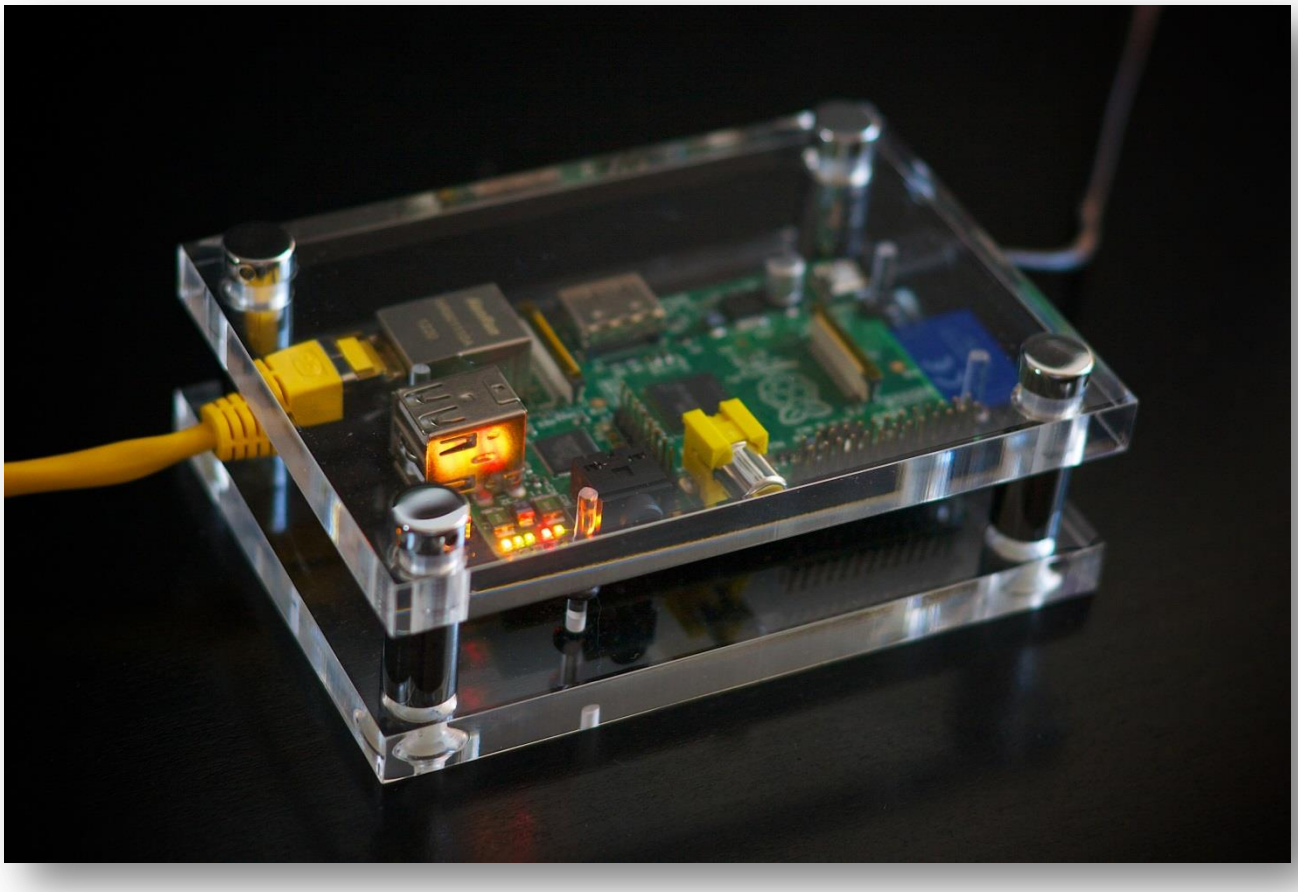
I.T.I. Aldini -Valeriani
2013/2014 (Bologna)

Via Bassanelli 9/11
40129 Bologna
TEL 0514156211

# *Raspberry Pi*

Andrea Sannino
5° INFORMATICA



# Sommario

**Introduzione** ..... **5**
Storia ..... 5
Fondazione ..... 5
Progetto Scuola 2.0 ..... 5

**Componenti Hardware** ..... **6**
Circuito System-on-Chip ..... 6
CPU ..... 6
GPU ..... 6
SDRAM ..... 6
Porte USB ..... 6
Output Video ..... 6
Output Audio ..... 6
Memoria ..... 6

Collegamenti di Rete ..... 6
Periferiche di basso livello ..... 6
Real-Time Clock ..... 6
Alimentazione ..... 6
Corrente Assorbita ..... 6
Dimensioni ..... 6
Considerazioni ..... 6
Altro ..... 6
DSI ..... 6
CSI ..... 6

**Sistema Operativo ..... 7**
Distro ..... 7
Raspbian Wheezy ..... 7
Kali Linux ..... 7

**Installazione e Configurazione iniziale ..... 8**
Formattazione SD Card ed Installazione Raspbian ..... 8
Windows ..... 8
Mac OS ..... 9
Raspi-Config ..... 10
Expand Filesystem ..... 10
Change User Password ..... 10
Enable Boot to Desktop/Scratch ..... 11
Internationalization Option ..... 11
Enable Camera ..... 11
Add to Rastrack ..... 11
Overclock ..... 12
Advanced Options ..... 12
About raspi-config ..... 13
startx ..... 13

**Amministrazione del Sistema** ..... **14**

IP Statico ..... 14

IP Dinamico ..... 15

DHCP ..... 15

DNS ..... 16

Protocollo Secure SHell ..... 17

Hard Disk Esterno ..... 19

File System ..... 19

Tabella Partizioni: MBR ..... 19

Tabella Partizioni: GPT ..... 20

Journaling ..... 20

mount ..... 21

Auto-mount ..... 23

Risparmio Energetico ..... 24

Espansione Memoria Swap ..... 25

**SAMBA** ..... **27**

Introduzione ..... 27

Demone ..... 27

Installazione ..... 27

Configurazione ..... 28

Gestione Permessi ..... 30

Utenti e Gruppi di Utenti ..... 30

File e Directory ..... 31

**Python** ..... **33**

Introduzione ..... 33

Analisi del Problema ..... 33

Script ..... 35

Esempio di Output ..... 35

Controlli Effettuati ..... 36

Temperatura ..... 36

Velocità/Utilizzo CPU ..... 36
Memoria RAM Libera/Utilizzata ..... 36
Memoria Swap Libera/Utilizzata ..... 36
Memoria HDD Libera/Utilizzata ..... 36
Memoria SD Libera/Utilizzata ..... 37
Sistema di Raffreddamento ..... 37

**Crontab ..... 38**
Come Funziona ..... 38
Operatori ..... 38
Campi ..... 39
Note ..... 39

**Altro ..... 40**
no-ip ..... 40
Installazione ..... 40
Configurazione ..... 42
Port Forwarding ..... 43
Esecuzione del demone all'avvio ..... 44

**Bibliografia ..... 46**

**Ringraziamenti ..... 47**

**Script Python ..... 48**

# ➢ INTRODUZIONE:

**Storia:**

Raspberry Pi è un low-cost single-board computer sviluppato all'inizio del 2012 nel Regno Unito dalla Raspberry Pi Foundation concepito per stimolare l'insegnamento di base dell'informatica e della programmazione nelle scuole e non solo:

Il prezzo del dispositivo ne permette la distribuzione anche in paesi in via di sviluppo dove quasi nessuno potrebbe permettersi l'hardware dei computer moderni.

**Fondazione:**

La Raspberry Pi Foundation è un'organizzazione di beneficenza il quale scopo è quello di promuovere lo studio dell'informatica (e argomenti attinenti) divertendosi, soprattutto nelle scuole.

La Fondazione promuove principalmente linguaggi come il Python e il Perl, notoriamente facili da imparare, ma sostiene anche l'uso del C, BASIC e di molti altri linguaggi supportati da Linux e ARM.

**Progetto Scuola 2.0**:

Nel mese di Gennaio per due settimane abbiamo partecipato ad un progetto, che potrebbe esser definito "in Beta Test", in cui siamo stati uniti ai Grafici del Professionale e suddivisi in gruppi, ognuno dei quali aveva un progetto diverso da portare a termine.

Il progetto portato avanti da me e dai miei compagni Scita Fabio e D'angelo Simone consisteva nel familiarizzare con Raspberry Pi installando il sistema operativo e facendo qualche test con GPIO, led e un termometro.

Il tempo a nostra disposizione è bastato appena per fare quanto descritto e, a me in particolare, per farmi "innamorare" di questo dispositivo, al punto di continuare ad utilizzarlo e a studiarlo a casa (per gentile concessione dei miei professori di Informatica), fino a decidere di portarlo come oggetto della mia Tesina.

## ➢ COMPONENTI HARDWARE:

| | |
|---|---|
| **Circuito System-on-Chip**: | Broadcom BCM2835 (CPU + GPU + DSP + SDRAM) |
| **CPU**: | 700 MHz ARM1176JZF-S core (famiglia ARM11) |
| **GPU**: | Broadcom VideoCore IV, OpenGL ES 2.0, 1080p30 H.264 high-profile decode |
| **SDRAM**: | 256 o 512 Megabytes (condivisa con la GPU) |
| **Porte USB**: | 2 (attraverso un hub USB integrato) |
| **Output Video**: | Connettore RCA per il video composito, HDMI |
| **Output Audio**: | 3,5 mm jack, HDMI |
| **Memoria**: | SD / MMC / SDIO card slot |
| **Collegamenti di Rete**: | Ethernet 10/100 (RJ-45) o WiFi (tramite adattatore wireless USB) |
| **Periferiche di basso livello**: | 2x13 header pins per GPIO, SPI, I²C, UART, +3,3 Volt, +5 Volt |
| **Real-Time Clock**: | No, però può essere "aggiunto" tramite una batteria tampone. |
| **Alimentazione:** | 5 V via MicroUSB o GPIO header pin |
| **Corrente** (potenza) **assorbita**: | 700 mA, (3,5 W) |
| **Dimensioni**: | 85,60 mm × 53,98 mm |

**Considerazioni:**

Tenendo conto che si hanno a disposizioni soli 512Mb di RAM (Modello B) e un processore da 700MHz, Raspberry Pi riesce comunque a fornire ottime prestazioni grazie ad un sistema operativo GNU/Linux che, combinato ad un processore ARM, è in grado di ottimizzare al meglio l'uso delle risorse del sistema.

