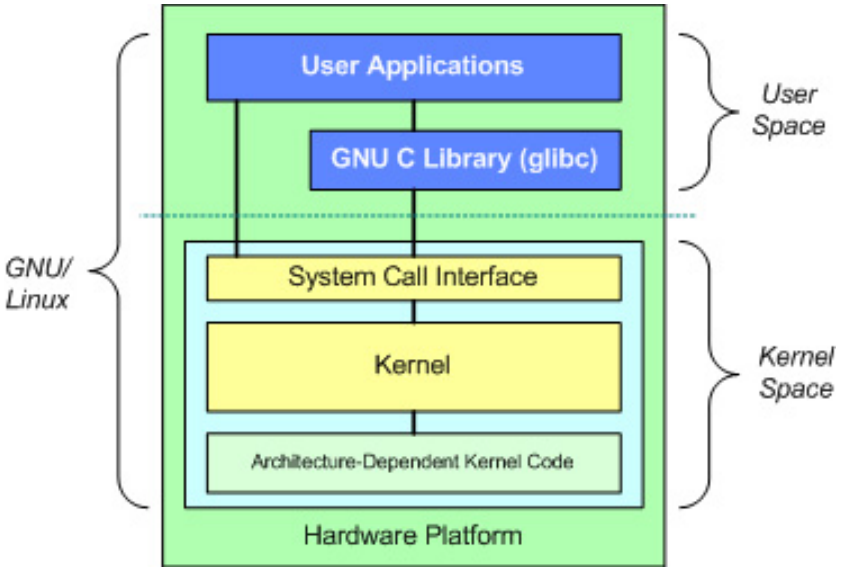
Computer Architecture

LAB2: Building Linux Operating System Environment

Linux Kernel Introduction

- The Linux kernel is a monolithic Unix-like computer operating system kernel.
- It is a open source software that suit for researcher to develop experiment environment with OS without any cost.



Linux Device-Tree Introduction

- Unlike x86 platform, ARM and others embedded system have various architecture, hardware and platform.
- In early stage of linux kernel(before 2012), each vender need to design specific board code for their own platform and merge into linux kernel, and it produced mess, same functional codes and made kernel hard to be maintained.
- After that Linux ARM core team use Device Tree data structure(derived from SPARC-based system) to descript platform architecture, read by kernel and generate corresponding information

Initrd ramdisk Introduction

- Initrd ramdisk (initrd) is is a scheme for loading a temporary root file system(RFS) into memory.
- In booting process, kernel will start to prepare environment before enter init process of Linux (not init in initrd) by programs in initrd.

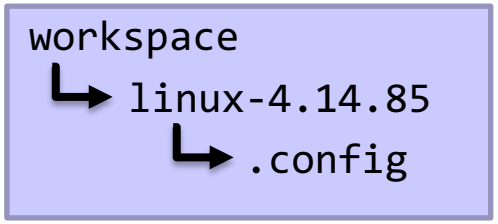
Building ARM Linux Kernel(1/3)

- Get Linux kernel source code
  - https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.14.85.tar.xz
- Extract
  - 在家目錄下新增linux-kernel資料夾並將下載的kernel放入此資料夾中
  - 此時Linux kernel才可編譯成可用的Image檔

- Command:
  - cd ~/workspace/Src
  - wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.14.85.tar.xz
  - tar -xvf linux-4.14.85.tar.xz -C ~/workspace

Building ARM Linux Kernel(2/3)

- Configure
  - 下載完kernel source code之後，我們必須設定linux kernel的執行平台環境，由於每個平台的環境不盡相同，我們直接提供realview eb最簡單的設定檔(config-4.14.85-realview-arm1136)。
  - 將config-4.14.85-realview-arm1136檔案放進workspace資料夾，並更改檔名
  - # cp config-4.14.85-realview-arm1136 .config
  - # mv .config linux-4.14.85/
  - linux-4.14.85]# make ARCH=arm menuconfig(選擇EXIT即可離開設定畫面)
  - 檔案結構示意圖：(上方指令僅供參考，只要符合此檔案結構圖即可)

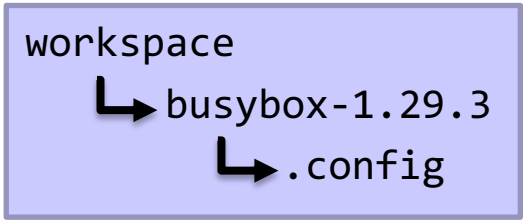


Building ARM Linux Kernel(3/3)

- Compile
  - 編譯前必須檢查是否有export cross compiler的路徑
  - linux-4.14.85]# make ARCH=arm CROSS_COMPILE=arm-none-linux-gnueabi-zImage
  - linux-4.14.85]# make ARCH=arm CROSS_COMPILE=arm-none-linux-gnueabi-dtbs
  - 編譯完成之後，我們所要使用的zImage檔將會產生在linux-4.14.85 /arch/arm/boot裡面、Device Tree會產生在linux-4.14.85 /arch/arm/boot/dts裡面，之後我們會透過此檔案來進行Linux作業系統開機的功能。

