2013/01/21

目標:BT下載機

1.移植U-boot

2.移植LINUX(rootfs)

3.移植下載軟體(**transmission**)

http://www.abokuo.com/25-my-artcles/pc/software/74-rtorrent-web-p2p.html

4.移植PPPOE(rp-pppoe &ppp)

5.USB掛載

2013/09/30 linux driver

所以開發環境要NFS才會快

2013/10/23

目標1:重回BT下載機 移植使用無線網卡

不使用PPPOE

2014/03/06

完成視頻監控系統

<http://blog.csdn.net/sg131971/article/details/6932237>

step1:u-boot 移植 2014/03-2014/07

step2:linux移植 2014/07-2014/11

step3: 視頻監控系統 2014/07-2015/02

2014/04/15

跟韋東山那本書來練

建立一個簡單的裸機架構(4/15-5/15)

u-boot(5/17-7/15)含移植

linux driver

目標2:u-boot-2009.08在mini2440上的移植

<http://blog.chinaunix.net/uid/26765074/cid-172658-list-3.html>

<http://blog.chinaunix.net/uid/26552187/cid-153521-list-1.html>

<http://blog.csdn.net/ce123_zhouwei/article/category/849226>

<http://www.it165.net/embed/html/201206/1998.html>

<http://www.it165.net/embed/html/201206/1999.html>

<http://www.it165.net/embed/html/201206/2000.html>

<http://www.it165.net/embed/html/201206/2001.html>

<http://www.it165.net/embed/html/201206/2002.html>

<http://www.it165.net/embed/html/201206/2003.html>

嵌入式Linux之我行

目標3Linux-2.6.32.2内核在mini2440上的移植

<http://blog.csdn.net/ssdsafsdsd/article/category/1376245/1>

2014/06/18-2015/2/05 linux driver

學習linux driver框架 移植

VMWARE:192.168.191.128

PC: 192.168.191.99

MINI2440: 192.168.191.33

**U-boot開機訊息**

U-Boot 2010.03 (Feb 05 2013 - 03:42:24)

modified by tekkamanninja (tekkamanninja@163.com)

Love Linux forever!!

=

I2C: ready

DRAM: 64 MB

Flash: 2 MB

NAND: 128 MiB

Video: 240x320x16 20kHz 62Hz

In: serial

Out: serial

Err: serial

USB slave is enable!

Net: dm9000

U-Boot 2010.03 (Feb 05 2013 - 03:42:24)

modified by tekkamanninja

(tekkamanninja@163.com)

Love Linux forever!!

Hit any key to stop autoboot: 0

# 1/21筆記 開發環境建立

**Yum加入proxy support**

1.修改 /etc/yum.conf

2.找個地方加入這一行

proxy=http://proxy\_ip:port

**安裝SSH**

yum –y install openssh

service sshd restart

記得關掉防火牆

確認IP

**安裝VSFTP**

Yum –y install vsftpd

Service vsftpd restart

setsebool ftp\_home\_dir=1

**Fedora不進入圖形化**

(舊版)在/etc/inittab中修改缺省的启动级别，id:5:initdefault: 把5改成3。

(新版 fedora15以上版本)

# ln -sf /lib/systemd/system/multi-user.target /etc/systemd/system/default.target  #文字模式  
# ln -sf /lib/systemd/system/graphical.target /etc/systemd/system/default.target   #圖形模式

參考http://www.wretch.cc/blog/iii9/377660

**開機自動執行**

(舊版)/etc/rc.d/rc.local

(新版)

systemctl enable sshd.service

systemctl enable vsftpd.service

**VSFTPD錯誤問題**

500 OOPS: priv\_sock\_get\_cmd

解决方法：

在vsftpd.conf 中添加 "pasv\_promiscuous=YES" 重启服务就可以了。

# 2012/01/22 筆記

交叉工具鏈

使用友善之臂編譯好的

若想自己編交叉工具鏈 也可以參考

<http://blog.chinaunix.net/uid-20543672-id-94268.html>

<http://blog.chinaunix.net/uid/20543672/cid-6406-list-1.html>

交叉工具解壓後會放在

opt/FriendlyARM/toolschain/4.4.3

U-boot

<http://blog.chinaunix.net/uid-20543672-id-94379.html>

samba搭建

參考doc下文章

解壓縮 Unzip \*.zip

Samba 暫時只能在/tmp

WHY? 權限問題

一樣是權限問題!!!!!!

