CUBIEBOARD
http://cubie.cc
Fucking cool，深刻的嵌入式技术和应用讨论社区

Cubieboard 上使用 ARM Streamline

ARM-DS-5

Website:http://cubieboard.org/
Support: support@cubietech.com

文档版本	作者	修改内容	审核
V-0.1-20140115	Parker	初始版本	

内容目录

1.概述

本文档将描述如何在 cubieboard 上使用 ARM-Streamline™调试 Android&Linux 系统， 调试的方式包括网络调试和 ADB 调试（只支持 android），另外本文以 linux 操作系统为例，windows 的使用方法类似。

2.ARM Streamline 介绍

2.1.什么是 ARM Streamline？

ARM Streamline™ 性能分析器是 ARM DS-5™ 工具链的一部分，它使软件开发人员能够充分利用基于 ARM 处理器的系统上的可用资源，以创建高性能和高能效的产品。它配有直观的图形用户界面，可显示从 CPU 和 GPU 性能计数器到源代码热点再到实际功耗等信息，这样，开发人员就可方便地缓解性能瓶颈，改进代码并行度，延长电池寿命并增强用户体验。Streamline 以系统跟踪点、硬件和软件性能计数器、基于样本的分析和用户注释为基础，提供了用于软件优化的功能强大而灵活的系统分析环境。更多的介绍和细节请访问其主页：http://ds.arm.com/zh-cn/

2.2.为何使用 Streamline ？

提高代码速度

- 找出 CPU 耗费时间较多的位置
- 改进多核平台的代码并行度
- 调整代码以实现最优高速缓存使用、向量化等

降低能耗

- 使用 ARM 能量探测器来监视实际功耗、电流和电压
- 发现改进电源管理方案的机会
- 优化计算任务以实现最佳能效

有效利用系统资源

- 分析和优化 Mali™ GPU 利用率以及 CPU 代码
- 监视 CPU 和 Mali GPU 高速缓存使用情况和系统内存
- 检查跨多个内核的负载分配情况

针对系统进行自定义

- 将自己的数据连接到 Streamline 分析视图
- 扩展开源驱动程序以监视变量和组件
- 检测用于向 Streamline 发送类似 printf 的消息的代码

3.开发前的准备

1）Ubuntu12.04 操作系统的电脑

2）cubietruck 开发板一块

3）USB-MiniUSB 数据线，用于主机和开发板的数据交互

（linux 系统没有 adb 工具，则通过网络进行数据交互）

4）下载 cubieboard android 或 linux 源码

5）下载 DS-5 源码包

4.下载源码和工具

Linux 源码：

$ mkdir linux-sdk-card

$ cd linux-sdk-card

1) kernel-source:

$ git clone https://github.com/cubieboard/linux-sdk-kernel-source.git

$ mv linux-sdk-kernel-source linux-sunxi

2) tools:

$ git clone https://github.com/cubieboard/linux-sdk-card-tools.git

$ mv linux-sdk-card-tools tools

3) products:

$ git clone https://github.com/cubieboard/linux-sdk-card-products.git

$ mv linux-sdk-card-products products

4) rootfs&u-boot:

$ git clone https://github.com/cubieboard/linux-sdk-binaries.git

$ mv linux-sdk-binaries binaries

Get file from:

