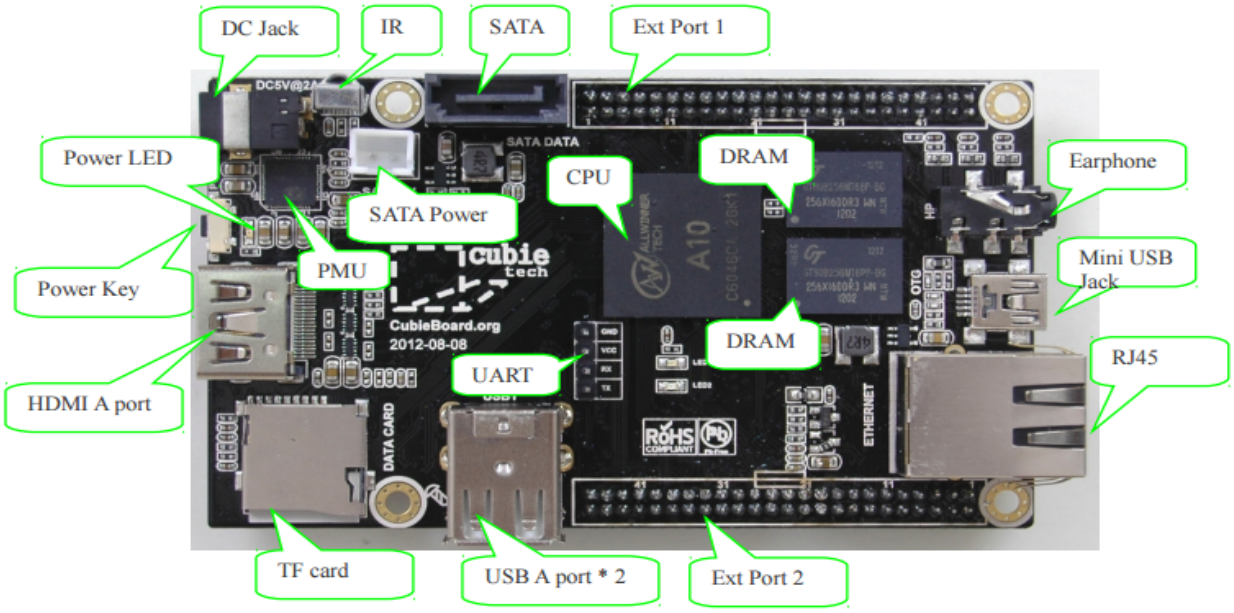
Lab 4

What is Cubieboard?

Cubieboard is a single-board computer

- CPU: 1G ARM cortex-A8 processor, NEON, VFPv3, 256KB L2 cache
- GPU: Mali400, OpenGL ES GPU
- Memory: 512MB/1GB DDR3 @480MHz
- Video Output: HDMI 1080p Output
- Networking: 10/100M Ethernet
- Storage: 4GB Nand Flash
- IO: 2 USB Hosts, 1 micro SD slot, 1 SATA, 1 IR sensor
- Extended interfaces: 96 extend pin including I2C, SPI, RGB/LVDS, CSI/TS, FM-IN, ADC, CVBS, VGA, SPDIF-OUT, R-TP..
- Software: Running Android, Ubuntu and other Linux distributions
- Power: requires regulated 5VDC 2A power supply