暫時解法: setenforce 0

關閉SELinux

#system-config-securitylevel

選擇DISABLE後按OK，必須重開機

Q:如何暫時關閉SELinux而不用重開機?  
A:命令列下執行setenforce 0就可以暫時關閉功能，執行setenforce 1就可以打開功能

vi /etc/sysconfig/selinux

SELINUX=enforcing 改成--> SELINUX=disabled

SElinux啟用中，在Windows檔案總管無法連到 Samba 所分享出來的目錄時，

在Linux 中，可執行下列指令：

**setsebool samba\_enable\_home\_dirs=1**

參考文件： /etc/samba/smb.conf

#---------------   
# SELINUX NOTES:   
# 分享群組   
# If you want to use the useradd/groupadd family of binaries please run:   
# **setsebool -P samba\_domain\_controller on**   
#

# 分享home目錄   
# If you want to share home directories via samba please run:   
# **setsebool -P samba\_enable\_home\_dirs on**#

# 2012/01/23(編譯u-boot)

設定交叉編譯路徑(參考2440手冊)

vi ~/.bashrc

export PATH=/home/kain/opt/FriendlyARM/toolschain/4.4.3/bin:${PATH}

export 確認

修改根目錄下Makefile

CROSS\_COMPILE = arm-tekkaman-linux-gnueabi-

改為

CROSS\_COMPILE = arm-none-linux-gnueabi-

Make mini2440\_config

Make

或執行/home/kain/u.sh

Build完之後 燒寫文件

-rwxr-xr-x. 1 root root 260308 Jan 23 04:05 u-boot.bin

先確認flash中位址

在mini2440中，由于bin文件大小已经大于了0x40000(256K)，所以我也修改了ENV的 OFFSET。现在的分区是这样的：  
**U-boot        ：0x0~0x60000  
u-boot参数区  ： 0x60000~0x80000  
内核和文件系统 ：0x80000之后**

注意 需修改蜂鳴器部分!!!!!

# 2012/01/28(修改u-boot)

修改蜂鳴器部分(GPB0)

\\192.168.191.128\kain\u-boot\u-boot-2010.03-tekkaman-master\board\tekkamanninja\mini2440\mini2440.c

Z:\lib\_arm\board.c

關掉蜂鳴器

# 2012/01/29(正式燒寫u-boot)

使用dnw

1.先切到nor

2.打開工具 Send選檔案連上去按a

# 2013/02/05

熟悉uboot命令 都忘光了!!!!

PC(WIN7): 192.168.199.99

VMWARE(Fedora):192.168.199.88

BOARD:192.168.199.128

gatewayip 192.168.199.1

printenv 顯示env

[u-boot@MINI2440]# setenv serverip 192.168.199.99

[u-boot@MINI2440]# setenv ipaddr 192.168.199.88

[u-boot@MINI2440]# setenv gatewayip 192.168.199.1

[u-boot@MINI2440]# saveenv

Saving Environment to NAND...

Erasing Nand...

Erasing at 0x6000000000002 -- 0% complete.

Writing to Nand... done

[u-boot@MINI2440]# printenv

baudrate=115200

ethaddr=08:08:11:18:12:27

netmask=255.255.255.0

tekkaman=bmp d 70000

stdin=serial

stdout=serial

stderr=serial

ethact=dm9000

bootdelay=10

filesize=1821E0

bootcmd=nand read 30008000 0x80000 300000;bootm 30008000

bootargs=noinitrd root=/dev/mtdblock3 console=ttySAC0,115200 init=/linuxrc

serverip=192.168.199.99

ipaddr=192.168.199.88

gatewayip=192.168.199.1

Environment size: 372/131068 bytes

# 2013/02/06

先想辦法把uboot下網路弄通

Dm9000

因為暫時無網路線 所以先移植linux&Yaffs

# 2013/02/07

移植linux 2.6.38.8 開不了機…

所以目前還是使用友善的版本

5.直接做yaffs支持，注意老版本的yaffs已经不支持linux-2.6.36以上内核了，需要最新版本yaffs，

首先下载git,然后用git下载，为内核打上补丁。

#apt-get install git-core

#git clone git://www.aleph1.co.uk/yaffs2

#cd yaffs2

#patch-ker.sh c m [linux-kernel]