http://dl.cubieboard.org/model/commom/linux-sdk-binaries

android4.2 源码：

git clone https://bitbucket.org/cubietech/a20-android4.2_lichee.git

git clone https://bitbucket.org/cubietech/a20-android4.2_android.git

请到 ARM 官网下载 DS-5 工具：

https://silver.arm.com/browse/browse_eval.tm

关于如何编译和构建 cubieboard 固件，请参考如下的教程：

android: http://pan.baidu.com/s/1dDF5cVR

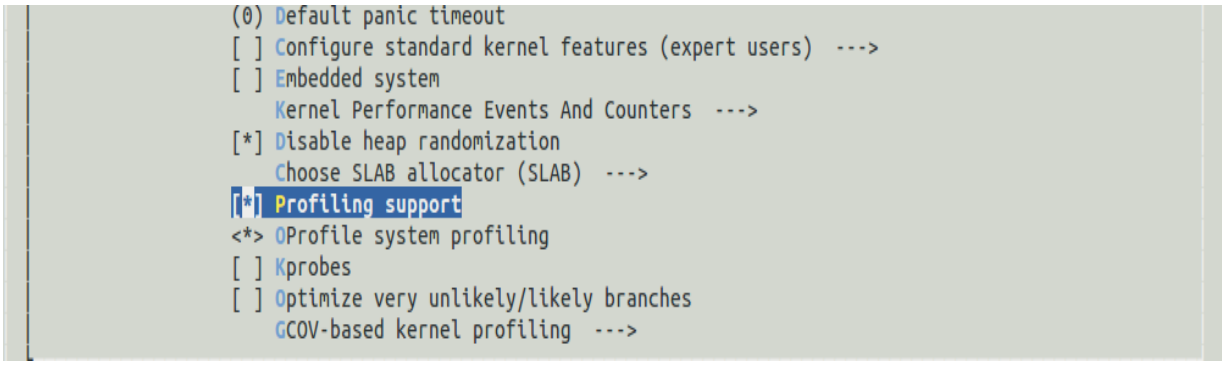
linux: http://pan.baidu.com/s/1o6LYsDs

5.添加内核选项

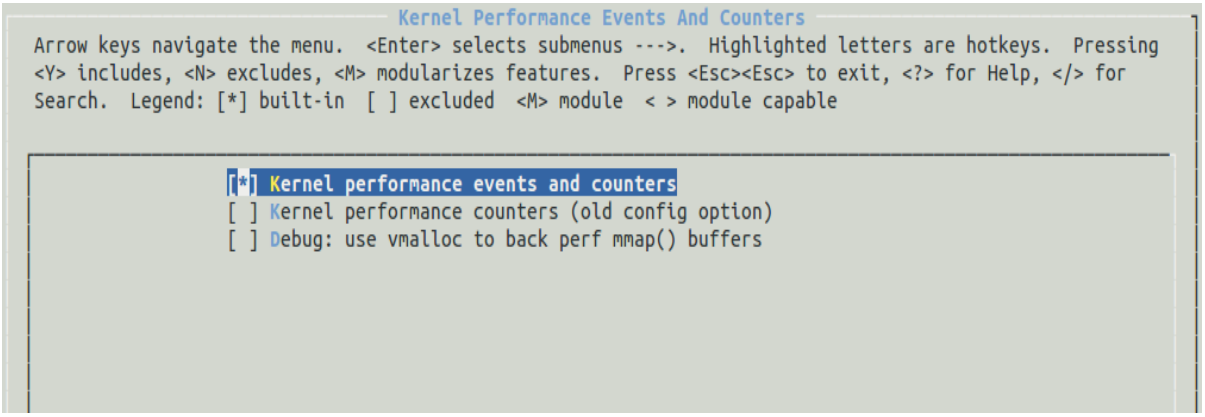
需要重新编译内核以支持 ARM Streamline™,在公版的 SDK 里面，android 内核目录为：lichee/linux-3.4，linux 的内核目录为 linux-sunxi 。