TOP



Lab4.1燒錄Debian

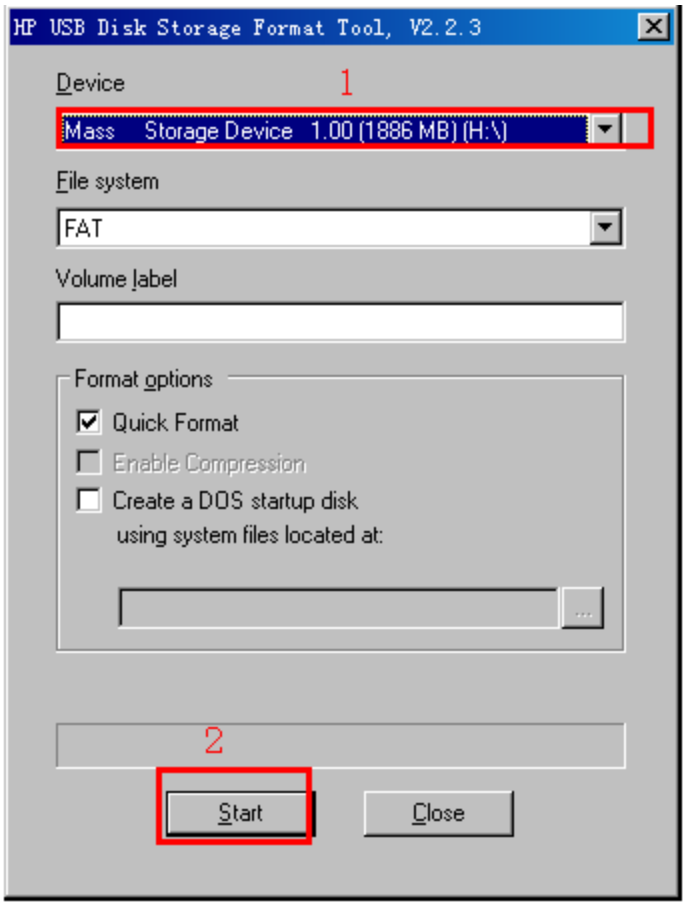
使用環境為Windows

測試的時候首先燒寫我們提供的鏡像

方法如下:

1. 首先燒寫前先要格式化SD卡

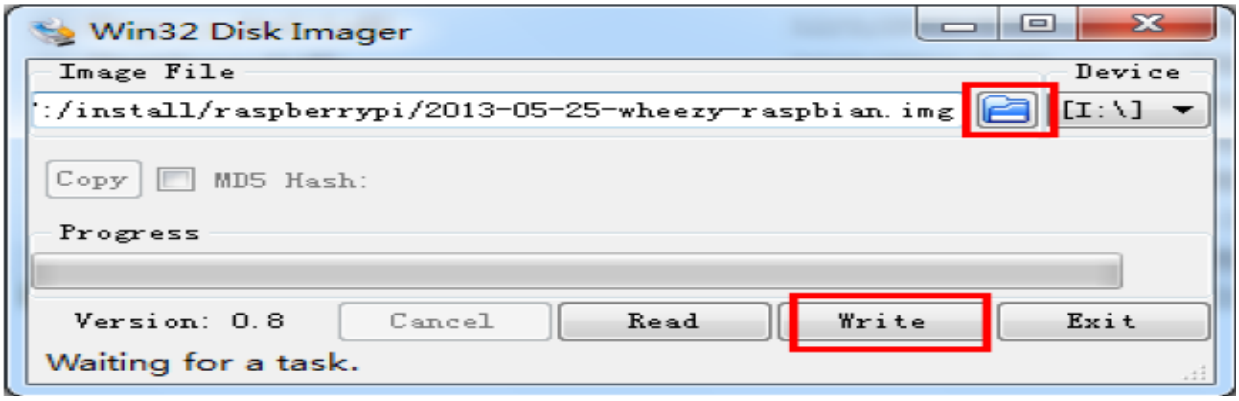
   打開**HPUSBDisk.exe**,按照如下操作:



2. 燒錄映像檔

   使用Win32DiskImager.exe

   選擇燒寫鏡像（選取我們配置好的鏡像**debian.img**）， 並點擊“Write”進行燒寫， 如下圖



3. 燒錄完成後，把SD卡插入CubieBoard中，接上USB鍵盤，最後插電，便會可以進入系統，登入使用者為root，密碼為cubieboard，以上步驟為實驗4.1。

/*
其他系統請參考
mac:http://www.ubuntu.com/download/desktop/create-a-usb-stick-on-mac-osx
Linux:Like mac
*/