**Altro:**

Sulla scheda sono inoltre presenti due connettori:

- connettore **DSI** (**Serial Digital Interface**): è l'interfaccia video che viene utilizzata nei monitor lcd / touchscreen di smartphone e tablet.
- Connettore **CSI** (**Camera Serial Interface**): è l'interfaccia utilizzata per collegare il modulo telecamera.

## ➢ SISTEMA OPERATIVO:

**Linux Distribution (Distro):**

A Linux distribution is a set of software components, assembled into a working whole, that constitute an operating system. In this specific case, it is based on a Linux Kernel.

Una distribuzione Linux è una collezione di programmi, relativi ad uno o più campi di applicazione, che vanno a costituire un sistema operativo, in questo caso basato su Kernel Linux.

**Raspbian:**

Raspbian is a free operating system based on Debian and optimized for the Raspberry Pi hardware. It comes with over 35.000 packages, with pre-compiled software that can be installed rapidly and effectively on your Raspberry Pi.

It was the first operating system available for Raspberry Pi when it was put on the market (on 2012) and it still remain one of the most installed operating system on this device because it provides periodic updates that improve the stability and performance of as many Debian packages as possible.

Raspbian è un sistema operativo gratuito, basato su Debian ed ottimizzato per l'architettura di Raspberry Pi, che viene rilasciato con oltre 35.000 pacchetti, completi di software pre-compilato per essere installato in maniera rapida ed efficacie.

E' stato il primo sistema operativo disponibile per Raspberry Pi quando, nel 2012, venne finalmente messo in commercio e rimane in ogni caso uno dei sistemi operativi più installati su questo dispositivo in quanto fornisce periodici aggiornamenti atti a migliorare la stabilità e le performance della board con il maggior numero di pacchetti Debian possibile.

**Kali Linux:**

Kali is a complete and free re-build of BackTrack Linux, developed according to Debian development standard, born to Penetration Testing and Security Auditing.

It count more than 300 penetration testing Tools and it support as many wireless devices as the Development Team possibly can. For more adventurous users, the entire Distro is customizable, from "which tools to install" all the way down to the kernel!

Since ARM-based systems are becoming more and more prevalent and inexpensive, the Development team made Kali's ARM support as robust as they can.

Kali è una ricostruzione completa e gratuita di BackTrack Linux, sviluppata secondo gli standard di sviluppo Debian, nata per fare test di penetrazione e controllo sicurezza.

Conta più di 300 penetration testing tools e supporta il maggior numero di dispositivi wireless possibile al team di sviluppo. Per gli utenti più avventurosi, l'intera Distro è personalizzabile, da "quali tool installare" fino ad arrivare al kernel! Dato che i sistemi ARM-based stanno diventando sempre più diffusi ed economici, il team di sviluppo ha reso il supporto di Kali ai sistem ARM il più robusto possibile.

## ➢ INSTALLAZIONE E CONFIGURAZIONE INIZIALE:

Per utilizzare Raspberry Pi è necessario installare una Distro Linux su di una scheda SD (Secure Digital). Nel mio caso utilizzo una **Sony SDHC UHS-I Classe 10** da **8Gb** con installato Raspbian.


**Formattazione della SD Card ed Installazione di Raspbian:**

Prima di installare il sistema operativo, è necessario scaricarne l'immagine e formattare la SD con un file system che Raspberry possa leggere (nel nostro caso FAT32, compatibile con qualsiasi sistema operativo).

Per scaricare l'immagine di **Raspbian**: Download Raspbian

Usando **Windows**:

- scaricare **Win32DiskImager**: Download Win32DiskImager
- selezionare la **SD** dal menù a tendina
- selezionare l'**immagine** da installare sulla SD
- cliccare su **Write** e attendere il termine del processo, dopo il quale sarà subito possibile utilizzare Raspberry Pi.

Usando **Mac OS**:

- Aprire **Terminal**
- Digitare diskutil list
- Identificare il **disco** (es. disk4) e non la partizione (es. disk4s1)
- Digitare diskutil unmountDisk /dev/disk4
- Digitare sudo dd if=/path/immagineRaspbian.img of=/dev/disk4 bs=1M

  dove path sta ad indicare il percorso in cui è contenuta l'immagine.

  immagineRaspbian.img è il nome completo dell'immagine scaricata dal link sopra indicato.
- Espellere infine la SD ed utilizzarla con Raspberry: sudo diskutil eject /dev/disk4

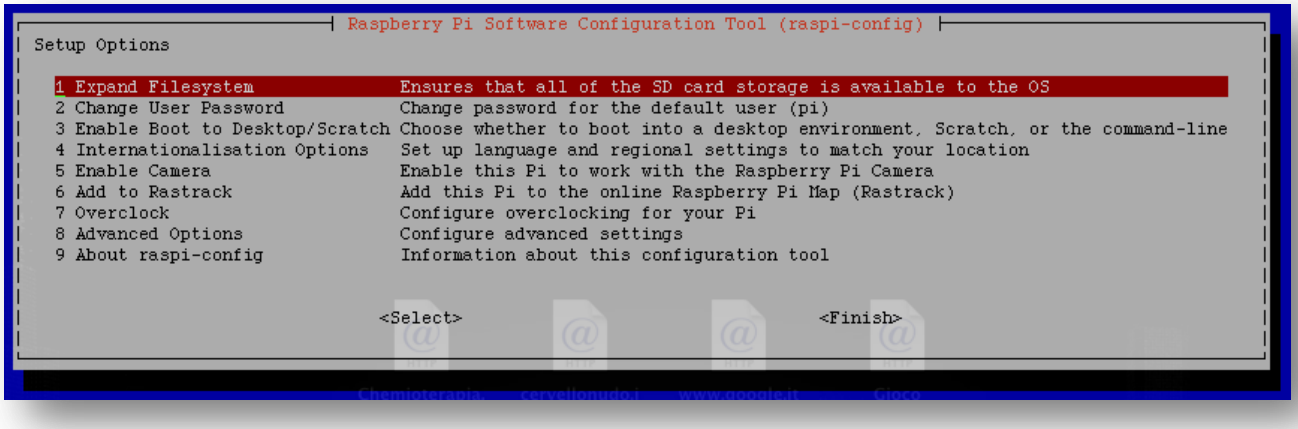
**Raspi-Config:**

Collegato uno schermo tramite ingresso HDMI / DVI, una tastiera tramite Hub USB, l'alimentazione tramite MicroUSB e, ovviamente, la SD nello slot apposito, Raspberry effettuerà il boot del sistema operativo, al termine del quale sarà chiesto di inserire:

**username**: pi

**password**: raspberry

Una volta effettuato il login, digitare subito 'sudo raspi-config' per utilizzare il tool di configurazione in interfaccia grafica.




Utilizzando le frecce direzionali 'su' e 'giù' è possibile scorrere il menù.

Premendo il tasto 'Invio' è possibile entrare nel secondo livello di una delle opzioni selezionate, all'interno della quale valgono queste stesse regole.

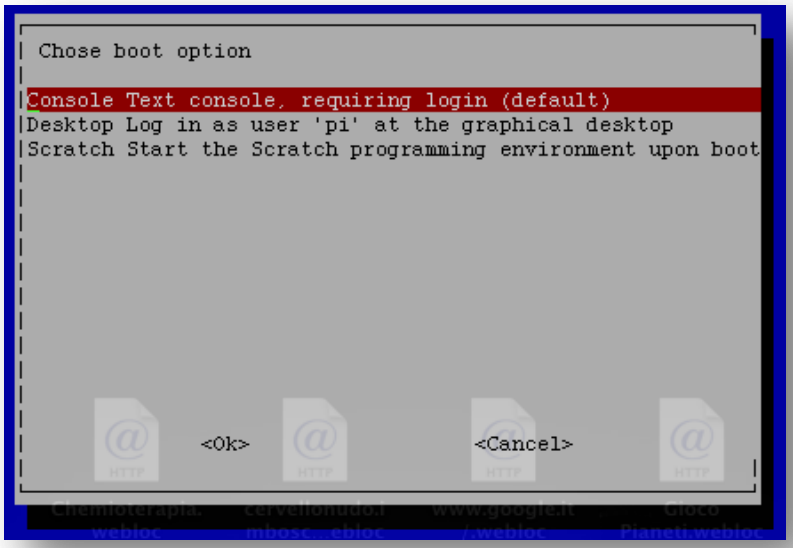
Utilizzando le frecce direzionali 'destra' e 'sinistra' è possibile selezionare <Select> / <Finish> o tornare al menù sopra menzionato.

- **Expand Filesystem:** Per permette a Raspberry Pi di utilizzare l'intera scheda di memoria e non solo la partizione creata per installare il sistema operativo (es. Raspbian 'pesa' circa 2gb, quindi la partizione ad esso riservata è di questa stessa dimensione, lasciando, nel mio caso, i restanti 6 gb totalmente inutilizzabili ed inaccessibili).

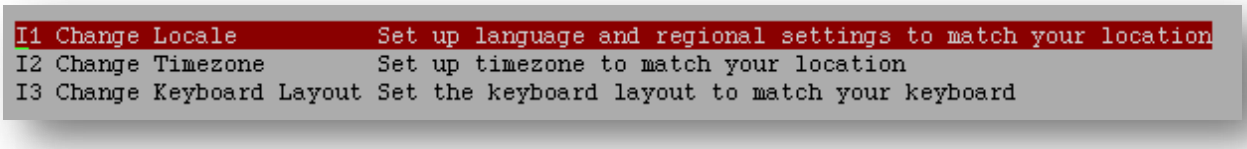
Il processo di espansione del File System inizierà immediatamente.

- **Change User Password**: Per modificare la password dell'utente '**pi**'.

- **Enable Boot to Desktop/Scratch**: Permette di modificare la modalità di avvio.