Building Initrd ramdisk

- Get Busybox
  - http://busybox.net/
  - We use busybox 1.29.3 as example
- Configure and compile
  - 和kernel一樣，busybox也需要先經過設定後才可以編譯。我們一樣提供最簡單的config檔(config-busybox-1.29.3)，記得將config-busybox-1.29.3 修改檔名成.config。
  - 將此config檔放入解壓縮後的busybox資料夾
  - 檔案結構示意圖：



  - busybox-1.29.3]# make ARCH=arm menuconfig
  - 設定結束後即可進行編譯
  - busybox-1.29.3]# make ARCH=arm CROSS_COMPILE=arm-none-linux-gnueabi-
  - busybox-1.29.3]# make install CROSS_COMPILE=arm-none-linux-gnueabi-

Building Initrd ramdisk (Cont.)

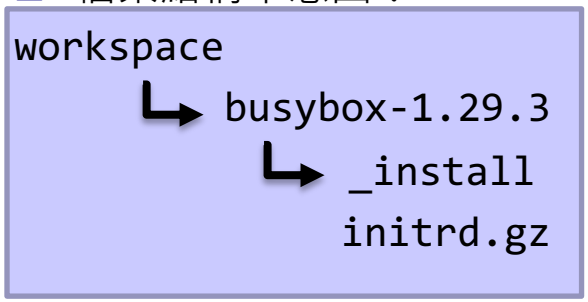
- File system setting
  - 安裝成功之後，busybox-1.29.3資料夾下會多一個”_install”資料夾，裡面只包含最基本的資料夾(bin,sbin,usr)還有執行檔linuxrc。此執行檔為linux kernel開機完成後所要執行的第一個程式。我們先將其名字改為init。
  - 接著要新增必要的資料夾： dev, etc, lib, opt, proc, sys, tmp, var, home
  - 檔案結構示意圖：

```
busybox
└─ busybox-1.29.3
   └─ _install
      └─ bin,sbin,usr,dev,etc,lib,opt,proc,sys,tmp,var,home
```

  - 注：教材有附上etc.tar.gz，解壓縮之後有etc資料夾可直接使用。
  - 其中最重要的就是etc，裡面放置著各個必要的環境設定檔。
  - (以下為lib資料夾內容詳細說明，可自行練習)
  - lib內則是要放置cross compiler的函示庫檔，也就是LAB1中所提到的qemu-arm執行時後面必須接上-L {library-path}。將library-path中所有的*.so檔案複製到lib資料夾下即可。
  - _install$ cp -P ~/workspace/arm-2014.05/arm-none-linux-gnueabi/libc/lib/* ./lib/

Building Initrd ramdisk (Cont.)

- Build Initrd ramdisk
  - 按照上述步驟編譯完成之後，我們即可將這些檔案與資料夾壓縮成Initrd ramdisk以提供開機使用。
  - 對於設定檔有興趣的，可以繼續研究etc中各個檔案。
  - 注意: /etc/init.d資料夾中的rcS檔案為init執行後會去讀入的script檔，也是整個環境的初始設定檔。因此必須修改檔案格式為可執行檔 (# chmod 755 rcS)
  - # cd _install
  - _install]# find . | cpio –H newc –o > ../initrd
  - _install]# gzip -f ../initrd
  - busybox-1.29.3資料夾下產出 initrd.gz，此檔即為我們所要使用的Initrd ramdisk。
  - 檔案結構示意圖：