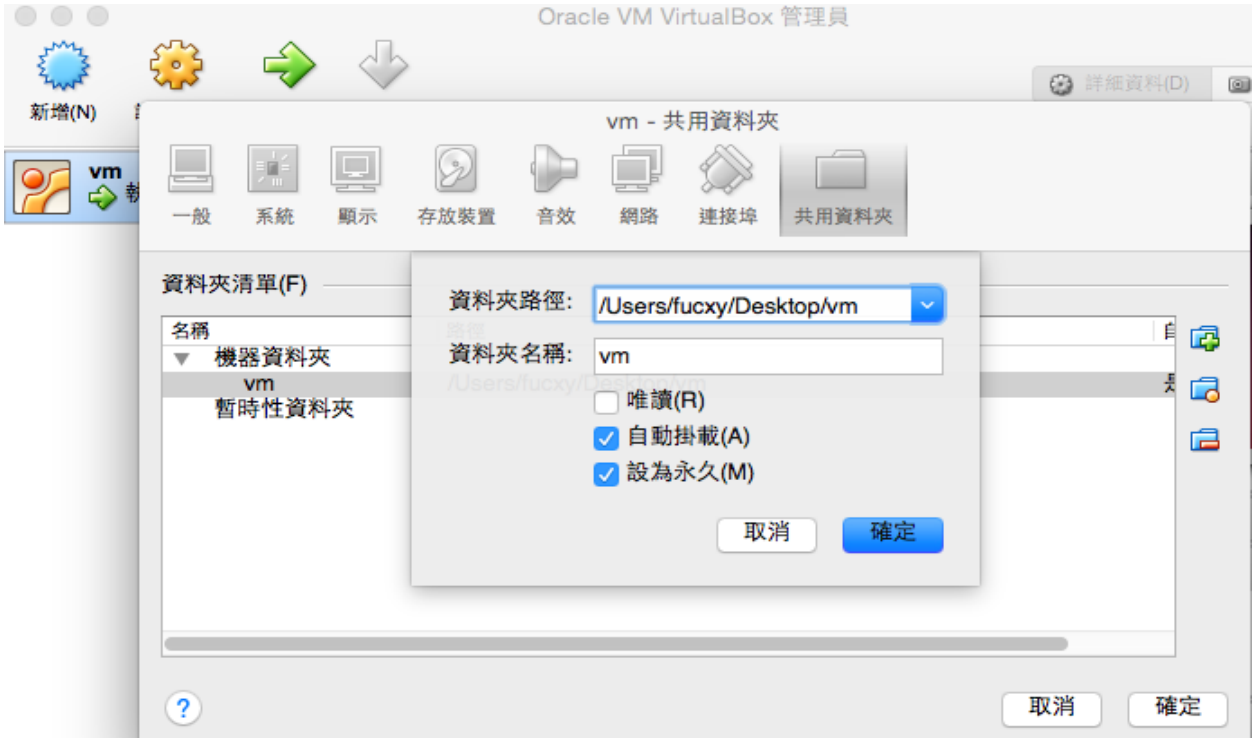
Lab4.2實驗環境架設

這一個部分需要大家把編譯環境架設完成，分為以下幾個步驟：

1. 首先先下載virtualbox安裝
2. 安裝完成後，建立一個新的虛擬機器，類型選擇Linux，而版本選擇ubuntu 32 bit(可以選擇自己習慣的版本)，再來就一直下一步，這裏我們就建成了我們的虛擬機器
3. 然後我們去ubuntu的官方網站，把相對應的iso檔下載下來，在這裡我們使用的是ubuntu desktop 14.04 32bit 的版本來安裝。
4. 安裝完成後，開啟終端機(terminal)
5. 我們要使用apt-get這一個套件管理系統安裝arm-linux專用的gcc，使用以下指令
   sudo apt-get update
   sudo apt-get upgrade
   sudo apt-get install gcc-arm-linux-gnueabihf
   便可以安裝成功
6. 然後寫一個簡單的c程式如下圖
   #include<stdio.h>
   int main(){
   printf(“Hello World!\n”);
   }
   檔名為hello.c
7. 然後使用以下指令編譯
   arm-linux-gnueabihf-gcc hello.c (-o 你想要的檔名) //不設的話會命名為a.out
   如此一來就可以得到編譯好的程式了，以及我們的編譯環境這樣就完成了我們所需要的實驗環境

