

INTEL Galileo 开发板实验

一、实验目的

- 1、了解 INTEL Galileo 开发板及其构成
- 2、掌握用 Arduino IDE 程序控制 Galileo 程序的方法
- 3、掌握编写 Linux 程序控制 Galileo 开发的方法

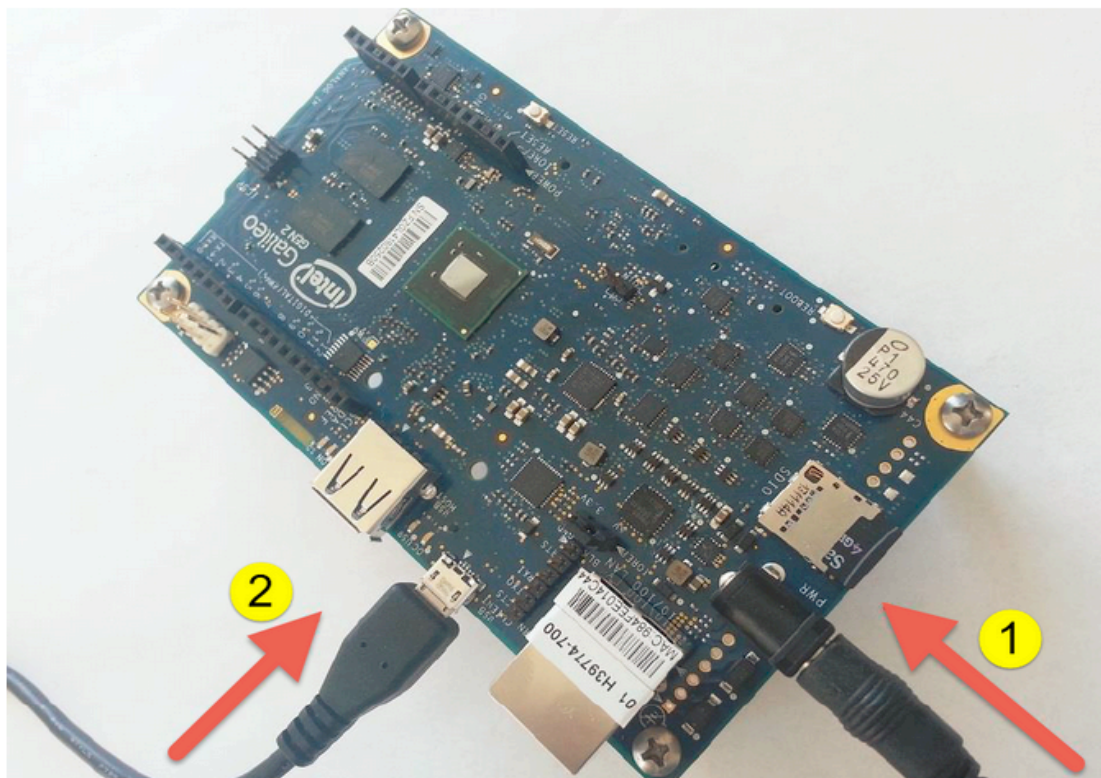
二、实验条件

- ✓ IBM-PC 兼容机
- ✓ Windows、Ubuntu11.04 或其他兼容的 Linux 操作系统
- ✓ JDK（建议安装 JDK8 及其以上版本）、Android Studio 或 Eclipse with ADT
- ✓ INTEL ATOM 平板

三、实验原理

Intel Galileo Gen2 开发板，二代的开发板性能相比一代提升很多，具体的区别可参考背景材料中所给的网址。当你拿到板子，先别急着上电，先看一下下面的内容，特别是注意事项，首先我们先认识一下板子。

Gen2 板子：



首次拿到板子，如果有插着 SD 卡的先拔下来，黄色标注 1 的为电源线，黄色标注 2 的为 usb 数据线（我们平时用的手机数据线即可），这里说明一点注意事项：上电时要先插电源线，再插数据线；断电时要先拔数据线，再拔电源线。（否则板子容易损坏）；

