quale è visibile in **Fig. 5**,

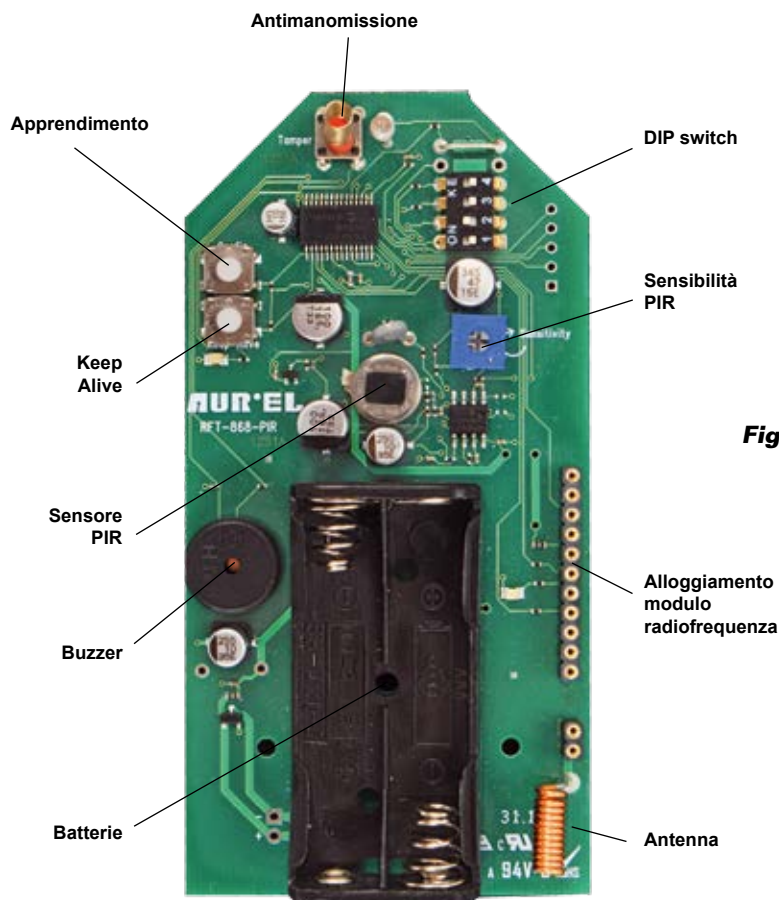


Fig. 5

Tabella 2

Tabella codici di allarme PIR (Byte 2) 0: Non attivo - 1: Attivo	
Bit 0	PIR
Bit 1	Tamper
Bit 6	Supervisione
Bit 7	Radio mode

Tabella 3

Tabella Stati della batteria (Byte 2)	
0	Batteria carica
1	Batteria scarica

permette di rilevare una presenza o, più precisamente, una sorgente di calore, nell'ambiente antistante il suo angolo di visione. A fronte della rilevazione di una presenza il sensore invia una serie di "pacchetti" il cui contenuto e la cui cadenza può essere personalizzata mediante i DIP Switch presenti sul circuito stampato.

Nell'applicazione UDOiT questi pacchetti sono recepiti ed elaborati in base alle "regole" impostate nell'applicazione stessa. A seconda dei casi può essere attivata una procedura di segnalazione dell'evento, per esempio attivando una sirena per un certo tempo ed inviando un messaggio e-mail o su un altro sistema di messaggistica, oppure può essere attivata un'utenza, come per esempio accendere una o più lampade. Ovviamente le diverse funzioni possono essere combinate. Il sensore PIR segnala anche i tentativi di manomissione e lo stato di carica della batteria. Queste segnalazioni vengono visualizzate sull'interfaccia grafica dell'applicazione di controllo. Sul dispositivo, oltre al sensore PIR ed ai DIP switch di configurazione sono presenti:

- Un microinterruttore da utilizzare per il processo di "Learning";
- Un microinterruttore da utilizzare per "risvegliare"

il dispositivo dallo stato di "sleep". Questa funzione è utile per mettere in condizione il dispositivo di ricevere pacchetti di configurazione dal sistema di gestione. Approfondiremo questa funzionalità in seguito;

- Un microinterruttore con la funzione di antimanomissione: se per caso il dispositivo viene manomesso, quest'ultimo entra in allarme ed invia pacchetti che avvertono dell'accaduto;
- Un trimmer per la regolazione della sensibilità del sensore PIR;
- Un buzzer che permette di emettere un segnale acustico in caso la batteria di alimentazione scenda sotto la soglia di carica minima; quest'ultima è una delle funzionalità impostabile con un opportuno pacchetto di configurazione;
- Infine, ben visibili, sono il modulo a radiofrequenza RFTide, l'antenna a spirale in rame smaltato e l'alloggiamento per le due batterie da 1,5V in formato AAA.

Utilizzando i DIP-Switch sono possibili le seguenti impostazioni:

- Dip 1: Radio mode (per la verifica della copertura radio);
- Dip 2: Supervisione (verifica presenza sensore);
- Dip 3: Pulse counter (allarme con una o più variazioni del sensore);
- Dip 4: Tempo inibizione (tempo di inibizione tra due allarmi, 4 secondi o 4 minuti).

Le impostazioni dei parametri tramite DIP-Switch sono lette per 10 minuti dopo aver inserito le batterie. Dopo tale tempo un cambio stato del DIP-Switch non viene preso in considerazione. La corretta modalità di impostazione dei DIP-Switch,

quindi, consiste nel rimuovere le batterie (o almeno una delle batterie), impostare i DIP-Switch nella configurazione desiderata e riposizionare le batterie, ridando tensione al dispositivo. In caso il dispositivo rilevi una situazione anomala, allarme o manomissione, il dispositivo si risveglia ed invia una serie di pacchetti con tutte le condizioni di anomalia che sono state rilevate.

In caso di allarme il PIC che gestisce il dispositivo risveglia dallo stato di sleep il modulo RFTide e manda un pacchetto così composto

- Comando: Codice SEND PACKET (0x00 0x0A);
- Destinazione: Indirizzo del dispositivo, codificato in fase di associazione alla rete e contestuale learning;
- Payload byte 0: 0x01 (Tipo sensore PIR);
- Payload byte 1: 0x32 (Pacchetto di allarme proveniente da un sensore PIR);
- Payload byte 2: Codici di allarme e anomalie (Vedi **Tabella 2**);
- Payload byte 3: Stato della batteria (Vedi **Tabella 3**);
- Payload byte 4-5: Non Utilizzati;
- Payload byte 6-7: Indice incrementale (l'applicazione si occuperà di fare il controllo sull'indice).



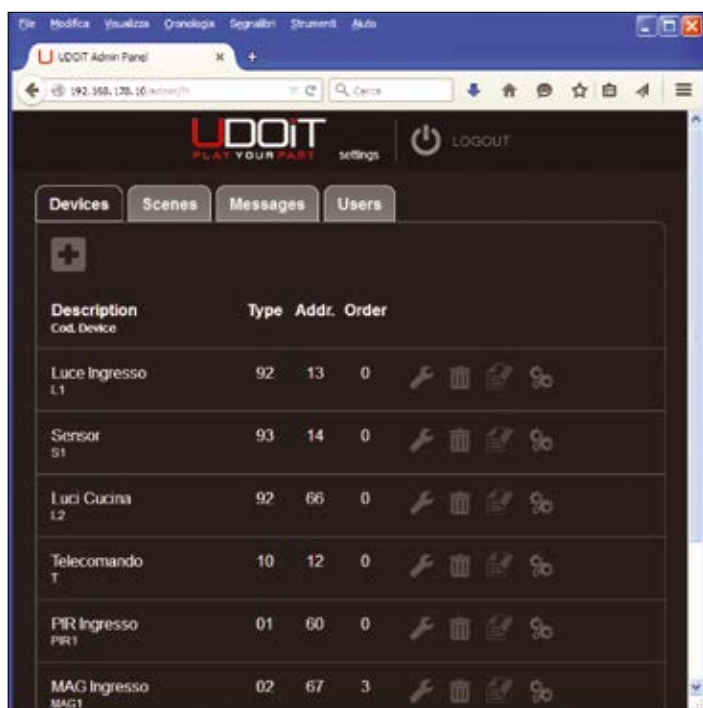


Fig. 6

Il protocollo RFTide prevede che, dopo aver inviato un pacchetto, il dispositivo attenda un pacchetto di risposta (ANSWER_MESSAGE) e a quel punto il sensore si può spegnere, altrimenti si spegne comunque dopo un secondo. Mediante la funzionalità di configurazione avanzata disponibile nella applicazione di gestione, che descriveremo nei prossimi articoli, è possibile impostare le seguenti funzionalità:

- Abilitazione buzzer per segnalazione batteria scarica;
- Abilitazione del LED durante la trasmissione di un allarme;
- Abilitazione del LED durante la trasmissione in supervisione;
- Tempo di supervisione (30-60-90 min).

Per aggiungere alla rete un dispositivo sensore PIR si utilizza la funzione "Add Device" (Fig. 6) della applicazione di gestione, che si trova nella pagina "Settings", come visibile in Fig. 7. La sezione "Settings" è protetta da USERID e PASSWORD.

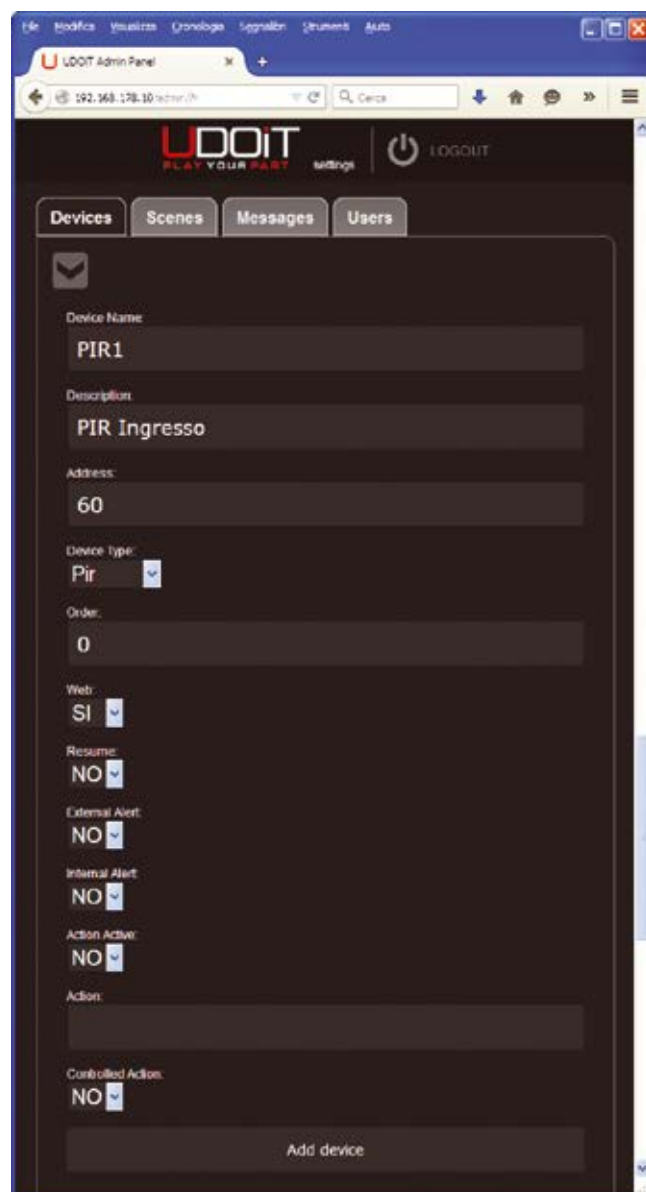


Fig. 7

Nella stessa figura si vede che il dispositivo è caratterizzato da un nome, di lunghezza massima quattro caratteri, una descrizione, che apparirà nelle pagine web di gestione dei dispositivi della rete, un indirizzo, che deve essere univoco nell'intero sistema, ed il tipo di dispositivo, nel nostro caso "PIR". Abbiamo poi la possibilità di scegliere se vedere o meno l'entrata del dispositivo nella pagina web di gestione dell'applicazione ed in caso positivo scegliere una posizione nell'elenco (campo "Order"). I

campi successivi servono per fare entrare o meno il dispositivo nella rete di controllo e sicurezza e per pilotare, in caso di attivazione, l'esecuzione di un processo personalizzato. Queste ultime funzionalità sono un po' più complesse da configurare e personalizzare e le affronteremo in seguito in un articolo dedicato.

SENSORE MAGNETICO

Il sensore "Magnetico", visibile in Fig. 8 e Fig. 9, permette di rilevare l'apertura di una porta, una finestra, un cassetto o presso-

Fig. 8

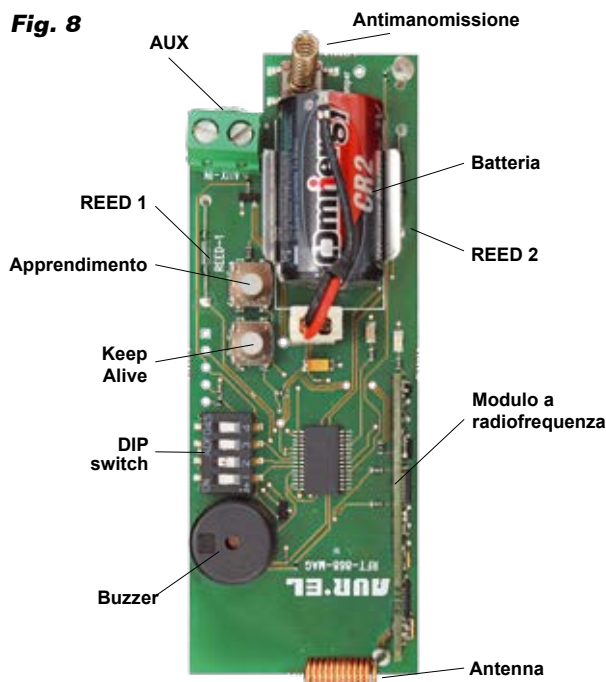
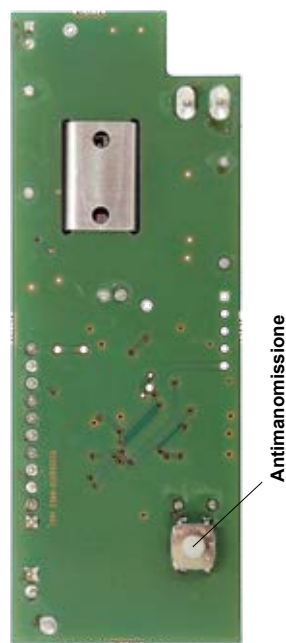


Fig. 9



ché qualsiasi “arredo” del quale si voglia tenere sotto controllo il meccanismo di accesso. Il funzionamento del dispositivo si basa sull’apertura e la chiusura di un interruttore magnetico di tipo “Reed”. L’interruttore, anzi gli interruttori, in quanto il dispositivo è dotato di due interruttori Reed, uno per lato, selezionabili mediante DIP-Switch, sono montati sul circuito stampato, mentre il magnete che attiva il contatto Reed è contenuto in uno “scatolino” separato da montare sulla parte mobile del meccanismo di accesso che si vuole tenere sotto controllo. A fronte della apertura o chiusura dell’accesso sotto controllo il dispositivo invia una serie di “pacchetti” il cui contenuto e la cui cadenza possono essere personalizzate mediante i DIP Switch presenti sul circuito stampato. Nell’applicazione UDOiT questi pacchetti sono recepiti ed elaborati in base alle “regole” impostate nell’applicazione stessa. Anche per il dispositivo a sensore magnetico, come nel caso del PIR, a seconda dei casi può essere

attivata una procedura di segnalazione dell’evento, per esempio attivando una sirena per un certo tempo ed inviando un messaggio e-mail o su un altro sistema di messaggistica, oppure può essere attivata un’utenza, come per esempio l’accensione di una o più lampade. Ovviamente le diverse funzioni possono essere combinate. Anche il dispositivo a sensore magnetico segnala i tentativi di manomissione e lo stato di carica della batteria. Queste segnalazioni vengono visualizzate sull’interfaccia grafica dell’applicazione di controllo. Sul dispositivo, oltre ai sensori magnetici reel ed ai DIP switch di configurazione sono presenti:

- Un microinterruttore da utilizzare per il processo di “Learning”;
- Un microinterruttore da utilizzare per “risvegliare” il dispositivo dallo stato di “sleep”. Questa funzione è utile per mettere in condizione il dispositivo di ricevere pacchetti di configurazione dal sistema di gestione. Approfondiremo questa funzionalità in seguito;

- Un microinterruttore con la funzione di antimanomissione, se per caso il dispositivo viene manomesso, quest’ultimo entra in allarme ed invia pacchetti che avvertono dell’accaduto;
- Un ulteriore microinterruttore nella parte posteriore del circuito stampato, che si attiva quando il dispositivo viene rimosso dal suo alloggiamento;
- Un connettore per un ingresso ausiliario, al quale può essere collegata l’uscita relè di un qualsiasi altro dispositivo con uscita digitale, per esempio un sensore di fumo o di acqua;
- Un buzzer che può essere configurato per emettere un segnale acustico in caso la batteria di alimentazione scenda sotto la soglia di carica minima; questa è una delle funzionalità impostabile con un opportuno pacchetto di configurazione;
- Infine, ben visibili, sono il modulo a radiofrequenza RFTide, l’antenna a spirale in rame smaltato e l’alloggiamento per la batteria al litio da 3V di tipo CR2.

Utilizzando i DIP-Switch sono

Tabella 4

Tabella codici di allarme Magnetico (Byte 2) 0: Non attivo - 1: Attivo	
Bit 0	Reed
Bit 1	Tamper
Bit 2	Antistrappo
Bit 3	Ausiliario
Bit 6	Supervisione
Bit 7	Radio mode

Tabella 5

Tabella Stati della batteria (Byte 3)	
0	Batteria carica
1	Batteria scarica

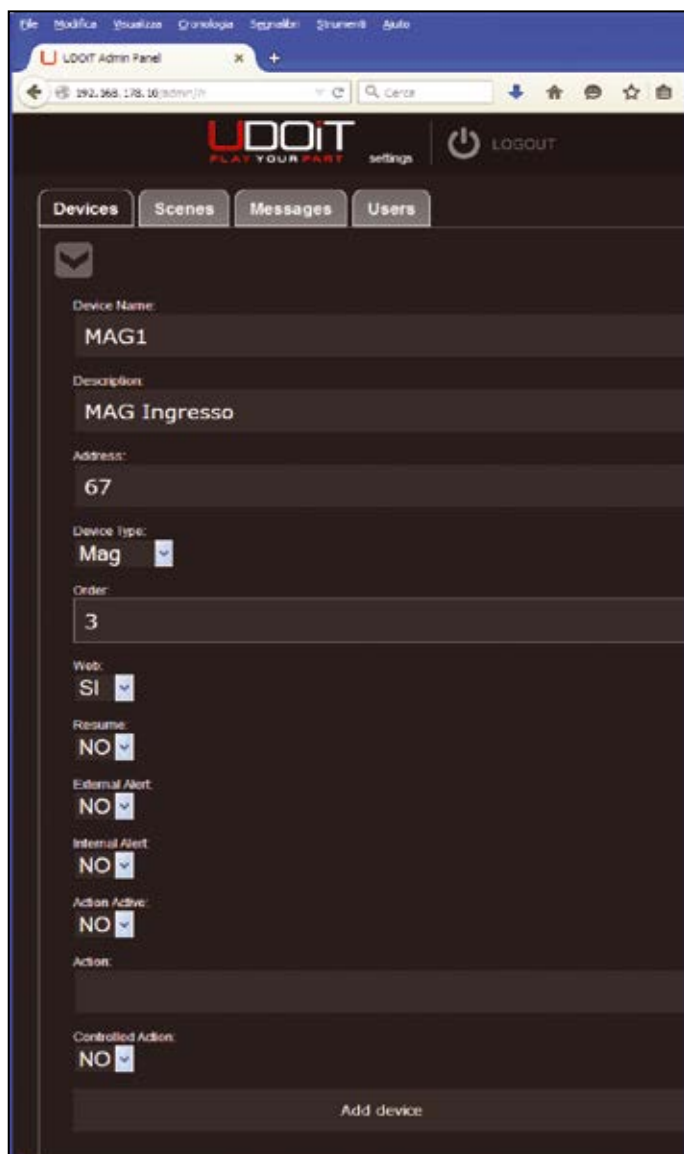


Fig. 10

possibili le seguenti impostazioni:

- Dip 1: Radio mode (per la verifica della copertura radio);
- Dip 2: Supervisione (verifica presenza sensore);
- Dip 3: Reed 1 o 2 (scelta lato Reed attivo);
- Dip 4: Set magneti (per messa a punto posizione magneti).

Le impostazioni dei parametri tramite DIP-Switch sono lette per 10 minuti da quando vengono inserite le batterie.

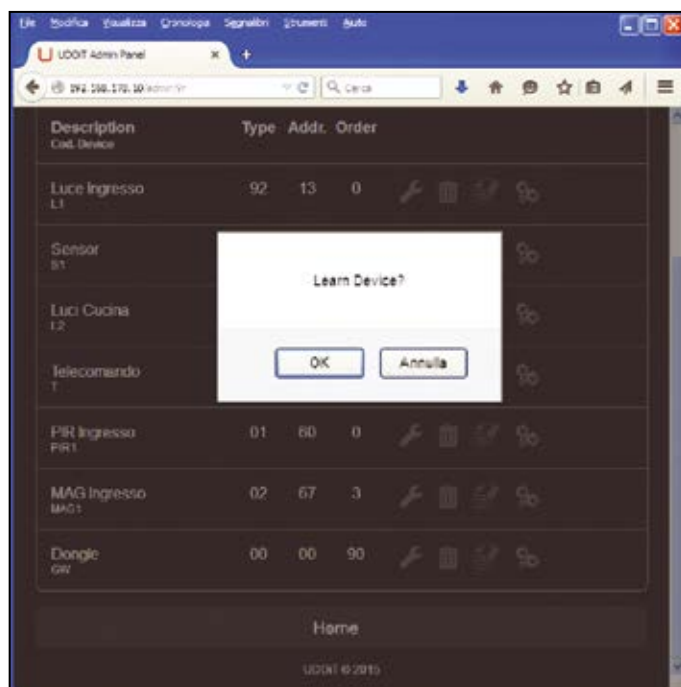
Dopo tale tempo un cambio stato del DIP-Switch non viene preso in considerazione. La corretta

modalità di impostazione dei DIP-Switch, è la stessa descritta per il dispositivo con il sensore PIR.

In caso di allarme il microcontrollore che gestisce il dispositivo risveglia dallo stato di sleep il modulo RFTide e manda un pacchetto così composto:

- Comando: Codice SEND PACKET (0x00 0x0A);
- Destinazione: Indirizzo del dispositivo, codificato in fase di associazione alla rete e contestuale learning;
- Payload byte 0: 0x02 (Tipo sensore Magnetico);
- Payload byte 1: 0x31 (Pacchet-

Fig. 11



to di allarme proveniente da un sensore Magnetico);

- Payload byte 2: Codici di allarme e anomalie (Vedi **Tabella 4**);
- Payload byte 3: Stato della batteria (Vedi **Tabella 5**);
- Payload byte 4-5: Non Utilizzati;
- Payload byte 6-7: Indice incrementale (l'applicazione si occuperà di fare il controllo sull'indice).

Anche per aggiungere alla rete un dispositivo sensore Magnetico si utilizza la funzione "Add Device" dell'applicazione di gestione, che si trova nella pagina "Settings", come visibile in **Fig. 10**. La procedura rispecchia quella adottata per il dispositivo sensore PIR, tranne che per il tipo dispositivo, dove questa volta sceglieremo l'opzione "MAG".

LEARNING DEI DISPOSITIVI

Una volta aggiunti i dispositivi



Fig. 12

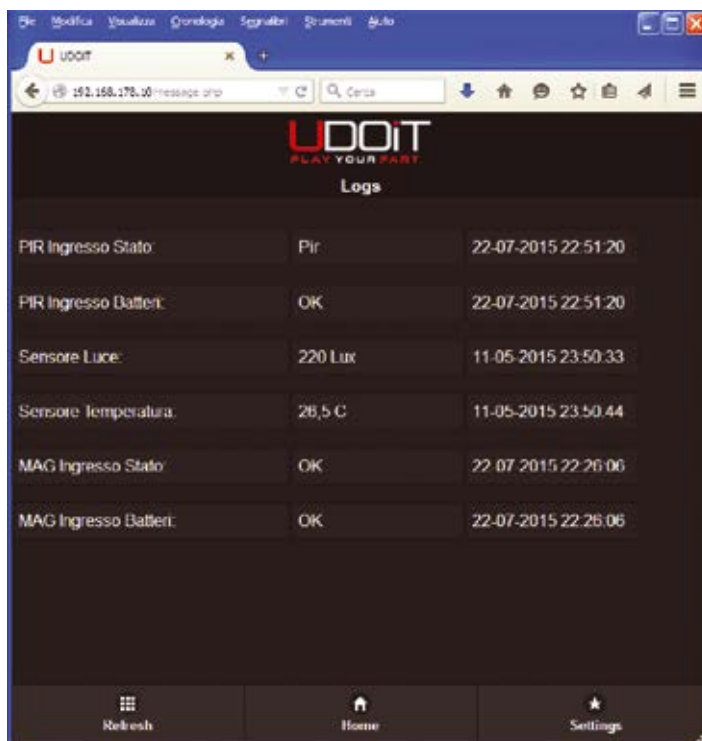


Fig. 13

desiderati nella pagina "Devices" del pannello "Settings" è necessario "associare" alla rete i dispositivi descritti, mediante un processo di apprendimento, durante il quale il dispositivo viene personalizzato con il Network ID di appartenenza (quello del Dongle, Master della rete) e l'indirizzo univoco di identificazione del dispositivo stesso, una sorta di cognome e nome che permetteranno di riconoscere univocamente il dispositivo nella rete e impediranno ai dispositivi presenti su reti adiacenti di interferire. Il processo di apprendimento avviene come segue :

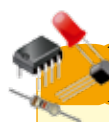
- Nella pagina "Devices" del pannello "Settings" si "pre-nota" l'operazione di apprendimento cliccando sull'icona a forma di foglio di carta e matita ("Learn Device") in corrispondenza del dispositivo per il quale si vuole eseguire l'associazione **Fig. 11**;
- A questo punto, per i dispositivi PIR e Magnetico, si preme

il tasto "Config" sul circuito stampato per circa cinque secondi, al rilascio si accende un LED rosso;

- Entro 15 secondi cliccare sul pulsante "OK" della finestra "Pop-Up" sulla pagina web;
- Se tutto ha avuto successo il LED lampeggia per qualche secondo;
- A questo punto il dispositivo è in grado di essere riconosciuto ed operare sulla rete;
- In caso non venga avviata la procedura di apprendimento il dispositivo esce dalla stato di apprendimento dopo circa cinque secondi.

Se è stata scelta l'opzione WEB "SI" lo stato del dispositivo sarà visibile nella pagina di gestione web del sistema (**Fig. 12**). In ogni caso lo stato del sensore e della sua batteria sono visibili nella pagina "Logs" dell'applicazione web di gestione (**Fig. 13**). Ci fermiamo qui, nei prossimi numeri approfondiremo le carat-

teristiche di ulteriori dispositivi e ulteriori funzionalità dell'applicazione di gestione. ■



per il MATERIALE

I prodotti presentati in questo progetto possono essere acquistati presso Futura Elettronica. Il dongle USB (cod. RFT868USB) costa 39,00 Euro, il sensore magnetico (cod. RFT868MAG) è disponibile a 49,00 Euro, il sensore di movimento (cod. RFT868PIR) è in vendita a 55,00 Euro, il modulo a relé (cod. RFT868REL) costa 55,00 Euro e il modulo a dimmer (cod. RFT868DIM) è disponibile a Euro 55,00.

Il radiocomando a 4 canali (cod. RFT8684CH) è in vendita a Euro 39,00 Euro, e la Raspberry Pi 2 Tipo B (cod. 8326274RS) a Euro 44,00.

Il materiale va richiesto a:
Futura Elettronica, Via Adige 11,
21013 Gallarate (VA)
Tel: 0331-799775 • Fax: 0331-792287
<http://www.futurashop.it>



Windows 10



Sperimentiamo con la versione embedded del nuovo sistema operativo di Microsoft installandola in Raspberry Pi e costruendo le nostre prime applicazioni.

MICROSOFT ASSAGGIA IL LAMPONE

..... di FABRIZIO GUGLIELMINO

Abbiamo visto, nel fascicolo precedente, alcuni dei movimenti che stanno avvicinando Microsoft al mondo open source e anticipato il rilascio da parte di Microsoft di una versione di Windows 10 specificatamente pensata per Raspberry Pi: si tratta del solo "core" privato dell'interfaccia grafica cui gli utenti Microsoft si sono abituati. In questo articolo vedremo di cosa si tratta e che possibilità offre. Prima di buttarci "anima e cuore" negli aspetti tecnici elenchiamo i requisiti necessari a provare in pratica il sistema operativo:

- Windows 10 Insider Preview installato su un PC;
- Raspberry Pi 2 e relativo alimentatore;
- micro SD da 8 GB;
- cavo HDMI;
- cavo ethernet;
- monitor al quale connettere la scheda Raspberry Pi.