Git 使用http proxy

export http\_proxy=http://proxy1.cgmh.org.tw:8080

export https\_proxy=http://10.60.36.146:3128

之後git clone <http://github.com/qemu/qemu.git>

 tar -zxvf yaffs2-HEAD-5dce9cd.tar.gz -C linux-2.6.38.8/yaffs2

加-C可以指定目錄

mv -f yaffs2-HEAD-5dce9cd yaffs2

目錄改名

打yaffs2 patch

./patch-ker.sh c m /home/kain/linux/linux-2.6.38.8

make ARCH=arm CROSS\_COMPILE=arm-none-linux-gnueabi- zImage

vi mkkernel.sh

cp arch/arm/boot/zImage /home/kain/linux/linux-2.6.32.2

./mkimage -n 'linux-2.6.32' -A arm -O linux -T kernel -C none -a 0x30008000 -e 0x30008040 -d zImage uImage

usb start

usb storage

fatload usb 0:1 0x30008000 uImage

nand erase 0x80000 0x300000

nand write 30008000 0x80000 300000

ROOT FS 目前使用ROOT FS QT

Supervivi> part show

Number of partitions: 4

name : offset size flag

------------------------------------------------------------

vivi : 0x00000000 0x00040000 0

param : 0x00040000 0x00020000 0

kernel : 0x00060000 0x00500000 0

root : 0x00560000 0x3fa80000 0

要利用SUPER VIVI燒錄 要修改partitions

NAND

Creating 4 MTD partitions on "NAND 128MiB 3,3V 8-bit":

0x000000000000-0x000000060000 : "u-boot"

0x000000060000-0x000000080000 : "uboot-env"

0x000000080000-0x000000580000 : "Kernel"

0x000000580000-0x000008000000 : "root"

Qt透過gui 執行shell

http://www.360doc.com/content/12/0604/10/9934259\_215779485.shtml

# 2013/03/07移植Transmission 2.77

./configure –help 決定下的參數

Xd –d XXX.tar.xz

Tar –xvf XXX.tar

需要先交叉openssl 和libcurl

**交叉openssl**

./Configure --prefix=/home/kain/linux/software/build no-asm no-shared os/compiler:arm-linux-gcc

Make

Make install

**交叉libcurl**

CC=arm-linux-gcc CXX=arm-linux-g++ \

./configure --prefix=/home/kain/linux/software/build/ \

--host=arm-linux \

--disable-shared \

--enable-static \

--enable-ipv6 --without-random \

--with-ssl=/home/kain/linux/software/build/

**交叉libevent**

CC=arm-linux-gcc CXX=arm-linux-g++

./configure --prefix=/home/kain/linux/software/build/ --host=arm-linux --disable-shared

make

make install

**交叉Zlib**

CC=arm-linux-gcc CXX=arm-linux-g++

./configure --static --prefix=/home/kain/linux/software/build/

Make

Make install

**交叉transmission**

<http://jyhshin.pixnet.net/blog/post/39399725-build-transmission-2.77-for-mips>

主要參考此篇

<http://www.arm9home.net/simple/index.php?t1803.html>

CC=arm-linux-gcc CXX=arm-linux-g++

OPENSSL\_LIBS=/home/kain/linux/software/build/lib

OPENSSL\_CFLAGS=/home/kain/linux/software/build/include

LIBCURL\_LIBS=/home/kain/linux/software/build/lib

LIBCURL\_CFLAGS=/home/kain/linux/software/build/include

LIBEVENT\_LIBS=/home/kain/linux/software/build/lib

LIBEVENT\_CFLAGS=/home/kain/linux/software/build/include

PKG\_CONFIG\_PATH="/home/kain/linux/software/build/lib/pkgconfig"

./configure --prefix=/home/kain/linux/software/build/transmission-2.77 --enable-static --host=arm-linux --enable-daemon --disable-nls --disable-mac --disable-gtk --enable-lightweight --with-zlib=/home/kain/linux/software/build

爛方法…

把所有的LIB都放在同一個目錄下…..