②更新板子的固件：

先插上电源线，然后插上你的 USB 数据线，你可以在设备管理器中看到，



这时候我们需要安装一下板子相应的驱动，右击选择更新驱动程序软件，

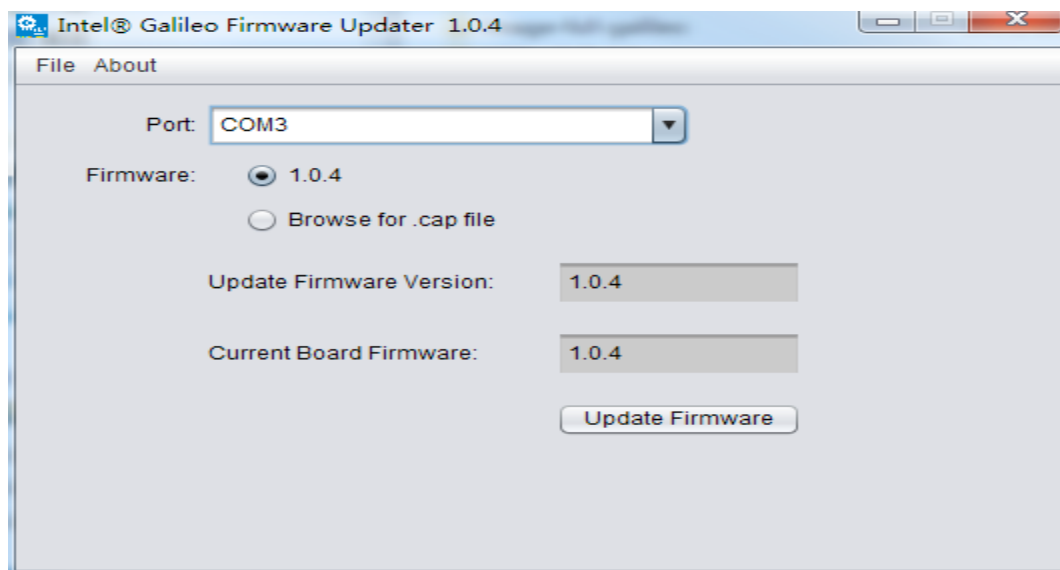


选择手动查找从官网可以下载到驱动程序 Galileo Driver，在附件一 Galileo Driver。安装完成后如下图：

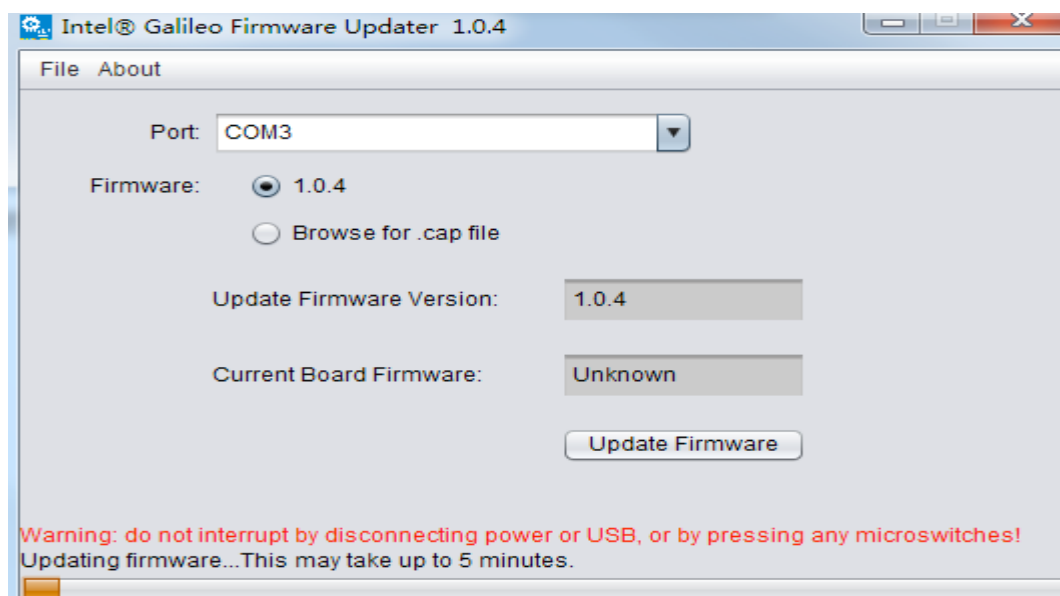


②. 更新固件：

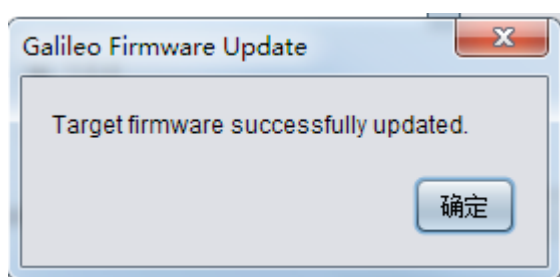
首先从官网下载固件更新工具 firmware-updater-1.0.4，在附件二 firmware-updater-1.0.4。打开：



选择你相应的端口，点击 Update Firmware



好了，等待 5 分钟左右吧，期间不可中断，所以要小心，不要断电。



Ok.

任务三：通过 tf 卡启动 Intel Galileo 开发板的环境配置

步骤：

附件二是 SD 卡中的文件系统，你们可以去 intel 的官网下载，为了节约时间，也可以到助教这边来拷贝（附件二-SD 卡文件系统），里面有这几个文件，请将下图所示的几个文件拷贝到 SD 卡中，然后插到板子上。

boot	2015/6/10 星期...	文件夹	
bzImage	2014/7/1 星期二 ...	文件	2,126 KB
core-image-minimal-initramfs-clanto...	2014/7/4 星期五 ...	WinRAR 压缩文件	2,734 KB
grub.efi	2014/7/1 星期二 ...	EFI 文件	274 KB
image-sdk-clanton.ext3	2014/7/4 星期五 ...	EXT3 文件	3,072,000...