Running Linux System on QEMU

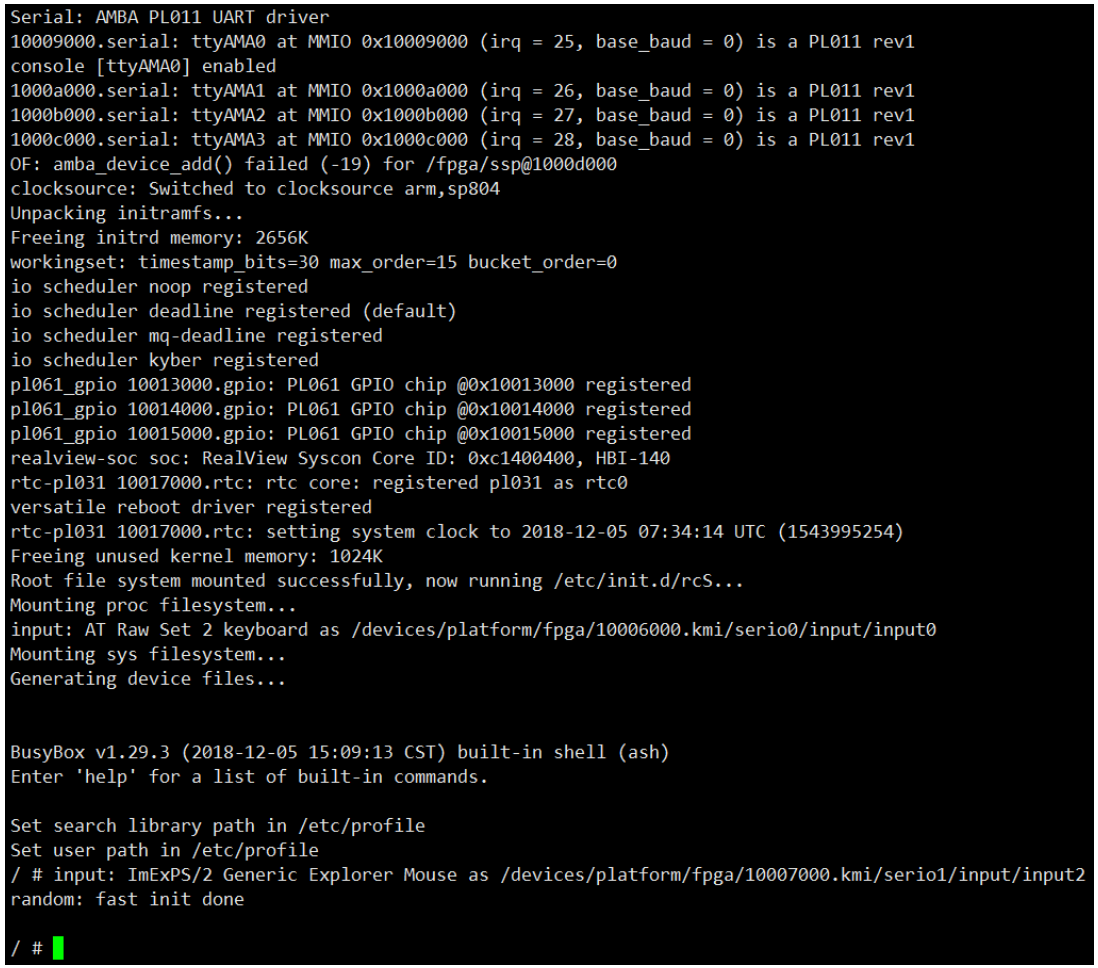
- Necessary Files: qemu-system-arm, zImage, initrd.gz arm-realview-eb.dtb
  - 在經過前面的步驟後，我們可以取得這些必要的檔案。先新增一個資料夾: qemu-test,然後將這些必要檔案擺放到此資料夾中(可用複製或是使用 ln -s 做鏈結)。
  - 檔案路徑:
    - qemu-system-arm: qemu-bin-3.0.0/bin
    - zImage: linux-4.14.85/arch/arm/boot/
    - Initrd.gz: buzybox-1.29.3/
    - Device tree: linux-4.14.85/arch/arm/boot/dts/

Running Linux System on QEMU

- qemu-test]# ~/workspace/qemu-bin-3.0.0/bin/qemu-system-arm -M realview-eb -m 128M -cpu arm1136 -kernel zImage -initrd ./initrd.gz -nographic -serial mon:stdio -dtb arm-realview-eb.dtb
  - 上述指令代表意義為：
    - 我們選取了realview emulation board當作模擬平台
    - 平台上的CPU我們選擇了ARM1136
    - linux kernel則為我們所編譯出來的zImage
    - Linux Device tree是Kernel 提供的arm-ralview-eb.dtb
    - root file system也是我們編譯的initrd.gz
    - 不開啟圖形化介面
  - 此外根據不同的平台與Linux核心設定，模擬所下的參數也會不一樣。

Running Linux System on QEMU(Cont.)

- 正常執行後，可於終端機看到開機畫面：



Running Application in Linux System

- 前面我們提過採用qemu-arm執行cross compiler編譯出來的程式。現在，我們將這隻程式放入busybox的_install/home資料夾中並且再次產出initrd.gz
  - 必須注意_inistall資料夾中是否有前次殘留的initrd.gz，必須移除，否則會造成initrd.gz過大的情況。
  - 執行結果如下張投影片所示。
  - 必須注意lib內是否有必要的*.so檔，因為我們編譯的程式是採用動態函示庫編譯。

Execution Result

- 執行結果如下

```
BusyBox v1.29.3 (2018-12-05 15:09:13 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

Set search library path in /etc/profile
Set user path in /etc/profile
/home # random: fast init done
input: ImExPS/2 Generic Explorer Mouse as /devices/platform/fpga/10007000.kmi/serio1/input/input2

/home # pwd
/home
/home # ls
test.o
/home # ./test.o
COURSE : Computer Architechure
/home #
```