La procedura di installazione ufficiale richiede l'utilizzo di una macchina con installato Windows 10; esistono in realtà dei metodi non ufficiali (docu-

mentati sul web) per installare Windows 10 sulla Raspberry Pi anche disponendo di versioni di Windows precedenti, OS X o Linux, ma in questa sede ci limiteremo al metodo ufficiale.

PREPARAZIONE DELL'AMBIENTE

Partiamo quindi preparando la nostra Raspberry Pi all'installazione: la prima operazione da compiere è il download dei tool e del sistema operativo, che va fatto dal link web <http://go.microsoft.com/fwlink/?LinkId=616847>, da cui verrà scaricata una ISO che si chiama "IOT Core Rpi"; un doppio clic sul file monterà l'immagine assegnandole un drive virtuale, dove troveremo il file di installazione (Fig. 1). Lo step successivo consiste nell'eseguire il file di installazione "Windows_10_IoT_Core_Rpi2" che installerà i tool e deposerà l'immagine del sistema vera e propria in "C:\Program Files (x86)\Microsoft IoT\FFU\RaspberryPi2".

Il file che ci interessa si chiama "flash.ffu" e rappresenta l'immagine del sistema operativo.

Il formato FFU è stato creato da Microsoft apposta

This PC > DVD Drive (H:) J_ARPI_A32FRE0_EN-US_DVS

Name	Date modified	Type	Size
Windows_10_IoT_Core_RPi2	6/17/2015 2:04 PM	Windows Installer ...	472,800 KB

Fig. 1

per il deploy di Windows.

Per creare la scheda SD necessaria ad avviare la Raspberry Pi è necessario eseguire "WindowsIoTImageHelper" che presenterà un'interfaccia con la quale specificare la scheda SD sulla quale scrivere e il percorso del file FFU (Fig. 2). Indicate le due opzioni basterà selezionare il tasto "Flash" e attendere che la procedura termini.

PRIMO AVVIO DI WINDOWS 10 IoT CORE

Completata la procedura di trasferimento dell'immagine avremo la nostra SD pronta da inserire nella Raspberry Pi per avviare Windows 10. Il primo avvio richiede un po' di tempo e qualche rapida impostazione (lingua ed eventuali parametri per il WiFi se disponibile). Dopo il boot ci si troverà di fronte ad una schermata simile a quella in Fig. 3.

Se qualcuno si aspettava il familiare start menu o i "tile" alla Windows 8 rimarrà deluso: Windows 10 IoT Core è infatti una versione "ridotta all'osso" di Windows 10 e tra le cose eliminate c'è proprio

tutta l'interfaccia utente. Lo scopo di questa versione non è poter installare pacchetti applicativi tipo Office, ma piuttosto disporre di tutta la potenza del motore di Windows per sviluppare Universal App. Di fatto la schermata che vediamo, altro non è che la Universal App predefinita eseguita all'avvio; col tempo potremo sostituirla con una nostra.

Non va dimenticato che questa versione del sistema operativo Microsoft è specificamente pensata per gli sviluppatori, non per niente il banner in basso alla schermata principale recita "Visit windowsondevices.com to start developing".

Quello che dobbiamo fare ora è proprio seguire l'indicazione che ci viene fornita: dobbiamo infatti spostarci temporaneamente sul nostro PC con installato Windows 10 "full" e scaricare l'ambiente di sviluppo con il quale realizzeremo il nostro primo esperimento.

L'ambiente di sviluppo ufficiale di Microsoft è Visual Studio: si tratta di un IDE molto potente e completo. Fortunatamente Microsoft ha

reso disponibile Visual Studio in edizione Community, che è gratuito e perfettamente utilizzabile per i nostri scopi, scarichiamolo quindi da www.visualstudio.com e procediamo con l'installazione.

Se abbiamo seguito correttamente i passi precedenti dovremmo disporre dei seguenti strumenti:

- Raspberry Pi con installato Windows 10 IoT Core, meglio se collegata a monitor, mouse e tastiera;
- PC con Windows 10 con installato Visual Studio Community edition e le componenti Microsoft IoT installate.

PRONTI PER SPERIMENTARE

Terminata la fase di preparazione dell'ambiente, siamo pronti a sperimentare; è importante notare che per eseguire gli esempi è necessario che la Raspberry Pi e il PC con Windows 10 siano collegati alla stessa rete. Se questa condizione è soddisfatta dovremmo vedere elencata la Raspberry Pi nel tool Windows IoT Core Watcher. Questo tool, installato in automatico quando abbiamo inizialmente installato il contenuto di "IoT Core Rpi", si occupa di monitorare la rete alla ricerca di dispositivi con



Fig. 2

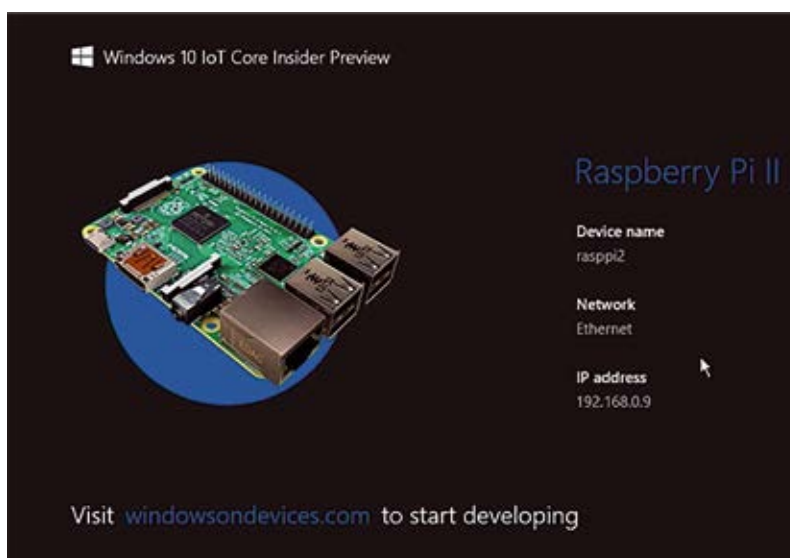


Fig. 3

installato Windows 10 IoT Core; il fatto di veder apparire la scheda in questo tool ci conferma che tutto è pronto e correttamente configurato (Fig. 4).

Avviamo quindi Visual Studio e creiamo il primo progetto, per “scaldarci” inizieremo con una variante del classico “Hello World”. Serve fare una premessa: quando svilupperemo App per Windows 10 IoT Core avremo due possibilità: creare App headed o headless. Come intuibile dal nome, le prime sono quelle che interagiscono con l’utente tramite una user interface: in sintesi, necessitano almeno del monitor e probabilmente di mouse e tastiera.

Le applicazioni headless sono invece più simili a quanto siamo abituati a fare, per esempio, con Arduino, ovvero applicazioni che non usufruiscono di interfaccia utente standard ma che interagiscono direttamente con l’hardware e sensori. Ci soffermeremo per ora sulle App headed che permettono di interagire con l’utente mediante una user interface.

Avviato Visual Studio creiamo un nuovo progetto, dobbiamo scegliere “Windows Universal” e creare un progetto partendo dal modello

Listato 1

```
<Page
  x:Class="App2.MainPage"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:local="using:App2"
  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
  mc:Ignorable="d">

  <Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <TextBlock Foreground="Blue" HorizontalAlignment="Center"
      VerticalAlignment="Center" FontSize="64">Hello World</TextBlock>

  </Grid>
</Page>
```

“Blank App” (Fig. 5).

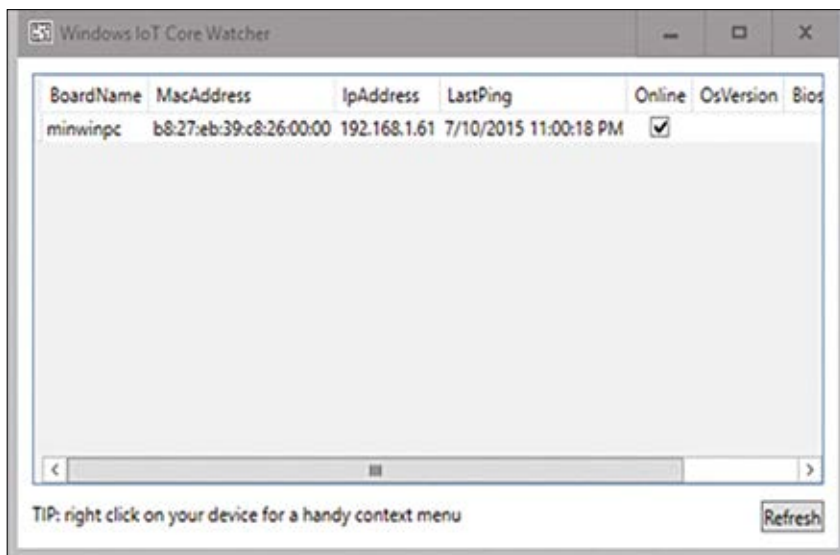
Visual studio creerà il nostro progetto che, pur partendo dal template “Blank App”, risulta composto da un discreto numero di file. Tra i file generati troveremo MainPage.xaml, i file XAML sono file XML usati per descrivere l’interfaccia utente. Nel nostro caso il file è composto solo dallo scheletro di base ma non ha di fatto contenuti. Per il primo test vogliamo semplicemente far apparire la scritta “Hello World” al centro dello schermo; modifichiamo quindi MainPage.xaml come nel Listato 1.

Rispetto al file originale abbiamo aggiunto un TextBlock, ovvero un controllo in grado di visualizzare

un testo. Impostato il suo colore al blu (Foreground=“Blue”) e allineato al centro della pagina (HorizontalAlignment=“Center” e VerticalAlignment=“Center”). La nostra prima Universal App è pronta: possiamo provarla. Nella toolbar di Visual Studio sono presenti i comandi per controllare l’esecuzione dell’App; come primo test la eseguiamo sul nostro computer, allo scopo è sufficiente fare clic sul pulsante run (Fig. 6) ossia la freccina simile ad un “play”. Se non abbiamo commesso errori vedremo una pagina bianca con un bel “Hello World” blue al centro: è un buon risultato ma il nostro obiettivo era quello di fare un’App per la Raspberry Pi. Il passo per arrivarci è molto breve. Sempre dal menu di prima, abbiamo una tendina che indica per quale architettura compilare la nostra App. Come è noto la Raspberry Pi è basata su tecnologia ARM, quindi scegliamo questa architettura dal menu. Vicino al pulsante che abbiamo usato per eseguire l’App c’è un’etichetta che recita “Computer locale” (o “Local computer” se abbiamo Visual Studio in inglese) e anche qui l’iconcina per aprire un menu a tendina (Fig. 7).

Visto che vogliamo eseguire il nostro codice sulla Raspberry Pi, dobbiamo scegliere “Computer remoto”, ci apparirà una finestra di

Fig. 4



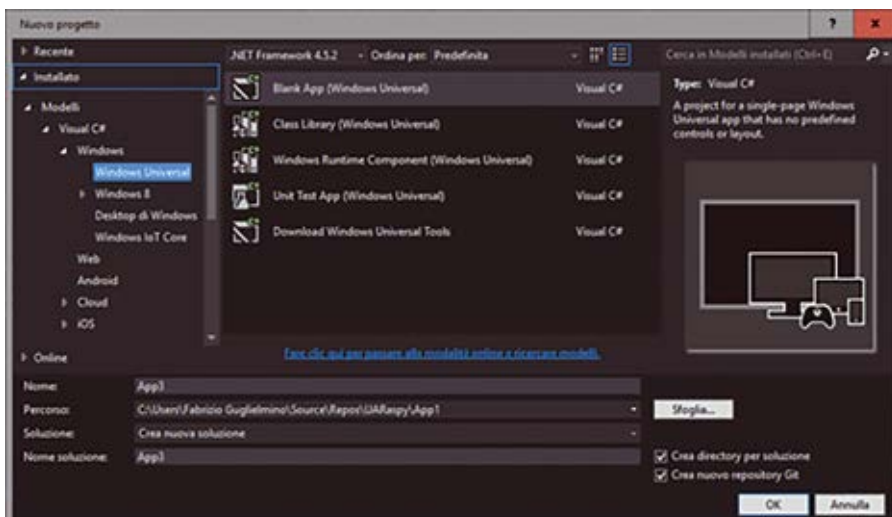


Fig. 5



Fig. 6

dialogo dove andare ad impostare l'IP della scheda e l'autenticazione (per ora scegliamo *nessuna*). Siamo nuovamente pronti a premere il tasto di esecuzione ma questa volta l'App verrà eseguita direttamente sulla Raspberry, vedremo la nostra etichetta blue "Hello Word" al centro del monitor collegato alla scheda.

Anche se ciò che abbiamo sviluppato è funzionalmente assai modesto abbiamo raggiunto un punto molto importante. Il poter eseguire l'App sia sul computer locale che sul device remoto è una cosa molto importante in fase di prototipazione, permette di essere molto più rapidi, per esempio, nelle fasi di design dell'interfaccia. Ovviamente man mano che aggiungeremo all'App funzionalità per interagire con hardware della scheda starà a noi gestire condizionalmente la presenza/assenza dei dispositivi richiesti.

Proseguiamo quindi espandendo un po' l'esempio iniziale per interagire con l'hardware; del resto uno degli scopi principali di quanto stiamo vedendo è proprio fornire uno strumento semplice e

potente ai Maker.

Quello che vogliamo fare ora è provare ad accendere un LED agendo con un clic su un tasto visualizzato nella schermata sul monitor. Per questo esempio dovremo collegare un LED ad uno dei GPIO della Raspberry Pi: useremo il pin 18 (Fig. 8).

Visto che non abbiamo ancora interagito con l'hardware della scheda, non abbiamo incluso il pacchetto necessario per farlo, come prima cosa, in "Esplora Soluzioni" di Visual Studio facciamo clic con il pulsante destro del mouse su "Riferimenti" e facciamo apparire il menu contestuale, dal quale selezioniamo "Aggiungi riferimento", allorché appare una finestra di dialogo con a destra un elenco di voci. Selezioniamo "Windows Universal" e mettiamo nella casella di testo per la ricerca in alto a destra "IoT"; in questo modo l'elenco dovrebbe mostrarci un solo elemento: "Windows IoT Extension SDK" (Fig. 9). Selezioniamolo e facciamo clic su OK per installarlo.

Il pacchetto "Windows IoT Extension SDK" ci offre tutte le estensioni specifiche per lavorare

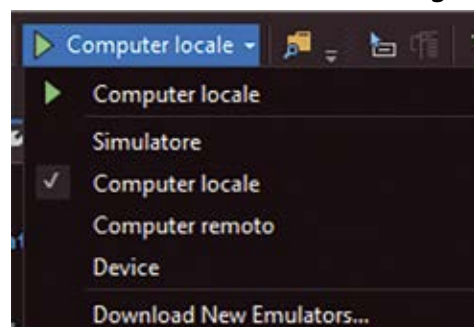
con l'hardware; di esse useremo le classi che permettono di interagire con i GPIO della scheda.

Iniziamo quindi a modificare il codice per adattarlo alla nuova esigenza: per prima cosa modifichiamo il file XAML per creare l'interfaccia che vogliamo; il contenuto di *MainPage.xaml* diventa come mostrato nel **Listato 2**.

Abbiamo aggiunto uno StackPanel, che è un controllo contenitore necessario quando si vogliono disporre più elementi nell'interfaccia. Lo StackPanel, in particolare, permette di disporre gli elementi in colonna uno di seguito all'altro (può anche lavorare con orientamento orizzontale se necessario). Dentro allo StackPanel abbiamo lo stesso TextBlock incontrato nel primo esempio, che funge semplicemente da titolo per la pagina. Sotto al TextBlock inseriamo un Button, il suo scopo è creare un pulsante nell'interfaccia. Il button è un controllo attivo, nel senso che è in grado di scatenare un'azione a fronte di un evento. L'attributo Click="Button_Click" indica che quando il bottone verrà premuto verrà eseguito il codice di un metodo chiamato Button_Click.

Vediamo quindi il codice che viene eseguito quando si fa clic sul pulsante come riportato nel **Listato 3**. Il pulsante agisce da switch che accende e spegne il LED in funzione della variabile "state". Allo stesso tempo viene modificata l'etichetta del bottone impostandola ad "ON"

Fig. 7



o "OFF" per indicare lo stato del LED, come visibile dal codice per alterare l'etichetta del bottone si valorizza la proprietà "Content". L'interazione con il GPIO avviene mediante l'oggetto "pin", ovviamente da qualche parte questo oggetto va definito e configurato specificando su quale pin lavorare e se in input o output. Tutto ciò avviene in un metodo chiamato all'avvio dell'App, come mostra il **Listato 4**.

InitGPIO fa esattamente quanto descritto, per prima cosa ricava un'istanza del GpioController, che come immaginabile rappresenta la classe di gestione dei GPIO.

Ricavata l'istanza del controller la utilizza per creare l'oggetto "pin" specificando su quale pin dovrà agire (nel nostro caso LED_PIN, ovvero 18). Configurando il modo operativo in Output e impostato al livello basso il pin, l'inizializzazione è completa. Ovviamente InitGPIO andrà chiamata all'avvio dell'App.

L'esempio completo del programma di controllo del LED (**Fig. 10**) può essere scaricato da <https://github.com/guglielmino/rpi-universal-app-ledsample>.

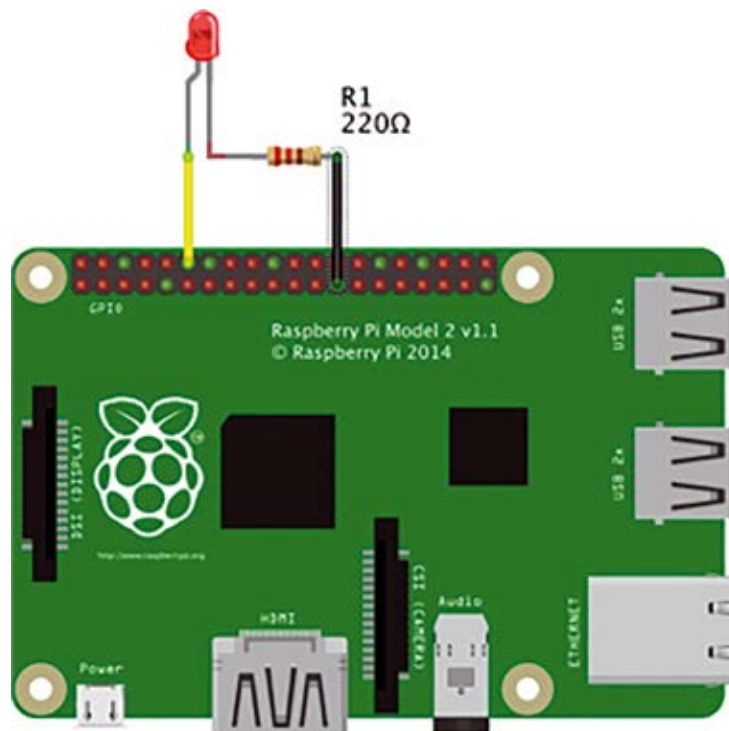


Fig. 8

PANNELLO DI CONFIGURAZIONE

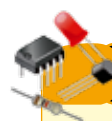
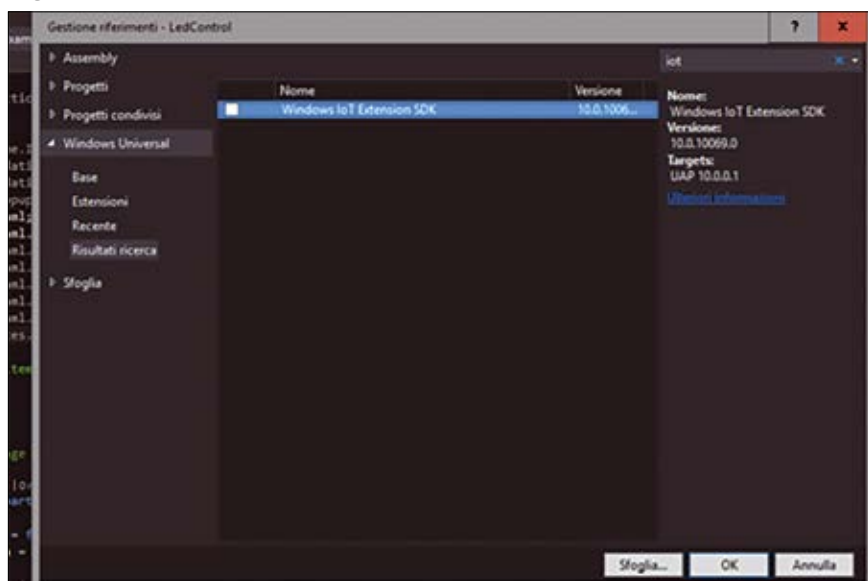
Abbiamo sperimentato con la scheda e scoperto che realizzare App è molto semplice. Progredendo però verso applicazioni più complesse, ci si scontrerà con la necessità di controllo dei parametri funzionali del sistema. Windows 10 IoT Core offre come predefinita un'interfaccia web di gestione, questi i parametri raggiungibili all'indirizzo web [http://](http://minwinpc.local:8080)

minwinpc.local:8080.

L'indirizzo è dato dal nome predefinito assegnato alla scheda; ovviamente è possibile modificarlo e nel caso cambia anche l'url, alternativamente è possibile usare l'IP address al posto di *minwinpc.local*.

Le credenziali predefinite per accedere sono "Administrator" con password "p@ssw0rd" (che ovviamente è bene cambiare se si

Fig. 9



per il MATERIALE

Tutto il materiale necessario per realizzare questo progetto è disponibile presso Futura Elettronica. La board Raspberry Pi 2 Tipo B (cod. 8326274RS) è in vendita al prezzo di 44,00 Euro. Questa board è anche disponibile in starter kit: starter kit Raspberry PI 2 modello B (cod. RASPKITV4) a 72,00 Euro.

Il materiale va richiesto a:
Futura Elettronica, Via Adige 11,
21013 Gallarate (VA)
Tel: 0331-799775 • Fax: 0331-792287
<http://www.futurashop.it>

Listato 2

```
<Page
  x:Class="App2.MainPage"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:local="using:App2"
  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
  mc:Ignorable="d">

  <Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}" >
    <StackPanel HorizontalAlignment="Center" VerticalAlignment="Center">
      <TextBlock Foreground="Blue" HorizontalAlignment="Center"
        FontSize="48">LED Control</TextBlock>
      <Button Click="Button_Click" HorizontalAlignment="Center" Margin="0,50,0,0" FontSize="64" >OFF</Button>
    </StackPanel>
  </Grid>
</Page>
```

utilizza la scheda al di fuori del laboratorio).

L'interfaccia è molto semplice, ma non per questo meno efficace: permette di vedere la lista dei processi in esecuzione, analizzare il carico CPU ed il consumo RAM in realtime, eseguire o stoppare le App e tante altre interessanti funzioni (Fig. 11).

Una funzionalità interessante è quella che permette di fare il deploy di applicazioni senza passare da Visual Studio, che diventa di gran aiuto quando la scheda non è collegata all'ambiente di sviluppo.

CONCLUSIONI

Nonostante una visione superficiale di Windows 10 IoT Core si percepiscono delle potenzialità che definiremmo entusiasmanti. La prima cosa da sottolineare è

che quando parliamo di Universal App parliamo di applicazioni che possono essere eseguite virtualmente su tutti i dispositivi supportati da Windows 10. I semplici esempi realizzati nell'articolo possono essere eseguiti, per esempio, su un telefono Windows Phone, ovviamente con gli opportuni controlli per l'utilizzo di hardware specifico come i GPIO. Rimanendo nell'ambito delle embedded board non c'è solo la Raspberry Pi ma per esempio troviamo la Intel Galileo, anche in questo caso la nostra App può essere eseguita su piattaforme diverse con modifiche minime o in certi casi nulle.

Un altro fattore distintivo è l'efficienza di sviluppo data dall'uso di uno strumento professionale come Visual Studio e dal ciclo di lavoro che permette sviluppo,

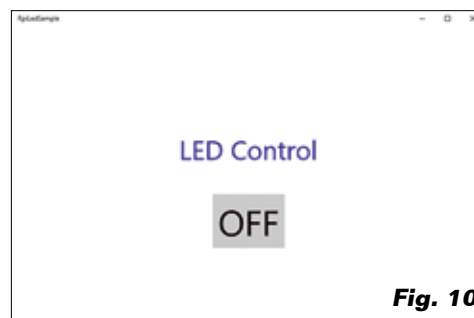


Fig. 10

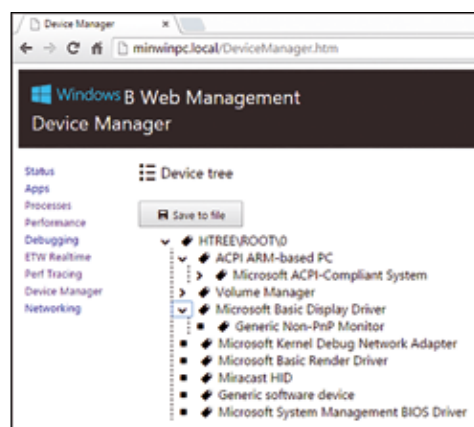


Fig. 11

debug e deploy totalmente integrato. Non abbiamo affrontato le possibilità offerte dall'interazione con il cloud di Microsoft, Azure, che permette di scalare le applicazioni da un livello hobbistico ad uno professionale mantenendo la stessa semplicità di lavoro. Per concludere siamo certi che Windows IoT Core si ritaglierà presto uno spazio importante nell'ormai affollato mondo dei tool usati dalla comunità dei Maker, non possiamo che apprezzare gli sforzi che Microsoft sta facendo nell'offrire un bagaglio di competenze e tecnologie altrimenti inarrivabili.

Listato 3

```
bool state = false;
const int LED_PIN = 18;
GpioPin pin;

private void Button_Click(object sender, RoutedEventArgs e)
{
  state = !state;

  var pinValue = state ? GpioPinValue.High : GpioPinValue.Low;
  pin.Write(pinValue);

  Button btn = (Button)sender;
  btn.Content = state ? "ON" : "OFF";
}
```

Listato 4

```
private void InitGPIO()
{
  var gpio = GpioController.GetDefault();

  pin = gpio.OpenPin(LED_PIN);

  pin.SetDriveMode(GpioPinDriveMode.Output);
  pin.Write(GpioPinValue.Low);
}
```




DOMOTICA OPEN SOURCE con i moduli UDOiT

Facciamo conoscenza con il telecomando e l'attuatore relè della soluzione di domotica Open Source di UDOiT basata sui moduli RFTide ed introduciamo l'utilizzo delle "Scene".

di MARCO MAGAGNIN

Una delle principali funzioni che ci si aspetta da un sistema di automazione domestica è la possibilità di attivare delle "utenze", siano esse semplici lampade, elettrodomestici od ulteriori sistemi di gestione dell'abitazione, come antifurti, controllo accessi, luci notturne, irrigazione del giardino e dell'orto. La collezione UDOiT prevede una serie di attuatori, alcuni disponibili ed altri in via di sviluppo oltre ad un modulo telecomando che permette di svincolare alcuni comandi dall'interfaccia web.

ATTUATORE RELÉ

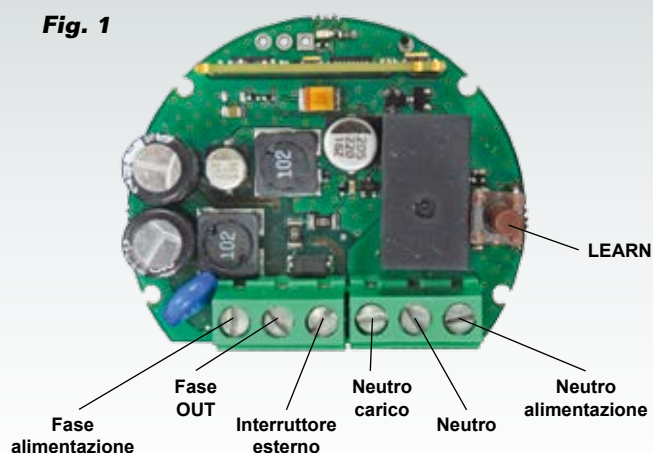
Il primo attuatore disponibile è il dispositivo "Relè", che permette di attivare carichi fino a 10 A con tensione fino a 220 V. A questo proposito è bene precisare che quando si decide di installare un impianto

domotico la parte in bassa tensione non è soggetta a certificazione mentre lo è tutta la parte di impianto che in qualche modo si interfaccia alla rete a 220 V, che pertanto deve essere installata e certificata da un installatore autorizzato.