USB掛載

<http://blog.csdn.net/star2523/article/list/2>

# 2013/09/13安裝NFS

未裝完

先用yum search nfs

yum –y install nfs-utils.i686

<http://blog.csdn.net/feiniao1221/article/details/6948453>

<http://blog.csdn.net/zzf0707/article/details/5318846>

<http://www.arm9home.net/read.php?tid=5521>

<http://www.jyguagua.com/?p=335>

# 2013/10/01安裝QT

<http://blog.csdn.net/vacole/article/details/6873314>

mini6410 手冊

1.echo $PATH 查看PATH

 2.通过以下命令查看arm-none-linux-gnueabi编译工具是否安装成功  
   a. 执行：arm-none-linux-gnueabi-g++ -v  
   b. 执行：arm-none-linux-gnueabi-gcc -v  
   如果以上两步都能正确打印对应的版本信息，则表明arm-none-linux-gnueabi编译工具  
   已安装成功！

3. ./configure -prefix /home/kain/QT/qt-everywhere-opensource-src-4.7.4/build -opensource -embedded arm -xplatform qws/linux-arm-g++ -no-webkit -qt-libtiff -qt-libmng -qt-mouse-tslib -qt-mouse-pc -no-mouse-linuxtp -no-neon

4.Make;make install

5.修改環境變數

vi ~/.bashrc

#Set QT Tool

export ARMQTDIR=/home/kain/QT/qt-everywhere-opensource-src-4.7.4/build

export PATH=$ARMQTDIR/bin:$PATH

export MANPATH=$ARMQTDIR/man:$MANPATH

export LD\_LIBRARY\_PATH=$ARMQTDIR/lib:$LD\_LIBRARY\_PATH

6.测试 ARM 版 QT 是否安装并配置成功：  
    a. 执行：qmake -v  
        打印以下对应信息：  
                QMake version 2.01a  
                Using Qt version 4.7.3 in /usr/local/qt-4.7.3-linux-arm-install/lib  
    b. 执行：which qmake  
                打印以下对应信息：  
                 /usr/local/qt-4.7.3-linux-arm-install/bin/qmake

7.

顺序执行以下操作：  
  @1. 执行：qmake -project hello1.cpp  
        生成 hello1.pro 工程文件  
  @2. 执行：qmake hello1.pro  
        生成 Makefile 文件   
  @3. 执行：make  
        生成 hello.o, hello 文件   
    
       如果执行@2步骤时出现以下错误：  
           QMAKESPEC has not been set, so configuration cannot be deduced.  
           Error processing project file: hello1.pro  
       则是 QMAKESPEC 变量没有设置好，执行：  
      export QMAKESPEC=/usr/local/qt-4.7.3-linux-arm-install/mkspecs/qws/linux-arm-g++

# 2013/10/23繼續BT下載機

掛載SD卡 沒做成

# 2014/05/30 linux driver開始

安裝NFS

2014/06/18

<http://www.cnblogs.com/lufee/archive/2011/09/04/2166525.html>

<http://bbs.csdn.net/topics/390169744>

<http://www.cnblogs.com/feisky/archive/2010/05/29/1746885.html>

<http://blog.csdn.net/tigerjibo/article/details/6453332> //不用mknod的方法 misc

yum install kernel-PAE-devel

yum install kernel-PAE

之後還需要重新開機!!!!!!!!!