Lab4.3 img檔編輯教學

1. 首先，我們必須設置共享資料夾，將在上一個實驗的環境跟你的實體機連結起來，若使用的是與實驗相同的virtulbox，可以參考以下連結設置。
   https://help.ubuntu.com/community/VirtualBox/SharedFolders
   以下是參考作法
   在virtual box中，可以在設定裡設定共享資料夾，



所以我們這裡的資料夾叫做vm，把唯獨取消其他都選取
設置完成後開啟虛擬機器
進入虛擬機器後從選單進入Devices->Insert Guest Additions CD Images
虛擬機便會看到跳出程式安裝
安裝完後重新開機
然後開啟終端機打入以下指令

```
sudo mount -t vboxsf -o uid=$UID,gid=$(id -g) vm
~/anywhere-you-wants-folders
```

如此一來資料夾就掛載上去了
要解除掛載請用

```
sudo umount ~/anywhere-you-wants-folders
```

便可以看到我們的實體機與虛擬機連結起來了

2. 假設在本實驗中，我們把shared folder放在/vmshared底下，並且也將4.1的debian.img檔也放進去了，我們現在要做的就是要將剛剛編譯好的程式放入img檔中，首先使用以下指令

```
cd /vmshared
ls
```

我們就可以看到我們的img檔

3. 在img檔中有許多磁區，但是我們要掛載的磁區只有一個，所以執行以下指令來看出我們要掛載的磁區位置。

```
sfdisk -uS -l the-imgane-file-you-use.img
```

如下圖

```
fucxy@fucxy-VirtualBox:/vmshared$ sfdisk -uS -l debian-server-cb2-card0-hdmi-v1.1.
img
Disk debian-server-cb2-card0-hdmi-v1.1.img: cannot get geometry

Disk debian-server-cb2-card0-hdmi-v1.1.img: 140 cylinders, 255 heads, 63 sectors/t
rack
Warning: The partition table looks like it was made
  for C/H/S=*/64/32 (instead of 140/255/63).
For this listing I'll assume that geometry.
Units = sectors of 512 bytes, counting from 0

   Device Boot    Start       End   #sectors  Id  System
debian-server-cb2-card0-hdmi-v1.1.img1          2048     26623      24576  83  Lin
ux
debian-server-cb2-card0-hdmi-v1.1.img2         26624   2205199    2178576  83  Lin
ux
debian-server-cb2-card0-hdmi-v1.1.img3             0         -          0   0  Emp
ty
debian-server-cb2-card0-hdmi-v1.1.img4             0         -          0   0  Emp
ty
fucxy@fucxy-VirtualBox:/vmshared$
```

其中他會有關鍵的兩行為

Units = sectors of XXX bytes 這邊為我們所使用的img檔一個區塊大小為多少

再來第二行為下面的磁區分布表

由於我們所要掛載的磁區為第二磁區（第一磁區為開機磁區）所以在這邊為debian-server-cb2-card0-hdmi-v1.1.img2這一個磁區，記住這一個Start的位置在此範例為26624

4. 在這一步我們必須要把img2這一個磁區mount起來，mount 在/mnt（確定為空資料夾）下，指令為

   mount -o loop,offset=$((sector大小＊磁區start)) the-imgane-file-you-use.img /mnt

   而在此範例中，sector大小為512，磁區start為26624，所以實際指令為下

   mount -o loop,offset=$((512*26624)) the-imgane-file-you-use.img /mnt

   如此一來就可以在/mnt底下看到img的東西了。
5. 再來再將4.2編譯好的程式檔案放入/mnt底下

   mv /where/is/your/proc.out /mnt/where/you/want/

   如此一來就將程式安裝到我們所使用的img上了
6. 再來我們必須要將mount的磁區解除，使用以下指令

   umount /mnt
7. 將img檔如4.1燒入進sd卡，開啟CubieBoard。
8. 登入系統後，執行以下指令

   /path/to/where/your/prog.out

   便可以看到

   Hello World!!

   顯示在螢幕中

Demo:

顯示Hello world

Bonus:

開機時直接執行Hello Workd程式！