Torniamo al dispositivo relè, visibile nella foto di **Fig. 1**, che dispone di un contenitore esterno disegnato in modo da poter essere installato in una normale cassetta da incasso per impianti elettrici di tipo 503E. A tale scopo per questo dispositivo è stato realizzato un modulo a radiofrequenza miniaturizzato, e tutta la realizzazione è anch'essa particolarmente compatta.

Sempre nella **Fig. 1** sono visibili le posizioni dei connettori e del tasto di learning presenti sul dispositivo, in particolare:

Fig. 1



- Un microinterruttore da utilizzare per il processo di "Learning";
- Un connettore con morsetti a vite che riportano le uscite del relè;
- Un connettore con morsetti a vite per alimentare il modulo dalla rete a 220 V;
- Un connettore con morsetti a vite per collegare un pulsante di comando esterno (questo connettore è stato predisposto per usi futuri, al momento il firmware non implementa questa funzionalità).

Il firmware del modulo implementa i comandi di gestione del dispositivo che utilizzano il protocollo RFTide standard e quindi il dispositivo relè si inserisce perfettamente nell'architettura UDOiT che vi abbiamo presentato negli articoli precedenti. Sono previste diverse possibilità di comando e gestione del modulo. Il comando di attivazione o disattivazione del relè, prevede un pacchetto di risposta con lo stato del relè. Il pacchetto di comando prevede la configurazione visibile in Fig. 2

All'invio di un pacchetto di attivazione o disattivazione del relè fa seguito un pacchetto di risposta da parte del modulo con il contenuto visibile in Fig. 3.

Nel caso il relè venga attivato mediante l'interruttore esterno, che funziona in modo bistabile, ovvero con l'interruttore collegato alla fase il relè è attivato mentre lasciato "floating" il relè è disattivato, l'even-

Comando	Indirizzo	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
0x00	0x00 Indirizzo in esadecimale	0x30	0x02	On/Off 0x01/0x00	0x00	0x00	0x00	0x00	0x00

Fig. 2

Comando	Indirizzo	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
0x00	0x00 Indirizzo in esadecimale	0x30	0x01	On/Off 0x01/0x00	0x00	0x00	0x00	0x00	Vers. Firmware

Fig. 3

to viene comunicato alla rete RFTide con l'invio di un pacchetto nello stesso formato del pacchetto di risposta visibile in Fig. 3, che riporta lo stato finale del dispositivo gestito dall'applicazione di controllo ospitata dal gateway. Per aggiungere alla rete un dispositivo attuatore Relè si utilizza la funzione "Add Device" della applicazione di gestione, che si trova nella pagina "Settings", come visibile in Fig. 4. Alla pagina "Settings", come già descritto nell'articolo precedente sul numero 198 della rivista, viene richiesta un'autenticazione con userid e password.

Nella Fig. 5 si nota che il dispositivo è caratterizzato da un nome, di lunghezza massima quattro caratteri, una descrizione, che apparirà nelle pagine web di gestione dei dispositivi della rete, un indirizzo, che deve essere univoco nell'intero sistema, ed il tipo di dispositivo, nel nostro caso "Relè".

Abbiamo poi la possibilità di scegliere se vedere o meno l'entrata del dispositivo nella pagina web di gestione dell'applicazione, cosa abbastanza ovvia per un dispositivo di attuazione carichi e, in questo caso scegliere una posizione nell'elenco (campo "Order").

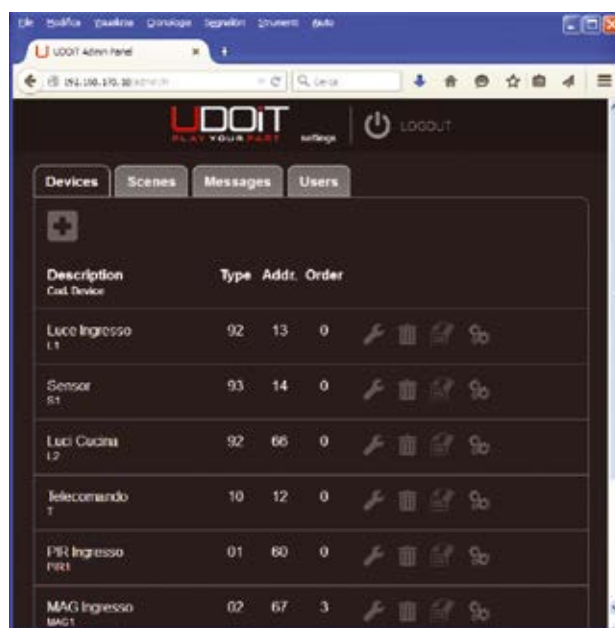


Fig. 4

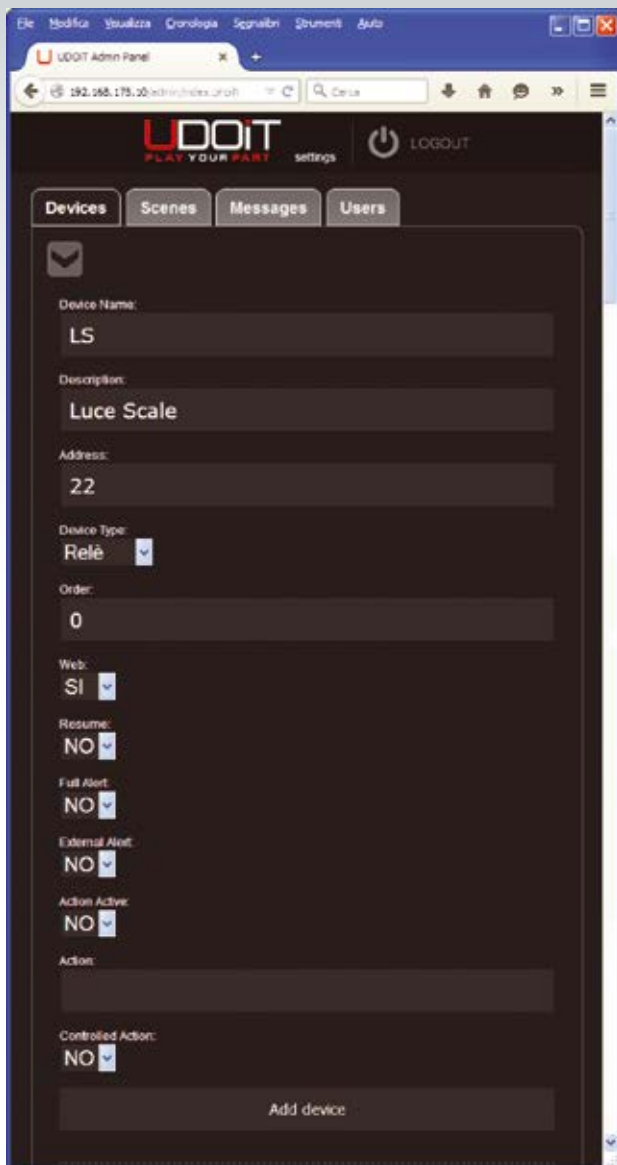


Fig. 5

I campi successivi servono per fare entrare o meno il dispositivo nella rete di controllo e sicurezza e per pilotare, in caso di attivazione, l'esecuzione di un processo personalizzato. Queste ultime funzionalità sono un po' più complesse da configurare e personalizzare e le affronteremo in seguito in un articolo dedicato. Una volta configurata l'anagrafica del dispositivo e salvata con il pulsante "Add device" possiamo vedere in **Fig. 6** come si presenta sulla pagina web: un'entrata di una riga con il nome ed il pulsante per l'attivazione e disattivazione del relè. Il codice colore e la dicitura ci informano anche sullo stato attuale del relè stesso. Per cambiare lo stato del relè si "clicca" sul pulsante di stato corrispondente al dispositivo o su un dispositivo mobile si "tappa" sullo stesso. Questo però non prima di avere eseguito l'operazione di "learning" del dispositivo stesso. L'operazione di learning è necessaria per associare alla rete il dispositivo appena

descritto mediante un processo di apprendimento, durante il quale il dispositivo viene personalizzato con il Network ID di appartenenza (quello del Dongle, Master della rete) e l'indirizzo di identificazione del dispositivo stesso, una sorta di cognome e nome che permetteranno di riconoscere univocamente il dispositivo nella rete e impediranno ai dispositivi presenti su reti adiacenti di interferire.

Il processo di apprendimento avviene come segue :

- Nella pagina "Devices" del pannello "Settings" si "prenota" l'operazione di apprendimento cliccando sull'icona a forma di foglio di carta e matita ("Learn Device" in corrispondenza del dispositivo per il quale si vuole eseguire l'associazione **Fig. 7**);
- A questo punto si preme il pulsante presente sul circuito stampato per circa cinque secondi, al rilascio si accende un LED rosso;
- Entro 15 secondi cliccare sul pulsante "OK" della finestra "Pop-Up" sulla pagina web;
- Il LED lampeggia per qualche secondo;
- Il dispositivo è in grado di essere riconosciuto ed operare sulla rete;
- In caso non venga avviata la procedura di apprendimento il dispositivo esce dallo stato di apprendimento dopo circa cinque secondi.

Se è stata scelta l'opzione WEB "SI" lo stato del dispositivo sarà visibile nella pagina di gestione web del sistema (**Fig. 6**).

TELECOMANDO

Il telecomando è un dispositivo particolare, che permette di eseguire delle azioni sugli attuatori. Ma solo per il tramite del gateway e dell'applicazione di gestione, che deve essere configurata in modo opportuno. Il telecomando, come visibile in **Fig. 8** dispone di quattro tasti di comando ai quali possono essere associate quattro diverse funzioni, un LED di segna-

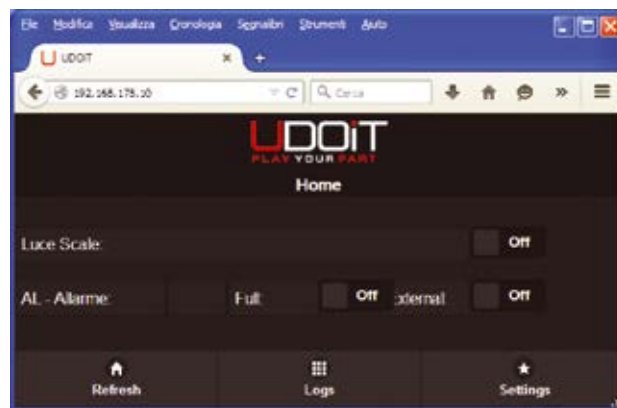


Fig. 6

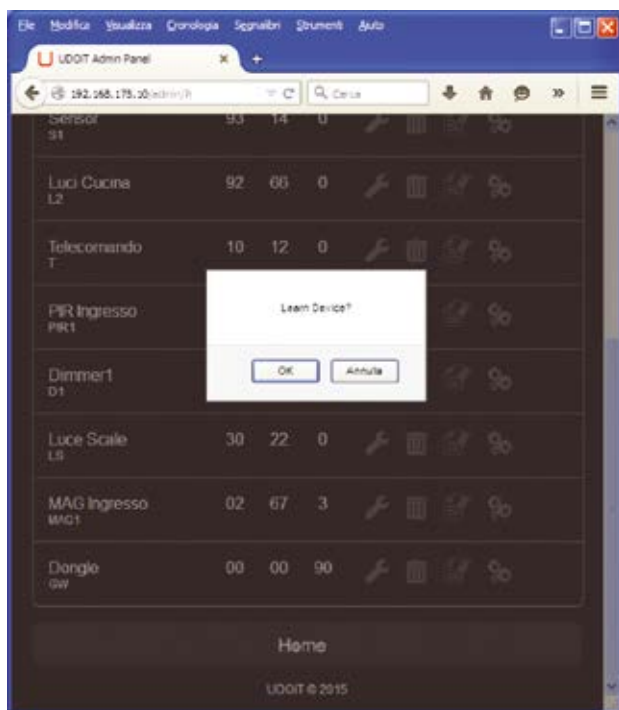


Fig. 7

zione ed è alimentato da una batteria a bottone da 3V. Lo schema di funzionamento complessivo è visibile in Fig. 9. Ovviamente anche il modulo RF del telecomando utilizza il protocollo di comunicazione RFTide con i contenuti visibili in Fig. 10.

A parte i soliti dati di riconoscimento del dispositivo, tipo di device nel byte 0 ed indirizzo nel byte indirizzo sono presenti due importanti informazioni, il numero del tasto premuto nel byte 1 e il codice univoco di identificazione nei byte 5 e 4 espressi secondo la convenzione Big-Endian. Il codice univoco di identificazione viene assegnato in fabbrica e serve a "certificare" la provenienza dei comandi inviati. Facciamo

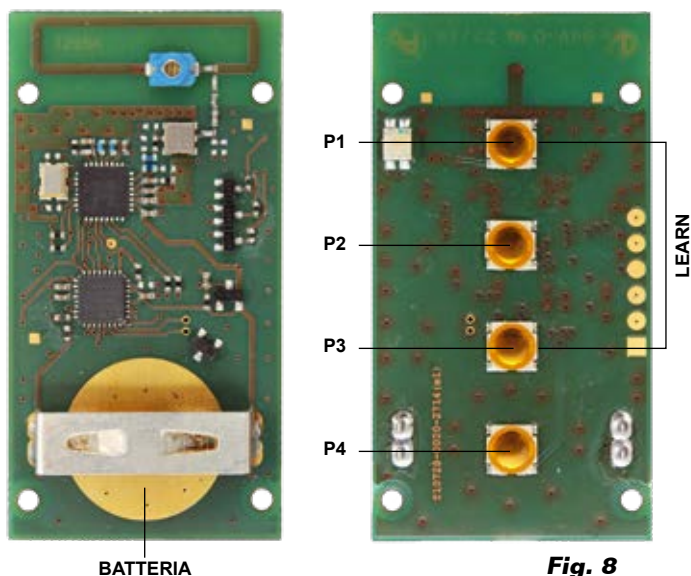
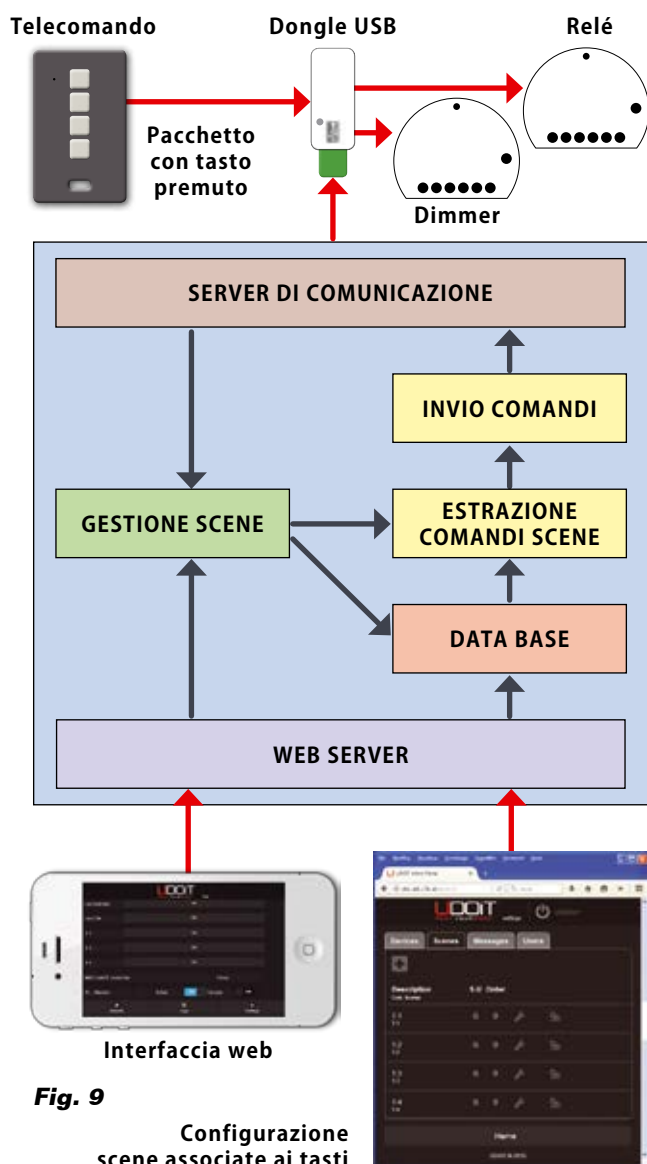


Fig. 8

il caso di un telecomando correttamente associato all'applicazione con il proprio indirizzo. In caso di "smarrimento" del telecomando la soluzione più logica sembra essere quella di procurarsi un nuovo telecomando ed eseguire su quest'ultimo l'operazione di learning mantenendo l'indirizzo e le configurazioni associate al vecchio telecomando. Questa soluzione ha una controindicazione da non sottovalutare; chiunque ritrovi il vecchio telecomando potrà utilizzarlo per impartire i comandi esattamente come con quello nuovo, con gravi controindicazioni in termini di sicurezza. L'alternativa di cancellare il vecchio telecomando e configurare quello nuovo con un indirizzo diverso costringe ad un grosso lavoro di riconfigurazione. La soluzione adottata è quella di "personalizzare" ciascun telecomando con un proprio codice identificativo e di verificare ad ogni comando che questo provenga dal telecomando desiderato. In questo modo i comandi provenienti dal vecchio telecomando saranno semplicemente ignorati. Nel seguito descriveremo il processo per fare in modo che l'applicazione riconosca il telecomando corretto. Tornando allo schema di Fig. 9 notiamo che quando viene premuto uno dei tasti viene inviato un pacchetto contenente il numero del tasto premuto. Il pacchetto viene letto e recepito dal Dongle, master della rete, che lo consegna all'applicazione di gestione. In un'apposita tabella dell'applicazione di gestione sono configurate delle "scene". Nel mondo della domotica, ciascuna "scena", o "scenario", è composto da una lista di comandi elementari che devono essere eseguiti in un unico processo, come per esempio accendere tre lampade o spegnere tutte le luci e attivare l'antifurto prima di uscire. Di solito, come anche nel caso dell'applicazione di gestione della soluzione UDOiT si possono inserire quante scene si vogliono e configurarle in modo molto flessibile per gestire gli attuatori presenti. Nel caso del telecomando, quando viene aggiunto all'applicazione, il processo di aggiornamento genera automaticamente quattro scene, una per ogni tasto disponibile, che possono poi successivamente essere personalizzate in modo da eseguire i comandi desiderati. Una volta ricevuto il pacchetto inviato dal telecomando, l'applicazione lo elabora in modo da "lanciare" la scena corrispondente al tasto premuto. Il lancio di una scena consiste nel reperire la lista dei comandi che costituiscono la scena stessa ed inviarli in sequenza nella rete per mezzo del "Dongle", master della rete. Per aggiungere un telecomando alla rete si utilizza sempre la funzione "Add Device" dell'applicazione di gestione, che si trova nella pagina "Settings". In questo caso si dovrà scegliere come tipo dispositivo "Remote" (Fig.11). Ovviamente scegliamo di non visualizzare il dispositivo sulla pagi-



na web. Si consiglia di utilizzare un nome corto per il telecomando in quanto lo stesso nome verrà utilizzato come prefisso per i nomi delle scene. Un nome di una sola lettera va benissimo: se assegnamo al telecomando il nome "T" le scene associate assumeranno i nomi da T-1 a T-4. Andando alla pagina "Scenes" della funzione "Settings", come nella Fig. 12, possiamo vedere le quattro scene generate automaticamente ed associate ai tasti del telecomando. Il processo di learning del telecomando include anche la memorizzazione del codice identificativo. La prima parte del processo è identica a quella degli altri dispositivi. Si prenota la funzione di learning per il teleco-

mando. Per portare il telecomando in stato di learning si devono premere contemporaneamente i tasti 1 e 3 per qualche secondo finché il LED da blu diventa rosso. Cliccando su OK vedremo il LED lampeggiare per qualche attimo, al termine l'operazione di learning è conclusa. Il codice identificativo viene assegnato con il pacchetto di risposta al comando di learning, un pacchetto con la configurazione di Fig. 10 ma con il numero tasto (byte 1) uguale a zero. Per associare un

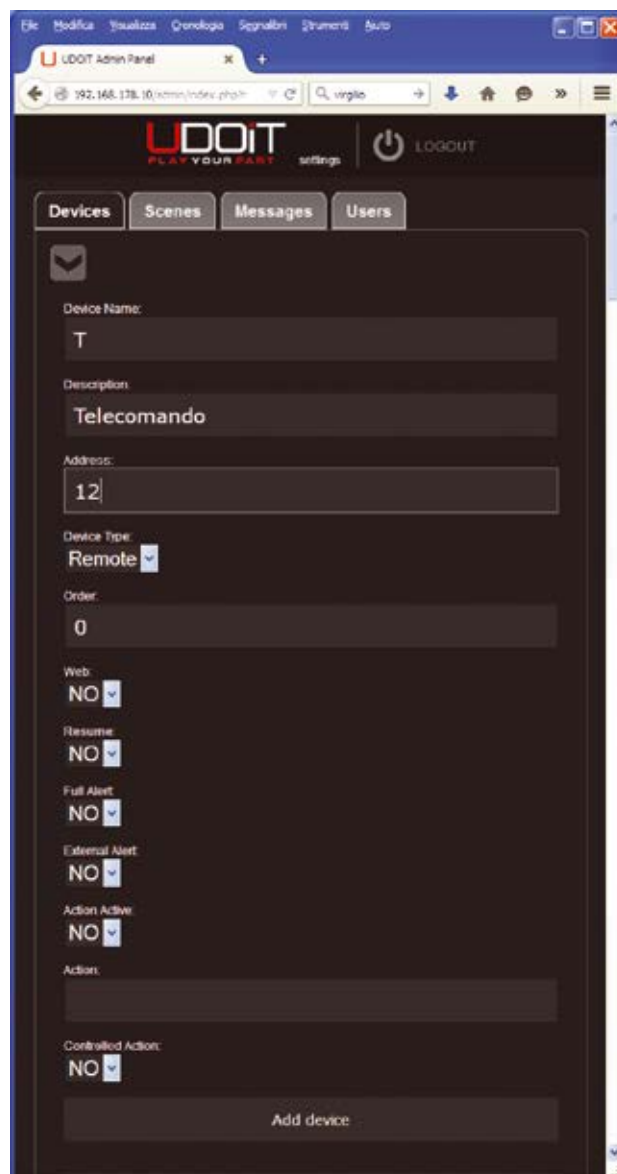


Fig. 11

Comando	Indirizzo	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
0x00	0x00	Indirizzo in esadecimale	0x10	Tasto Premuto 0x01/0x04	Livello batteria	Livello batteria	SerialID LSB	SerialID MSB	Vers. Firmware
									0x00

Fig. 10



Fig. 12

diverso telecomando allo stesso indirizzo, come nel caso di smarrimento o furto, basta ripetere l'operazione descritta. Un nuovo processo di learning disassocia dall'applicazione l'identificativo seriale del vecchio telecomando ed identificativo. Il vecchio telecomando non avrà più effetti sull'applicazione. Ora possiamo configurare le scene in modo da personalizzare ciascun tasto del telecomando per fargli eseguire i comandi desiderati. Prima, però è opportuno approfondire il processo di configurazione ed il funzionamento delle scene.

LE SCENE

Nei sistemi di domotica e di automazione in generale per "scene" o "scenari" si intende la creazione di sequenze di operazioni preconfigurate che possono essere innescate ed eseguite a fronte di un evento, come l'attivazione di un sensore, o di una specifica richiesta dell'operatore, come la pressione di un tasto del telecomando. Dal punto di vista operativo le scene vengono configurate utilizzando la funzione "Scenes", anch'essa raggiungibile dalla pagina "Settings". Prima però è necessaria una premessa per descrivere con quali "mattoni" si costruiscono le sequenze di comandi che costituiscono le scene. Torniamo alla descrizione dei dispositivi che entrano a far parte della rete. Come abbiamo visto in precedenza il protocollo di comunicazione dei dispositivi "attuatori" prevede dei comandi in grado di variare lo stato degli attuatori stessi, nel caso del Relè, la possibilità di attivarlo e disattivarlo. Questi "comandi" sono gestiti dall'applicazione di controllo. Per facilitare la gestione

da parte dell'utente, a ciascun "comando", in fase di descrizione dell'attuatore, viene associato un nome mnemonico, che ne faciliti l'identificazione e che viene memorizzato, assieme al comando corrispondente, in una delle tabelle del database dell'applicazione. Per esempio se identifichiamo un attuatore relè con il nome "REL1", in fase di salvataggio (o di aggiornamento, nel caso eseguiamo delle modifiche ad un attuatore esistente), verranno create due entrate nella tabella dei comandi; uno con nome REL1_ON al quale verrà associato il comando di attivazione del relè ed uno con nome REL1_OFF al quale verrà associato il comando di disattivazione del relè. Fatta questa premessa, andando alla pagina "Settings" e cliccando sulla tab "Scenes" si apre la funzione di gestione delle scene. Se abbiamo già configurato un telecomando, come nel nostro caso, vedremo quattro scene già presenti; quelle che sono state generate automaticamente quando abbiamo descritto il dispositivo telecomando. Per aggiungere una scena indipendente dal telecomando si deve cliccare sull'icona a forma di "+". Le scene associate ai tasti del telecomando possono essere attivate sia utilizzando i tasti del telecomando che dall'interfaccia web dell'applicazione. Tutte le scene possono anche essere attivate mediante script che a loro volta possono essere schedulati a tempo, associati all'attivazione di sensori o utilizzati come ponte di comunicazione con applicazioni esterne. Questo è un argomento avanzato che non affrontiamo al momento



Fig. 13



Fig. 14

ma tratteremo in seguito. Nella pagina di sintesi che espone la lista delle scene, visibile in Fig. 13, ciascuna scena è identificata dal nome e dall'ordine nel quale viene presentata nella pagina web di gestione dell'applicazione. Le icone a fianco di ciascuna entrata, permettono la "manutenzione" della scena.

In particolare la prima icona permette di modificare la descrizione ed il comportamento della scena, la seconda permette di comporre la lista di comandi da associare alla scena e la terza permette di eliminare la scena dal sistema. Si può notare che l'icona di eliminazione non è disponibile per le scene associate ai tasti del telecomando. Queste e le altre scene associate a funzioni di "sistema" sono create e cancellate in automatico e non possono essere eliminate dall'utilizzatore. Per esempio quando si elimina l'entrata associata ad un telecomando nella pagina "Devices" vengono cancellate anche le quattro scene associate a quest'ultimo. Per aggiungere una scena nel sistema si utilizza la funzione "Add Scene" (Fig. 14) dell'applicazione di gestione.

Nella stessa figura si vede che la scena è caratterizzata da un nome, di lunghezza massima quattro caratteri, una descrizione, che apparirà nelle pagine web di gestione del sistema. Abbiamo poi la possibilità di scegliere se vedere o meno l'entrata del dispositivo nella pagina web di gestione dell'applicazione

e in questo caso scegliere una posizione nell'elenco (campo "Order"). Scegliamo l'opzione "SI". I campi successivi servono per pilotare, in caso di attivazione, l'esecuzione di un processo personalizzato. Queste ultime funzionalità sono più complesse da configurare e personalizzare e le affronteremo in seguito in un articolo dedicato.

Per "salvare" la scena nel sistema clicchiamo sul pulsante "Add Scene". Una volta inserita una nuova scena è possibile associare la lista dei comandi che devono essere eseguiti quando viene attivata. Dalla finestra "Scenes" clicchiamo sull'icona a forma di tre ingranaggi (Configure Scene) in corrispondenza della scena che vogliamo configurare. In alto si apre la scheda visibile in Fig. 15. Nella riga in basso possiamo scegliere quale comando associare utilizzando il campo a tendina che presenta la lista di tutti i comandi disponibili nel sistema. Il primo campo permette di stabilire la posizione che vogliamo assegnare al comando all'interno della lista. L'ultimo campo permette di introdurre un ritardo tra l'esecuzione di un comando ed il successivo. Dato un comando, il ritardo viene applicato prima dell'esecuzione del comando. Per chiarire, se abbiamo una lista di due comandi e vogliamo introdurre un ritardo tra i due, dobbiamo specificare il valore del ritardo nel secondo comando. Questo dato è espresso in millesimi di secondo, ad esempio 2000 indica due secondi. I valori sono incrementali, quindi per aggiungere un ulteriore ritardo di due secondi



Fig. 15



Fig. 16

tra il secondo ed il terzo comando bisogna indicare il valore 4000. Una volta definito un comando lo aggiungiamo alla lista cliccando sull'icona a forma di "+" (Fig. 16). Se vogliamo modificare un comando già inserito nella lista lo facciamo direttamente sull'entrata della lista e lo confermiamo cliccando sull'icona a forma di chiave inglese.

Per cancellare un comando della lista clicchiamo sull'icona a forma di bidone della spazzatura in corrispondenza del comando da eliminare. Per confermare il tutto clicchiamo sull'icona in alto a forma di freccia orientata verso il basso.