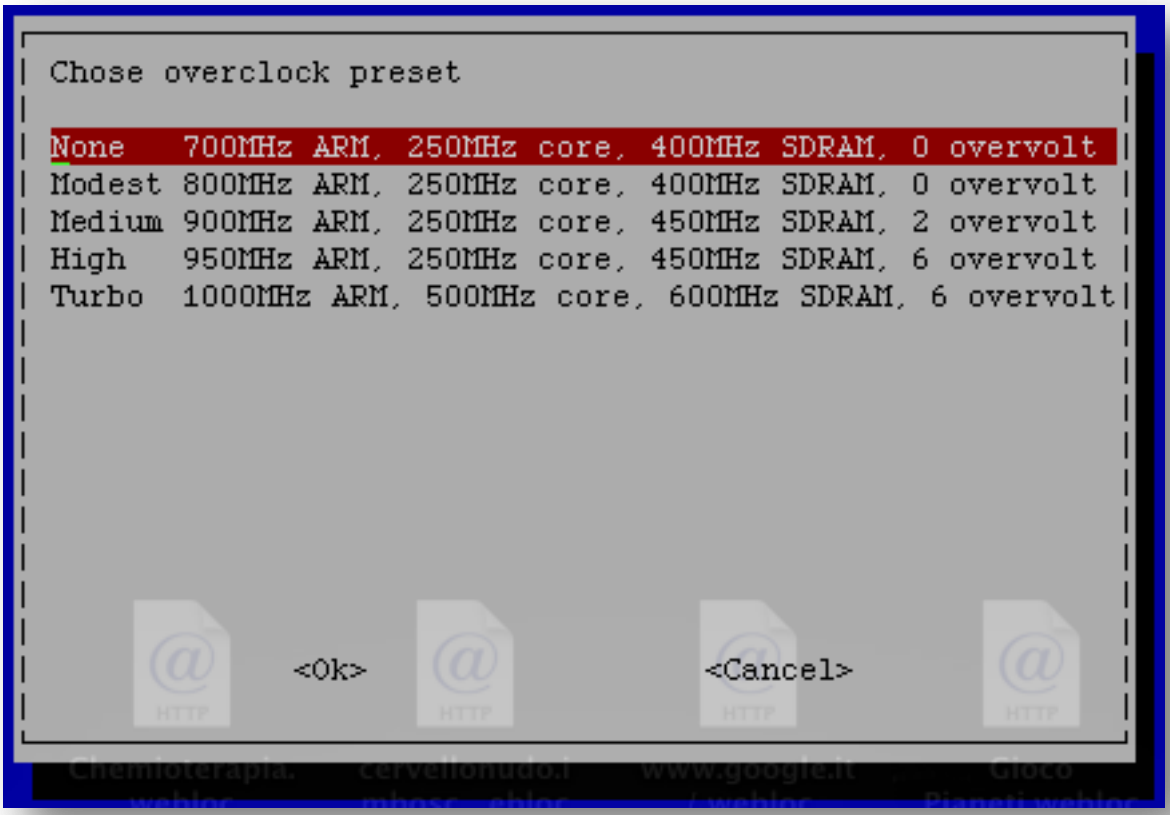



- **Internationalisation Options**: Permette di modificare i seguenti parametri

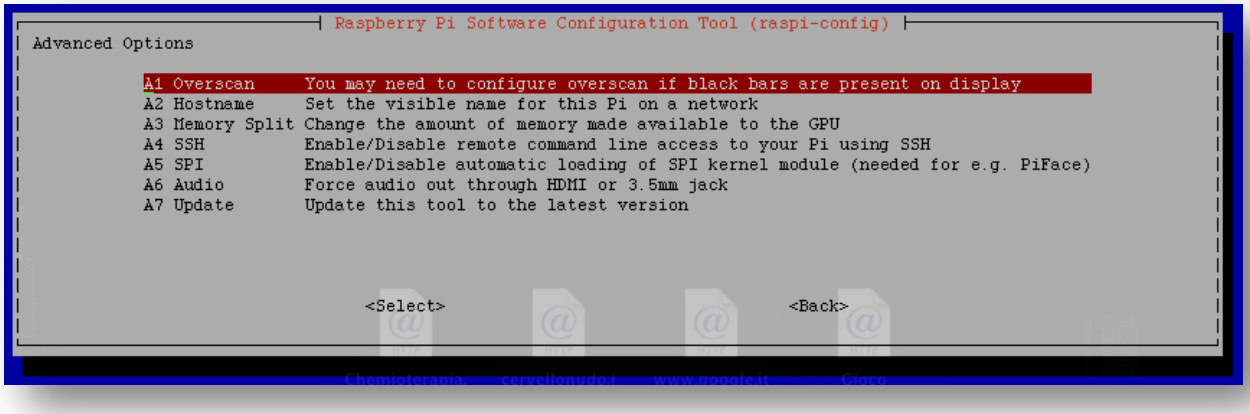



- **Enable Camera**: Permette di utilizzare il modulo della video-camera di Raspberry assicurando 128Mb della RAM alla GPU.
- **Add to Rastrack**: Permette di aggiungere la posizione di Raspberry (se connesso ad internet) sulla mappa di Rastrack (Mappa di Google Maps sulla quale sono indicati tutti i Raspberry che hanno questa opzione attivata).

- **Overclock**: Permette di overclockare la CPU fino ad 1000 MHz.




- **Advanced Options**: Permette di navigare attraverso il seguente menù:




  - **Overscan**: Per attivare/disattivare il "bordo nero" necessario ai vecchi schermi per visualizzare l'immagine completa (e non "tagliata").
  - **Hostname**: Permette di impostare in nome del dispositivo sulla rete locale.
  - **Memory Split**: Permette di modificare la quantità di memoria RAM riservata alla GPU.

- **SSH**: Permette di attivare/disattivare il protocollo SSH.
- **SPI**: Permette di attivare/disattivare il caricamento automatico del modulo Kernel SPI (per utilizzare dispositivi come PiFace).
- **Audio**: Per forzare le applicazioni ad utilizzare un'uscita audio piuttosto che un'altra (o di reimpostare "Auto").
- **Update**: Permette di aggiornare il tool "raspi-config" all'ultima versione disponibile.

- **About raspi-config**: Mostra le informazioni relative al tool "raspi-config".

Dopo aver settato Raspberry Pi attraverso questo tool, effettuare il Reboot (digitare 'sudo reboot') per attivare le modifiche.

**Startx:**

Dopo aver effettuato la configurazione iniziale ed effettuato il **Reboot** per confermare le modifiche, digitare startx per utilizzare l'interfaccia grafica di Raspberry Pi.

## ➢ AMMINISTRAZIONE DEL SISTEMA:

L'IP del dispositivo può essere settato a seconda delle necessità e della configurazione della rete.

Di fabbrica Raspberry Pi viene configurato per dipendere dal **DHCP**, quindi con indirizzamento dell'**IP dinamico**.

È consigliabile impostare un **IP Statico** se si utilizza il dispositivo in remoto per ovviare al rischio che l'IP venga cambiato dal DHCP della rete, impossibilitando l'accesso ad dispositivo stesso.

**IP Statico:**

Per configurare un'IP Statico, controllare l'**IP**, la **maschera di rete** e l'**IP di Broadcast** della propria rete "domestica", assicurarsi che l'IP scelto sia compreso nello **spazio di indirizzamento** del proprio Modem/Router (Contattare l'Amministratore della Rete) e modificare il file **interfaces** (sudo nano /etc/network/interfaces) come segue:

```
auto lo
#/etc/network/interfaces
iface lo inet loopback
#iface eth0 inet dhcp
auto eth0
iface eth0 inet static
        address 192.168.1.68
        netmask 255.255.255.0
_       broadcast 192.168.1.255

allow-hotplug wlan0
iface wlan0 inet manual
wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
iface default inet dhcp
```

Modificare opportunamente gli IP indicati dopo **address**, **netmask** e **broadcast** con gli IP adatti alla rete utilizzata.

**IP Dinamico:**

Per impostare un'IP Dinamico, settare lo stesso file **interfaces** (sudo nano /etc/network/interfaces) come segue:

```
auto lo
#/etc/network/interfaces
iface lo inet loopback
iface eth0 inet dhcp
auto eth0
#iface eth0 inet static
#       address 192.168.1.68
#       netmask 255.255.255.0
#       broadcast 192.168.1.255

allow-hotplug wlan0
iface wlan0 inet manual
wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
iface default inet dhcp
```

**NB:** In entrambi i casi, per confermare le nuove impostazioni, effettuare il **reboot** del sistema o, se non si è connessi tramite Protocollo SSH, **disattivare** e **riattivare l'interfaccia ethernet** digitando:

- sudo ifdown
- sudo ifup

**DHCP** (**D**inamic **H**ost **C**onfiguration **P**rotocol):

Impostando un **IP Dinamico**, si è in "balìa" del DHCP della rete, ovvero:

Ad ogni richiesta di accesso ad una rete IP (quale ad esempio **Internet**), al dispositivo sarà assegnato un IP **univoco** e tutte le informazioni necessarie per comunicare con la rete verranno auto-configurate dal DHCP stesso.

Al dispositivo verranno inoltre condivise le informazioni necessarie al corretto funzionamento del **servizio DNS** quali "Indirizzi dei Server DNS" e "Nomi di Dominio DNS".

**DNS** (**D**omain **N**ame **S**ystem):

Il DNS è un sistema utilizzato per la "traduzione" dei **nomi a dominio** degli **host** della rete in **indirizzi IP** e viceversa.

Per facilitare all'utente l'arduo compito che è la memorizzazione degli IP che identificano i siti internet, anziché utilizzare degli indirizzi IP (difficili da ricordare), agli **host** sono stati associati dei **nomi a dominio** (es. www.google.it) che poi un DNS tradurrà in indirizzo IP (es. 173.194.113.247), permettendo così all'applicazione utilizzata (es. Browser) di contattare l'host associato al nome a dominio indicato.