任务四：用 linux 超级终端 minicom 来观察一下板子的启动过程

步骤：

前面你们看视频应该已经了解了板子是怎么启动的，当你插上 SD 卡是板子就会启动 SD 卡里面的文件系统。这里就不再说明了，接下去我们需要验证一下板子启动的是否为 SD 卡内的文件系统，于是我们可以利用 linux 超级终端 minicom 来完成（如果你想在 windows 下面看，也可以选用其他 win 下面的超级终端如 putty，只是 minicom 在后面的开发中还会经常用到，所以这里直接介绍 minicom）；

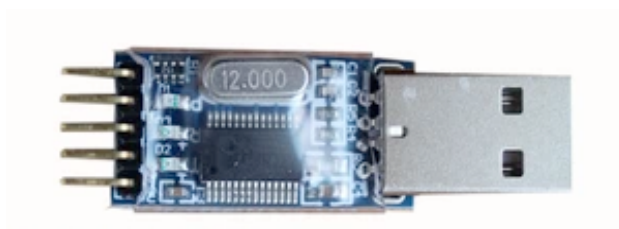
打开虚拟机启动 Ubuntu。

在命令行输入 `sudo apt-get install minicom`

```
lc@lc-virtual-machine:~$ sudo apt-get install minicom
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  minicom
0 upgraded, 1 newly installed, 0 to remove and 704 not upgraded.
Need to get 297 kB of archives.
After this operation, 1,208 kB of additional disk space will be used.
WARNING: The following packages cannot be authenticated!
  minicom
Install these packages without verification [y/N]? █
```