Con queste funzionalità creiamo due scene, una con il comando che attiva il relè ed una con il comando che lo disattiva. In alternativa, se abbiamo un telecomando descritto nel sistema, possiamo configurare le scene associate ai pulsanti. Andiamo ora alla pagina principale di gestione del sistema, dove troviamo i pulsanti associati alle scene create, come visibile in Fig. 17. Se clicchiamo sui pulsanti associati alle scene vedremo attivarsi e disattivarsi il relè associato. Se le scene corrispondono ai tasti del telecomando otterremo lo stesso effetto premendo i corrispondenti tasti del telecomando. Ovviamente il tutto funziona in modo disgiunto. Possiamo attivare il relè premendo un tasto del telecomando e disattivarlo dall'interfaccia web.

FUNZIONE ALLARME

La gestione degli allarmi è un primo esempio preconfezionato della integrazione possibile tra sensori, attuatori, scene e script di controllo personalizzati. Lo

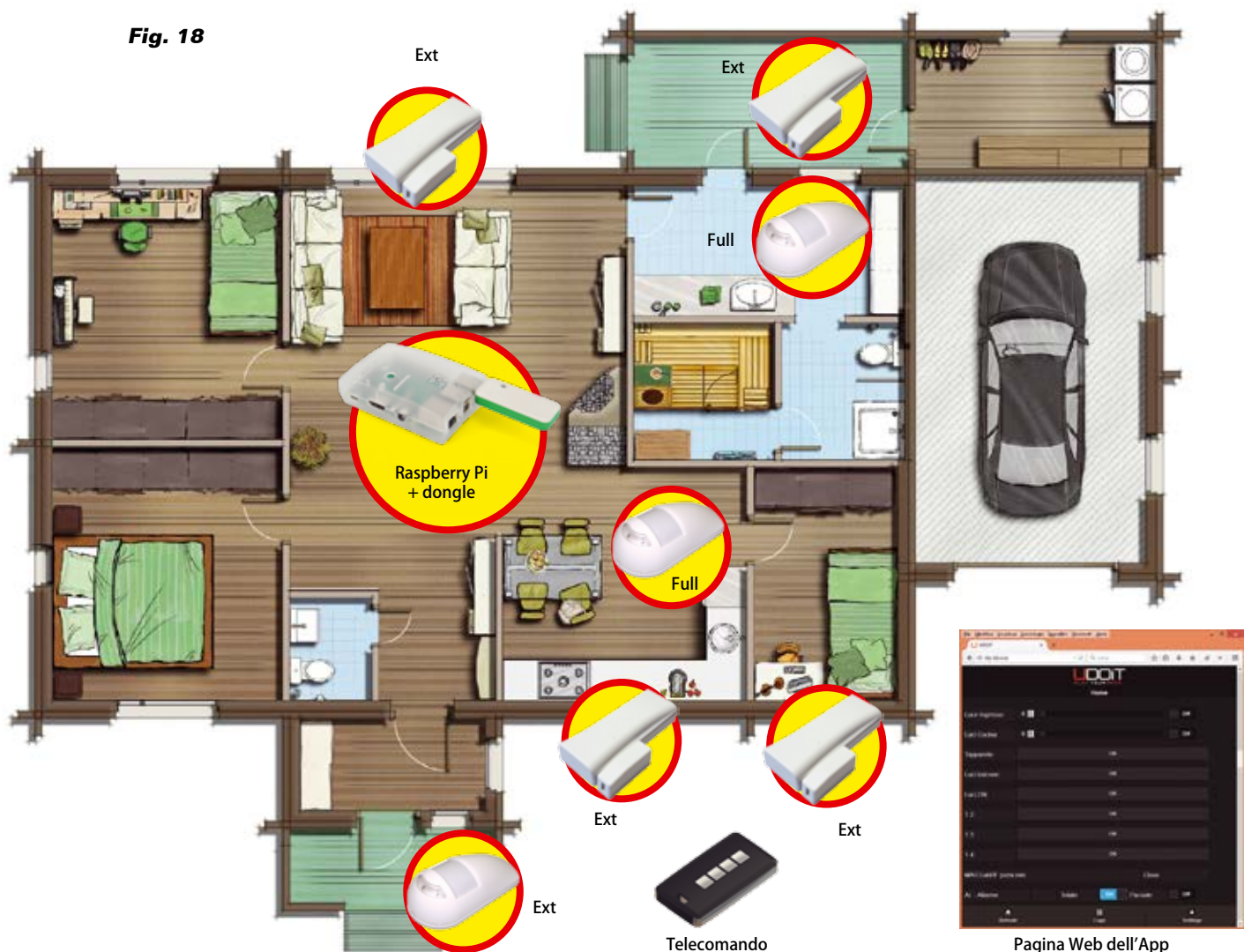
scopo è quello di proteggere o comunque di tenere sotto controllo un'ipotetica abitazione con una rete di sensori opportunamente predisposti. Questa funzionalità preconfezionata prevede due livelli di "controllo", un livello di controllo periferico, o "esterno", al quale possiamo associare i sensori collegati a porte e finestre esterne in modo da intercettare eventuali entrate ed uscite dall'abitazione ed un livello di controllo "totale" al quale possiamo associare anche sensori in grado di intercettare eventuali movimenti all'interno dell'abitazione. Come nell'esempio visibile in Fig. 18 possiamo prevedere dei sensori magnetici per rilevare l'apertura di porte e finestre esterne e dei sensori PIR per rilevare movimenti all'interno dell'abitazione. Come visibile in Fig. 19 l'interfaccia grafica prevede un'entrata predefinita per l'attivazione e disattivazione delle funzionalità di controllo totale ed esterno. Si attiva o disattiva ciascuna funzionalità cliccando sul corrispondente cursore ON/OFF. Nel caso sia attiva la funzionalità di controllo esterno e si decide di attivare il controllo totale, automaticamente verrà disattivata la funzionalità di controllo esterno. Lo stesso comportamento nel caso opposto.

Come abbiamo già descritto per il telecomando, anche alle funzionalità di controllo sono associate delle scene predefinite che vengono innescate quando i sensori si attivano. In particolare a ciascuna funzionalità sono



Fig. 17

Fig. 18



associate due scene. La logica dell'applicazione prevede che quando uno dei sensori associati alla funzionalità si attiva viene eseguita la prima scena associata alla funzionalità quella che ha suffisso -ON nel nome, ovvero ALT-ON per il controllo totale e ALP-ON per il controllo interno. Successivamente, sempre la logica predefinita dell'applicazione prevede che dopo un intervallo di un minuto vengono attivate le corrispondenti scene con il suffisso -OFF nel nome, rispettivamente ALT-OFF e ALP-OFF (Fig. 20). In questo modo, per esempio, controllando uno o più attuatori Relè, è possibile attivare una sirena per un minuto, oppure innescare l'ingresso ausiliario di un secondo sistema antifurto o ancora accendere luci.

Utilizzando poi le funzionalità avanzate di personalizzazione mediante script, che descriveremo nei prossimi articoli, è possibile inviare messaggi mail, o via applicazioni social, o interagire con applicazione esterne, sia locali che disponibili per mezzo della rete internet. Per attivare la struttura di controllo è necessario utilizzare la pagina "Devices" della funzione "Settings".

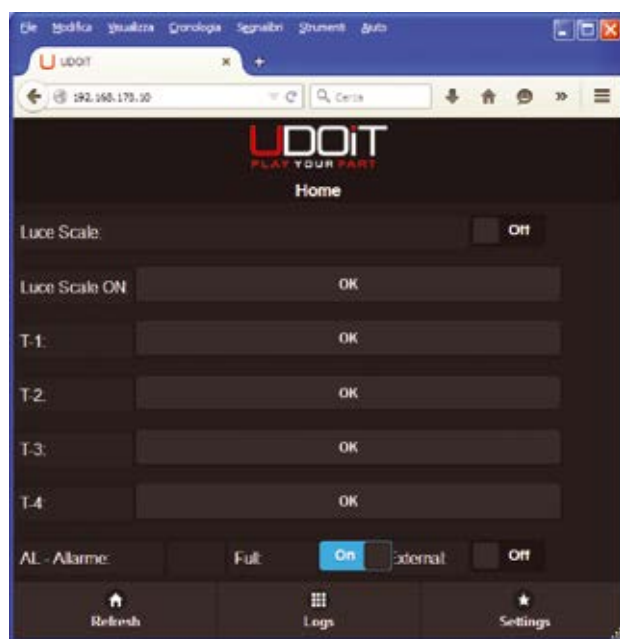


Fig. 19



Fig. 20

Per ciascuno dei sensori che si vuole far partecipare alla struttura di controllo è necessario aprire la funzione di update del sensore stesso, cliccando sull'icona a forma di chiave inglese, in corrispondenza del sensore desiderato ed impostare a "SI" uno o entrambi i campi "Total alert" e "External alert". Questa operazione deve essere eseguita per tutti i sensori che si vogliono far partecipare alla rete di controllo. Successivamente è necessario configurare le scene associate alle funzioni attivate, per esempio per attivare e disattivare uno o più relè, secondo le modalità descritte nel paragrafo precedente dedicato alle scene (Fig. 21 e Fig. 22).

Fig. 21

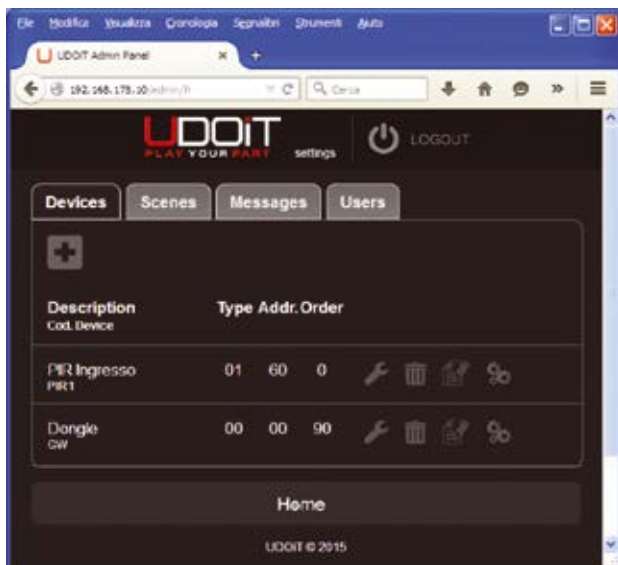


Fig. 22



per il MATERIALE

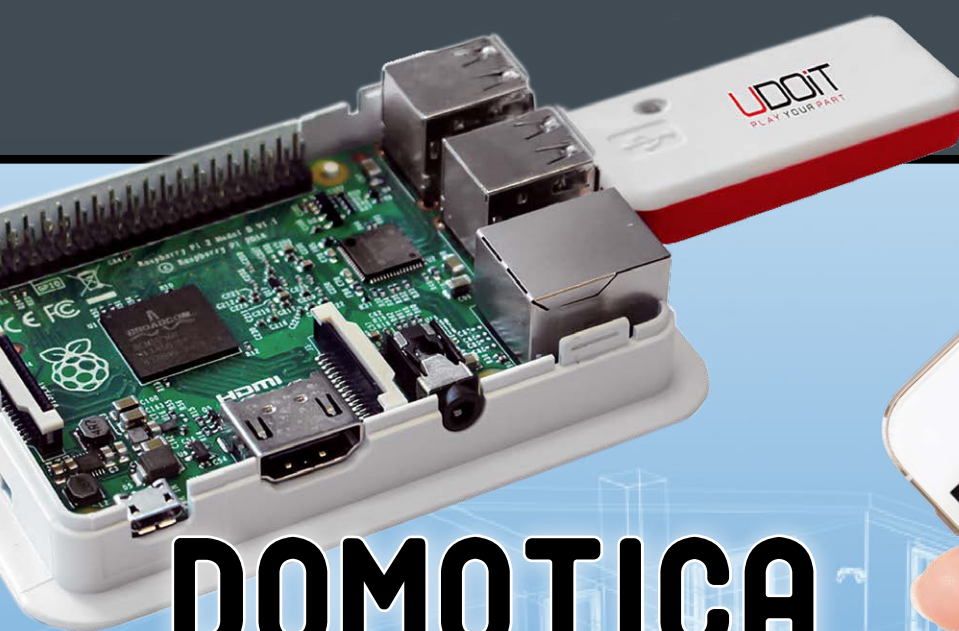
I prodotti presentati in questo progetto possono essere acquistati presso Futura Elettronica. Il dongle USB (cod. RFT868USB) costa 39,00 Euro, il sensore magnetico (cod. RFT868MAG) è disponibile a 49,00 Euro, il sensore di movimento (cod. RFT868PIR) è in vendita a 55,00 Euro, il modulo a relé (cod. RFT868REL) costa 55,00 Euro e il modulo a dimmer (cod. RFT868DIM) è disponibile a 55,00 Euro. Il radiocomando a 4 canali (cod. RFT8684CH) è in vendita a 39,00 Euro, e la Raspberry Pi 2 Tipo B (cod. 8326274RS) a 44,00 Euro.

Il materiale va richiesto a:

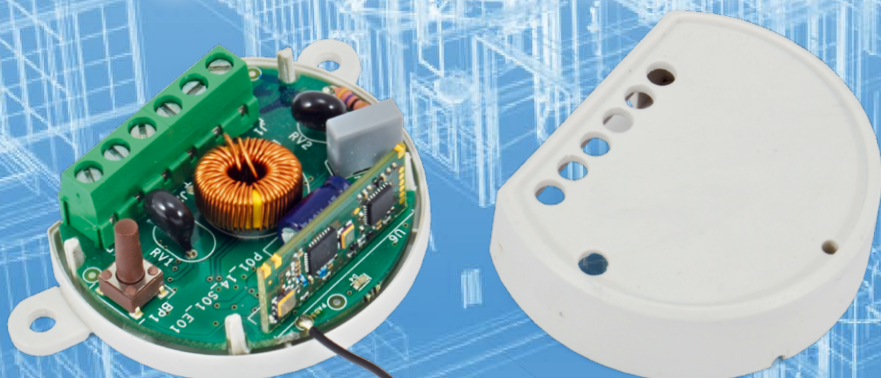
Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)

Tel: 0331-799775 - Fax: 0331-792287

<http://www.futurashop.it>



DOMOTICA OPEN SOURCE con i moduli UDOiT



Presentiamo il dimmer della serie di attuatori e vediamo come installare l'applicazione necessaria a dare vita a tutto il sistema.

di MARCO MAGAGNIN

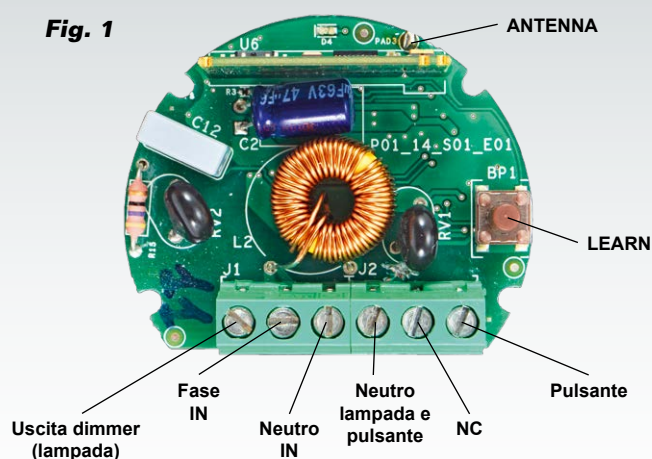
Una delle funzionalità maggiormente apprezzate in un sistema di automazione domestica è la possibilità di gestire il livello di luce emessa dai dispositivi di illuminazione. Gli attuatori relè possono unicamente accendere o spegnere le lampade collegate, mentre per regolare la luminosità è necessario un attuttore differente, in grado di "parzializzare" l'alimentazione verso la lampada, regolando l'intensità della luce emessa. Per curiosità il termine dimmer viene dall'inglese "to dim" che significa "abbassare". Oltre all'attuttore dimmer, in questo articolo descriviamo le modalità di installazione e di primo utilizzo dell'applicazione di gestione su Raspberry Pi.

ATTUATORE DIMMER

Il dispositivo "Dimmer", permette di controllare carichi resistivi, quindi lampade ad incandescen-

za, con assorbimento fino a 200W e tensione di 220V. A questo proposito è sempre bene precisare che quando si decide di installare un impianto di automazione domestica, è necessario rispettare le normative in fatto di certificazione e sicurezza, la parte di impianto alimentato in bassa tensione non è soggetta a certificazione mentre lo è tutta la parte di impianto che in qualche modo si interfaccia alla rete a 220 V, che pertanto deve essere installata e certificata da un installatore autorizzato. Torniamo al dispositivo dimmer, visibile nella foto di **Fig. 1**, che dispone di un contenitore esterno disegnato in modo da poter essere installato in una normale cassetta da incasso per impianti elettrici di tipo 503E. A tale scopo per questo dispositivo è stato realizzato un modulo a radiofrequenza miniaturizzato, e tutta la realizzazione è anch'essa particolarmente com-

Fig. 1



patto. Sempre nella **Fig. 1** sono visibili le posizioni dei connettori e del tasto di learning presenti sul dispositivo, in particolare:

- Un microinterruttore da utilizzare per il processo di "Learning";
- Un connettore con morsetti a vite che riporta le uscite del dimmer;
- Un connettore con morsetti a vite per alimentare il modulo dalla rete a 220 V;
- Un connettore con morsetti a vite per collegare un pulsante di comando esterno.

Il firmware del dispositivo dimmer implementa i comandi di gestione del modulo che utilizzano il protocollo RFTide standard e quindi il dispositivo dimmer si inserisce perfettamente nella architettura UDOiT che vi abbiamo presentato negli articoli precedenti.

Il comando di accensione o spegnimento della "lampada", prevede un pacchetto di risposta con lo stato del dispositivo. Il pacchetto di comando prevede la configurazione visibile in **Fig. 2**. Diversamente dal dispositivo relè il pacchetto di risposta

ai comandi di gestione prevedono, nel byte 3 , il ritorno del livello di dimmer.

Questo livello corrisponde all'intensità con la quale si vuole che venga accesa la lampada a fronte di un comando di accensione. Lo stesso livello di intensità viene utilizzato anche quando si accende la lampada utilizzando il pulsante esterno.

Il pulsante esterno prevede diverse modalità di impiego. L'impiego normale è di accendere e spegnere la lampada collegata. Se la lampada è accesa una breve pressione del pulsante la spegne, oppure, se spenta, la accende. Ovviamente questo comportamento è integrato con l'utilizzo dell'interfaccia web e del telecomando. Ovvero si può accendere la lampada da web e spegnerla col pulsante, o viceversa. Una pressione continua sul pulsante permette di impostare il livello di luminosità.

Col pulsante mantenuto premuto la luminosità varia, dapprima a crescere e poi a diminuire finché non si rilascia il pulsante.

A questo punto la lampada rimane accesa con la luminosità impostata. Lo stato impostato con il pulsante viene comunicato alla rete RFTide con l'invio di un pacchetto nello stesso formato del pacchetto di risposta visibile in **Fig. 3**, che riporta lo stato finale del dispositivo all'applicazione di controllo ospitata dal gateway.

Per impostare il livello di intensità alla quale si vuole impostare la lampada al momento dell'accensione è disponibile un comando che utilizza un pacchetto configurato come in **Fig. 4**.

Come si vede, nel byte 2 si imposta il livello di intensità in un intervallo tra 0 (lampada spenta) e 49 (massima intensità). Quando viene inviato un pacchetto di impostazione dell'intensità, se quest'ultima presenta un valore maggiore di zero, l'intensità viene impostata nella memoria del modulo e la lampada viene accesa all'intensità impostata.

Se il valore dell'intensità è impostato a zero la lam-

Comando	Indirizzo	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
0x0d	0x00	Indirizzo in esadecimale	0x20	0x02	On/Off 0x01/0x00	0x00	0x00	0x00	0x00

Fig. 2

Comando	Indirizzo	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
0x0d	0x00	Indirizzo in esadecimale	0x20	0x01	On/Off 0x01/0x00	Dimmer level 0-49	0x00	0x00	0x00
									Vers. Firmware

Fig. 3

Comando	Indirizzo	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
0x0d	0x00	Indirizzo in esadecimale	0x20	0x04	Dimmer level 0-49	0x00	0x00	0x00	0x00
									Vers. Firmware

Fig. 4

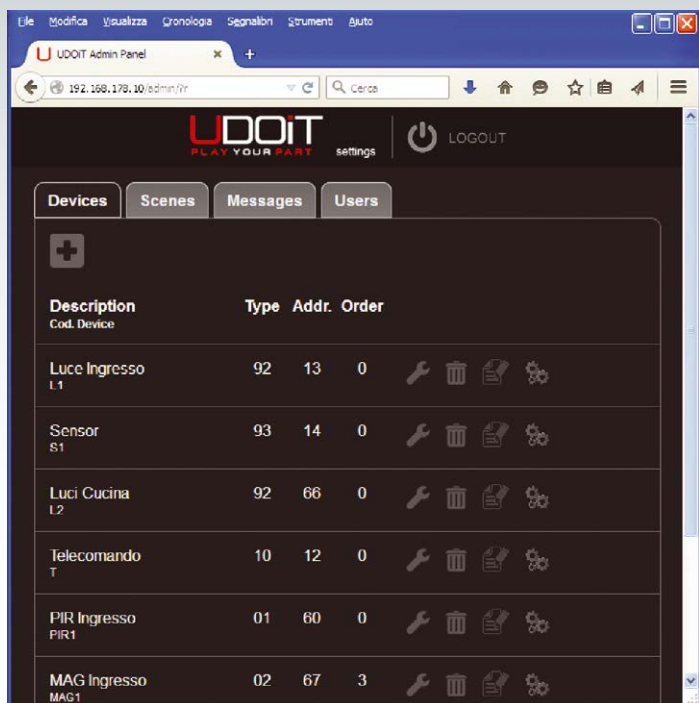


Fig. 5

pada viene spenta ma il valore non viene memorizzato della memoria del modulo. Questo per evitare anomalie di comportamento che potrebbero generare perplessità. Con l'intensità impostata a zero, un comando di accensione della lampada, in realtà, non produrrebbe alcuna luce. Come vedremo in seguito, nella pagina di controllo web, l'entrata corrispondente al dispositivo dimmer è composta da uno slider per impostare la luminosità ed un interruttore per l'accensione e lo spegnimento. Per aggiungere alla rete un attuatore dimmer si utilizza la funzione "Add Device" della applicazione di gestione, che si trova nella pagina "Settings". Alla pagina "settings", come già descritto negli articoli sui numeri 198 e 199 della rivista si accede dopo la richiesta di autenticazione con userid e password (Fig. 5). Nella Fig. 6 si nota che il dispositivo è caratterizzato da un nome, di lunghezza massima quattro caratteri, una descrizione, che apparirà nelle pagine web di gestione dei dispositivi della rete, un indirizzo, che deve essere univoco nell'intero sistema, ed il tipo di dispositivo, nel nostro caso "Dimmer". Abbiamo poi la possibilità di scegliere se vedere o meno l'entrata del dispositivo nella pagina web di gestione dell'applicazione, cosa abbastanza ovvia per un dispositivo di gestione lampade e, in questo caso scegliere una posizione nell'elenco (campo "Order"). I campi successivi servono per fare entrare o meno il dispositivo nella rete di controllo e sicurezza e per pilotare, in caso di attivazione, l'esecuzione di un processo perso-

nalizzato. Queste ultime funzionalità sono un po' più complesse da configurare e personalizzare e le affronteremo in seguito in un articolo dedicato. Una volta configurata l'anagrafica del dispositivo e averla salvata con il pulsante "Add device" possiamo vedere in Fig. 7 come si presenta sulla pagina web: un'entrata di una riga con il nome, uno slider per l'impostazione dell'intensità luminosa ed il pulsante per l'accensione e lo spegnimento della lampada. Il codice colore e la dicitura ci informano anche sullo stato attuale della lampada (intensità impostata e stato acceso/spento). Per cambiare lo stato della lampada si "clicca" sul pulsante di stato corrispondente al dispositivo o, su un dispositivo mobile, si "tappa" sullo stesso. Questo però non prima di avere eseguito l'operazione di "learning"

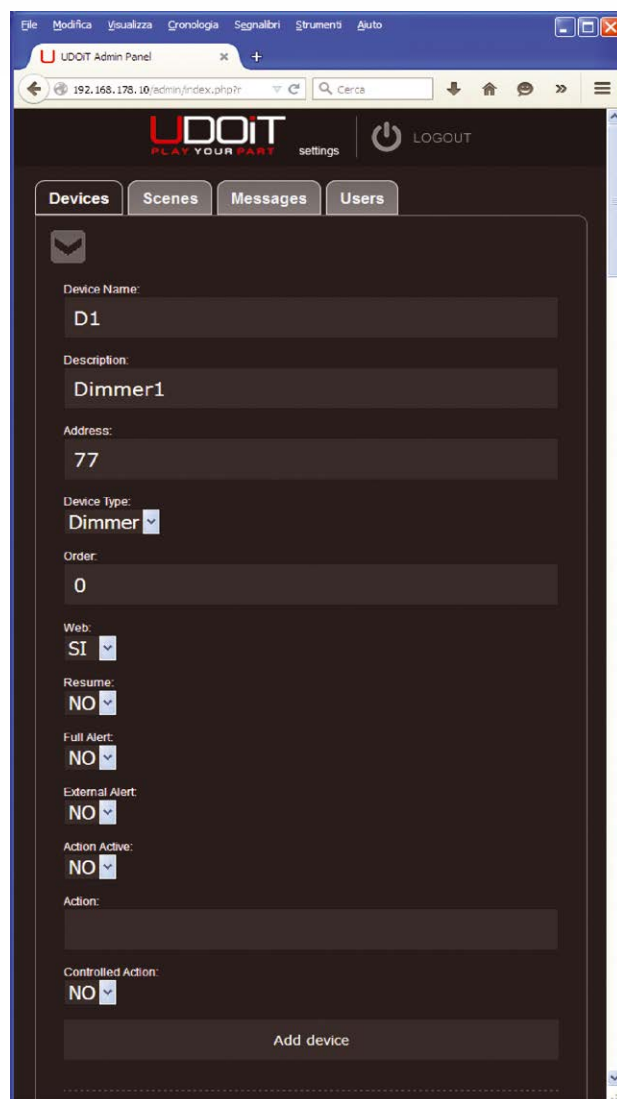


Fig. 6

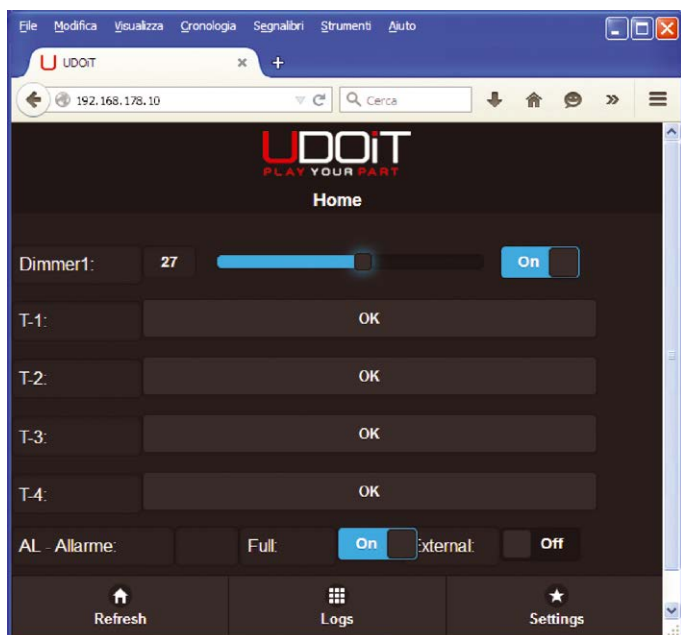


Fig. 7

del dispositivo stesso. L'operazione di learning è necessaria per associare alla rete il dispositivo appena descritto mediante un processo di apprendimento, durante il quale il dispositivo viene personalizzato con il Network ID di appartenenza (quello del Dongle, Master della rete) e l'indirizzo univoco di identificazione del dispositivo stesso, una sorta di cognome e nome che permetteranno di riconoscere univocamente il dispositivo nella rete e impediranno ai dispositivi presenti su reti adiacenti di interferire con quelli della nostra rete.

Il processo di apprendimento avviene come segue :

- Nella pagina "Devices" del pannello "Settings" si "prenota" l'operazione di apprendimento cliccando sull'icona a forma di foglio di carta e matita ("Learn Device") in corrispondenza del dispositivo per il quale si vuole eseguire l'associazione (Fig. 8);
- A questo punto si preme il pulsante presente sul circuito stampato per circa cinque secondi, al rilascio si accende un LED rosso;
- Entro 15 secondi cliccare sul pulsante "OK" della finestra "Pop-Up" sulla pagina web;
- Se tutto ha avuto successo il LED lampeggia per qualche secondo;
- Il dispositivo è in grado di essere riconosciuto ed operare sulla rete;
- In caso non venga avviata la procedura di apprendimento, il dispositivo esce dallo stato di apprendimento dopo circa cinque secondi.