Per modificare manualmente la lista dei DNS Server utilizzati da Raspberry Pi, digitare:

sudo nano /etc/resolv.conf e modificare questo contenuto:

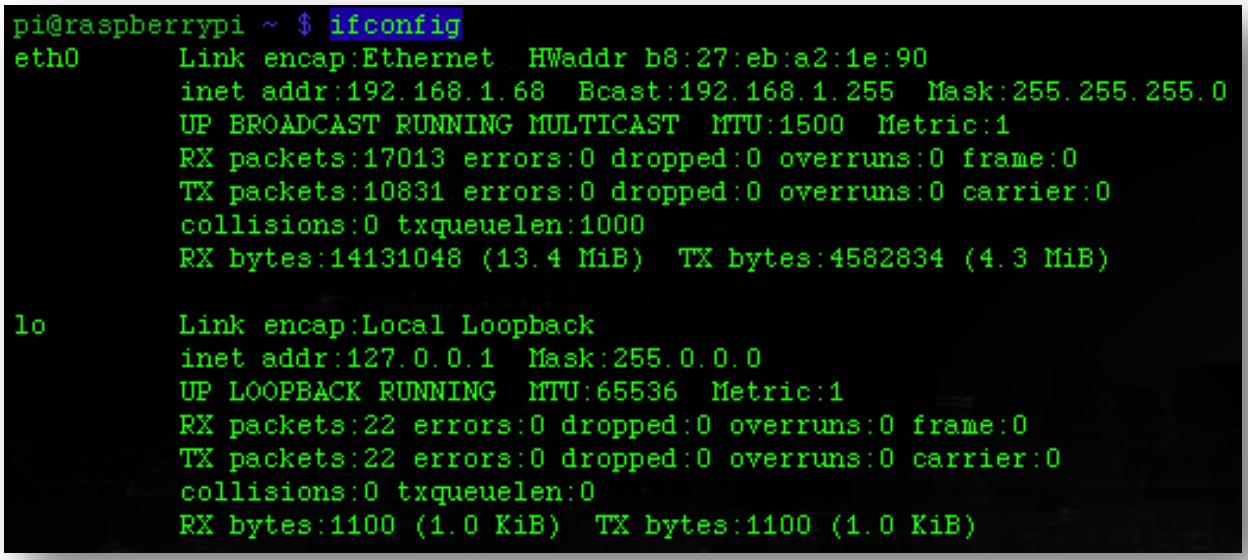
```
domain lan
search lan
nameserver 192.168.1.254
```

**Protocollo Secure SHell (SSH)**:

Per poter utilizzare l'interfaccia testuale di Raspberry Pi da un altro computer, e quindi in remoto, bisogna abilitare il **protocollo SSH** da **raspi-config**, conoscere l'IP di Raspberry Pi e username e password per accedervi.

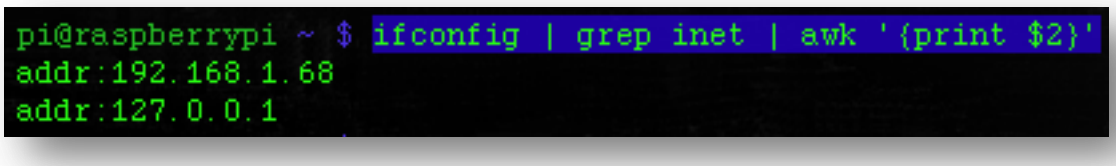
Una volta abilitato il protocollo e riavviato il dispositivo:

- Digitando ifconfig verranno mostrate le informazioni sullo stato corrente di una o più interfacce attive, in questo caso "eth0" (ethernet interface) e "lo" (localhost, o loopback interface).




- Per estrapolare l'IP della scheda di rete, digitare:

ifconfig | grep inet | awk '{print $2}'

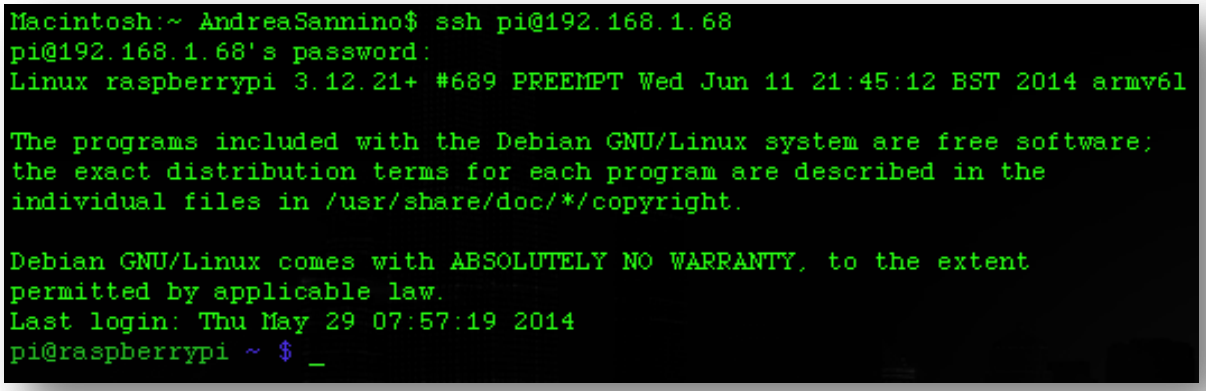



**eth0** è il primo risultato.

**lo** è il secondo risultato.

- E' possibile collegarsi a Raspberry Pi in remoto (trovandosi sulla stessa LAN) utilizzando **Putty** (Download Putty) da Windows, o **Terminal** da Mac OS.

  In questo caso ho usato il **Terminale Mac OS** per collegarmi, digitando la stringa 'ssh pi@192.168.1.68' e inserendo 'raspberry' come password:




Ora che la connessione è stabilita, si può utilizzare Raspberry Pi così come se si stesse digitando direttamente dalla tastiera ad esso collegata.

L'unica cosa che non è possibile fare da SSH, è avviare l'**interfaccia grafica** tramite 'startx'.

NB Se dovesse cadere la connessione o si dovesse spegnere / riavviare Raspberry Pi (anche se da comando BASH), la connessione SSH cadrà e sarà possibile ri-effettuarla solo dopo il ripristino della connessione alla rete (Locale).

**Hard Disk Esterno:**

Data la scarsa quantità di memoria di massa a disposizione (tutta quella rimasta inutilizzata dal sistema operativo, ergo pochi gigabyte), ho deciso di installare un Hard Disk esterno autoalimentato (Raspberry Pi non è in grado di reggere anche l'alimentazione di un'Hard Disk, rischia di spegnersi).

Queste sono le sue **Specifiche Tecniche**:

| | |
|---|---|
| Memoria: | 2000Gb |
| Standard USB: | 2.0 |
| Tabella di Partizioni: | GPT |
| File System: | HFS+ |

Montare una periferica è spesso facile.. spesso, non sempre!

Prima di parlare di "mount" del disco, facciamo un passo indietro e ripassiamo un po la teoria:

**File System:**

**HFS Plus** è un File System sviluppato da Apple per sostituire il precedente HFS come file system primario sui computer Macintosh e rientra nella **tabella di partizione GPT**.

Esso dispone della funzione (all'occorrenza disattivabile) di **Journaling** per migliorare l'affidabilità dei dati.

Il problema di questo file system è che non è leggibile né da Windows (se non utilizzando dei software a pagamento) né da Linux (se non installando dei pacchetti GRATUITI appositi).

**Tabella di Partizione MBR** (**M**aster **B**oot **R**ecord):

MBR è il vecchio standard per la gestione della partizione del disco rigido, ed è ancora ampiamente utilizzato da molte persone. L'MBR risiede proprio all'inizio del disco rigido e detiene le informazioni su come le partizioni logiche sono organizzate nell'Hard Disk.

Inoltre, l'MBR contiene l'eseguibile per caricare il codice / procedura di boot (avvio) per il sistema operativo .

Per un disco MBR si possono avere solo quattro partizioni primarie. Per creare più partizioni è possibile impostare la quarta partizione come partizione estesa e così si sarà in grado di creare più sotto-partizioni ( o unità logiche ) all'interno di esso.

Ogni partizione può essere di massimo di 2 Tb, per un totale massimo non superiore a 8Tb.

**Tabella di Partizione GPT** (**G**UID **P**artition **T**able)**:**

GPT è il più recente standard per la gestione delle partizioni di un disco rigido.

Si avvale di identificatori univoci globali ( **GUID** ) per definire la partizione ed è parte dello standard **UEFI**. Un sistema basato su UEFI ( che è richiesto per Windows 8 in funzione Secure Boot ) , è un obbligato ad utilizzare GPT.

Con GPT è possibile creare un numero di partizioni teoricamente illimitato sul disco fisso, anche se è generalmente viene limitato a 128 partizioni dalla maggior parte dei sistemi operativi.

Ogni partizione può contenere fino a 2^64 Byte, il che equivale a qualche miliardo di Terabyte!!

**Journaling:**

Il journaling è una tecnica utilizzata da molti file system moderni per preservare l'**integrità dei dati** da eventuali cadute di tensione. È una tecnologia derivata dal mondo dei **Database**.

Il journaling si basa sul concetto di **transazione**: ogni scrittura su disco è interpretata dal file system come una transazione.

Se la transazione ha "**successo**", il risultato dell'operazione è permanente.

Se la transazione ha "**insuccesso**", si deve tornare allo stato precedente all'inizio della transazione.