选择 y 回车即可。这样就安装好了 minicom。

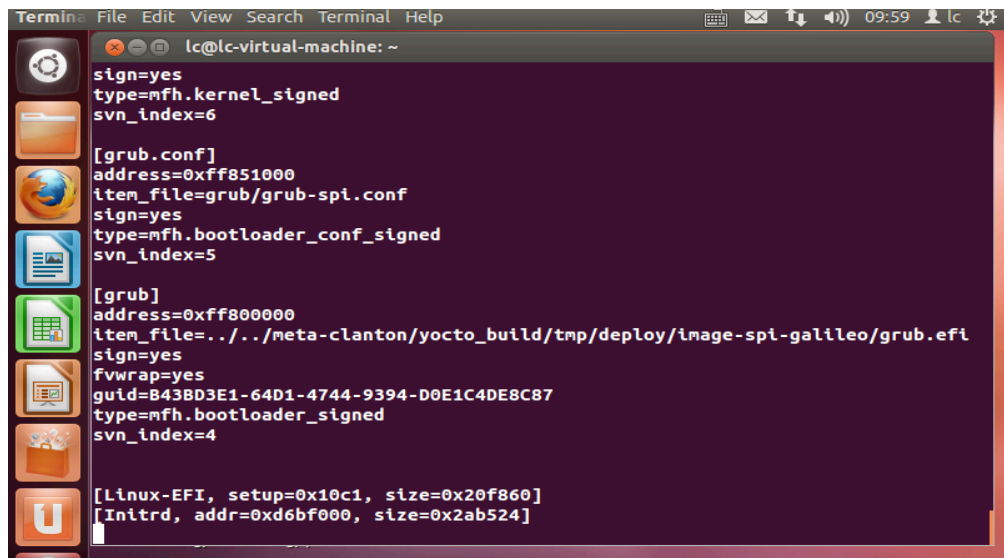
现在我们打开 minicom 来观察一下 SD 卡文件系统的启动过程



上图是 usb 转串口模块，三根线和板子相连就可以了，GND 接板子的 GND，TX 接板子的 RX，RX 接板子的 TX；

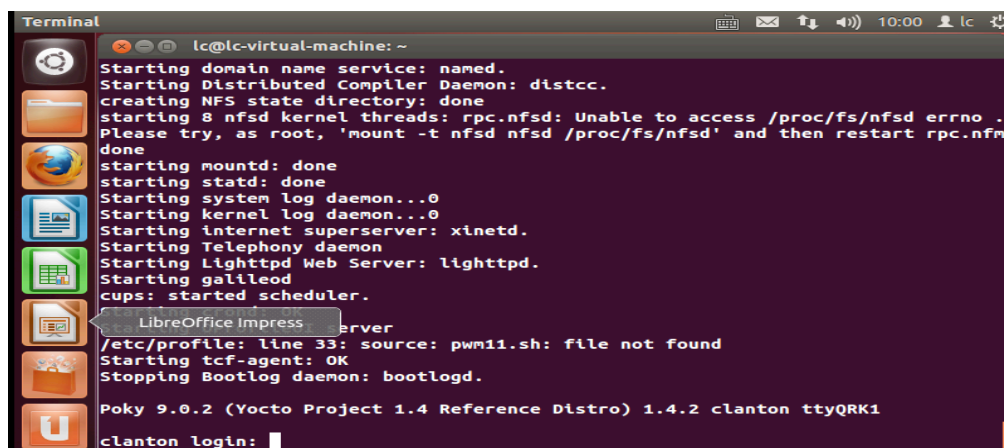
在终端输入 `sudo minicom` 进入超级终端

接好插上板子的电源，板子开始启动



```
Termin: File Edit View Search Terminal Help
lc@lc-virtual-machine: ~
sign=yes
type=mfh.kernel_signed
svn_index=6
[grub.conf]
address=0xff851000
item_file=grub/grub-spi.conf
sign=yes
type=mfh.bootloader_conf_signed
svn_index=5
[grub]
address=0xff800000
item_file=../../meta-clanton/yocto_build/tmp/deploy/image-spi-galileo/grub.efi
sign=yes
fvwrap=yes
guid=B43BD3E1-64D1-4744-9394-D0E1C4DE8C87
type=mfh.bootloader_signed
svn_index=4
[Linux-EFI, setup=0x10c1, size=0x20f860]
[Initrd, addr=0xd6bf000, size=0x2ab524]
```

过差不多几十秒后，启动完毕如下图：



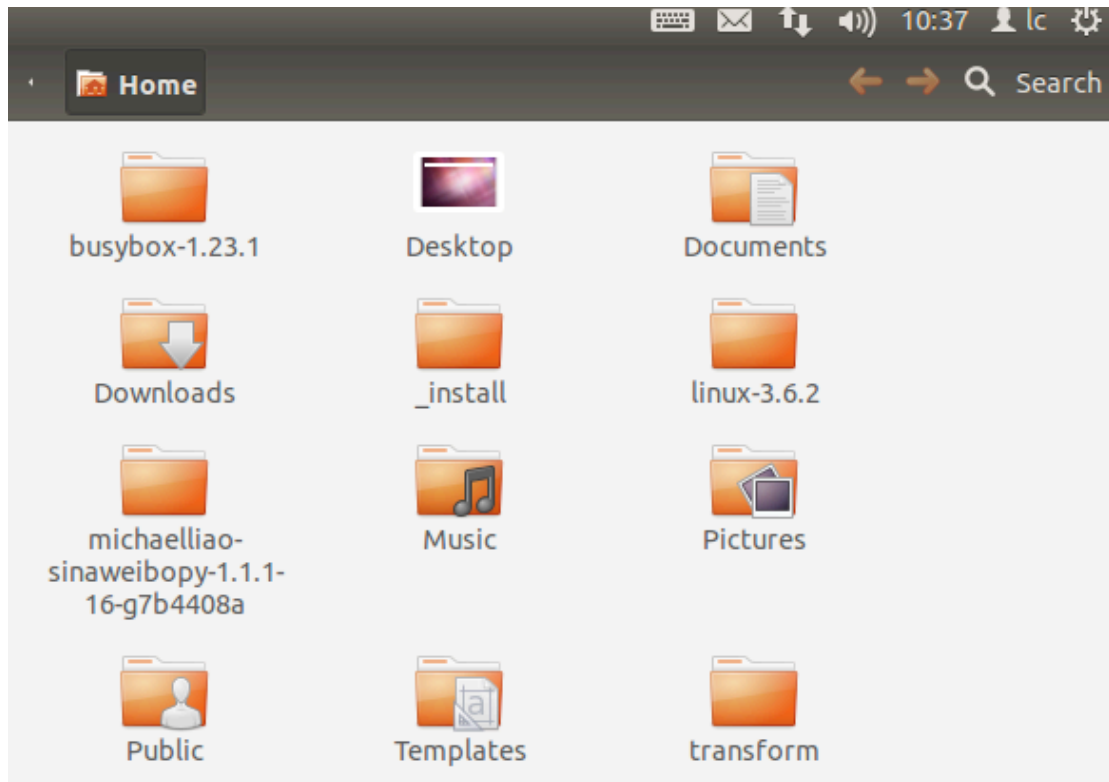
```
Terminal
lc@lc-virtual-machine: ~
Starting domain name service: named.
Starting Distributed Compiler Daemon: distcc.
creating NFS state directory: done
starting 8 nfsd kernel threads: rpc.nfsd: Unable to access /proc/fs/nfsd errno .
Please try, as root, 'mount -t nfsd nfsd /proc/fs/nfsd' and then restart rpc.nfsd
done
starting mountd: done
starting statd: done
Starting system log daemon...0
Starting kernel log daemon...0
Starting internet superserver: xinetd.
Starting Telephony daemon
Starting Lighttpd Web Server: lighttpd.
Starting galileod
cups: started scheduler.
LibreOffice Impress server
/etc/profile: line 33: source: pwm11.sh: file not found
Starting tcf-agent: OK
Stopping Bootlog daemon: bootlogd.
Poky 9.0.2 (Yocto Project 1.4 Reference Distro) 1.4.2 clanton ttyQRK1
clanton login: █
```