Se è stata scelta l'opzione WEB "SI" lo stato del dispositivo sarà visibile nella pagina di gestione web del sistema (Fig. 7).

INSTALLAZIONE DELL'APPLICAZIONE DI GESTIONE

Con i dispositivi descritti sino a questo momento è possibile coprire una grande varietà di automazioni domestiche. Per "orchestrare" il loro comportamento in modo da soddisfare le esigenze personali è necessario utilizzare l'applicazione di gestione, che funge da "collante" dell'intero sistema. Nella presentazione dei dispositivi abbiamo anticipato alcune delle funzionalità disponibili per la gestione dei singoli dispositivi e per poterli far interagire tra di loro, come nel caso della gestione di avvisi di intrusione. In questo paragrafo descriviamo le modalità di installazione dell'applicazione e le configurazioni di base per integrarla nella rete ethernet di casa. Prima di iniziare è opportuno riprendere alcune delle condizioni di utilizzo e di configurazione dell'applicazione stessa, oltre che raccomandazioni e consigli per evitare ... sorprese.

Licenza d'uso

Prima di proseguire permettiamo alcune precisazioni doverose riguardo l'applicazione software che vi presentiamo, valide del resto per tutte le applicazioni open source in genere ed anche, a leggere bene le clausole di licenza, anche per le applicazioni con licenza a pagamento. Trattandosi di un'applicazione open source rilasciata in base alla licenza Creative Common senza oneri economici da parte degli utilizzatori, in altre parole gratuita, è bene sottolineare che viene rilasciata "AS IS" ovvero nello stato in cui si trova, e non è prevista né può essere pretesa alcuna forma di garanzia o di assistenza all'utilizzo. L'installazione, la personalizzazione e la gestione dell'applicazione è a totale carico dell'utilizzatore e non è previsto alcun supporto in caso di malfunzionamento, richieste di configurazioni, personalizzazioni, modifiche o nuove funzionalità. La disponibilità dei codici sorgenti permette, a chi ha le conoscenze adatte, di risolvere in autonomia eventuali situazioni anomale o di apportare le modifiche desiderate per personalizzare la propria installazione. A nostro avviso è necessario che gli utilizzatori di queste applicazioni abbiano già acquisito un certo grado di conoscenza e dimestichezza con la gestione, amministrazione e programmazione di sistemi GNU/Linux. L'applicazione di gestione del sistema di automazione domestica UDOiT è disponibile come applicazione Open Source, regolamentata dalla licenza CC BY-NC-SA 3.0, una licenza comune nel mondo Open Source. La sigla sta per Creative Commons Attributions (BY)-NotCommercial (NC)-ShareAlike (SA) 3.0 Unported con la clausole "AS IS" e "DISCLAIMER OF

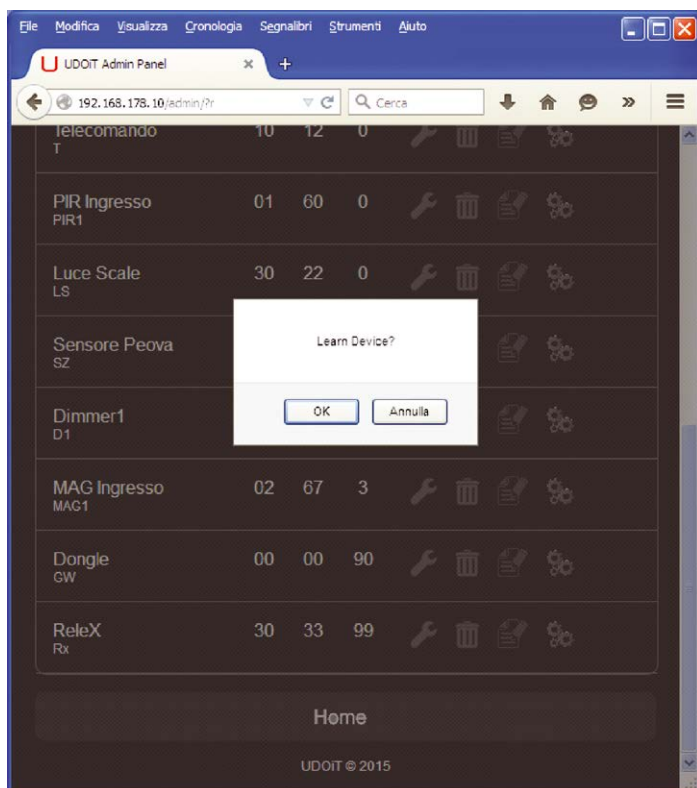


Fig. 8

WARRANTY". Per un maggiore approfondimento consultate i siti:

<http://creativecommons.org/licenses/by-nc-sa/3.0/>.
https://it.wikipedia.org/wiki/Creative_Commons

Le licenze Creative Common, rilasciate dall'organizzazione omonima con sede a Mountain View e succursali in tutto il mondo, Italia compresa, consentono agli autori di un'opera di condividere con il pubblico alcuni diritti, riservandone per sé altri. Nel caso dell'applicazione UDOiT le clausole sopracitate hanno i seguenti significati:

- **BY – Attribution (Attribuzione):** In qualsiasi utilizzo dell'opera bisogna sempre indicare l'autore della stessa (attributo obbligatorio) in modo che sia possibile attribuirne la paternità;
- **NC – Non Commercial (Non Commerciale):** Non sono consentiti usi commerciali dell'opera creativa. Chiunque può riprodurre, trascrivere, eseguire e distribuire l'opera, purché non a scopo di lucro, attribuendo sempre la paternità come definito nel primo attributo;
- **SA – Share Alike (Condividi allo stesso modo):** si può modificare l'opera ma l'opera modificata deve essere rilasciata secondo le stesse condizioni scelte dall'autore originale e citando l'autore originale;
- **AS IS – Nello stato in cui si trova:** l'opera viene rilasciata nello stato in cui si trova, sia a livello di

sviluppo che per quanto riguarda eventuali bug o malfunzionamenti;

- **DISCLAIMER OF WARRANTY – (Dichiarazione di non responsabilità):** non è prevista né richiedibile alcuna forma di garanzia o di supporto per eventuali malfunzionamenti o per l'utilizzo del software. L'utilizzo, la configurazione, la personalizzazione e l'eventuale modifica e del software è di esclusiva responsabilità dell'utilizzatore.

Tutto quanto sopra non deve sorprendere, in quanto si tratta di applicazioni delle quali viene rilasciato il codice sorgente ed il cui utilizzo è gratuito per fini non commerciali. È sempre comunque possibile contattare l'autore del software per ottenere personalizzazioni o assistenza a pagamento. Come è d'uso nel mondo Open Source anche le donazioni sono particolarmente gradite in quanto il supporto, anche economico, da parte degli utilizzatori permette di dare continuità e sviluppo all'applicazione.

Download e installazione

L'applicazione di gestione si basa sul sistema operativo Raspbian ed è disponibile come immagine da scaricare dal sito UDOiT (<http://udoit.eu/>), previa registrazione gratuita. È stata scelta questa modalità per evitare un lungo processo di installazione e configurazione di pacchetti sulla distribuzione di base, che potrebbe portare ad errori se non eseguita in modo rigoroso. L'applicazione utilizza il server web nginx con le estensioni cgi per l'esecuzione di programmi php, il server database Mysql, l'ambiente "node.js" con una serie di pacchetti aggiuntivi, i linguaggi di programmazione php, python, javascript (nativo e come linguaggio di sviluppo di node.js), alcuni script bash. Per quanto riguarda lo sviluppo dell'interfaccia web sono stati utilizzati HTML5, jquery-mobile, i fogli di stile css, il linguaggio javascript e moduli php eseguiti come cgi. Inoltre è necessario configurare opportunamente l'interfaccia ethernet ed installare i pacchetti necessari a trasformare Raspberry Pi in un access point; oltre a qualche strumento di gestione come phpadmin per il database. È abbastanza comprensibile che una descrizione passo-passo dell'installazione richiederebbe ben più di un articolo sulla rivista. Per questo si è adottata la soluzione di fornire l'applicazione come immagine preinstallata che permette di partire con una soluzione funzionante senza sforzi .. sovrumani. Per quanto riguarda gli aggiornamenti successivi del software, tutti i programmi dell'applicazione sono contenuti in due cartelle principali. I programmi di

gestione dell'ambiente web sono contenuti nella cartella `/var/www` mentre i programmi che realizzano i server in background sono contenuti nella cartella `/home/RF`. Nel sito che ospita la distribuzione oltre all'immagine della SD Card ed alle due cartelle che contengono i programmi è presente anche un file di configurazione del database, da utilizzare per ripristinare eventuali situazioni dalle quali è ... difficile uscirne. Per riassumere, la prima installazione si esegue scaricando l'immagine e creando da questa una SD Card da utilizzare con Raspberry Pi. Gli aggiornamenti successivi, salvo diverse indicazioni, si eseguono sostituendo le due cartelle di programmi con quelle contenenti la versione aggiornata. Se non avete dimestichezza con l'ambiente GNU/Linux in generale e con la distribuzione Raspbian in particolare, vi raccomandiamo di approfondire gli argomenti che trattiamo in questi articoli consultando gli articoli dedicati a Raspberry Pi sui numeri precedenti della rivista o consultando il libro "Raspberry Pi – Il mio primo Linux embedded" reperibile presso Futura Elettronica. Una volta scaricata l'immagine della SD Card la scompattiamo in una cartella a piacere. Come SD Card è raccomandabile utilizzare un modello da almeno 8GB di classe 10 e di ottima qualità, del tipo utilizzato nelle macchine fotografiche e nelle cineprese. L'applicazione esegue un numero molto alto di letture e scritture sul database e le SD Card hanno dei limiti fisici sul numero massimo di letture/scritture. Nel capitolo "Raccomandazioni" approfondiremo l'aspetto legato alle memorie di massa da utilizzare con l'applicazione. Descriviamo la procedura di download ed installazione riferendoci ad un PC con sistema operativo Windows in quanto riteniamo che quanti utilizzano un PC con Linux non abbiano bisogno di queste note. Dunque scarichiamo dall'indirizzo citato il file "RPI_UDOiT_XXXXXX.zip" dove XXXXXX indica la data di rilascio della distribuzione. Il file scaricato contiene la copia immagine in formato binario, del sistema operativo Raspian e di tutti i pacchetti e programmi per eseguire l'applicazione di gestione del sistema di automazione domotica. Scaricato il file in una qualsiasi cartella del PC (nel nostro caso abbiamo chiamato la cartella UDOiT), dobbiamo trasferirlo su SD Card. Per fare ciò utilizziamo il programma Win32DiskImager che può essere scaricato gratuitamente dal link:

<http://sourceforge.net/projects/win32diskimager/?source=dlp>

Lo installiamo sul PC semplicemente scompattando il file compresso in una cartella a scelta ed otteniamo il file "RPI_UDOiT_XXXXXX.img". Inseriamo la

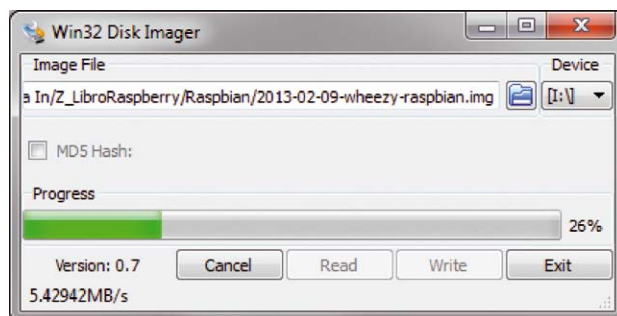


Fig. 9

SD Card nell'apposito slot del PC, se disponibile, altrimenti utilizziamo un Multi Media Card Reader. Apriamo Win32DiskImager, e utilizzando la lista di scelta sotto la dicitura "Device", scegliamo come destinazione l'unità che identifica la SD Card. Facciamo molta attenzione durante questa operazione perché una volta lanciato, il programma inizializza inesorabilmente il device indicato, fosse anche il disco fisso principale del PC. Ora premiamo il pulsante con l'icona che contiene il simbolo della "cartella" e carichiamo il file .img che avevamo scompattato in precedenza (Fig. 9). Infine, premiamo "Write" e poi diamo l'OK al messaggio di richiesta conferma. La barra di progressione indica l'avanzamento del trasferimento su SD Card. Come notate, esiste anche un pulsante "Read" che permette di fare un backup della SD Card ogni volta che vogliamo, ad esempio per creare una SD Card per una seconda scheda con le stesse configurazioni, oppure per salvarne una copia prima di eseguire installazioni, aggiornamenti o modifiche importanti. Usate queste funzioni per fare copie di backup di situazioni consolidate. Vi permetterà di tornare indietro da eventuali operazioni "maldestre" senza dover rifare tutto da capo. Lo spazio occupato dalle informazioni trasferite occupa circa 4GB, l'eventuale spazio rimasto libero sulla SD Card non potrà essere al momento utilizzato; vedremo tra poco come fare per rendere utilizzabile dal sistema tutta la capacità della SD Card. Al termine del trasferimento dell'immagine del sistema operativo otterremo il messaggio "Write successful". Diamo OK. Poi usciamo dal programma Win32DiskImager premendo il pulsante "Exit": il primo passo è stato fatto. Senza togliere la SD Card dal suo supporto sul PC passiamo a configurare l'interfaccia ethernet per assegnare un indirizzo IP statico a Raspberry Pi che sia coerente con lo schema di indirizzamento della rete domestica. Sulla SD Card si trovano due partizioni, delle quali una, che contiene la partizione di boot e quindi il kernel del sistema operativo Raspian è formattata FAT32 mentre l'altra, che contiene il filesystem di "lavoro" è formattata ext4. La prima è visibile dal file manager del PC. In questa parti-

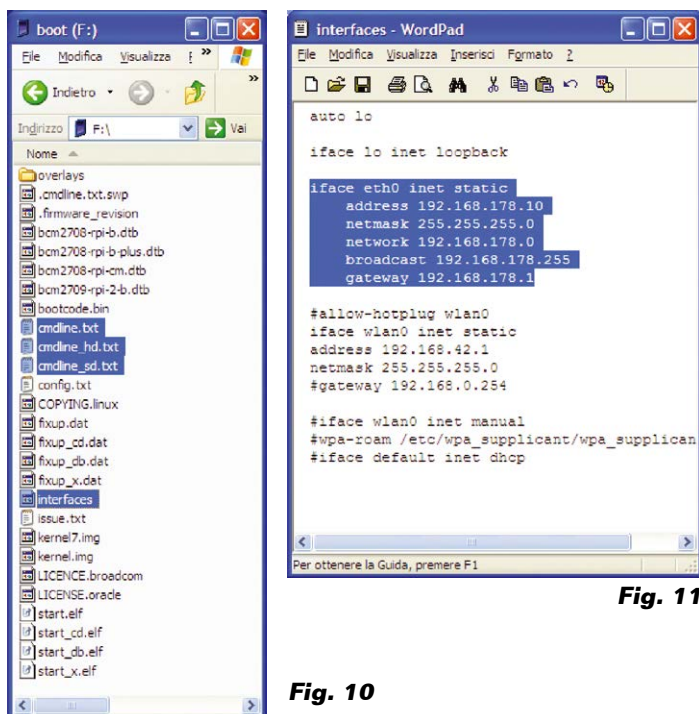


Fig. 11

Fig. 10

zione, tra gli altri file, troviamo un file di configurazione di nome interfaces (Fig. 10). Lo apriamo con WordPad e modifichiamo la configurazione degli indirizzi per renderli compatibili con la nostra rete domestica (Fig. 11). Per l'indirizzo IP statico da assegnare a Raspberry Pi dobbiamo scegliere un indirizzo non utilizzato da un altro dispositivo ed al di fuori dell'intervallo assegnato al server DHCP. Prendiamo nota dell'indirizzo assegnato, salviamo, chiudiamo il file e "smontiamo" la SD Card dal PC con lo strumento "Rimozione sicura dell'hardware". A questo punto, preleviamo la scheda SD Card dal PC e la inseriamo nello slot di Raspberry Pi al quale dobbiamo collegare il dongle USB che sarà il dispositivo master della nostra rete di moduli a radiofrequenza. La scheda non dispone di interruttore di accensione e per attivarla si dà alimentazione, utilizzando un alimentatore da almeno 2A a 5V. Lasciamo a Raspberry Pi il tempo di "partire" (un paio di minuti, per sicurezza) e poi possiamo provare ad accedere all'applicazione da un browser su PC compatibile con lo standard HTML5, si raccomanda Firefox, digitando l'indirizzo IP assegnato a Raspberry Pi. Dovrebbe apparire la pagina "Home" dell'applicazione (Fig. 12). Tutto OK? Speriamo di sì. In questo caso potete già procedere ad associare i vostri sensori ed attuatori oltre a personalizzare il funzionamento generale del sistema. Per iniziare cliccate sul menu "Settings". Nella videata di richiesta inserite l'userid "admin" e la password "admin" (Fig. 13). Potrete poi modificare l'utente e la password a vostro piacimento cliccando sulla

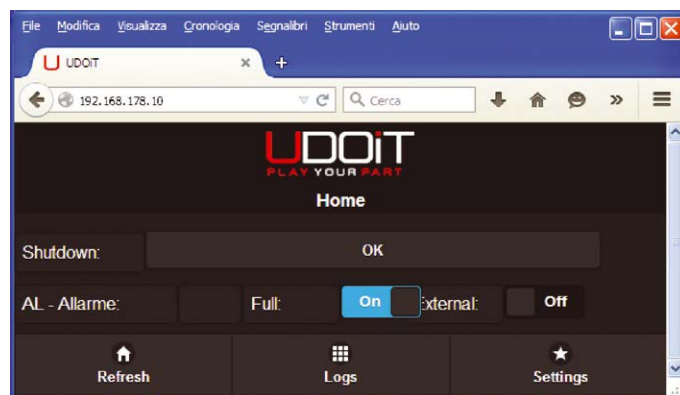


Fig. 12

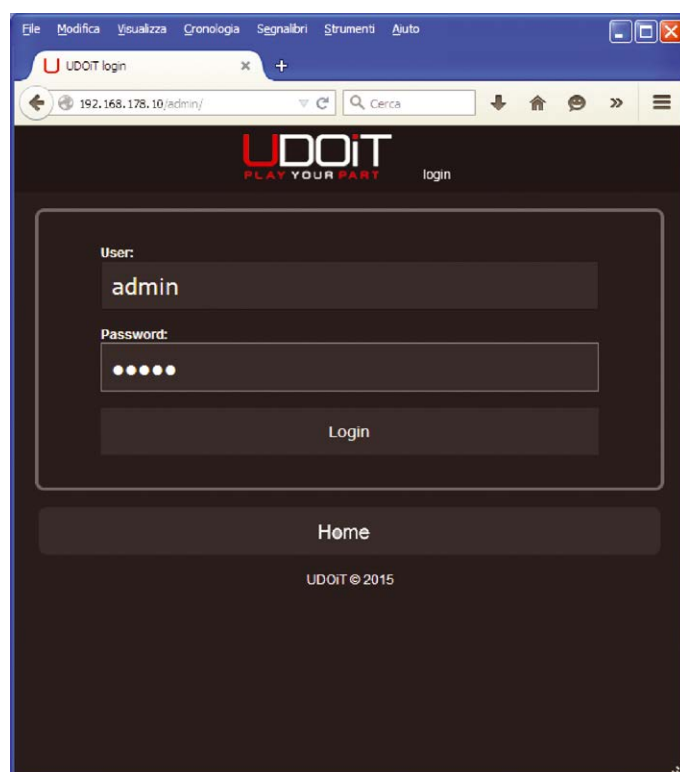


Fig. 13

tab "Users". Una prima serie di descrizioni su come associare e configurare i sensori, gli attuatori e le scene la trovate negli articoli sui numeri precedenti della rivista 197, 198 e 199.

Configurazione iniziale e aggiornamenti

Per intervenire sulla configurazione e per gestire gli aggiornamenti e le personalizzazioni all'applicazione è stato previsto di utilizzare l'accesso remoto via SSH, sia per quanto riguarda la finestra di terminale che per la gestione del file system. In ambiente Windows utilizziamo i soliti strumenti, che abbiamo presentato più volte, PuTTY e WinSCP. PuTTY si scarica gratuitamente in licenza MIT da: <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>

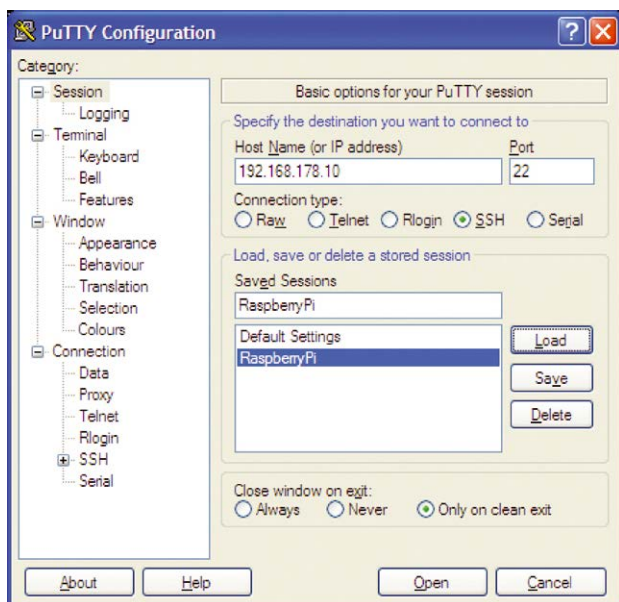


Fig. 14

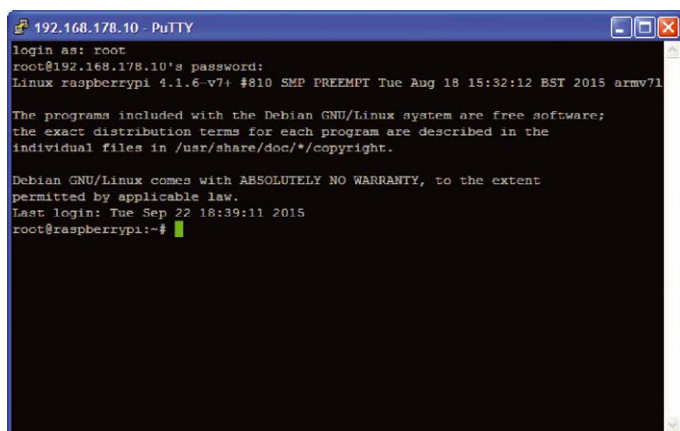


Fig. 15

PuTTY non ha bisogno di installazione: è un semplice eseguibile pronto per essere avviato con un doppio clic. Lo scarichiamo e lo configuriamo per la prima volta. Nella videata di apertura iniziamo a configurare una sessione dedicata a Raspberry Pi e a salvarla, in modo da poterla riutilizzare. Per prima cosa digitiamo RaspberryPi (o un altro nome a vostra scelta) nella casella "Saved Session". Ora inseriamo l'indirizzo IP di Raspberry Pi nella casella "Host Name (or IP address)" l'indirizzo IP è quello che avete assegnato a Raspberry Pi (Fig. 14). Premete "Save" e poi "Open". Appare la finestra del terminale a linea di comando, eventualmente preceduta da una finestra di avviso di sicurezza alla quale rispondete "ON". A questo punto è come se stessimo usando Raspberry Pi in locale (Fig. 15). Alla richiesta di login autenticatevi con l'utente "root", e la password "pi". Per inciso, l'altro utente configurato nella distribuzione è l'utente standard di Raspbian, "pi" con password "raspberrypi". Come

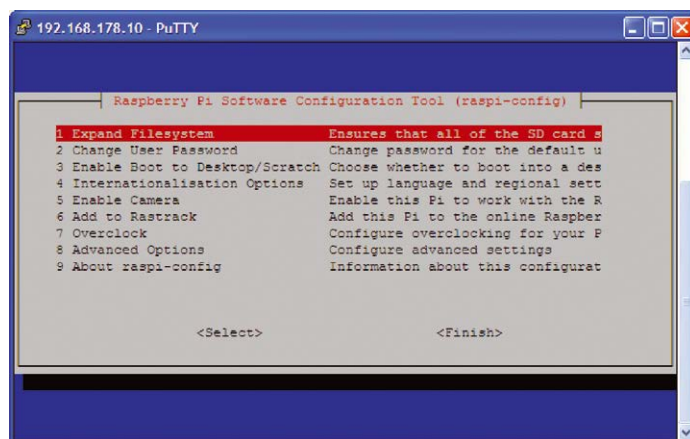


Fig. 16

prima operazione facciamo in modo di espandere la partizione di root in modo da utilizzare tutto lo spazio disponibile sulla SD Card. Utilizziamo lo strumento di configurazione Raspi-config, che lanciamo con il comando:

raspi-config

Selezionate la voce "Expand Filesystem" per espandere la partizione di root, ovvero lo spazio occupato da Raspbian sulla SD Card per i file di sistema ed i dati utente, fino a sfruttare l'intera capacità della SD Card stessa; in altre parole, se abbiamo caricato l'immagine di Raspbian da 4 GB su una SD Card da 8 GB, il sistema continuerà a vedere solo 4 GB fino a quando non avremo utilizzato questa funzione (Fig. 16). Diamo invio, dopo avere selezionato la voce di menu, diamo OK nella schermata di avviso che la partizione verrà espansa all'avvio successivo. Al termine della configurazione (Finish) ci viene chiesto se riavviare Raspberry Pi subito, per espandere lo spazio sulla SD Card, o rimandarlo ad un

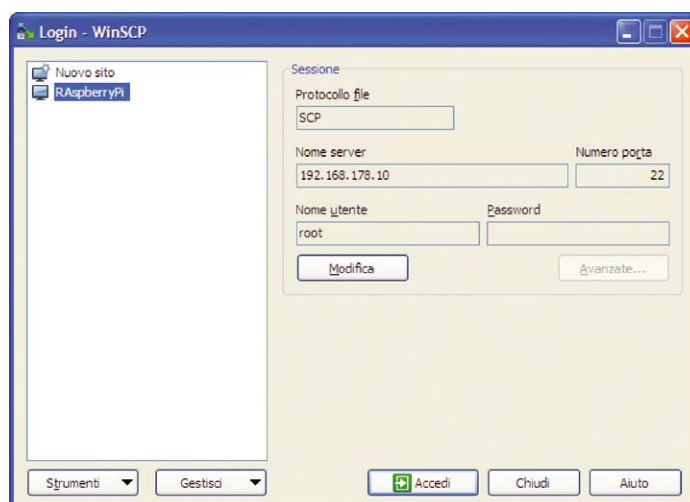


Fig. 17

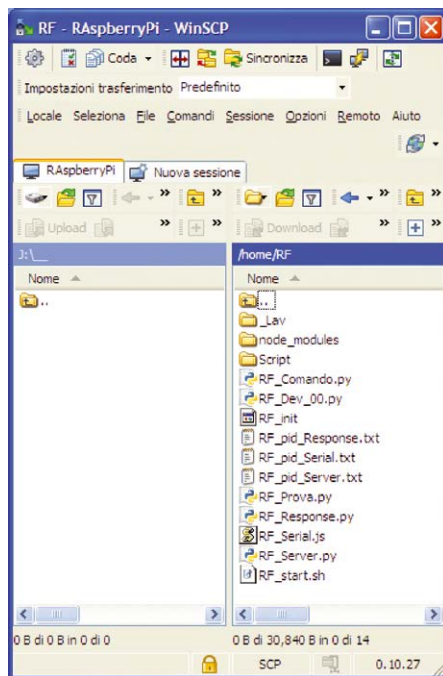
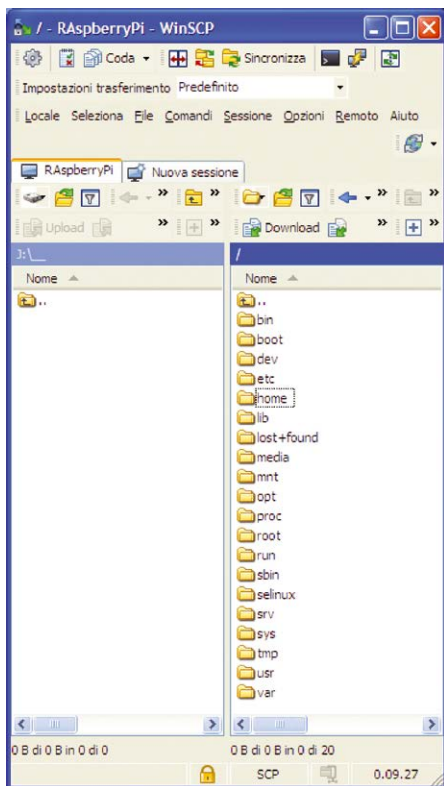


Fig. 19

Fig. 18

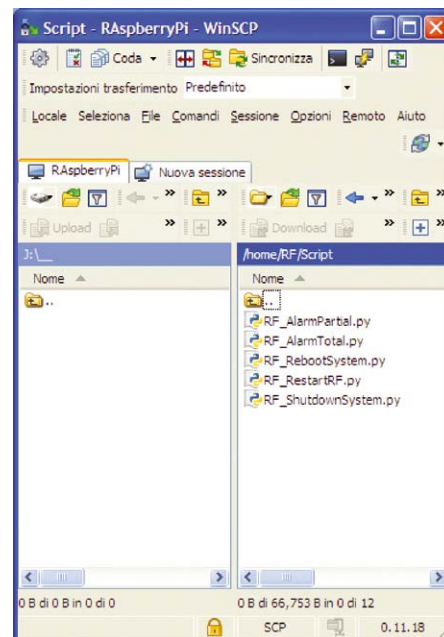


Fig. 20

momento successivo. Scegliete “Si” ed attendete il “reboot”. Descriviamo anche la possibilità di accedere con WinSCP al filesystem di Raspberry Pi in quanto ci verrà utile per eseguire i backup e gli aggiornamenti dell’applicazione. Potete scaricare gratuitamente WinSCP all’indirizzo: <http://winscp.net/eng/download.php>, scaricate l’ultima versione stabile, evitando le versioni beta. WinSCP può fare tutte le operazioni fondamentali sui file, come download e upload. Consente anche di editare i file, rinominare file e cartelle, creare nuove cartelle, modificare le proprietà di file e cartelle e creare collegamenti. Lanciate il setup; e seguite il processo d’installazione accettando le scelte predefinite. Premendo “Fine” si lancia il programma.