**Mount**:

Ora che abbiamo appreso la differenza tra MBR e GPT e tra Journaled e non Journaled, possiamo finalmente montare il disco su Raspberry Pi:

- Stampare l'elenco dei dispositivi connessi con le relative partizioni installate usando:

```
pi@raspberrypi /etc/network $ sudo fdisk -l
```

La parte che interessa a noi è questa:

```
WARNING: GPT (GUID Partition Table) detected on '/dev/sdb'! The util fdisk doesn't support GPT. Use GNU Parted.


Disk /dev/sdb: 2000.4 GB, 2000398934016 bytes
255 heads, 63 sectors/track, 243201 cylinders, total 3907029168 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x6622e4b0

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1  3907029167  1953514583+  ee  GPT
```

- In fondo al Warning si trova un consiglio molto utile: **Use GNU Parted.**
- Digitare sudo parted e successivamente help per l'elenco dei comandi utilizzabili:

```
pi@raspberrypi /etc/network $ sudo parted
GNU Parted 2.3
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) list
  align-check TYPE N                        check partition N for TYPE(min|opt) alignment
  check NUMBER                             do a simple check on the file system
  cp [FROM-DEVICE] FROM-NUMBER TO-NUMBER   copy file system to another partition
  help [COMMAND]                           print general help, or help on COMMAND
  mklabel,mktable LABEL-TYPE               create a new disklabel (partition table)
  mkfs NUMBER FS-TYPE                      make a FS-TYPE file system on partition NUMBER
  mkpart PART-TYPE [FS-TYPE] START END     make a partition
  mkpartfs PART-TYPE FS-TYPE START END     make a partition with a file system
  move NUMBER START END                    move partition NUMBER
  name NUMBER NAME                         name partition NUMBER as NAME
  print [devices|free|list,all|NUMBER]     display the partition table, available devices, free
space, all found partitions, or a particular partition
  quit                                     exit program
  rescue START END                         rescue a lost partition near START and END
  resize NUMBER START END                  resize partition NUMBER and its file system
  rm NUMBER                                delete partition NUMBER
  select DEVICE                            choose the device to edit
  set NUMBER FLAG STATE                    change the FLAG on partition NUMBER
  toggle [NUMBER [FLAG]]                   toggle the state of FLAG on partition NUMBER
  unit UNIT                                set the default unit to UNIT
  version                                  display the version number and copyright information
of GNU Parted
(parted) _
```

- Dando il print list, all si ottiene l'elenco di tutti i dispositivi connessi con le relative partizioni installate.

Questo è quello che ci interessa, **l'Hard Disk da 2Tb** con **Tabella di Partizioni GPT**:

```
Model: ST2000DL 001-9VT156 (scsi)
Disk /dev/sdb: 2000GB
Sector size (logical/physical): 512B/512B
Partition Table: gpt

Number  Start   End     Size    File system  Name                  Flags
 1      20.5kB  210MB   210MB   fat32        EFI System Partition  boot
 2      210MB   2000GB  2000GB  hfs+         LaCie
```

- Ora che sappiamo che il file system è l'**HFS+**, bisogna installare i pacchetti necessari ad interpretarlo:

```
pi@raspberrypi /etc/network $ sudo apt-get install hfsplus hfsutils hfsprogs
```

- E riparare il disco (per disattivare la funzione di Journaling) usando:

```
sudo fsck.hfsplus -f /dev/sdb1
```

Ora che il disco è riparato sarà possibile montarlo digitando semplicemente:

```
sudo mount -o force /dev/sdb1 /mnt/HFS/
```

**Auto-Mount**:

Così com'è impostato ora Raspberry Pi, ad ogni avvio saremo nuovamente obbligati a "cercare" il disco tra i dispositivi collegati, identificare la partizione e inviare il comando di mount indicato sopra rendendolo finalmente disponibile.

Per ovviare a questa noioso procedura:

- Trovare il codice **identificativo univoco universale** (**UUID**) del disco digitando blkid:

```
pi@raspberrypi /etc/network $ blkid
```

- Trovando questo codice:

```
/dev/sdb1: UUID="84315750-d0cc-4f4d-8987-514a89d7746c"
```

- Modificare il file fstab digitando sudo nano /etc/fstab:

```
pi@raspberrypi /etc/network $ sudo nano /etc/fstab
```

- E aggiungendo la linea sotto evidenziata (indicando al posto di 'LaCie' il punto di mount da voi preferito):

```
proc            /proc           proc    defaults          0       0
/dev/mmcblk0p5  /boot           vfat    defaults          0       2
/dev/mmcblk0p6  /               ext4    defaults,noatime  0       1
UUID=C4DC-08E5  /mnt/USB        vfat    rw,auto,users     0       0
UUID=84315750-d0cc-4f4d-8987-514a89d7746c       /mnt/LaCie      hfsplus rw,auto,users   0       0

# a swapfile is not a swap partition, so no using swapon|off from here on, use  dphys-swapfile s$
/mnt/USB/swapRaspy.img  none    swap    sw      0       0
```

facendo attenzione a separare le colonne con un **tab**.

- Effettuare il **reboot** del sistema.

**Risparmio Energetico:**

Essendo l'Hard Disk costantemente acceso per essere sempre a disposizione di Raspberry Pi, i consumi aumentano e di conseguenza la "vita" dello stesso si accorcia.

Per risolvere questo problema ho trovato un pacchetto precompilato che permette di gestire l'alimentazione e lo stato del disco: **hdparm**.

- Per installarlo ho utilizzato:

```
pi@JONES ~ $ sudo apt-get install hdparm
```

- Per settare un tempo di attesa prima dello spindown (stato di Stand-By) del disco pari a 5 minuti:

```
pi@JONES ~ $ sudo hdparm -S 60 /dev/sdb1
```

**NB**: Per indicare un tempo differente, consultare la pagina del manuale del tool usando il comando:

```
pi@JONES ~ $ man hdparm
```

**Espansione Memoria Swap:**

La memoria di Swap è un'estensione della capacità della memoria volatile complessiva del computer, oltre il limite imposto dalla quantità di RAM installata, attraverso l'utilizzo di uno spazio su un altro supporto fisico di memorizzazione, ad esempio il disco fisso.

E' consigliabile disporre di una memoria di Swap almeno doppia rispetto alla RAM installata.

Nei normali Hard Disk la Swap è un'intera partizione del disco, noi ci limiteremo a creare un file e a salvarlo sull'HDD appena installato.

- Spostarsi all'interno dell'HDD:

```
pi@raspberrypi ~ $ cd /mnt/LaCie/
```

- Creare un file immagine da 1024 Mb (il doppio della RAM) "riempiendolo" di zero binari:

```
pi@raspberrypi /mnt/LaCie $ dd if=/dev/zero of=/mnt/LaCie/swapfile.img bs=1M count=1024
```

- Renderlo swap:

```
pi@raspberrypi /mnt/LaCie $ mkswap /mnt/LaCie/swapfile.img
```

- Attivarlo lo swap su quel file:

```
pi@raspberrypi /mnt/LaCie $ swapon /mnt/LaCie/swapfile.img
```

- Per disattivare lo swap su quel file:

```
pi@raspberrypi /mnt/LaCie $ swapoff /mnt/LaCie/swapfile.img
```

**NB**: per eliminarlo basta un semplice rm /mnt/LaCie/swapfile.img, ricordarsi però di disattivare lo swap su quello stesso file prima di eliminarlo!!

- Per attivare lo swap su quel file ad ogni avvio:
  - modificare il file **/etc/fstab**:

```
pi@raspberrypi /etc/network $ sudo nano /etc/fstab
```

  - aggiungere la seguente stringa in fondo al file:

```
/mnt/LaCie/swapfile.img    none    swap    sw    0    0
```

- Reboot per confermare le modifiche.

## ➢ SAMBA:

**Introduzione**:

Dopo tutta la fatica fatta per installare l'Hard Disk esterno su Raspberry Pi posso finalmente condividerlo con gli altri dispositivi connessi alla rete "casalinga".

Per farlo ho scelto di usare Samba, un "demone" che permette appunto la condivisione di risorse nella rete locale LAN con pochi e semplici passaggi.

> **Demone**: programma che viene eseguito in background senza che sia sotto il controllo diretto dell'utente. Spesso avviato automaticamente al boot del sistema.

**Installazione:**

- Iniziare aggiornando il sistema:

```
pi@raspberrypi ~ $ sudo apt-get update && sudo apt-get upgrade
```

- Installare i pacchetti necessari:

```
pi@raspberrypi ~ $ sudo apt-get install samba samba-common samba-common-bin smbclient
```

- Fermare il demone di samba:

```
pi@raspberrypi ~ $ sudo service samba stop
```

**Configurazione**:

- Modificare il file di configurazione di Samba:

```
pi@raspberrypi ~ $ sudo nano /etc/samba/smb.conf
```

- Inserendo infondo al file:

```
[LaCie]
comment = Raspberry's HDD
browseable = yes
path = /mnt/LaCie
public = yes
guest ok = no
writable = yes
```

  - [**LaCie**] = nome visibile che identificherà il disco
  - **comment** = commento che descrive, ad esempio, cosa contiene il disco
  - **path** = percorso in cui si trova la cartella di mount del disco
  - **public** = **yes** per renderlo visibile tra i dispositivi di rete ad ogni dispositivo connesso alla stessa. **no** per non renderlo visibile.
  - **guest ok** = **yes** per condividere il disco e i permessi sotto-specificati a chiunque sia connesso allo stesso. **no** per disabilitare questa opzione.
  - **writable** = **yes** per abilitare la scrittura sul disco. **no** per renderlo Read-Only.

- Ora creiamo un nuovo utente che sarà usato come login per ogni nuovo collegamento al disco attraverso la rete:

```
pi@raspberrypi ~ $ sudo adduser --shell /bin/false nuovoUtente
```

- per abilitarlo anche alla shell:

```
pi@raspberrypi ~ $ sudo adduser --shell /bin/true nuovoUtente
```

- Inserire la password e seguire le istruzioni indicate fino a terminare il processo:

```
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for userna
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
```

- Impostare la password da utilizzare con l'utente appena creato per accedere al disco:

```
pi@raspberrypi ~ $ sudo smbpasswd -a nuovoUtente
```

- Effettuare il reboot di Raspberry Pi per confermare la nuova configurazione.

**PS**: per riattivare il demone di samba:

```
pi@raspberrypi ~ $ sudo service samba start
```

**Gestione Permessi**:

Ispirandomi al sistema presente in questo stesso istituto, ho deciso di creare una cartella "riservata" per ogni componente della famiglia, cosicché i dati importanti quali backup, password e qualunque altra cosa (a discrezione dell'utente stesso) possano sempre essere a sua disposizione all'interno delle mura casalinghe e da qualunque dispositivo.

Per far questo ho dovuto armeggiare un po' con i permessi riservati ad ogni utente nella cartella madre dell'Hard Disk seguendo questa procedura:

- **Utenti e Gruppi di Utenti**:
  - Prima di tutto ho dovuto creare il gruppo di utenti riservato a quelli utilizzati dai miei familiari. Ho scelto il comando:

```
pi@JONES ~ $ groupadd home
```

  - Dopodiché ho creato ogni utente, l'ho aggiunto al gruppo appena creato e vi ho associato una password per abilitarlo all'utilizzo dell'HDD attraverso il demone samba:

```
pi@JONES ~ $ sudo adduser --shell /bin/false andrea
```

```
pi@JONES ~ $ sudo usermod -g home andrea
```

```
pi@JONES ~ $ sudo smbpasswd -a andrea
```

- **File e Directory**:

Passiamo ora all'Hard Disk.