最后可以看到这个文件系统的版本。

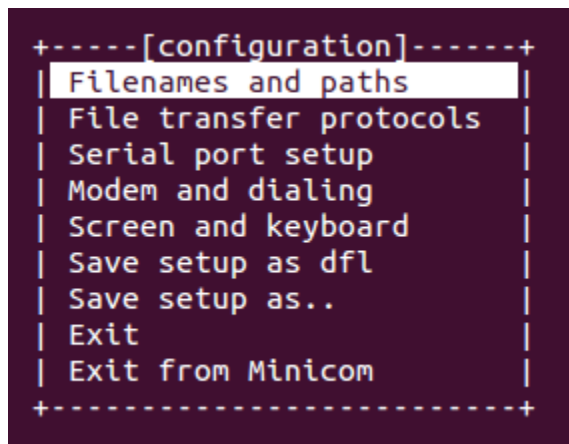
好了，接下去写一下怎么将电脑上的文件通过串口传送到板子上，后面后面我们需要把程序编译好可执行文件拷贝到 SD 卡中在板子运行。所以通过 `minicom` 来传送可执行代码会方便很多

首先在可以

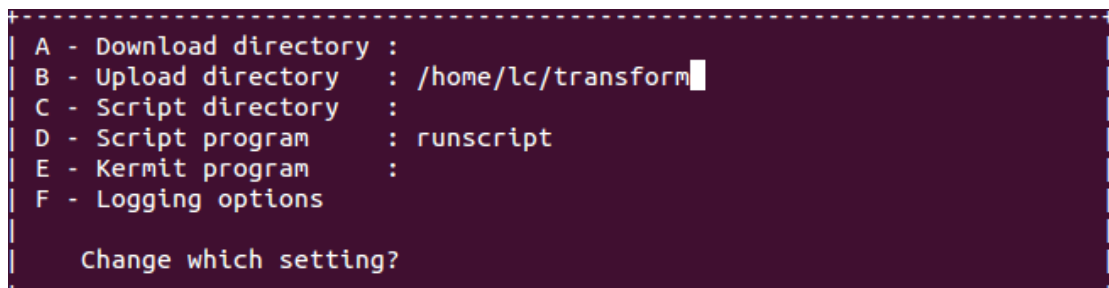
Home 下建一个文件夹叫做 `transform`，今后可以把要传送的文件放在这个文件下，