Per il futuro vi consigliamo di creare sul desktop due collegamenti, uno con Putty e l’altro con WinSCP per facilitarvi il compito. Lanciato il programma apparirà una videata che avvisa che non è stata impostata una connessione sicura e crittografata tra il vostro PC e Raspberry Pi; GNU/Linux è molto rigido per la sicurezza e, d’altra parte, quelli che stiamo usando sono strumenti di amministrazione. Per ora non mettiamo troppa carne al fuoco e rispondiamo “SI”. Il risultato è la videata di apertura di WinSCP che configuriamo come in Fig. 17, con indirizzo IP, utente “root” e scegliendo il protocollo “SCP”. Infine salviamo le impostazioni con “Salva”. Ora premiamo “Accedi”, digitiamo la password, “pi”, quando ci viene richiesta e finalmente il nostro

ambiente di gestione remota è pronto ed appare come in Fig. 18, con le due sessioni aperte e connesse. Nella finestra di sinistra dell’interfaccia grafica di WinSCP abbiamo il file system del PC ed in quella di destra quello di Raspberry Pi, posizionato sulla cartella personale corrispondente all’utente “root”. L’interfaccia familiare dovrebbe togliervi ogni dubbio sul suo utilizzo: cliccando sull’icona a freccia in cima all’elenco delle cartelle risalite fino alla radice del filesystem “/”. Facciamo doppio clic sulla cartella “home” e, all’interno di questa, sulla cartella “RF”. In quest’ultima cartella sono contenuti i programmi che vengono eseguiti in “background” o come servizi (server) dell’applicazione di gestione (Fig. 19).

Date un’occhiata anche alla cartella “Script” dove sono contenuti i programmi per la gestione “personalizzata” dell’applicazione. Alcuni di questi sono già stati resi disponibili per la gestione delle condizioni di gestione delle intrusioni e per alcune funzioni di servizio (Fig. 20).

Torniamo indietro e andiamo alla cartella “var” e poi “www”. In questa cartella sono contenuti tutti i programmi necessari alla gestione dell’interfaccia web dell’applicazione (Fig. 21).

All’indirizzo di download è presente anche un file compresso che contiene le due cartelle appena descritte. Per eseguire l’aggiornamento dell’applicazione si scarica il file, lo si scompatta in una cartella del PC e poi si sostituiscono, con WinSCP, i programmi presenti sulla SD Card con quelli contenuti nelle cartelle appena ottenute. Leggete prima

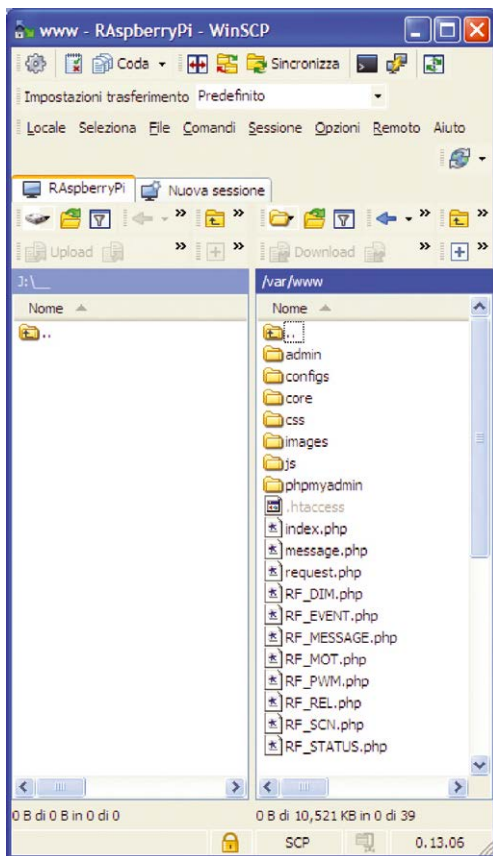


Fig. 21

come eseguire dei backup nel seguente capitolo “Raccomandazioni”. Segnaliamo che le liste di file e cartelle mostrate in WinSCP non vengono aggiornate automaticamente in ogni situazione. Potrebbe accadere di non vedere immediatamente nuovi file aggiunti o eliminati. In questi casi clicchiamo col tasto destro del mouse in uno spazio vuoto e selezioniamo “aggiorna”. Per terminare, è il momento buono per aggiornare la distribuzione Raspbian con i comandi:

```
apt-get update
apt-get upgrade
apt-get dist-upgrade
```

Raccomandazioni

La prima raccomandazione, nel caso vogliate utilizzare l'applicazione in modo massiccio e continuativo, è quello di non fidarsi delle SD Card ma di prevedere l'utilizzo di un disco fisso. Le SD Card sono soggette ad un numero fisso, ed anche, entro certi termini, abbastanza limitato di cicli di scrittura. Una SD Card che può durare una vita se installata in una macchina fotografica può “rompersi” dopo poche settimane di utilizzo nella applicazione UDOiT. Lo stesso dicasi per l'utilizzo di unità SSD. La scelta migliore è di affidarsi ad un hard disk tradizionale con connessione USB. La scelta di

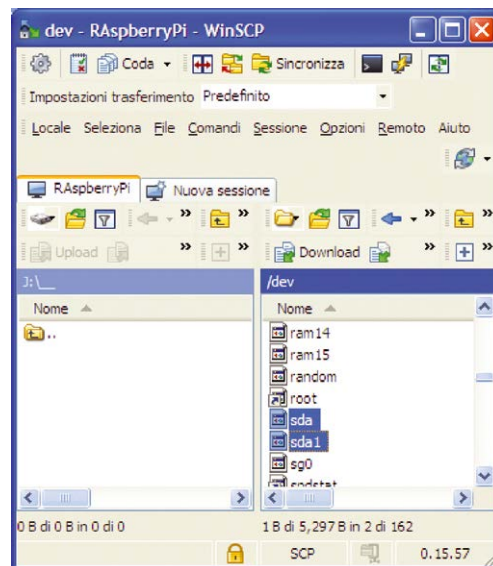


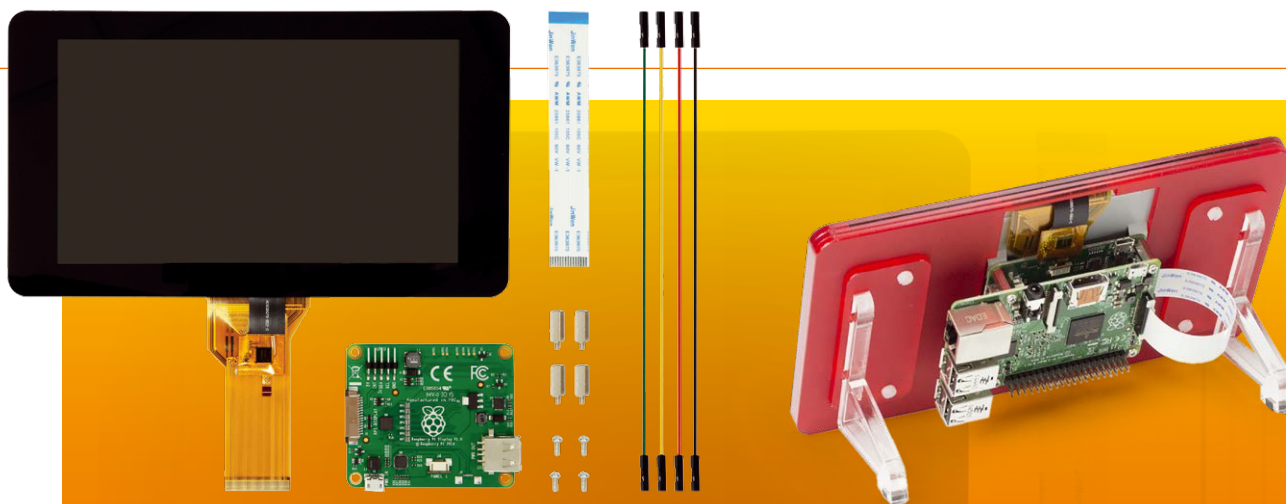
Fig. 22

un'unità con connessione USB è l'unica possibile, in quanto Raspberry Pi non è dotato di un connettore SATA (Serial Advanced Technology Attachment) per il collegamento di dischi rigidi. Ricordiamo che alla fine dell'operazione avremo la nostra partizione di root sull'unità di memoria esterna, mentre la partizione di boot deve restare sulla SD Card, in quanto è l'unico device abilitato al boot iniziale del sistema. Vediamo ora come trasferire ed attivare la partizione di root su disco fisso. Colleghiamo il disco rigido USB a una presa USB libera su Raspberry Pi. Se il disco non ha la possibilità di essere alimentato dall'esterno, utilizziamo un Hub USB autoalimentato per connettere il disco rigido. Una volta collegato il disco rigido possiamo vedere il buon esito dell'operazione verificando la presenza del file descrittore del disco nella cartella /dev. Se è l'unico disco collegato dovremmo trovare il file /dev/sda1 (Fig. 22).

Probabilmente ci troviamo in presenza di un disco con formattazione FAT o NTFS. Dobbiamo formattarlo con una tipologia di formattazione che sia ottimale per GNU/Linux; nel nostro caso ext4. Ext4 (fourth extended file system) è una delle strutture di file system supportate da GNU/Linux (che include anche FAT e NTFS per compatibilità con il mondo Windows) di rilascio abbastanza recente e che è in grado di indirizzare singoli file di dimensioni fino a 16 Tebibyte e memorie di massa fino a un Exbibyte (1.152.921.504.606.846.976 di byte); un quadriliardo di byte sarà sicuramente sufficiente per la nostra applicazione. Formattiamo il nostro disco con un comando molto semplice ma allo stesso tempo molto ... distruttivo. Quindi, fate attenzione!

```
mkfs.ext4 /dev/sda1
```

Creiamo un punto di mount e montiamo la nuova



Display per Raspberry Pi

Raspberry Pi ha anche un **display touch ufficiale** che lo trasforma in un vero e proprio dispositivo stand-alone. Il display è prodotto dall'azienda inglese Inelco Hunter, ha una diagonale da **7 pollici**, con risoluzione di **800 x 480 pixel** e colori a **24 bit**; è **touch capacitivo a 10 punti di contatto** ed ha un **angolo di visione di 70 gradi**. Lo schermo è stato progettato per poter essere collegato al

connettore DSI (Display Serial Interface) di Raspberry Pi, lasciando libera la porta HDMI per un secondo display. E' inoltre possibile montare lo schermo direttamente al Raspberry Pi, creando di fatto un PC all-in-one o un tablet senza batteria. Lo schermo viene alimentato attraverso un cavo USB ed è possibile anche utilizzare una sola fonte di alimentazione sia per il display che per Raspberry Pi. Non è il primo display touch disponibile per Raspberry Pi, ma è l'unico ufficialmente approvato dal team di sviluppo e che non richiede modifiche e personalizzazioni al kernel del sistema operativo per essere utilizzato.

partizione di root, vuota, in modo da averla visibile nel filesystem

```
mkdir /mnt/temp
mount /dev/sda1 /mnt/temp
```

Copiamo tutta la partizione di root da SD Card sulla partizione di root del disco fisso; lasciate lavorare Raspberry Pi in pace perchè l'operazione richiede abbastanza tempo (**Fig. 18**). Anche qui Attenzione!

```
cp -Rpvx / /mnt/temp
```

cp è il comando di copy e i parametri significano:

- R = copia tutte le cartelle ricorsivamente;
- p = conserva gli attributi dei file;
- v = verbose, spiega cosa sta succedendo;
- x = rimani all'interno del file system specificato.

Viene copiato tutto il filesystem di root (/) sul device specificato al punto di mount /mnt/temp, cioè sul nostro disco fisso sda1. Grazie al parametro "x" non viene copiato il file system di boot. Ora dobbiamo configurare i file utilizzati durante il boot per indirizzare correttamente la nuova par-

tizione di root su disco e non più quella sulla SD Card, in particolare i file:

```
/etc/fstab
/boot/cmdline.txt
```

Per questo sono stati predisposti dei file con le configurazioni opportune che basta copiare sui file originali con i comandi:

```
cp /etc/fstab_hd /mnt/temp/etc/fstab
cp /boot/cmdline_hd.txt /boot/cmdline.txt
```

Terminate le operazioni diamo il comando "reboot" e vedremo che come partizione di root viene utilizzata quella presente sul disco fisso. Per tornare alla configurazione iniziale, con la partizione di root sulla SD Card date il comando:

```
cp /boot/cmdline_sd.txt /boot/cmdline.txt
```

La seconda raccomandazione riguarda i backup dell'applicazione. Terminata l'installazione vi raccomandiamo di eseguire una copia della SD Card, sempre utilizzando

lo strumento Win32DiskImager e conservarla per eventuali necessità di ... ricominciare da capo.

Non è necessario eseguire spesso la copia della SD Card. Una volta ottenuta una copia, in caso di necessità, sarà possibile ricreare la situazione esistente a partire dai backup delle cartelle di programmi e del database, questi sì da eseguire con frequenza maggiore, almeno in occasione di aggiunta di dispositivi alla rete, di creazione di scene e di personalizzazioni al funzionamento del sistema.

Per eseguire queste copie vi consigliamo di utilizzare lo strumento WinSCP e di creare una cartella di backup sul vostro PC. All'interno di questa cartella create ulteriori due cartelle "home" e "var" andate all'interno della cartella "home" poi, nella finestra del file system di Raspberry Pi, andate alla cartella "home" e, dall'interno di questa, trascinate la cartella "RF" nella finestra di sinistra.

Eseguite lo stesso procedimento per la cartella "www" all'interno di "var".

Per eseguire l'aggiornamento dei programmi eventualmente resi disponibili sul sito UDOiT si segue il processo opposto. Dopo aver scaricato il file compresso .zip che contiene gli aggiornamenti, scompattatelo in una cartella a vostra discrezione. Aprite questa cartella nella finestra di sinistra di WinSCP e trascinate le due cartelle con gli aggiornamenti dalla finestra di sinistra nelle rispettive cartelle della finestra di destra.

Prima di aggiornare le cartelle con una nuova versione fate un backup, non si sa mai. Un altro backup importante è quello relativo al contenuto del database dell'applicazione. Per fare questo tornate al browser del vostro PC (consigliato Firefox) e digitate

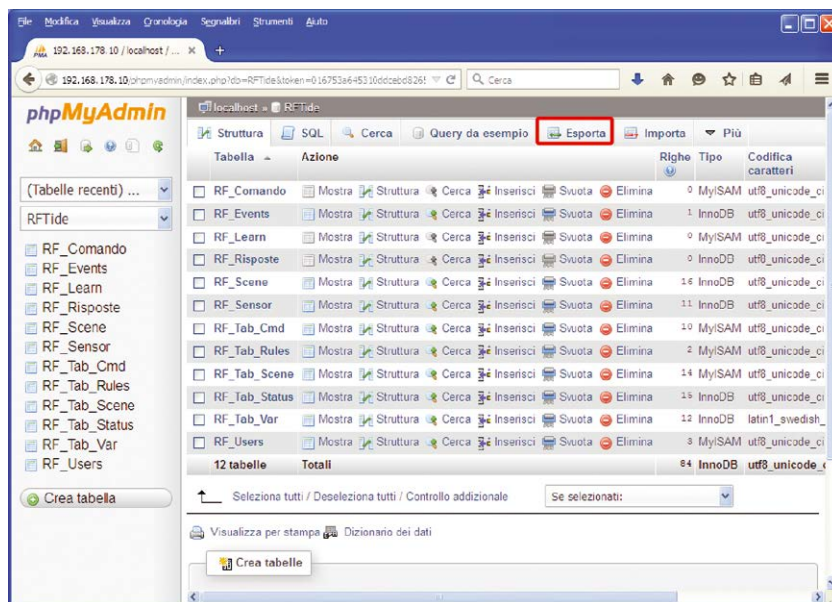


Fig. 23

l'indirizzo:

<http://<indirizzo-IP-Raspberry>/phpmyadmin>

nella finestra che si apre inserite userid "root" e come password "pi". Nella finestra successiva selezionate il database "RFTide" (Fig. 23) e poi, nella finestra che elenca tutte le tabelle contenute cliccate in alto a destra sulla tab "Esporta". All'interno della pagina selezionate l'opzione "Personalizzato - mostra tutte le possibili opzioni", andate alla sezione "Opzioni di creazione dell'oggetto" e spuntate la voce "Aggiungi l'istruzione DROP TABLE / VIEW / PROCEDURE / FUNCTION / EVENT" (Fig. 24). Finalmente cliccate sul pulsante "Esegui" in fondo alla pagina. Quando viene chiesto dove salvare il file di backup selezionate la cartella dove avete eseguito i backup precedenti. L'operazione inversa si esegue selezionando la tab "Importa" ed il file da importare nella finestra di dialogo che si apre cliccando su "Sfoglia". Terza raccomandazione. Tenete presente una grande differenza tra una linea telefonica tradizionale ed una connessione telefonica ADSL sia su doppino che in fibra ottica. In caso di caduta di

tensione all'interno dell'appartamento la linea telefonica tradizionale continua a funzionare, in quanto alimentata dalla centrale telefonica.

Un eventuale antifurto con batteria tampone è quindi in grado di avvisarci della caduta di tensione utilizzando la funzione di inoltro di messaggi vocali. Al contrario, in caso di caduta di tensione, il nostro modem/router ADSL cesserà immediatamente di funzionare.

Nel caso di una linea ADSL su doppino telefonico, il telefono continuerà a funzionare, mentre in caso di connessione in fibra ottica, nemmeno quello. Lo stesso dicasi per Raspberry Pi e quindi per i sistemi di controllo ad essa associati. Nel caso di Raspberry Pi, essendo un PC a tutti gli effetti, una caduta di tensione improvvisa rischia anche di rendere inutilizzabile la SD Card.

Quindi, in ogni caso fate i backup. In secondo luogo, volendo gestire un sistema di automazione domestica, dal quale vogliamo essere avvisati, magari per e-mail, di eventuali malfunzionamenti di qualche impianto o, per l'appunto, di una caduta della tensione, è bene pensare di

Esportazione delle tabelle dal database "RFTide" in corso

Metodo di esportazione:

☐ Rapido - mostra solo le opzioni minime

☒ Personalizzato - mostra tutte le possibili opzioni

Tabelle:

Selezione tutto / Deselezione tutto

RF_Learn
RF_Risposte
RF_Scene
RF_Sensor
RF_Tab_Cmd
RF_Tab_Rules
RF_Tab_Scene
RF_Tab_Status
RF_Tab_Var
RF_Users

Output:

☒ Salva l'output in un file

Modello per nomi dei file: DATABASE ☒ usa questo per esportazioni futura

Set di caratteri del file: Utf-8

Compressione: Nessuno

☐ Salva l'output come testo

Formato:

SQL

Opzioni specifiche al formato:

☒ Visualizza commenti (include informazioni come l'orario dell'esportazione, versione di PHP, e la versione del server)

Aggiunge un commento personalizzato al header (per tornare a capo)

☐ Includi l'orario della creazione, ultimo aggiornamento, ed ultimo controllo del database

☐ Includi esportazione in una transazione

☐ Disabilita i controlli sulle chiavi straniere

Sistema di database o server: MySQL un po' vecchio per massimizzare la compatibilità con:

NONE

Dump delle tabelle

☐ struttura

☐ dati

☒ struttura e dati

Opzioni di creazione dell'oggetto

Aggiungi istruzioni:

☒ Aggiungi istruzione DROP TABLE / VIEW / PROCEDURE / FUNCTION / EVENT

☒ Aggiungi istruzione CREATE PROCEDURE / FUNCTION / EVENT

☒ opzioni di CREATE TABLE:

☒ IF NOT EXISTS

☒ AUTO_INCREMENT

☒ Racchiudi i nomi dei campi e della tabelle in virgolette (Protegge i nomi dei campi e tabelle che contengono caratteri speciali o parole chiave)

Opzioni del dump dei dati

Invece delle istruzioni INSERT, usa:

☐ Istruzioni INSERT DELAYED

☐ Istruzioni INSERT IGNORE

Funzione da utilizzare per il dumping dei dati: INSERT

La sintassi da utilizzare per gli inserimenti dei dati:

☐ includi i nomi dei campi in ogni istruzione INSERT
Ad esempio: INSERT INTO tbl_name (col_1,col_2,col_3) VALUES (1,2,3)

☐ inserisci multiple righe in ogni istruzione INSERT
Ad esempio: INSERT INTO tbl_name VALUES (1,2,3), (4,5,6), (7,8,9)

☒ entrambi i precedenti
Ad esempio: INSERT INTO tbl_name (col_1,col_2,col_3) VALUES (1,2,3), (4,5,6), (7,8,9)

☐ nessuno dei precedenti
Ad esempio: INSERT INTO tbl_name VALUES (1,2,3)

Lunghezza massima di una query creata: 50000

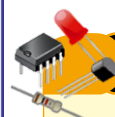
☒ Esegui il dump dei campi nella notazione esadecimale (ad esempio, "aoc" diventa 0x616f63)

☒ Esegui il dump dei campi TIMESTAMP in UTC (abilita il dump e la mancanza dei campi TIMESTAMP tra server in diversi fusi orari)

Esegui

Fig. 24

dotarsi di un sistema UPS (Uninterruptible Power Supply) che faccia da backup alla sorgente di alimentazione principale. Il sistema di backup della sorgente di alimentazione deve essere in grado di notificare l'evento al sistema di controllo, in modo da provvedere ad inviare i messaggi di allerta secondo le modalità previste come e-mail, SMS, ecc. Inoltre, in caso di caduta di tensione prolungata, dopo un periodo predefinito, il sistema deve essere in grado di eseguire uno "shut-down" controllato, in modo da preservare l'integrità di dati e applicazioni, per poi riaccendersi una volta ripristinate le condizioni di normalità. Per il momento sappiate che il problema esiste, e che può essere arginato adottando un buon UPS da PC, nel futuro vi indicheremo le soluzioni che meglio soddisfano le funzionalità richieste.



per il MATERIALE

I prodotti presentati in questo progetto possono essere acquistati presso Futura Elettronica. Il dongle USB (cod. RFT868USB) costa 39,00 Euro, il sensore magnetico (cod. RFT868MAG) è disponibile a 49,00 Euro, il sensore di movimento (cod. RFT868PIR) è in vendita a 55,00 Euro, il modulo a relé (cod. RFT868REL) costa 55,00 Euro e il modulo a dimmer (cod. RFT868DIM) è disponibile a 55,00 Euro. Il radiocomando a 4 canali (cod. RFT8684CH) è in vendita a 39,00 Euro, e la Raspberry Pi 2 Tipo B (cod. 8326274RS) a 44,00 Euro.

Il materiale va richiesto a:
Futura Elettronica, Via Adige 11,
21013 Gallarate (VA)
Tel: 0331-799775 - Fax: 0331-792287
<http://www.futurashop.it>

VIDEOGAME ARCADE

di ROBERTO
TESTI

RETROPIE
Electronica In



Ricostruiamo in casa un videogioco da bar per riscoprire il fascino dei giochi Arcade anni '80: basta una Raspberry Pi, uno shield qui descritto, uno schermo LCD e alcuni pulsanti.

Tante volte avrete sentito parlare di videogioco Arcade, ma forse, se non avete almeno 35-40 anni, non ne conoscete il significato. Un videogioco Arcade o semplicemente videogioco da bar, è un sistema elettronico inserito in un cabinato solitamente in legno, costituito inizialmente da una scheda elettronica e successivamente da un vero e proprio computer. Tale macchina da bar viene chiamata anche con il termine coin-op, abbreviazione di coin-operated (funzionamento a monete); in italiano è macchina a gettoni. I videogiochi Arcade conobbero il loro splendore, sia per qualità che per popolarità, dalla fine degli anni settanta alla fine degli anni ottanta del secolo scorso. I titoli che hanno scritto la storia dei videogiochi sono Space Invaders pubblicato nel 1979, Pac-Man (1980), Donkey Kong (1981), Burger Time (1982) e a seguire tanti, tanti altri. I primi cabinati prodotti erano specifici per il gioco in essi contenuto, ma la scelta si rivelò ben presto troppo dispendiosa, in quanto era difficile adattarli alle mutate esigenze del pubblico; per questo nacquero cabinati generici con connettore Jamma (Japan Amusement Machinery Manufacturers Association, un'associazione giapponese che raccoglieva i produttori di videogiochi Arcade). Jamma è uno standard della pin-out del connettore della scheda madre dei videogiochi Arcade, che permetteva di cambiare semplicemente la scheda del gioco anziché tutto il cabinato. Negli anni in cui impazziva la moda dei videogame da bar non esisteva una tecnologia casalinga a basso costo che permettesse alle masse di possederne uno e molto spesso per poter giocare ad un gioco da bar bisognava adattarsi a versioni malfatte e limitate

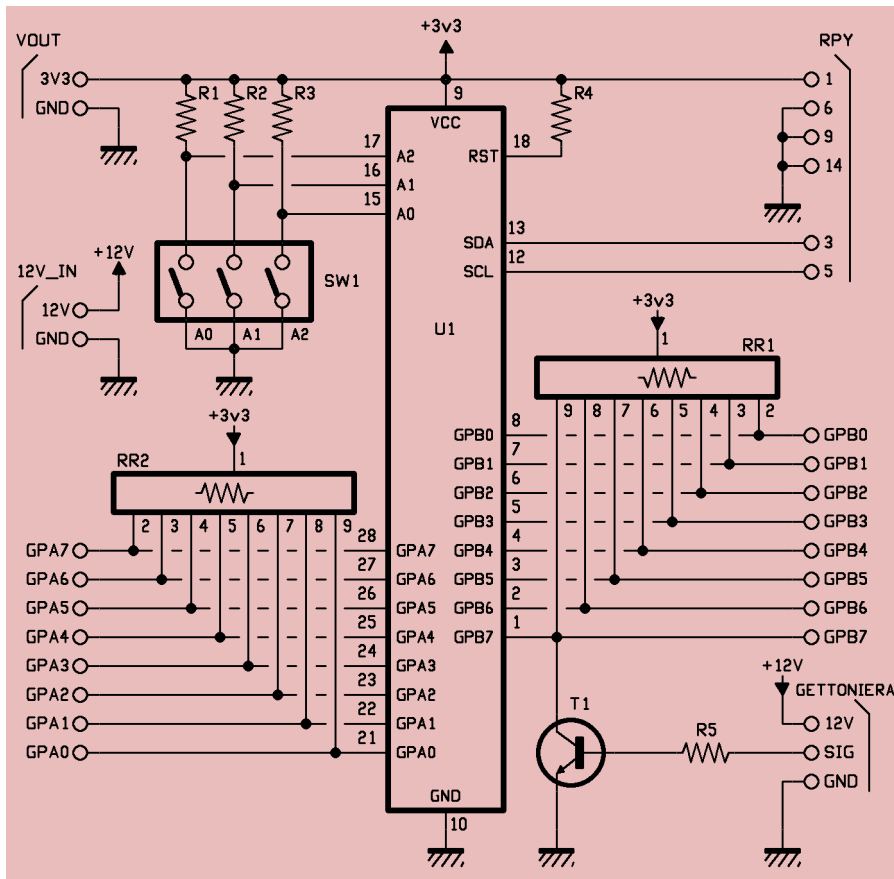
che funzionassero con le console e computer domestici, infinitamente meno potenti. Anche con l'avvento delle console più moderne, i giochi Arcade hanno mantenuto intatto il fascino di un tempo e sono ancor oggi apprezzati da un pubblico giovane, anche se le sale giochi con i cabinati Arcade ormai non esistono più. Con la dismissione dei cabinati da bar, tanti cultori hanno cercato di accaparrarsene un esemplare da restaurare per metterlo in taverna o in camera propria. A chi non è riuscito, proponiamo in questo articolo una versione che, sfruttando la disponibilità di schede a microprocessore potenti ma economiche come Raspberry Pi, permette di avere a casa un vero Arcade. Basta uno shield di interfaccia con pulsanti, joystick e gettoniera, per ricreare un cabinato da bar: noi vi diamo l'elettronica e un suggerimento per costruire il box; voi metteteci la vostra fantasia.