才可以build

[root@localhost linux\_driver]# insmod globalmem.ko

Error: could not insert module globalmem.ko: Device or resource busy

設備號衝突

修改設備號

[root@localhost linux\_driver]# cat /proc/devices

Character devices:

1 mem

4 /dev/vc/0

4 tty

4 ttyS

5 /dev/tty

5 /dev/console

5 /dev/ptmx

7 vcs

10 misc

13 input

14 sound

21 sg

29 fb

99 ppdev

116 alsa

128 ptm

136 pts

162 raw

180 usb

188 ttyUSB

189 usb\_device

202 cpu/msr

203 cpu/cpuid

226 drm

**250 globalmem**

251 hidraw

252 usbmon

253 bsg

254 rtc

Block devices:

259 blkext

7 loop

8 sd

9 md

11 sr

65 sd

66 sd

67 sd

68 sd

69 sd

70 sd

71 sd

128 sd

129 sd

130 sd

131 sd

132 sd

133 sd

134 sd

135 sd

253 device-mapper

254 mdp

[root@localhost linux\_driver]# mknod /dev/globalmem c 250 0

[root@localhost linux\_driver]# ls -al /dev/globalmem

-rw-r--r-- 1 root root 9 Jun 20 09:31 /dev/globalmem

[root@localhost linux\_driver]# echo "fuck you" > /dev/globalmem

[root@localhost linux\_driver]# cat /dev/globalmem

fuck you

# 2014/06/23

**把void (\*signal(int signum,void(\*handler)(int)))(int)分成两部分：**

**typedef void (\*sighandler\_t)(int);**

**sighandler\_t signal(int signum, sighandler\_t handler);**

此处由于加了typedef自定义了一个新类型sighandler\_t，所以第二行的函数原型看起来顺眼多了，形式跟int func(char c, int i)无异，但是如果看不懂typedef语句，这两句话仍然是噩梦。

要理解typedef，只要记住一句话就差不多了，那就是：typedef在语句中所起的作用只不过是把语句原先定义变量的功能变成了定义类型的功能而已。我们只消看几个例子立即明白。

例如语句 **typedef int \*apple;**理解它的正确步骤是这样的：先别看typedef，就剩下int \*apple; 这个语句再简单不过，就是**声明了一个指向整型变量的指针apple**(注意：定义只是一种特殊的声明)，加上typedef之后就解释成**声明了一种指向整型变量指针的类型apple**。

现在，回过来看上面的这个函数原型 typedef void (\*sighandler\_t)(int)，盖住 typedef不看 ，再简单不过，sighandler\_t就是**一个函数指针，指向的函数接受一个整型参数并返回一个无类型指针**。加上typedef之后sighandler\_t就是一种新的类型，就可以像int一样地去用它，不同的是它声明是**一种函数指针，这种指针指向的函数接受一个整型参数并返回一个无类型指针**。怎么样？简单吧。

再来做一个更酷的练习，请看：**typedef char \*(\* c[10])(int \*\*p);**

去 掉typedef就变成char \*(\* c[10])(int \*\*p)，先不管这个语句有多难看，它一定是声明了一个拥有10个元素的数组c对不对？okay没什么了不起的，只不过这个数组c的元素有点特别，它们都 是函数指针，并且它们所指向的这些函数统统都接受一个二级指针然后返回一直指向字符型的指针。加上typedef之后，c就不是一个数组了，而是一种类型 了 。

# 2014/06/26

Android ndk環境架設 參考此文

<http://www.cnblogs.com/zdz8207/archive/2012/11/27/android-ndk-install.html>

有ERROR 參考

<http://blog.csdn.net/netwalk/article/details/15507251>

<http://www.cnblogs.com/shadox/archive/2011/12/02/2272564.html>

編譯命令

進入cygwin

$ndk/ndk-build

K122906607@lkcgmh0082 /cygdrive/d/android/android-ndk-r9d/samples/hello-jni

$ ls libs/armeabi/

gdb.setup gdbserver libhello-jni.so

D:\android\android-ndk-r9d\samples\hello-jni\libs\armeabi\xxx.so

可以執行APP了

**2014/07/10**

先弄LED的部分 用新的架構 並用APP執行~~