Ora che gli utenti sono stati creati, aggiunti al gruppo home appositamente creato e associata ad ognuno di questi una password per utilizzare il disco tramite samba, possiamo passare alla configurazione dei permessi del disco stesso e alla creazione delle cartelle di rete dedicate ad ogni utente:

- Per prima cosa modificare i proprietari della cartella sulla quale il disco è montato all'interno di Raspberry Pi utilizzando come proprietari l'utente root e il gruppo home:

```
pi@JONES ~ $ sudo chown root:home /mnt/LaCie/
```

- Assegnare i permessi di lettura, scrittura ed esecuzione ad entrambi secondo questa tabella:

```
chmod abc nomefile

a = utente proprietario
b = gruppo proprietario
c = qualunque altro utente che
    non rientra tra quelli sopra

7 = rwx
6 = rw
5 = rx
4 = r
3 = wx
2 = w
1 = x
0 = nothing
```

```
pi@JONES ~ $ sudo chmod 770 /mnt/LaCie/
```

- Ed assegnare infine ad ogni utente la propria cartella di rete precedentemente creata all'interno di /mnt/LaCie e settiamo i permessi (al proprietario 'andrea' e al gruppo 'root') di questa cartella e di tutti i file e directory al suo interno archiviati:

```
pi@JONES ~ $ sudo chown andrea:root /mnt/LaCie/Andrea
```

```
pi@JONES ~ $ sudo chmod -R 770 /mnt/LaCie/Andrea
```

## ➢ PYTHON:

**Introduzione**:

Come già detto all'inizio di questo documento, Raspberry Pi è nato anche per promuovere linguaggi come il Python (da qui è stato inserito nel nome "**Pi**", ovvero **Python Interpreter**), un linguaggio appunto di facile comprensione e apprendimento.

Non essendo da me partire con le cose "facili e noiose", ho saltato il classico "Hello World!" per dedicarmi subito ad un programma funzionale!

**PS**: per far capire il livello di difficoltà di questo linguaggio, ecco **helloworld.py**:

```
#!/usr/bin/python
#-*- coding: utf-8 -*-
print "Hello World!"
```

**Analisi del Problema**:

Dopo aver Overclockato Raspberry Pi, ho notato che, anche senza fare assolutamente **nulla** (demoni Samba e MySQL disattivati e periferiche smontate), il processore, come anche la scheda ethernet, era comunque abbastanza caldo per un dispositivo a risparmio energetico, quasi in modalità **Stand-By**.

Ho fatto un po' di ricerche su internet e ho scoperto che:

1. È possibile stampare la temperatura della scheda usando questo comando:

```
pi@raspberrypi ~/script $ /opt/vc/bin/vcgencmd measure_temp
temp=35.8'C
```

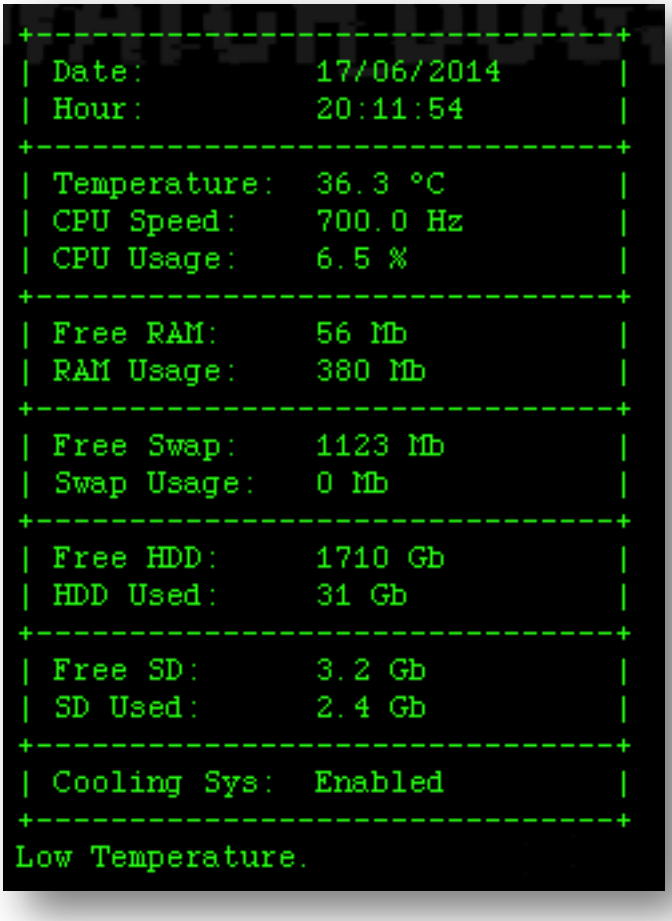
2. Quando il dispositivo viene sottoposto ad uno stress intenso, come un download che viene contemporaneamente zippato:
    - Il **Regolatore di Tensione in Ingresso** raggiunge temperature superiori a 60°C.
    - Il **Processore** raggiunge temperature superiori a 60°C.
    - La **Scheda Ethernet** raggiunge temperature superiori a 65°C.

Aspettando la consegna dei dissipatori appositamente tagliati per questi tre componenti, ho deciso di preservare l'integrità di Raspberry Pi scrivendo uno script python che, eseguito ogni minuto, controlla la temperatura della Board e "agisce di conseguenza".

**Script:**

*Per il sorgente dello script, andare all'ultima pagina di questo documento.*

**Esempio di Output:**



**Controlli Effettuati**:

- **Temperatura**:
  - Quando la temperatura rientra in un determinato range, viene considerata:

    Bassa / Media / Alta / Critica

  - Se è considerata "**Alta**" o "**Critica**", viene attivata una porta della GPIO che controlla un transistor. Quest'ultimo controlla l'alimentazione di una ventola di raffreddamento posta sopra al dispositivo.

    Verrà poi aggiornato un file di **log** all'interno del quale vengono specificate "Data", "Ora", "Temperatura" e "stato della ventola".

  - Se è considerata "**Media**" o "**Bassa**", la ventola viene spenta senza aggiornare il file di **log**.

  - Se è considerata "**Critica**" e rischia quindi di danneggiare il dispositivo, viene aggiornato un altro file di **log** con "Data", "Ora", "Temperatura", "Velocità CPU", "% Utilizzo CPU" e "stato della ventola" (per motivare lo **shutdown**) e infine viene spento Raspberry Pi utilizzando il comando sudo shutdown -h now.

- **Velocità** (Hz) e **Utilizzo** (%) della **CPU**:
  - Se la percentuale di utilizzo della CPU è tra il 100% e il 130%, viene scritto il file di **log** con le stesse informazioni della **temperatura Critica** senza fare altro **(no shutdown, no reboot)**.
  - Se supera il 130% (rendendo l'uso del dispositivo pressoché impossibile, confermo personalmente), viene scritto il file di **log** (come sopra) per poi esegui un **reboot** dell'intero sistema.

- **Memoria RAM Libera** e **Utilizzata** (Mb).

- **Memoria Swap Libera** e **Utilizzata** (Mb).

- **Memoria HDD Libera** e **Utilizzata** (Mb):
  - Se l'Hard Disk è stato montato correttamente (try), mostra il valore letto dal subprocess.check_output.
  - Se l'Hard Disk non è stato montato correttamente (o semplicemente non è connesso a Raspberry Pi) (except), mostra il valore "**N.A.**" (Not Available).

- **Memoria SD Libera** e **Utilizzata** (Mb):
    - Qui non ho effettuato il controllo "try/except" perché si presuppone che la scheda SD sia correttamente montato se si è in grado di utilizzare il dispositivo!

- **Sistema di Raffreddamento** (Stato pin4 GPIO):
    - Nonostante la temporanea assenza di un transistor che controlli l'accensione/spegnimento della ventola, lo script possiede già un algoritmo di controllo della porta GPIO assegnata.

Al momento, la ventola è collegata direttamente all'alimentazione del dispositivo, rimanendo quindi accesa, qualunque sia la temperatura dello stesso.

## ➢ CRONTAB:

Per rendere questi controlli automatici e soprattutto ciclici, ho inserito nel file di **crontab**:

```
pi@raspberrypi ~/script $ crontab -e
```

Questa stringa:

```
* * * * * sudo python /home/pi/script/temp.py
```

**Come Funziona**:

Raspberry Pi, come ogni altro sistema Unix / Unix-Like, dispone del comando 'crontab' che consente la pianificazione di comandi tramite la registrazione di questi presso il sistema per essere poi mandati in esecuzione periodicamente in maniera automatica dal suo demone (**crond**).

Quest'ultimo legge, una volta al minuto, il contenuto del crontab (per visualizzarlo: 'crontab -l') ed esegue i comandi pianificati il quale periodo di attesa si è esaurito.

**Operatori**:

Esistono diversi modi per specificare valori multipli in un campo:

- L'operatore virgola («,») specifica una lista di valori, ad esempio: «1,3,4,7,8».
- L'operatore trattino («-») specifica un intervallo di valori, ad esempio: «1-6», che equivale a «1,2,3,4,5,6».
- L'operatore asterisco («*») specifica tutti i possibili valori di un campo. Ad esempio, un asterisco nel campo dell'ora è equivalente a «ogni ora».

Esiste anche un operatore supportato da alcune versioni estese del cron, l'operatore slash («/»), che può essere usato per "saltare" un certo numero di valori. Ad esempio, «*/3» nel campo dell'ora equivale a «0,3,6,9,12,15,18,21»;

L'operatore «*» specifica «ogni ora», ma il «/3» indica che solo il primo (0), quarto (3), settimo (6), ecc. dei valori restituiti da «*» vengano usati.