然后再终端中输入 `sudo minicom -s`
来配置传送路径



回车



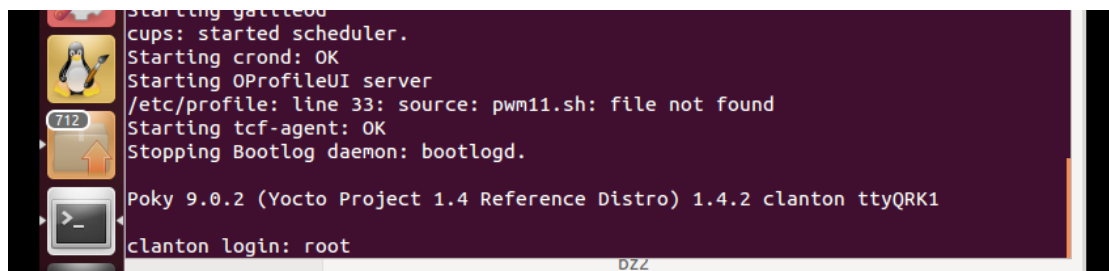
选择 B，然后填入你刚才创建的那个文件的路径即可。

其他的设置这里不再说明，可以参考网站的介绍使用 and 配置方法

接下去再说一下把文件传到板子上的一个过程

首先打开 minicom

sudo minicom



```
starting gattledb
cups: started scheduler.
Starting crond: OK
Starting OProfileUI server
/etc/profile: line 33: source: pwm11.sh: file not found
Starting tcf-agent: OK
Stopping Bootlog daemon: bootlogd.

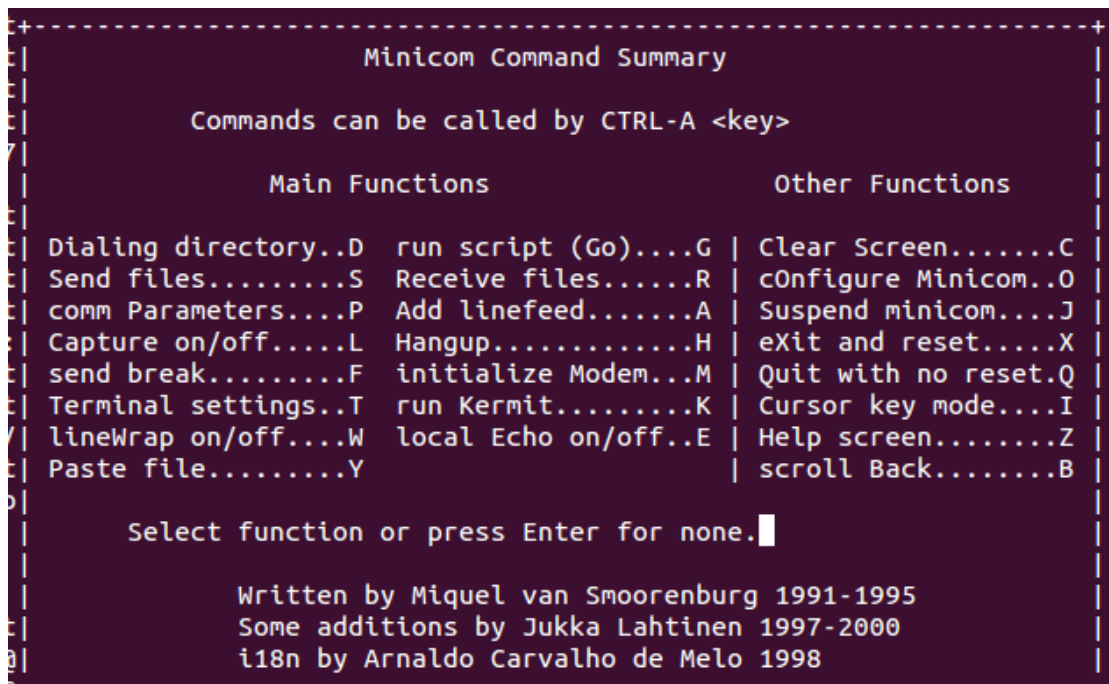
Poky 9.0.2 (Yocto Project 1.4 Reference Distro) 1.4.2 clanton ttyQRK1
clanton login: root
```

然后 root 进入超级用户权限

cd /media/realroot/

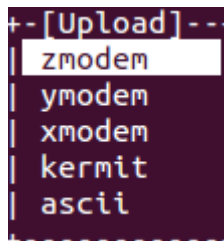
一般我们把文件传到这个路径下，你们可以自己选择

然后 Ctrl +A 键然后按 Z 进入



```
+-----+
|                                     |
|               Minicom Command Summary               |
|                                     |
|   Commands can be called by CTRL-A <key>             |
|                                     |
|           Main Functions           |           Other Functions           |
|                                     |                                     |
| Dialing directory..D   run script (Go)....G | Clear Screen.....C |
| Send files.....S     Receive files.....R | cOnfigure Minicom..O |
| comm Parameters....P  Add linefeed.....A | Suspend minicom....J |
| Capture on/off.....L Hangup.....H       | eXit and reset....X |
| send break.....F     initialize Modem...M | Quit with no reset.Q |
| Terminal settings..T  run Kermit.....K   | Cursor key mode....I |
| lineWrap on/off....W  local Echo on/off..E | Help screen.....Z   |
| Paste file.....Y     | scroll Back.....B |
|                                     |
| Select function or press Enter for none. |
|                                     |
| Written by Miquel van Smoorenburg 1991-1995 |
| Some additions by Jukka Lahtinen 1997-2000 |
| i18n by Arnaldo Carvalho de Melo 1998      |
|                                     |
+-----+
```

再按 S 选择传送文件。



```
+-[Upload]--
| zmodem
| ymodem
| xmodem
| kermit
| ascii
```

选 zmodem 回车

然后上下键选择你所要传送到板子的文件按空格键选中，然后回车就可以了。

I/O 基本操作

Blink/ 点亮 LED 灯

数字电平输出

```
int led = 13;           // Arduino 板子的 LED 一般是 pin13
pinMode(led, OUTPUT);   // 配置 IO 引脚的方向 (INPUT, OUTPUT, INPUT_PULLUP)
digitalWrite(led, HIGH); // 控制引脚, 输出高/低电平

delay(1000);           // wait for a second, 毫秒级延时
```

数字电平读取

```
int pushButton = 2;
pinMode(pushButton, INPUT);
int buttonState = digitalRead(pushButton);
```

模拟电平读取

```
int sensorValue = analogRead(A0); //这里的 A0是板子上 ANALOG IN 中的 A0~A5.
```

模拟电平输出

```
analogWrite(pin, value);
//value: the duty cycle: between 0 (always off) and 255 (always on).
```