Il software utilizzato impiega gli emulatori Arcade più diffusi disponibili da oltre 10 anni per i PC.

IL PROGETTO

Come "cervello" abbiamo scelto l'ultima versione di Raspberry Pi 2, e come software RetroPie 3.0, essendo l'immagine Linux più





completa in fatto di emulazione di giochi Arcade e più facile da utilizzare rispetto ad altre “immagini”, richiedenti una configurazione lunga e complessa. Con RetroPie 3.0 è possibile emulare vari tipi di console del passato, tra cui Nintendo, Game Boy, Sega Master System, Amiga, Commodore ecc. In questo caso, ci concentreremo sull'emulatore MAME (Multiple Arcade Machine Emulator), famoso in tutto il mondo perché può emulare praticamente tutti i giochi degli anni '80 e '90, che potevamo trovare nelle sale giochi. Nel progetto realizzeremo un “cabinato” MAME (Multiple Arcade Machine Emulator) per il quale occorre:

- Raspberry Pi 2;
- SD-Card da 8 Gb o superiore;
- alimentatore per Raspberry Pi (5V 2A microUSB);
- joystick di tipo Arcade;
- pulsanti Arcade;

- cavi per collegamento joystick e pulsanti con Raspberry Pi;
- case per il fissaggio del tutto;
- cavo HDMI;
- monitor CVBS o HDMI;
- cavo audio (opzionale, perché si possono utilizzare gli altoparlanti di un televisore con interfaccia HDMI);
- tastiera USB da PC (solo per la configurazione).

Questo per l'hardware; invece per il software ci occorre:

- RetroPie 3.0 ottimizzato per Raspberry Pi, derivato dalla distribuzione Debian;
- modifiche I²C al software MCP23017 per la gestione degli input;
- software per l'emulazione delle ROM;
- ROM dei giochi.

Precisiamo che le ROM sono i file
contenenti i giochi per un certo

emulatore; il nome richiama le memorie a sola lettura contenenti il codice che il microprocessore del videogioco faceva girare; quindi come nei vecchi videogame si riprogrammava l'unità sostituendo la ROM, negli emulatori si cambia gioco caricando un file chiamato ROM.

Prima di procedere concentriamo ci su RetroPie, dicendo innanzitutto che l'immagine si scarica gratuitamente dal sito <http://blog.petrockblock.com/retropie/retropie-downloads/>.

Il progetto RetroPie è iniziato con l'idea di trasformare la Raspberry Pi in una console di retro-gaming e nell'arco di poco tempo si è evoluto fino alla versione 3.0. Oltre agli emulatori, gli sviluppatori hanno installato un front-end, una propria struttura: un piccolo sistema operativo dedicato al retrogaming, dove c'è la possibilità di "navigare" attraverso un'interfaccia semplice ed intuitiva, tra i vari emulatori.

Il progetto utilizza Raspbian come sistema operativo di base ed integra un ampio set di emulatori per una vasta gamma di sistemi informatici degli ultimi decenni. Il RetroPie Setup Script è il cuore del software per l'installazione e la configurazione di tutti i componenti della distribuzione.

Lo script di installazione RetroPie consente di installare e configurare tutti gli emulatori, il front-end, driver e altri componenti per la nostra console retrogaming. Tecnicamente, si tratta di uno script “bash” che si compone di poche migliaia di righe di codice ed è gestito dalla linea di comando. I sorgenti completi possono essere scaricati da GitHub per sostenere la collaborazione, il feedback, per fare delle domande ed avere il supporto al sistema, oppure per potersi creare una propria immagine basandosi sugli script

disponibili (Fig. 2). Esiste già un'immagine con un'installazione completa per lo script di installazione RetroPie, che viene fornita con tutti gli emulatori e le funzioni supportate.

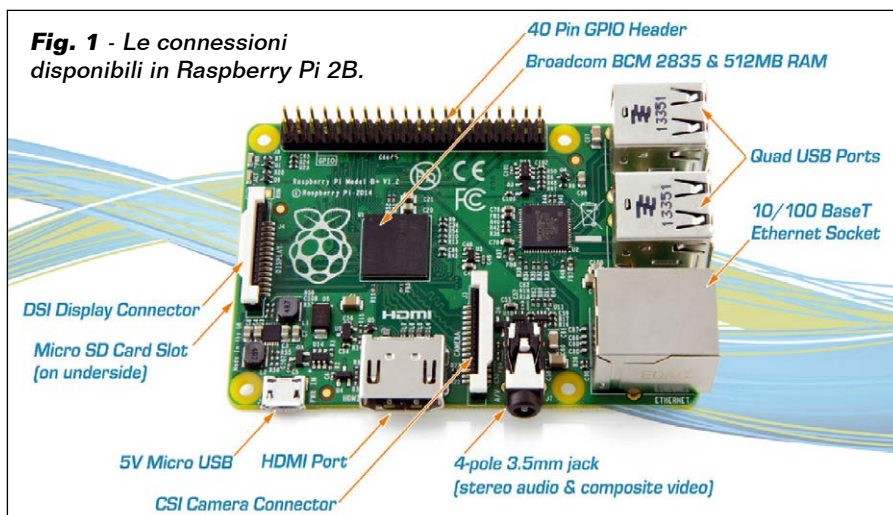
L'immagine RetroPie SD-card è pronta per l'uso e fornisce un'installazione completa di tutti i sistemi e le funzioni supportate dallo script di installazione RetroPie. Eccone le caratteristiche:

- installazione completa di tutti gli emulatori disponibili supportati dallo script di installazione RetroPie (vedere "Emulatori" tra qualche paragrafo);
- avvio automatico della stazione di emulazione, del front-end per la navigazione e l'esecuzione delle ROM; la stazione di emulazione è gestita da Aloshi (un front end che supporta la navigazione senza tastiera) che ospita anche le fonti di stazione di emulazione su GitHub;
- splash screen preconfigurato che può essere personalizzato con lo script RetroPie-Setup;
- azioni SAMBA (interoperabilità di programmi Windows per Linux) per ciascun sistema per copiare ROM sulla Raspberry Pi da rete;
- daemon USB per copiare ROM sulla Raspberry Pi da chiavetta USB.

EMULATORI

RetroPie permette di emulare molti giochi Arcade e le piattaforme gaming degli anni '90; incorpora i seguenti emulatori:

- 3do;
- Amstrad CPC;
- Apple II;
- Apple Macintosh;
- Atari 2600;
- Atari 5200 and 8 bit series;
- Atari 7800;
- Atari Jaguar;
- Atari Lynx;



- Atari ST/STE/TT/Falcon;
- Commodore 64;
- Commodore Amiga;
- FinalBurn Alpha;
- Intellivision;
- MAME;
- MSX;
- Neo Geo;
- Neo Geo Pocket (Color);
- Nintendo 64;
- Nintendo DS;
- Nintendo Entertainment System;
- Nintendo Game Boy;
- Nintendo Game Boy Color;
- Nintendo Game Boy Advance;
- Nintendo Virtual Boy;
- PC;
- PC Engine/TurboGrafx-16;
- PlayStation 1;
- ScummVM;
- Sega 32X;
- Sega CD;
- Sega Dreamcast;
- Sega Game Gear;
- Sega Megadrive/Genesis;
- Sega Master System;
- Sega Saturn;
- Sega SG-1000;
- Super Nintendo Entertainment System;
- Vectrex;
- Videopac or Odyssey2;
- WonderSwan (Color);
- Zmachine;
- ZX Spectrum.

Oltre a questi, esistono emulatori detti Ports che possono esservi installati a scelta:

- KODI;
- Minecraft Pi Edition;

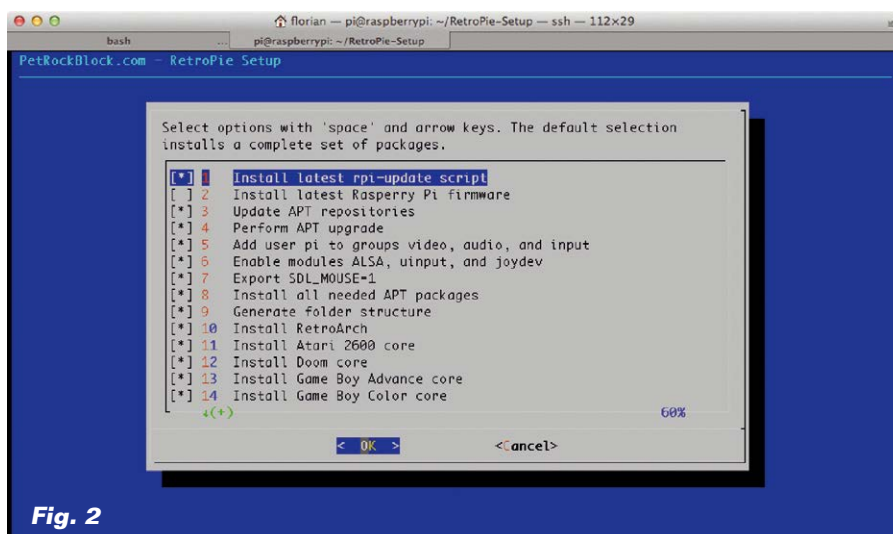


Fig. 2

- Cave Story;
- DOOM;
- Duke Nukem 3D;
- Quake Series;
- Descent;
- Super Mario War;

SHIELD MAME

Oltre al software RetroPie, il nostro gioco Arcade si compone di un hardware per il collegamento di game pad o joystick Arcade direttamente alla Raspberry Pi; nel progetto interfacciamo Raspberry Pi tramite una scheda realizzata apposta con l'integrato MCP23017, che abbiamo battezzato "shield MAME"; questa supporta il collegamento di un joystick Arcade e un massimo di 12 pulsanti.

Ricordiamo che per la maggior parte di ROM MAME servono 2 o 3 pulsanti, mentre se si vogliono emulare altre piattaforme consigliamo l'aggiunta di almeno 6 pulsanti "game" e 2 pulsanti "start e select". Lo shield non fa altro che collegarsi all'I²C-Bus

della Raspberry Pi attraverso l'integrato MCP23017 e trasmettere lo stato dei contatti dei pulsanti e del joystick. L'MCP23017 è un I/O Expander a 16 bit con interfaccia I²C che permette di gestire 2 gruppi da otto I/O ciascuno e serve perché con i soli GPIO della Raspberry Pi non potremmo gestire tutto. Invece così possiamo gestire fino a 8 joystick Arcade con 12 pulsanti ciascuno. A0-A1-A2 sono i pin per definire l'indirizzo I²C-Bus dello shield laddove se ne monti più d'uno (il massimo è 8); se montate due shield dovete impostare indirizzi differenti. L'indirizzo si imposta mediante i dip switch contenuti nell'SW1, collegato alle tre linee, ciascuna delle quali è provvista di un resistore di pull-up che la mantiene a livello logico alto quando il dip corrispondente è OFF (aperto).

I pin da GPB0 a GPB7 e GPA0 a GPA7, andranno collegati ai pulsanti (Fig. 3).

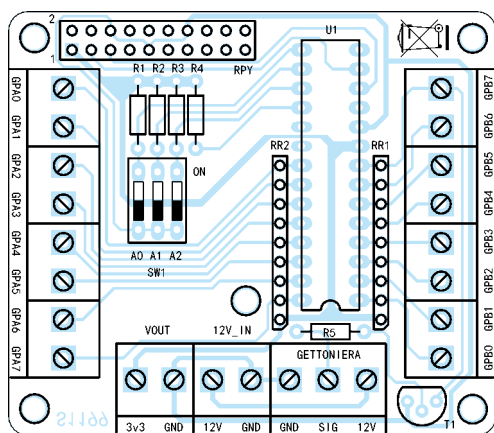
Dallo schema si può vedere che

abbiamo inserito un pin (SIG) per la gettoniera (trovate i dettagli su questo dispositivo nell'apposito riquadro che si trova in queste pagine) che è opzionale; infatti non fa altro che emulare la pressione di un tasto. Se non vogliamo utilizzare la gettoniera possiamo sostituirla con un pulsante da collegare bypassando il transistor T1, collegandolo quindi tra GPB7 dello shield e massa. Tutte le linee di input dell'MCP23017 sono dotate di resistore di pull up, grazie alle reti resistive RR1 ed RR2.

REALIZZAZIONE PRATICA

Per carrozzare il videogame potete scegliere tra il bartop (solo plancia e schermo) e il grande cabinato da sala giochi; noi abbiamo realizzato due tipi di case, il primo dei quali è una plancia con joystick, pulsanti e gettoniera e dietro ha una presa HDMI per poterlo collegare a qualsiasi televisore. È facilmente trasportabile e poco ingombrante.

[piano di MONTAGGIO]

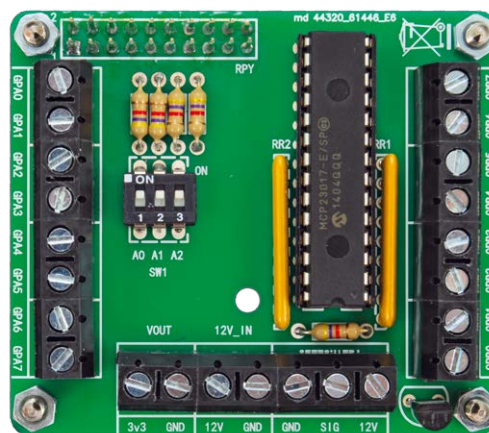


Elenco Componenti:

R1 ÷ R5: 4,7 kohm
 RR1, RR2: Rete resistiva 8x4,7kom
 + C
 U1: MCP23017

SW1: Dip-Switch 3 vie
 T1: BC547

Varie:
 - Zoccolo 14+14
 - Morsetto 2 vie (10 pz.)



- Morsetto 3 vie
 - Strip M/F 10 vie (2pz.)
 - Torretta M/F 15mm (4 pz.)
 - Vite 8 mm 3 MA (4 pz.)
 - Dato 3MA (4 pz.)
 - Circuito stampato S1199



Il progetto nasce da Nicola Salmoria, che ha iniziato a lavorare all'emulatore il 24 dicembre 1996 con l'idea di unire più emulatori trovati in rete (tra i quali un primitivo emulatore di Pac Man) in un unico programma in grado di eseguire più giochi. La prima versione, la 0.1, è stata rilasciata il 5 febbraio 1997; funzionante in ambiente MS-DOS da linea di comando, è in grado di eseguire cinque titoli. In breve tempo il progetto ha cominciato ad attirare l'attenzione di altri sviluppatori, che hanno iniziato a collaborare inserendo nuovi driver per il funzionamento dei videogiochi, grazie ad una

peculiare architettura del codice sorgente. Nel 1997 è cominciato lo sviluppo parallelo di MAME32 (ora MAMEUI), una versione adattata all'ambiente Windows a 32 bit con l'aggiunta di una GUI che ne facilita l'utilizzo. Nel 1998 la versione 0.34 raggiunge i 1.000 titoli supportati. Dal 2001 la versione ufficiale del MAME non viene più sviluppata sotto DOS ma sotto Microsoft Windows, sempre senza interfaccia grafica, come programma da linea di comando. L'ultima versione stabile di MAME è la 0.135, rilasciata l'1 novembre 2009: supporta 4.302 titoli (8.334 contando anche i vari cloni). Oggi il progetto è coordinato da Aaron Giles. MAME è composto da varie parti in grado di emulare completamente le architetture tipiche delle macchine Arcade: in pratica ricostruisce via software tutte quelle circuiterie interne che permettono di gestire gli input del joystick, la gestione del monitor e dei riproduttori di suoni. L'unica cosa che manca è la parte software delle macchine, cioè le ROM originali dei giochi, che per motivi di copyright non possono essere distribuite legalmente. Da un punto di vista logico MAME può

essere diviso in 3 livelli:

- il primo si occupa dell'emulazione dell'hardware vero e proprio;
- il secondo contiene tutte le funzioni generali e i moduli che fanno da collante tra il primo e il terzo livello;
- il terzo è quello che "presenta" l'emulatore all'utente ed è composto dall'interfaccia grafica (se presente) e da tutte quelle opzioni che permettono di avviare e pilotare l'emulatore.

I primi due livelli sono scritti in C in modo d'avere maggior velocità e portabilità. Nessuna libreria esterna è richiesta ed il codice è compilabile per qualunque piattaforma da un compilatore standard. Il terzo livello è definito OSD (Operating System Dependent) e contiene tutto quel codice specifico per le varie piattaforme; tutte le varie chiamate di sistema di Windows o di Linux si trovano in questo livello permettendo una facile migrazione del codice da una piattaforma all'altra modificando solo questo livello.

Le dimensioni della plancia sono 70x21 cm. Il secondo modello realizzato è il classico Bartop con il monitor incassato ...per i più nostalgici. La struttura può essere costruita con legno truciolare da 18 mm di spessore o MDF di pari spessore. Per quanto riguarda l'elettronica, lo shield per Raspberry Pi si ottiene preparando per prima cosa il circuito stampato di cui trovate le tracce sul nostro sito www.elettronica.in e poi montandovi i componenti, che sono tutti a montaggio tradizionale; preve-

dete uno zoccolo per l'integrato, la cui tacca di riferimento andrà orientata come mostrato nel piano di montaggio visibile in queste pagine. Il transistor T1 va montato solo se intendete collegare la gettoniera; idem per R5 e la morsettiere siglata GETTONIERA. Lo shield va completato montandogli le morsettiere richieste e, in corrispondenza delle piazzole siglate RPY, un connettore femmina a 20 poli su due file a passo 2,54x2,54 mm con terminali passanti (in modo da poter sovrapporre due shield), che può essere ottenuto semplicemente da due strip femmina da 10 poli a passo 2,54 mm. Terminato lo shield, inseritelo sul connettore GPIO della Raspberry Pi facendo combaciare i contatti 1 e 2 delle due schede, poi collegate, mediante le morsettiere, i pulsanti che nel frattempo avrete posizionato sul pannello frontale della console che avrete realizzato allo scopo. Fate altrettanto con il joystick e l'eventuale gettoniera

(o il pulsante da montare in sua sostituzione). Se montate due shield, la tensione di 12 V (necessaria solo se usate la gettoniera) va applicata solo a quello cui è connessa la gettoniera.

INSTALLAZIONE IMMAGINE SD

Una volta recuperato tutto il materiale necessario ed assemblato il cabinato, potremo iniziare l'installazione del software. Accendete prima di tutto il PC e scaricate il file immagine del RetroPie dal nostro sito (quello che forniamo contiene le configurazioni dei pulsanti secondo il layout di Fig. 5 e i BIOS degli emulatori, oltre alla gestione della gettoniera) quindi estraete il file zip e installate il file .img sulla scheda SD (state attenti che eventuali dati su di essa andranno persi). Da Windows potete utilizzare Win32DiskImager, su Mac RPI-sd card builder, mentre su Linux potete utilizzare, facendo molta attenzione al percorso della scheda, il comando:

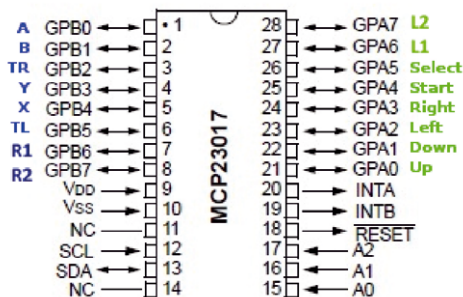


Fig. 3 - Pinout dell'I/O Expander e corrispondenza con pulsanti e movimenti del joystick.


```
sudo dd if=PERCOSO_FILE_IMG  
of=PERCOSO_SCHEDA_SD bs=1m
```

Consigliamo di utilizzare una scheda SD da 8 GB o superiore, in quanto l'immagine occupa circa 4 GB ed in seguito dovremo inserire le varie ROM dei giochi. A questo punto dovete (Fig. 4):

- 1) selezionare l'immagine che volete scrivere sulla scheda SD (se avete scaricato l'immagine prima dovete decomprimerla);
- 2) selezionare l'unità (device) della scheda SD; assicuratevi che sia il dispositivo corretto, perché la procedura cancella completamente tutti i dati sulla SD;
- 3) fare clic su "Write" e scollegare il dispositivo quando il processo è completo (il tempo dipende dalle prestazioni del vostro PC); ora disponete di una scheda SD di avvio per il vostro sistema.

Una volta completata la copia rimuovete la scheda SD dal PC e inseritela nella Raspberry Pi. Riprendiamo ora la Fig. 5, dove i nomi dei pulsanti sono quelli originari dei giochi Arcade. MAME utilizza i primi tre pulsanti in alto, mentre Nintendo usa i primi 6. In realtà aprendo i giochi, i pulsanti indicati nelle istruzioni sono A, B, C, che corrispondono in Fig. 5, rispettivamente a A, X e TR. Ci sono poi Start (bianco) che è in basso e Select (nero) che è sopra Start, duplicati per i due giocatori;

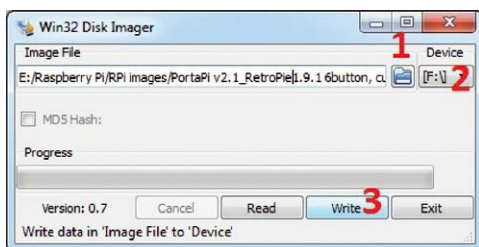


Fig. 4 - Creazione della SD-Card.



MAME è un software scaricabile e installabile gratuitamente sul proprio computer e distribuibile liberamente. Anche il codice sorgente è libero. È invece illegale vendere il MAME o il suo codice sorgente e trarre profitto dalla sua distribuzione. La norma recita: "La distribuzione di MAME sullo stesso mezzo fisico di copie illegali di ROM è rigorosamente proibita. Non è permesso distribuire MAME in alcun modo se si vendono o pubblicizzano CD-ROM illegali o altri mezzi contenenti ROM. Ciò vale anche se non si trae, direttamente o indirettamente, un guadagno da tale vendita. È permesso rendere disponibili ROM e MAME sullo stesso sito web, ma solo avvisando gli utenti sullo stato di copyright relativo alle ROM, e chiarendo che gli utenti non devono scaricare ROM se non hanno titolo per farlo." Per le ROM il discorso è diverso: il possesso di file immagine di memorie appartenenti a schede

logiche di giochi che non si possiedono può costituire una violazione del diritto d'autore. In pratica è come fare una copia di un gioco o un CD musicale di un conoscente invece di comprarlo. Per quanto ci riguarda, nel nostro sito inseriamo dei link alle sole ROM free e forniamo delle informazioni per la ricerca in rete delle altre ROM per chi avesse il diritto di usarle (autorizzazione dei detentori dei diritti, possesso delle schede logiche corrispondenti). Per acquisire il diritto di utilizzare una o più ROM si può chiedere l'autorizzazione ai legittimi proprietari, oppure si possono comprare delle schede logiche originali e utilizzare legalmente le ROM corrispondenti; i diritti li acquisite anche se comprate delle schede originali guaste. Un terzo sistema è acquistare il pannello di controllo HotRod Joystick prodotto dalla Hanaho, insieme al quale riceverete un set di ROM della Capcom, assolutamente legale.

premendoli entrambi si resetta il sistema e si esce forzatamente dal gioco. Per tutti i giochi, A seleziona, B esce; la combinazione di Start e Select fa retrocedere al menu dell'emulatore in cui ci si trova o al menu di riavvio del sistema; dal menu dell'emulatore, con B si va alla funzione ESCI e si torna al menu degli emulatori disponibili, scorribile con il joystick. Per effettuare i collegamenti tra lo shield e i pulsanti seguite la Tabella 1, dove sono già stati configurati i pulsanti degli emulatori che utilizzano RetroArch e i pulsanti del MAME; non ci resta che realizzare il cabinato

o controller Arcade, inserire le ROM dei giochi e iniziare a giocare. Nella nostra immagine su SD è attivata la gettoniera, ossia la lettura del pulsante collegato all'ingresso GPB7 dello shield.

PERSONALIZZARE L'IMMAGINE

Se non volete usare la nostra immagine ma creare un'installazione personalizzata, ad esempio per modificare il numero di giocatori, le impostazioni grafiche ecc., dovete:

- scaricare l'immagine "RetroPie" versione V3.0 dal sito <http://blog.petrockblock.com/retropie/retropie-downloads/>;

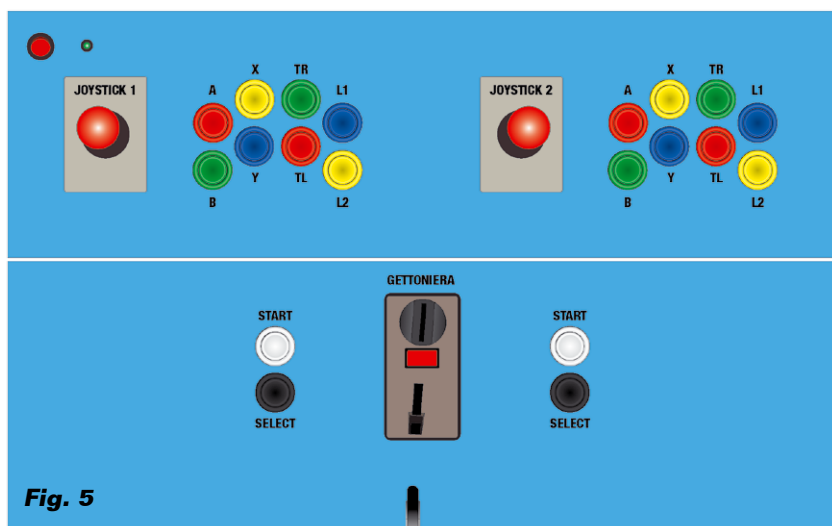


Fig. 5

- installare l'immagine su SD-Card come descritto nella precedente procedura;
- creata l'immagine, inserire l'SD-Card nella Raspberry Pi e connettere tastiera e cavo LAN (dopo l'avvio apparirà la schermata in **Fig. 6** e per il momento non sarà possibile gestire nessun joystick Arcade, ma solo la tastiera);
- premere due volte il pulsante F4 per entrare nel prompt dei comandi e verificate l'indirizzo IP della Raspberry Pi;
- con la tastiera, scrivere questo comando che vi permette di entrare nelle configurazioni avanzate della distro: *Sudo raspi-config*;
- cliccare su "Expand file system" per aumentare spazio sull'SD-Card (**Fig. 7**) altrimenti non ci sarà spazio per installare i vari pacchetti e le ROM; completata l'operazione, nella schermata che appare fare clic su OK;
- con i tasti freccia spostatevi verso destra della finestra e scegliete <Finish> (**Fig. 8**), allorché appare una nuova finestra in cui viene chiesto se procedere o meno al reboot; a questa richiesta dovrete rispondere scegliendo <Yes>.

Riavviate quindi la Raspberry

MCP21017 PIN (gpio)	Nome pulsanti	Equivalente in Retroarch
GPB0	A	8
GPB1	B	5
GPB2	TR	4
GPB3	Y	7
GPB4	X	3
GPB5	TL	6
GPB6	R1	//
GPB7	R2/SIG	1
GPA0	UP	trigger -1
GPA1	DOWN	trgger +1
GPA2	LEFT	trigger -0
GPA3	RIGHT	trigger +0
GPA4	START	11
GPA5	SELECT	10
GPA6	L1	0
GPA7	L2	9

Tabella 1 - Corrispondenza tra i GP I/O dello shield e i pulsanti; la colonna pulsanti in RetroArch indica il valore assegnato nel file di configurazione RetroArch al pulsante.

Pi e scollegate la tastiera; poiché una volta configurati i due controller, se avvieremo RetroPie con collegata la tastiera il software non riconoscerà i pulsanti e sarete costretti a riavviare nuovamente la Raspberry Pi. Se dovete effettuare delle modifiche mediante tastiera, questa va collegata solo dopo l'avvio di Emulstation.

Più avanti questo ci servirà per configurare i pulsanti del MAME. A questo punto su PC installate il software Mobaxterm, che è un emulatore di terminale, per controllare da remoto la Raspberry Pi; una volta che l'avete installato e aperto, fate clic su "session" in alto a sinistra e nella finestra appena aperta fate clic su SSH: si aprirà una schermata dove inserire l'indirizzo IP della vostra Raspberry Pi.

Dopo averlo inserito fate clic su OK (**Fig. 9**).

Nel campo sulla sinistra della schermata fate clic sull'indirizzo appena creato e inserite come user "pi" e per password: "raspberrypi".