**Campi**:

I primi cinque campi su ogni riga specificano **con che frequenza** e **quando** eseguire un comando.

```
.---------------- [m]inute: minuto (0 - 59)
|  .------------- [h]our: ora (0 - 23)
|  |  .---------- [d]ay [o]f [m]onth: giorno del mese (1 - 31)
|  |  |  .------- [mon]th: mese (1 - 12) OPPURE jan,feb,mar,apr...
|  |  |  |  .---- [d]ay [o]f [w]eek: giorno della settimana (0 - 6) (domenica=0 o 7)  OPPURE sun,mon,tue,wed,thu,fri,sat
|  |  |  |  |
*  *  *  *  *  comando da eseguire
```

**Note**:

- Per «giorno della settimana» (5° campo), sia 0 che 7 sono considerati il valore domenica.
- Se sia «giorno del mese» (3° campo) che «giorno della settimana» (5° campo) sono presenti sulla stessa linea, il comando viene eseguito quando almeno uno dei due è vero. Vedere l'esempio sotto.
- Il sesto campo e i successivi (ossia, il resto della linea) specificano il comando da eseguire.

## ➢ ALTRO:

Tra le tante cose che ho sperimentato su questo dispositivo straordinario, ho deciso di parlarvi anche della configurazione del **DNS no-ip**.

**no-ip**:

No-IP è un provider DNS dinamico che offre servizi a pagamento e gratuiti.

In particolare, offre servizi **DNS**, **e-mail**, il **monitoraggio della rete** e **certificati SSL**.

Nel mio caso specifico ho deciso di adottarlo per garantirmi un accesso remoto a Raspberry Pi dovunque mi trovi, anche se non sono connesso alla sua stessa rete LAN.

Dopo esserci iscritti sul sito www.no-ip.com e aver registrato uno dei tre domini disponibili gratuitamente (oltre i quali il servizio diventa a pagamento), possiamo dedicarci all'installazione e alla configurazione del software su Raspberry Pi.

- **Installazione**:
  - Creare una cartella da dedicare all'estrazione del software:

```
pi@raspberrypi ~/script $ mkdir /home/pi/noip
```

  - Scaricare il software:

```
pi@raspberrypi ~/noip $ wget http://www.noip.com/client/linux/noip-duc-linux.tar.gz
```

  - Ed estrarre:

```
pi@raspberrypi ~/noip $ tar vzxf /home/pi/noip/noip-duc-linux.tar.gz
```

- Compilare ed installare il software:

```
pi@raspberrypi ~/noip $ cd noip-2.1.9-1/
pi@raspberrypi ~/noip/noip-2.1.9-1 $ sudo make install
if [ ! -d /usr/local/bin ]; then mkdir -p /usr/local/bin;fi
if [ ! -d /usr/local/etc ]; then mkdir -p /usr/local/etc;fi
cp noip2 /usr/local/bin/noip2
/usr/local/bin/noip2 -C -c /tmp/no-ip2.conf

Auto configuration for Linux client of no-ip.com.
```

- Verrà poi chiesto di inserire uno username di login a no-ip o un indirizzo email usato per registrarsi sul sito stesso e la password associata all'account.

- **Configurazione:**
  - A questo punto, nel caso in cui abbiamo registrato più di un host sull'account indicato, verranno poste le seguenti domande (y/N):
  - Dopo aver affrontato questo passaggio, verranno chiesti ancora un paio di dati prima del termine della configurazione:

```
2 hosts are registered to this account.
Do you wish to have them all updated?[N] (y/N)  N
Do you wish to have host [host1.noip.me] updated?[N] (y/N)  N
Do you wish to have host [host2.noip.me] updated?[N] (y/N)  y
```

```
Please enter an update interval:[30] 30
Do you wish to run something at successful update?[N] (y/N)  N
Please enter the script/program name

New configuration file '/tmp/no-ip2.conf' created.
```

- **Port Forwarding**:

Nelle reti informatiche il Port Forwarding è l'operazione che permette il trasferimento dei dati da un computer ad un altro tramite una specifica **porta** di comunicazione.

Questa tecnica può essere usata per permettere ad un utente esterno di raggiungere un host con indirizzo IP privato (all'interno di una LAN) mediante l'apertura di una porta dell'IP pubblico dello stesso. Per compiere questa operazione si ha bisogno di un Router in grado di eseguire una traduzione automatica degli indirizzi di rete, detta **NAT**.

Per fare il Port Forwarding sul mio Modem/Router ho seguito una procedura che è differente per ogni Router in commercio.

In sintesi, mi sono connesso alla pagina di configurazione del mio router ed ho attivato l'applicazione **SSH** (Secure Shell: protocollo **TCP**/IP, Port **22**) sulla periferica **raspberrypi**.

| Gioco o applicazione | Periferica | Registro |
|---|---|---|
| Secure Shell Server (SSH) | raspberrypi | Acceso |

- **Esecuzione del demone all'avvio**:
  - Creare il demone:

```
pi@raspberrypi ~/noip/noip-2.1.9-1 $ sudo nano /etc/init.d/nomescript
```

  - Scrivere all'interno del file appena creato quanto segue:

```
#! /bin/sh
# /etc/init.d/noip2onoff
### BEGIN INIT INFO
# Provides:          noip
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Simple script to start a program at boot (from: www.stuffaboutcode.com)
# Description:       Script that start/stop a program at every boot/shutdown.
### END INIT INFO

# If you want a command to always run, put it here

# Carry out specific functions when asked to by the system
case "$1" in
  start)
    echo "Starting noip"
    # run application you want to start
    /usr/local/bin/noip2
    ;;
  stop)
    echo "Stopping noip"
    # kill application you want to stop
    killall noip2
    ;;
  *)
    echo "Usage: /etc/init.d/noip {start|stop}"
    exit 1
    ;;
esac

exit 0
```

  - Rendere lo script eseguibile:

```
pi@raspberrypi ~/noip/noip-2.1.9-1 $ sudo chmod 755 /etc/init.d/nomescript
```

- Testare per assicurarsi che tutto funzioni:

```
pi@raspberrypi ~/noip/noip-2.1.9-1 $ sudo /etc/init.d/noip2
Usage: /etc/init.d/noip {start|stop}
pi@raspberrypi ~/noip/noip-2.1.9-1 $ sudo /etc/init.d/noip2 start
Starting noip
pi@raspberrypi ~/noip/noip-2.1.9-1 $ sudo /etc/init.d/noip2 stop
Stopping noip
```

- Aggiungere lo script alla lista di programmi che vengono avviati in fase di boot:

```
pi@raspberrypi ~/noip/noip-2.1.9-1 $ sudo update-rc.d nomescript default
```

- Per eliminare lo script dalla lista:

```
pi@raspberrypi ~/noip/noip-2.1.9-1 $ sudo update-rc.d -f nomescript remove
```

## ➢ BIBLIOGRAFIA:

Le **Guide** da me utilizzate per fare quanto sopra descritto, così come le **pagine web** e i **manuali** utilizzati per reperire informazioni, definizioni e quant'altro, sono le seguenti:

Wikipedia: Raspberry Pi

Wikipedia: Distribuzione (Software)

Wikipedia: Distribluzione (Linux)

Raspbian

Word Reference

Google Translate

Kali Linux

Kali documentation

Raspberry Pi - Downloads

Raspberry Pi - Documentation

Wikipedia: DHCP

Partizioni

Wikipedia: Journaling

Rpy-Italia - Forum

Wikipedia: Port Forwarding

Wikipedia: Crontab

<Stuff about="code" />

## ➢ RINGRAZIAMENTI:

I miei ringraziamenti vanno:

- Agli organizzatori del progetto **Scuola 2.0** per avermi dato la possibilità di avvicinarmi a questo dispositivo.
- Ai professori **Santandrea Giovanni** e **Chou Li Zan** per avermi gentilmente concesso di continuare ad utilizzare il dispositivo anche oltre la fine del progetto Scuola 2.0,.

  Il tutto da casa, lasciandomelo "in comodato d'uso" [cit.] fino al termine degli Esami di Maturità.
- Speciali ringraziamenti vanno al **Forum Italiano Ufficiale di Raspberry Pi** e a tutti i suoi utenti, senza i quali avrei impiegato settimane per imparare quanto, grazie a loro, ho imparato in quasi due mesi di "**Hands-On**": Rpy-Italia - Forum

Ringrazio in particolare Turk e painbrain che si sono sempre resi disponibili e pazienti alle mie richieste di maggiori informazioni e delucidazioni.