DigitalReadSerial/ 串口输出

Intel Galileo 只有一个串口, 使用 USB 连接 Galileo 板子和电脑时, 板子上有转换电路。否则, 使用引脚 RX(0),TX(1)外接 TTL/USB 连接电脑。

串口配置

```
Serial.begin(9600);           // 配置波特率
```



```
Serial.println(buttonState);    // 串口输出
```

Millis/ 运行时间读取

返回当前的程序的运行毫秒数 绝对值没有多少意义，一般都是调用两次，计算差值。

```
unsigned long time = millis();  
//Returns Number of milliseconds since the program started (unsigned long)
```

Debounce/ 按键去抖

检测到按键按下之后，记录 millis()值。

等待一段之间(定义的去抖阈值),再判断按键,若仍为按下状态,才执行对应操作。

tonePitchFollower/ 模拟量转方波驱动扬声器

```
int sensorReading = analogRead(A0);  
int thisPitch = map(sensorReading, 400, 1000, 120, 1500);  
tone(9, thisPitch, 10);  
//tone()在一个引脚产生占空比50%的方波  
//第三个参数为持续时间(毫秒)，可选.未声明则直到 noTune()才停止。  
//统一时间只能有一个 tone()工作。
```

Serial output 串口输出

ASCIITable/ 输出语句

```
Serial.println("hello");  
Serial.write(thisByte);  
Serial.print(thisByte);  
Serial.print(thisByte, HEX);  
Serial.print(thisByte, OCT);
```

```
Serial.print(thisByte, BIN);
```

Serial_Read/ 串口交互

brightness = Serial.read(); //注意,串口那里需要设置成不自动添加回车, 否则发完数据会自动发一个回车.

Arrays/ 数组操作

所有位置都可以改为数组元素操作.

```
pinMode(ledPins[thisPin], OUTPUT);  
digitalWrite(ledPins[thisPin], HIGH);
```

以太网配置

ChatServer/

将 Galileo 配置成一个局域网中的聊天服务器.

其他电脑可以 telnet 到此服务器, 输出的消息会广播给所有连接此服务器的客户端。

使用 SPI 对应的引脚。 Ethernet shield attached to pins 10, 11, 12, 13

```
#include <SPI.h>  
#include <Ethernet.h>  
  
//配置 Galileo 的 Mac 地址,IP,gateway, subnet.  
byte mac[] = { //Galileo 的以太网端口地址, 板子上有标签98:4F:EE:00:2E:98  
0x98, 0x4F, 0xEE, 0x00, 0x2E, 0x98 };  
IPAddress ip(10,42,0, 177);  
IPAddress gateway(10,42,0, 1);  
IPAddress subnet(255, 255, 255, 00);  
  
// telnet defaults to port 23  
EthernetServer server(23);  
  
// initialize the ethernet device  
// 支持 DHCP -> Ethernet.begin(max);
```

```
//          Ethernet.begin(mac, ip);
Ethernet.begin(mac, ip, gateway, subnet);

// wait for a new client:
EthernetClient client = server.available();

// 判断
if(client)
{
  ...
}

server.write(val);
server.write(buf, len);

//具体应用看 Arduino - Ethernet library - Server class / Client class
```

EEPROM

Intel Galileo 的 EEPROM 是11KB, 所以地址是从0~(11 * 1024 - 1), 即 0~11263.

```
#include <EEPROM.h>

//the EEPROM can only hold a value from 0 to 255.
EEPROM.write(address,value);

value = EEPROM.read(address);
```

UART0串口

[How do I access /dev/ttyS0?](#)

参考 Programming_GPIO_From_Linux/的文档教程,学会 linux 下控制 Galileo 的 gpio.

ttyS0对应 UART0, 对应引脚 IO0,IO1.

查看 Galileo IO mapping 可知, 相关引脚在 linux 下为 gpio40,gpio41,gpio4.

gpio40是0-RX 的选择器控制端口, 其值为0时选择 ttyS0, 为1选择 gpio50.

gpio41是1-TX 的选择器控制端偶, 0->ttyS0, 1->gpio51; gpio4是 Level Shifter OE, 要配置其输出为1使能.

导出引脚40,41,4

```
echo -n 40 > /sys/class/gpio/export
echo -n 41 > /sys/class/gpio/export
echo -n 4 > /sys/class/gpio/export
```

配置为输出:

```
echo -n "out" > /sys/class/gpio/gpio40/direction
echo -n "out" > /sys/class/gpio/gpio41/direction
echo -n "out" > /sys/class/gpio/gpio4/direction
```

配置输出值:

```
echo -n "0" > /sys/class/gpio/gpio40/value
echo -n "0" > /sys/class/gpio/gpio41/value

echo -n "1" > /sys/class/gpio/gpio40/value
```

引脚配置完成, 现在 ttyS0与 UART0的 TX0,RX0相连接. 硬件上, 使用 TTL 转 USB 模块, RX0(Galileo IO 0) -> TTL(TX), TX0(Galileo IO 1) -> TTL(RX), Galileo GND -> TTL(GND);

USB -> 电脑.