A questo punto potrete inviare i comandi per installare e configurare i driver per i joystick con MCP23017; installate il kernel Linux DKMS (Dynamic Kernel Module Support) con:

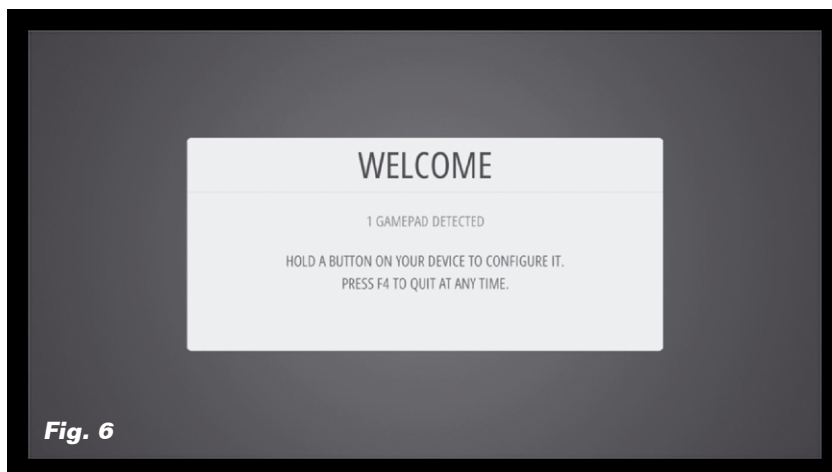


Fig. 6

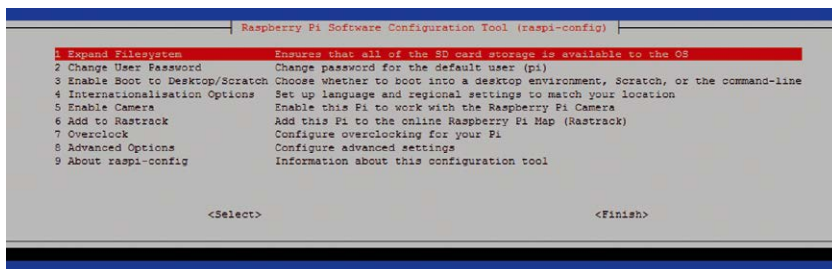


Fig. 7

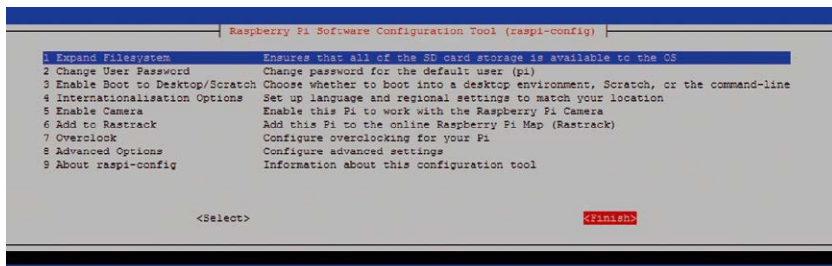


Fig. 8

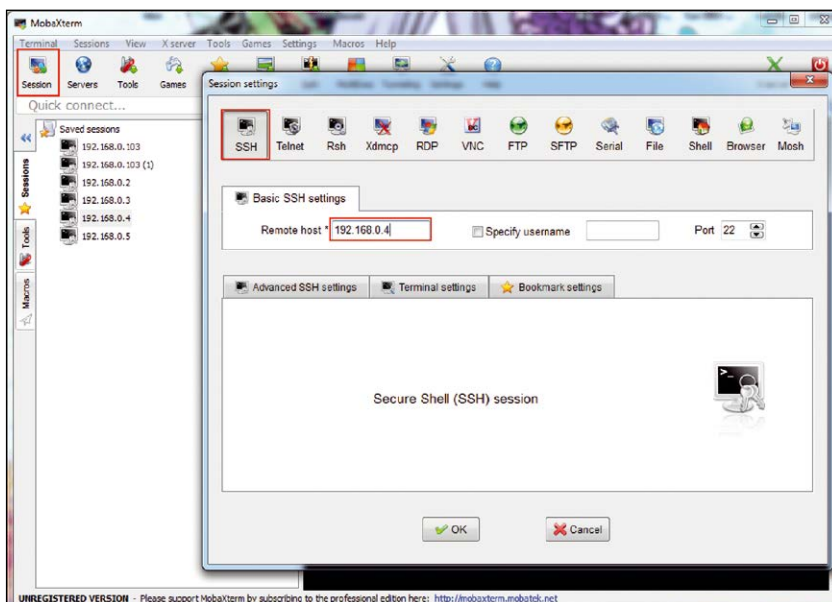


Fig. 9

```
sudo apt-get install -y --force-yes
dkms cpp-4.7 gcc-4.7 git joystick
```

Se verrà visualizzato un errore, impartite il comando:

```
sudo apt-get install -f
```

altrimenti proseguite con:

```
wget http://www.niksula.hut.
fi/~mhienka/Rpi/linux-headers-rpi/
linux-headers-`uname -r`_`uname
-r`_2_armhf.deb
a questo punto un messaggio
```

chiede se scaricare e installare i driver audio (AP-MODE), premete <Y> (sì) per eseguire. Digitate i comandi:

```
sudo dpkg -i linux-headers-`uname
-r`_`uname -r`_2_armhf.deb
sudo rm linux-headers-`uname
-r`_`uname -r`_2_armhf.deb
```

Ora dovete recuperare la cartella "build" che potete trovare sul sito della rivista. Troverete i file necessari al funzionamento dei joystick. Scompattata la cartella, copia-

tela attraverso MobaXterm nella cartella:

```
../home/pi
```

Entrate nella cartella "DEBIAN" con il comando:

```
cd /home/pi/build/mk-arcade-
joystick-rpi-0.1.4/DEBIAN/
```

cambiate i diritti d'amministratore di due file, altrimenti durante l'installazione non potrete configurare i joystick:

```
chmod 775 postinst
chmod 775 premm
```

Tornate ora alla cartella principale con il comando "cd" e successivamente impartite il comando:

```
cd build
```

Autorizzate l'installazione dei seguenti driver:

```
sed -i "s/\$MKVERSION/0.1.4/g" /home/pi/
build/mk-arcade-joystick-rpi-0.1.4/usr/
src/mk_arcade_joystick_rpi-0.1.4/* /home/
pi/build/mk-arcade-joystick-rpi-0.1.4/
DEBIAN/control /home/pi/build/mk-arcade-
joystick-rpi-0.1.4/DEBIAN/prerm
```

create il file di gestione dei due controller:

```
sudo dpkg-deb --build "mk-arcade-
joystick-rpi-0.1.4/"
```

```
sudo dpkg -i mk-arcade-joystick-
rpi-0.1.4.deb
```

e create l'indirizzo dei joystick:

```
sudo modprobe mk_arcade_joystick_
rpi map=0x20,0x21
```

In questo caso abbiamo fatto l'esempio con due controller, con indirizzo rispettivamente 0x20 e 0x21; l'indirizzo si decide

Chip Address	A2	A1	A0	I ² C Address
000	ON	ON	ON	0x20
001	ON	ON	OFF	0x21
010	ON	OFF	ON	0x22
011	ON	OFF	OFF	0x23
100	OFF	ON	ON	0x24
101	OFF	ON	OFF	0x25
110	OFF	OFF	ON	0x26
111	OFF	OFF	OFF	0x27

Tabella 2

mediante il dip-switch montato sullo shield, secondo la **Tabella 2**. Entriamo nell'editor testo del seguente file "modules":

```
sudo nano /etc/modules
```

e inseriamo le seguenti stringhe:

```
mk_arcade_joystick_rpi
map=0x20,0x21
i2c-bcm2708
i2c-dev
```

Salvate premendo "CTRL + O" e sovrascrivete il file. Premete CTRL+X per uscire. Entrate ora nell'editor di testo, aprite il seguente file "raspi-blacklist.conf" e inserite il cancelletto (#) nella stringa #i2c-bcm2708:

```
sudo nano /etc/modprobe.d/raspi-blacklist.conf
#i2c-bcm2708
```

Salvate premendo "CTRL + O" e sovrascrivete il file, quindi fate CTRL+X per uscire. Ora potrete testare i pulsanti e verificare che tutte le installazioni siano andate a buon fine, con il comando:

```
jstest /dev/input/js0
```

mettendo *js1* si testerà l'altro joystick. Uscite con CTRL+Z. Riavviate la Raspberry Pi con il comando `sudo reboot`.

PRIMO AVVIO E CONFIGURAZIONE CONTROLLER

A sistema completo ed SD inserita, accendete la Raspberry Pi; al primo avvio si presenterà

la schermata già vista in **Fig. 6** e dal menu si configureranno i controlli sia per Emulationstation (ovvero il front-end del sistema operativo) che per RetroArch Emulator, sistema che gestisce la configurazione di tutti gli emulatori (tranne MAME). Teniamo premuto un qualsiasi pulsante della console finché appare la schermata in **Fig. 10** e poi il menu di **Fig. 11**. Quando avrete configurato i movimenti del joystick e i pulsanti A-B-X-Y-LB-LT-RB-LB-START e SELECT, tenete pigiato un pulsante già configurato e apparirà la scritta "hold for 2S to skip": la configurazione terminerà e ulteriori comandi saranno ignorati. In questa fase la funzione dei pulsanti è:

- A: permette di confermare le scelte;
 - B: permette di uscire dagli emulatori e tornare indietro nelle schermate;
 - START: permette di entrare nel menu per cambiare pulsanti, spegnere riavviare il sistema;
 - SELECT: permette di entrare nel menu ed effettuare alcune impostazioni meno importanti, tipo inserire immagini per ogni ROM, oppure in caso di un lungo elenco di ROM, selezionarlo per lettera iniziale.
- Inoltre il software prevede degli

Hotkey, ossia scorciatoie da tastiera per attivare determinate funzioni (**Tabella 3**).

Una volta configurato il controller, avremo accesso alla schermata di selezione emulatore; non preoccupatevi se non ci trovate alcunché perché andranno inserite le ROM.

Soffermiamoci prima sull'emulatore RetroPie; in esso troviamo tutte le configurazioni per il nostro sistema operativo (**Fig. 12**).

Consigliamo di utilizzare una tastiera per le configurazioni, essendo più pratico e veloce, altrimenti si potrebbe utilizzare il controller Arcade appena installato. Vediamo una ad una le singole voci.

- **Configure audio setting:** permette di decidere se portare l'audio all'uscita HDMI (predefinita) o sulla presa jack, e selezionare il volume in uscita.
- **Configure retroarch/launch retroarch rgui:** modifica le impostazioni degli emulatori, audio, video, controller ecc. MAME non è contemplato perché bisognerà modificare manualmente le impostazioni.
- **Configure retroarch/joystick:** impostazioni da modificare solo se si utilizza per giocare una tastiera o un joystick analogico USB (controller PS3, PC,

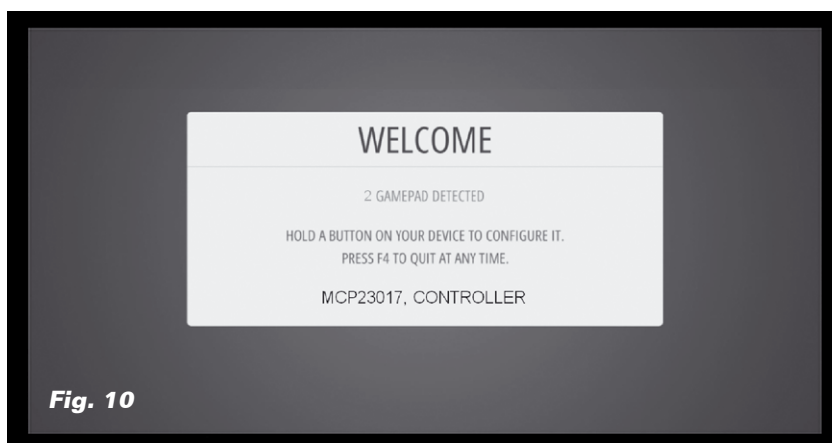
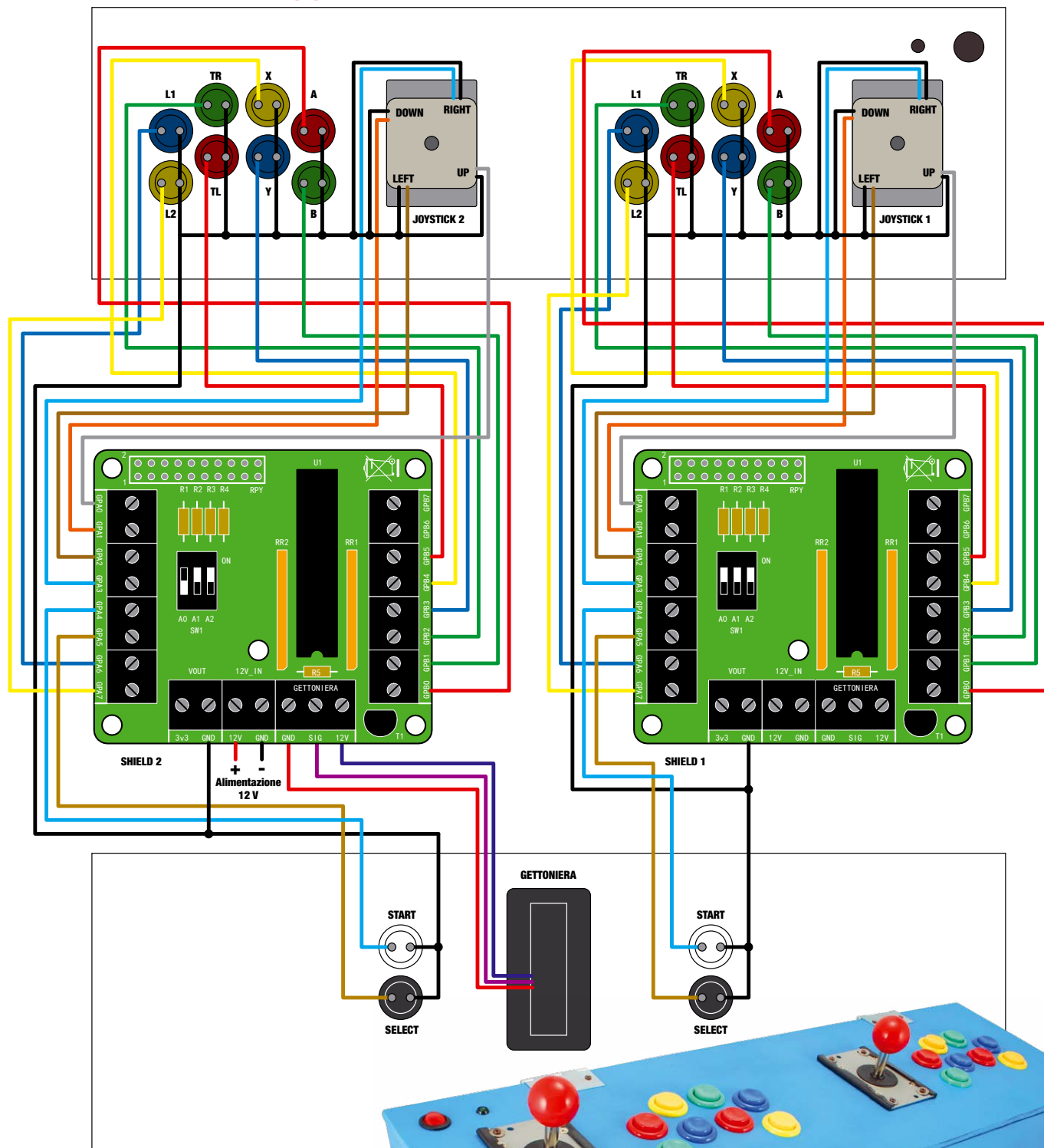


Fig. 10

Schema di cablaggio



XBOX, ecc.).

- **Configure retroarch net play:** si possono modificare le impostazioni della rete, solo in caso si tenga sempre collegata Raspberry Pi alla propria rete.
- **Configure splashscreen:**



Fig. 11

permette di scegliere, fra 30 temi, l'immagine all'avvio del sistema.

- **Configure wifi:** attraverso un dongle, la Raspberry Pi potrà essere collegata costantemente alla rete WiFi; scegliete la connessione desiderata e le credenziali per accedere.
- **Show Ip address:** visualizza l'indirizzo IP di Raspberry Pi.
- **Raspberry pi configuration tool raspi-config:** permette di accedere al menu di configurazione della Raspberry Pi, che normalmente si effettua con il comando `sudo raspi-config`. Vi troverete le voci seguenti.
 - **Expand Filesystem:** allarga la partizione sulla SD nei limiti dello spazio disponibile.
 - **Internationalisation Options:** per settare correttamente location, tastiera e time-zone.
 - **Advanced Options:** da questo menu potete modificare alcune impostazioni di default di Raspberry Pi, come il clock e i settaggi audio. Da qui potete anche abilitare l'SSH.

Le restanti impostazioni riguardano il cuore del sistema operativo e non vanno modificate.

Hotkey	Action
Select+Start	Exit
Select+X	RGUI Menu
Select+B	Reset

Tabella 3

TRASFERIMENTO DELLE ROM

Le ROM possono essere caricate:

- con una chiavetta USB;
- con una connessione SSH tramite un programma tipo Mobaxterm o PuTTY.

Nel primo caso procuratevi una chiavetta USB formattata e create una cartella con nome "retropie" quindi inseritela nella presa USB della Raspberry Pi. Riavviate quest'ultima e automaticamente nella Pen Drive saranno create le cartelle per ciascun emulatore. Estraiete la chiavetta USB ed inseritela nel vostro PC: notate che è stata creata una cartella *roms*, contenente tante sottocartelle quanti sono gli emulatori. Procuratevi il MAME romset versione 037b5, anche se potrebbero funzionare alcune ROM da altro romset.

Per gli altri emulatori è necessario fare delle prove di compati-

bilità. Trasferite le ROM che desiderate nelle rispettive cartelle. Una volta inserite le varie ROM nella chiavetta toglietela dal PC e inseritela nella Raspberry Pi, quindi accendete quest'ultima. Tutto il contenuto della chiavetta sarà trasferito sulla SD della Raspberry Pi.

Per sapere quando il caricamento è terminato, guardate il LED verde: quando smetterà di lampeggiare il processo sarà terminato. Non vi resta che selezionare l'emulatore e far partire un gioco. Vediamo adesso la procedura per caricare le ROM dall'emulatore di terminale tramite Mobaxterm (Fig. 13):

1. assicuratevi che la Raspberry Pi sia accesa e connessa al router;
2. dal PC installate e avviate Mobaxterm per poi connettersi al Raspberry Pi, indicando l'indirizzo IP (assegnato in DHCP) e immettendo Username: "pi" e password "raspberrypi";
3. raggiungete la cartella RetroPie/roms;
4. copiate le ROM nelle cartelle dei rispettivi emulatori;
5. riavviate la Raspberry Pi per visualizzare e utilizzare le ROM disponibili.

Notate che alcuni emulatori potrebbero non riconoscere le ROM perché necessitano del



Fig. 12

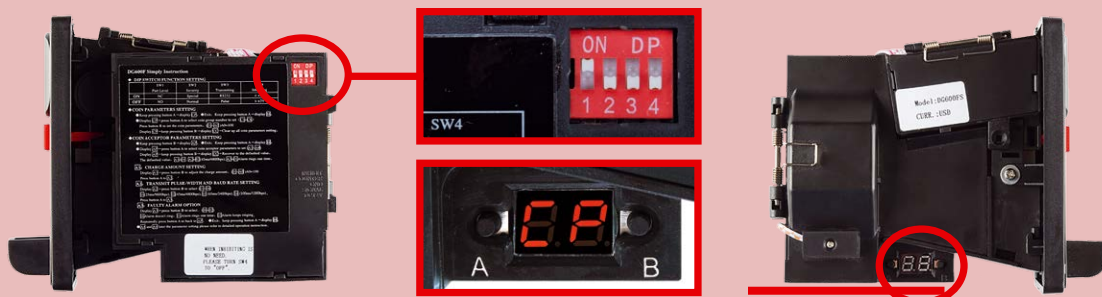
Installare la gettoniera



La gettoniera non è indispensabile ma l'abbiamo voluta per ricreare l'atmosfera delle sale giochi. Essa non fa altro che simulare la pressione di un pulsante ed inviare alla Raspberry Pi un numero di impulsi corrispondente al valore della moneta inserita. In questo progetto abbiamo programmato 3 monete: 50 cent = 1 credito, 1 euro = 2 crediti e 2 euro = 5 crediti. La gettoniera può lavorare con RS232 oppure come semplice contatto, esattamente come la usiamo noi. Sul lato sinistro ha due pulsanti e un piccolo display a 7 segmenti che ci permetteranno di programmare le monete a nostro piacimento; in alto ha un "dip-switch" per decidere come utilizzare la gettoniera: nel nostro caso dovremo mettere il dip 1 a ON e i restanti a OFF.

Fatto ciò iniziamo a programmare la prima moneta: tenendo premuto il pulsante "A" fino a quando non comparirà la scritta "CP", premiamo una volta il pulsante "A" e vedremo comparire "C1" (significa coin 1: prima moneta). A questo punto premiamo il pulsante "B" e vedremo incrementarsi il numero; questo equivale al numero di impulsi che la gettoniera invierà ogni qualvolta inseriremo quella moneta: mettiamo 01 ed inseriamo la moneta da 50 cent (si sentiranno dei "bip" vorrà dire che il valore della moneta è stato accettato). D'ora in poi ogni volta che inseriremo la moneta da 50 centesimi la gettoniera emetterà un impulso, l'equivalente nel gioco di un credito. Ripetiamo la procedura per tutte le altre monete:

C1:	01	equivale a un credito	(50 cent)
C2:	02	equivale a due crediti	(1 euro)
C3:	05	equivale a cinque crediti	(2 euro)



BIOS corrispondente: è il caso, ad esempio, della Playstation o di NeoGeo; per quest'ultimo salva-

te il file *bios NeoGeo.zip* (solo in questo caso, nella cartella *roms*). Altri emulatori come NES hanno bisogno di ROM con estensione .nes; SNES richiede ROM con estensione .smc e via di seguito. Trovate tutti i BIOS che vi occorrono alla pagina web <https://github.com/RetroPie/RetroPie-Setup/wiki/> con le istruzioni su dove salvarli, emulatore per emulatore. Qualora non visualizzate un emulatore su Emulationstation basta ripetere la procedura salvando almeno una ROM nella cartella di quell'emulatore. Al prossimo avvio sarà visibile e selezionabile, oppure attraverso connessioni SSH. In Fig. 14 vedete come appare la grafica di Emulationstation dopo aver caricato le ROM: con il joystick vi spostate a destra e a sinistra, mentre con il tasto "A" della vostra plancia entrate nell'emulatore. Selezionate un gioco e lanciatelo con il tasto "A". Il "B" è utilizzato dal front-end di Emulationstation per uscire

dal menu di un emulatore e per tornare alla schermata di selezione (schermata principale). Nel sito www.mamedev.org vengono messe a disposizione solo le ROM che è legale possedere e scaricare in quanto i detentori del copyright hanno dato l'autorizzazione o il copyright è decaduto.

Per provare l'emulatore scarichiamo ad esempio la ROM di STAR FIRE collegandoci a <http://www.mamedev.org/roms/starfire> e



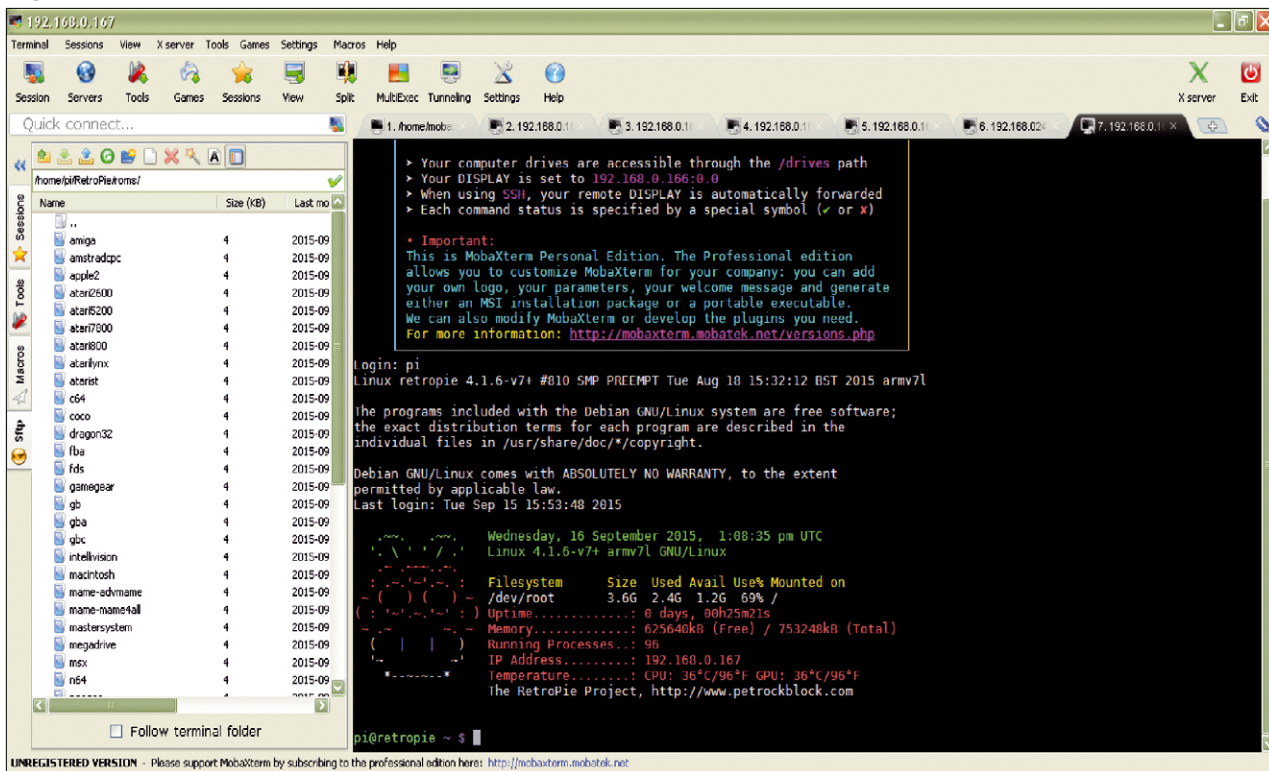
Guarda il nostro Retro Pie su YouTube!



Vuoi vedere in azione il Videogame Arcade con uno o due giocatori? Nulla di più semplice: collegati alla pagina <https://youtu.be/RqVdxCCqSDg>. Potrai anche vedere i video di moltissimi altri nostri progetti proposti in passato.



Fig. 13 - Caricamento delle ROM da MobaXterm.



iniziamo il download cliccando sulla voce *Download the Star Fire ROM images*, dopo aver spuntato l'opzione "I understand that these ROM images are for non-commercial use only".

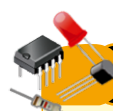
Ora carichiamo la ROM (*star-fire.zip*) nell'emulatore MAME copiandola nella cartella "mame-mame4all" tramite MobaXterm o chiavetta USB e riavviamo la Raspberry Pi. Dalla Emulationstation selezioniamo il MAME e con il joystick cerchiamo la voce Star Fire e avviamo il gioco premendo il tasto A.

Scegliamo ora i tasti da utilizzare:

1. premete TAB per entrare nel menu di configurazione;
2. andate in INPUT (General);
3. assegnate ai tasti le funzioni desiderate e impostare un tasto per terminare il gioco cambiando l'attuale UI Cancel, quindi uscite dal menu di configurazione.

Ora siete pronti per avviare il vostro videogame. Ricordiamo che molti siti mettono a disposizione un gran numero di ROM, ma per rimanere nella legalità (vedere il riquadro in queste pagine) scegliete quelle dei vostri giochi

preferiti assicurandovi di avere i diritti per farlo.
Buon divertimento!



per il MATERIALE

Presso Futura Elettronica è disponibile il kit completo di RetroPie per realizzare una postazione di gioco (cod. RETROPIEKIT) contenente un Raspberry Pi 2, uno shield di espansione, un joystick, nove pulsanti, un cavo di rete, un cavo HDMI, un alimentatore e una SD card con il software preinstallato a Euro 128,00. Il materiale necessario per realizzare questo progetto è acquistabile anche separatamente. Lo shield RetroPie per Raspberry Pi (cod. FT1199K) è in vendita a 13,00 Euro; la Raspberry Pi 2 (cod. 8326274RS) è disponibile a 44,00 Euro; il joystick arcade (cod. JOYSTICKASH) costa 17,00 Euro; i pulsanti (cod. COM093-disponibili in diversi colori) sono in vendita a 2,90 Euro ciascuno; la gettoniera (cod. GETTON1) a 61,00 Euro.

Il materiale va richiesto a:
Futura Elettronica, Via Adige 11,
21013 Gallarate (VA)
Tel: 0331-799775 • Fax: 0331-792287
<http://www.futurashop.it>



Fig. 14