## ➢ SCRIPT PYTHON:

```
#!/usr/bin/python
#-*- coding: utf-8 -*-
#-------------------------------------------------------------#
#   System Status Check Copyright (C) 2014 Andrea Sannino     |
#-------------------------------------------------------------#
#   This program is free software: you can redistribute       |
#   it and/or modify it under the terms of the                |
#   GNU General Public License as published by the            |
#   Free Software Foundation,                                 |
#   either version 3 of the License.                          |
#                                                             |
#   This program is distributed in the hope that it will      |
#   be useful, but WITHOUT ANY WARRANTY;                      |
#   without even the implied warranty of                      |
#   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.      |
#   See the GNU General Public License for more details.      |
#                                                             |
#   GNU General Public License:                               |
#   http://www.gnu.org/licenses/                              |
#                                                             |
#-------------------------------------------------------------+
#        System Status Check - Emergency Shutdown             |
#              & Cooling System Automation                    |
#-------------------------------------------------------------+
# Board:         Raspberry Pi*                                |
# Author:        Andrea Sannino                               |
# Site/Forum:    http://rpy-italia.org/                       |
# Date:          16/05/2014                                   |
# Version:       1.2.3                                        |
# License:       GNU General Public License v.3               |
#-------------------------------------------------------------+
# Update:        1.0.1                                        |
# Date:          18/05/2014                                   |
# Object:        Table View                                   |
#-------------------------------------------------------------+
# Update:        1.0.2                                        |
# Date:          19/05/2014                                   |
# Object:        CPU USAGE, CPU SPEED                         |
#-------------------------------------------------------------+
# Update:        1.0.3                                        |
# Date:          20/05/2014                                   |
# Object:        Various Updates by Turk                      |
#-------------------------------------------------------------+
# Update:        1.0.4                                        |
# Date:          21/05/2014                                   |
# Object:        Updates by Turk: Log Temperature             |
#-------------------------------------------------------------+
# Update:        1.1.0                                        |
# Date:          21/05/2014                                   |
# Object:        Log Temperature & CPU Speed, Usage           |
#-------------------------------------------------------------+
# Update:        1.1.1                                        |
# Date:          22/05/2014                                   |
# Object:        Various Updates + Temp_LOG                   |
```

```
#-----------------------------------------------------------+
# Update:       1.1.2                                       |
# Date:         23/05/2014                                  |
# Object:       RAM & Swap Check > Temp.log                 |
#-----------------------------------------------------------+
# Update:       1.1.3                                       |
# Date:         12/06/2014                                  |
# Object:       Bug Fixes: ts, Uso_CPUs, etc.               |
#-----------------------------------------------------------+
# Update:       1.1.4                                       |
# Date:         13/06/2014                                  |
# Object:       Bug Fixes: subprocess.call                  |
#-----------------------------------------------------------+
# Update:       1.2.0                                       |
# Date:         16/06/2014                                  |
# Object:       Free/Used Memory: HDD & SD                  |
#-----------------------------------------------------------+
# Update:       1.2.1                                       |
# Date:         17/06/2014                                  |
# Object:       Free/Used HDD: mounted/unmounted            |
#-----------------------------------------------------------+
# Update:       1.2.2                                       |
# Date:         17/06/2014                                  |
# Object:       \t Indentation                              |
#-----------------------------------------------------------+
# Update:       1.2.3                                       |
# Date:         25/06/2014                                  |
# Object:       Ottimizzazione output log (tshut.log)       |
#-----------------------------------------------------------#

import RPi.GPIO as GPIO
import subprocess
import time
import datetime

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(4, GPIO.OUT)

low=45
average=55
high=60
critical=65

shut=0    #livello allarme -> se =0 no problem, =1 reboot, =2
shutdown

Temp_LOG= open("/home/log/temp.log","a") #Log Temperature + Stato
Ventola
File_LOG= open("/home/log/tshut.log","a") #Log Shutdown: t,
Uso_CPU, Vel_CPU

Data= datetime.datetime.now().strftime('%d/%m/%Y') #es. 01/01/2001
Ora= datetime.datetime.now().strftime('%H:%M:%S') #es. 01:01:01
t= float(subprocess.check_output(["/opt/vc/bin/vcgencmd
measure_temp | cut -c6-9"],shell = True)[:-1]) #es. 41.1
ts=str(t)
```

```
Uso_CPU= ((float((subprocess.check_output(["uptime | awk '{ print
$10 }' | cut -c1-4"],shell= True)[:-1]).replace(',','.')))/2)*100
#media ultimi 5:1.2
Uso_CPUs=str(Uso_CPU)
Vel_CPU= float((subprocess.check_output(["cat
/sys/devices/system/cpu/cpu0/cpufreq/scaling_cur_freq"],shell=
True)[:-1]))/1000 #es. 700
Vel_CPUs=str(Vel_CPU)
Total_RAM= int(subprocess.check_output(["free -m | grep Mem | awk
'{print $2}'"],shell=True)[:-1]) #es. 437
Total_RAMs=str(Total_RAM)
Uso_RAM= int(subprocess.check_output(["free -m | grep Mem | awk
'{print $3}'"],shell=True)[:-1]) #es. 136
Uso_RAMs=str(Uso_RAM)
Free_RAM= int(subprocess.check_output(["free -m | grep Mem | awk
'{print $4}'"],shell=True)[:-1]) # es. 301
Free_RAMs=str(Free_RAM)
Total_Swap= int(subprocess.check_output(["free -m | grep Swap |
awk '{print $2}'"],shell=True)[:-1]) #es. 1024
Total_Swaps=str(Total_Swap)
Uso_Swap= int(subprocess.check_output(["free -m | grep Swap | awk
'{print $3}'"],shell=True)[:-1]) #es. 215
Uso_Swaps=str(Uso_Swap)
Free_Swap= (subprocess.check_output(["free -m | grep Swap | awk
'{print $4}'"],shell=True)[:-1]) #es. 785
Free_Swaps=str(Free_Swap)
try:
     Free_HDD= int(subprocess.check_output(["df -BG | grep LaCie |
awk '{print $4}'"],shell=True)[:-2]) #es. 1735
except:
     Free_HDD= "N.A."
Free_HDDs=str(Free_HDD)
try:
     Uso_HDD= int(subprocess.check_output(["df -BG | grep LaCie |
awk '{print $3}'"],shell=True)[:-2]) #es. 265
except:
     Uso_HDD= "N.A."
Uso_HDDs=str(Uso_HDD)
Free_SD= float(subprocess.check_output(["df -h | grep rootfs | awk
'{print $4}'"],shell=True)[:-2]) #es. 3.2
Free_SDs= str(Free_SD)
Uso_SD= float(subprocess.check_output(["df -h | grep rootfs | awk
'{print $3}'"],shell=True)[:-2]) #es. 4.7
Uso_SDs= str(Uso_SD)

print ""
print "+------------------------------+"
print "| Date:\t\t",Data+"\t|"
print "| Hour:\t\t",Ora+"\t|"
print "+------------------------------+"
print "| Temperature:\t",t,"°C\t\t|"
print "| CPU Speed:\t",Vel_CPU,"Hz\t|"
print "| CPU Usage:\t",Uso_CPU,"%\t\t|"
print "+------------------------------+"
print "| Free RAM:\t",Free_RAM,"Mb\t\t|"
print "| RAM Usage:\t",Uso_RAM,"Mb\t\t|"
print "+------------------------------+"
print "| Free Swap:\t",Free_Swap,"Mb\t\t|"
print "| Swap Usage:\t",Uso_Swap,"Mb\t\t|"
```

```
print "+------------------------------+"
print "| Free HDD:\t",Free_HDD,"Gb\t\t|"
print "| HDD Used:\t",Uso_HDD,"Gb\t\t|"
print "+------------------------------+"
print "| Free SD:\t",Free_SD,"Gb\t\t|"
print "| SD Used:\t",Uso_SD,"Gb\t\t|"
print "+------------------------------+"
if t>high:
     print "| Cooling Sys:\tEnabled\t\t|"
else:
     print "| Cooling Sys:\tDisabled\t|"
print "+------------------------------+"


#CONTROLLO Temp.log: TEMPERATURA
if t<=low:
     print "Low Temperature.\n"

if t>low and t<=average:
     print "Average Temperature.\n"

if t>average and t<=high:
     print "High Temperature.\n"

if t>high and t<=critical:
     print "Critical Temperature!\nActivating Cooling System . . .
\n"
     Temp_LOG.write (Data+" "+Ora+"\n"+"Temperature:\t"+str(t)+"
°C\n"+"Cooling System:\tEnabled\n")

if t>high: GPIO.output(4, True)              #accendo la ventola
if t<=high: GPIO.output(4, False)      #Spengo la ventola


#CONTROLLO Temp.log: CPU
if Uso_CPU>=100:
     print "Critical CPU Usage.\n"
     Temp_LOG.write (Data+" "+Ora+"\n"+"CPU Usage:\t"+Uso_CPUs+"
%\n")


#CONTROLLO Temp.log: RAM
if Uso_RAM>=Total_RAM:
     print "Extreme RAM Usage.\n"
     Temp_LOG.write (Data+" "+Ora+"\n"+"Total RAM:\t"+Total_RAMs+"
Mb\n"+"RAM Usage:\t"+Uso_RAMs+" Mb\n")


#CONTROLLO Temp.log: Swap
if Uso_Swap>=Total_Swap and Total_Swap!=0:
     print "Extreme Swap Usage.\n"
     Temp_LOG.write (Data+" "+Ora+"\n"+"Total
Swap:\t"+Total_Swaps+" Mb\n"+"Swap Usage:\t"+Uso_Swaps+" Mb\n")
if Total_Swap==0:
     print "Swap Disabled.\n"
     Temp_LOG.write (Data+" "+Ora+"\n"+"WARNING: Swap
Disabled\nTotal Swap:\t"+Total_Swaps+" Mb\n")
Temp_LOG.close ()
```

```
#CONTROL  LO File.log: SHUTDOWN

if t > high:
     File_LOG.write ("+------------------------------+\n"+"|
Date:\t\t"+Data+"\t|\n"+"| Hour:\t\t"+Ora+"\t|\n"+"|
Temperature:\t"+ts+"°C\t\t|\n"+"| CPU Speed:\t"+Vel_CPUs+"
Hz\t|\n"+"| CPU Usage:\t"+Uso_CPUs+" %\t\t|\n| Cooling
Sys:\tEnabled\t\t|\n+------------------------------+\n")
     File_LOG.write ("ALERT: Temperature\n")

if t < high and Uso_CPU > 100:
     File_LOG.write ("+------------------------------+\n"+"|
Date:\t\t"+Data+"\t|\n"+"| Hour:\t\t"+Ora+"\t|\n"+"|
Temperature:\t"+ts+"°C\t\t|\n"+"| CPU Speed:\t"+Vel_CPUs+"
Hz\t|\n"+"| CPU Usage:\t"+Uso_CPUs+" %\t\t|\n| Cooling
Sys:\tDisabled\t|\n+------------------------------+\n")

if t > critical:
     shut=2

if Uso_CPU > 100:
     File_LOG.write ("ALERT: CPU Usage\n")

if Uso_CPU > 130:
     if shut != 2:
          shut=1

#reboot
shut=1
if shut == 1:
     File_LOG.write ("The system is going down for reboot NOW!\n")
     File_LOG.close()
     subprocess.call(["sudo","shutdown","-r","now"])

#shutdown
if shut == 2:
     File_LOG.write ("The system is going down for system halt
NOW!\n")
     File_LOG.close()
     subprocess.call(["sudo","shutdown","-h","now"])

#+--------------------------+
#| Date:          21/05/2014 |
#| Hour:          19:38:20   |
#| Temperature:   65.8°C     |
#| CPU Speed:     900.0 Hz   |
#| CPU Usage:     103.5 %    |
#| Cooling Sys:   Enabled    |
#+--------------------------+
#ALERT: Temperature
#ALERT: CPU Usage
#The system is going down for system halt NOW!
```