硬件配置完成!

接下来, 在电脑终端中连接对应串口, 波特率默认为9600 Mac OS X 下如下:

```
$ screen /dev/tty.usbserial-A7027C8G 9600
```

然后, 在 Galileo 的控制台中,

Galileo -> ttyS0

```
root@clanton:~# echo "hello, ttyS0, this is Galileo" > /dev/ttyS0
```

ttyS0 -> Galileo

```
root@clanton:~# cat /dev/ttyS0
```

这样就变为了输入. 然后在 **ttyS0**端口,也就是电脑中的 **screen** 中,进行输出, 信息就会通过串口显示在 Galileo 中.

UART 串口 - Serial

/dev/ttyS0 is the first Quark UART (UART0) and it's the Digital 0 and 1 pins

/dev/ttyS1 is the second Quark UART (UART1) and it's the audio jack one

/dev/ttyGS0 is the virtual serial device created by the Linux cdc-acm driver via the Galileo USB port marked as "USB client".

the Arduino mappings

Serial is routed to the /dev/ttyGS0

Serial1 is routed to /dev/ttyS0

Serial2 is not used

Arduino sketch

```
void setup() {  
}  
void loop() {  
  delay(5000);  
  if(Serial) {  
    Serial.println("This is Serial, must be mapped to /dev/ ttyGS0, i.e. visible in
```

```
Arduino IDE's Serial Monitor");
    }
    else {
printf("Serial is not ready");
    }
    if(Serial1) {
Serial1.println("This is Serial1, must be mapped to /dev/ttyS0, i.e. visible on something
connected to the Digital pins 0 & 1");
    }
    else {
printf("Serial1 is not ready");
    }
    if(Serial2) {
Serial2.println("This is Serial2, unused, you shouldn't see this anywhere");
    }
    else {
printf("Serial2 is not ready");
    }
    }
```

SD

The library supports FAT16 and FAT32 file systems on standard SD cards and SDHC cards.

It uses short 8.3 names for files.

The communication between the microcontroller and the SD card uses SPI, which takes place on digital pins 11, 12, and 13 (on most Arduino boards) or 50, 51, and 52 (Arduino Mega). Additionally, another pin must be used to select the SD card.

SD 卡启动 linux

Booting your board from an SD card(Galileo_GettingStarted.pdf - Page8)

Note: Your SD card must meet the following requirements:

- SD card must be formatted as FAT or FAT32.
- SD card size must be less than 32GB.

you may need to add a boot partition to your SD card.

To do this on a Windows machine, perform the following:

Open a cmd.exe instance as an Administrator. Run diskpart.exe and run the following commands:

```
select vol <a>; (where <a> = the drive letter of the SD card)
```

```
clean;
create part primary;
active;
format quick label="BOOTME";
exit
```

Copy all files and directories from the zip file to your SD card.

```
drwxrwxrwx@ 1 gxp  staff      4096 Apr 13 09:56 boot
-rwxrwxrwx@ 1 gxp  staff  2113856 Oct  1  2013 bzImage
-rwxrwxrwx@  1 gxp  staff      1441609  Oct    1    2013
core-image-minimal-initramfs-clanton.cpio.gz
-rwxrwxrwx@ 1 gxp  staff  314572800 Oct  1  2013 image-full-clanton.ext3
```

Insert the SD card, then power on the board.

get Ethernet MAC address in a sketch

```
Serial.println(system("cat /sys/class/net/eth0/address"));
```

or

```
Serial.println(system("cat /sys/class/net/eth0/address > /dev/ttyGS0"));
```

system()

`system()`可以执行 linux 命令行的语句, 并且返回0/256. 成功返回0, 失败返回256.

```
void setup()
{
    // put your setup code here, to run once:
    Serial.begin(9600);

    Serial.println("\n\nHello world!\n");

    Serial.println(system("ifconfig -a"));
    Serial.println(system("ifconfig -all"));
    Serial.println(system("cat /sys/class/net/eth0/address"));
    Serial.println(system("mac_address"));
    Serial.println(system("/sbin/ifconfig"));
}

void loop()
{ }
```

开机自启动脚本设置

Galileo/Documents/boot_script

IO 速度研究

Galileo/Documents/IO_speed

linux GPIO

Galileo/Documents/Programming_GPIO_From_Linux

Sketch Size Limits

Galileo/Documents/sketch_size

Galileo PWM

Galileo/Documents/PWM

五、实验报告及要求

分别利用 Arduino IDE 和 Linux 完成 Galileo 开发板外设的控制，使板上的 LED 灯每隔一秒变换一次状态。

实验报告中要包含以下几个部分：

- 1、实验目的
- 2、实验条件
- 3、实验原理
- 4、实验步骤分析
- 5、实验结果与总结