5.1.General Setup

进入 General setup 把 Profiling support 选上，如下图：

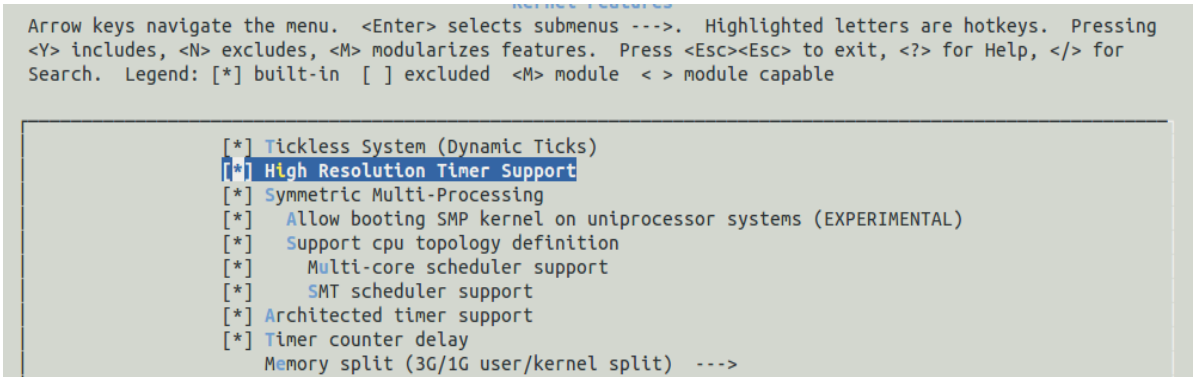


进入 General setup -> Kernel Performance Events And Counters 把 Kernel performance events and counters 选上，如下图：

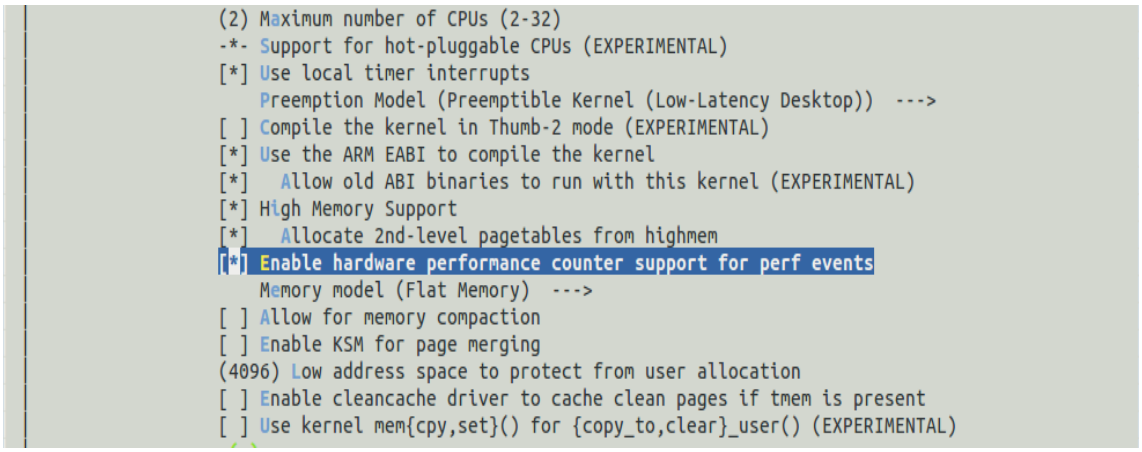


5.2.Kernel Features

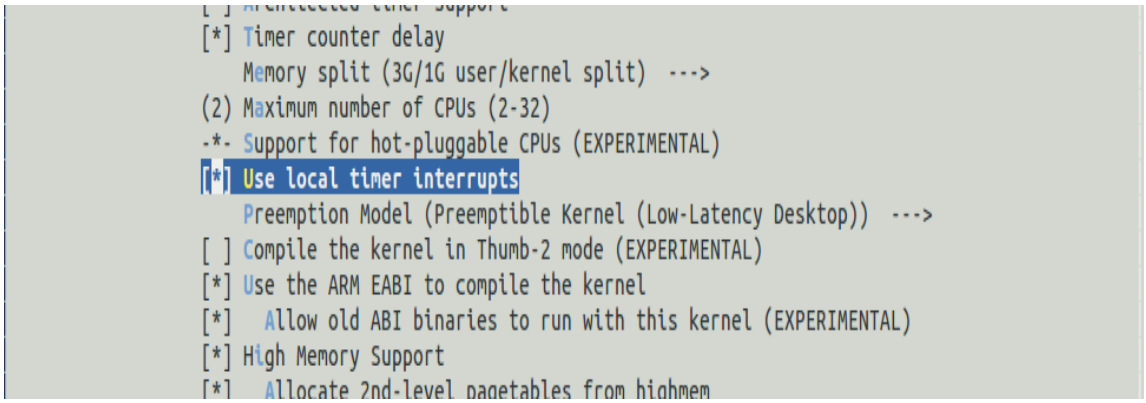
进入 Kernel Features 把 High Resolution Timer Support 选上，如下图：



进入 Kernel Features 把 Enable hardware performance counter support for perf events 选上，如下图：

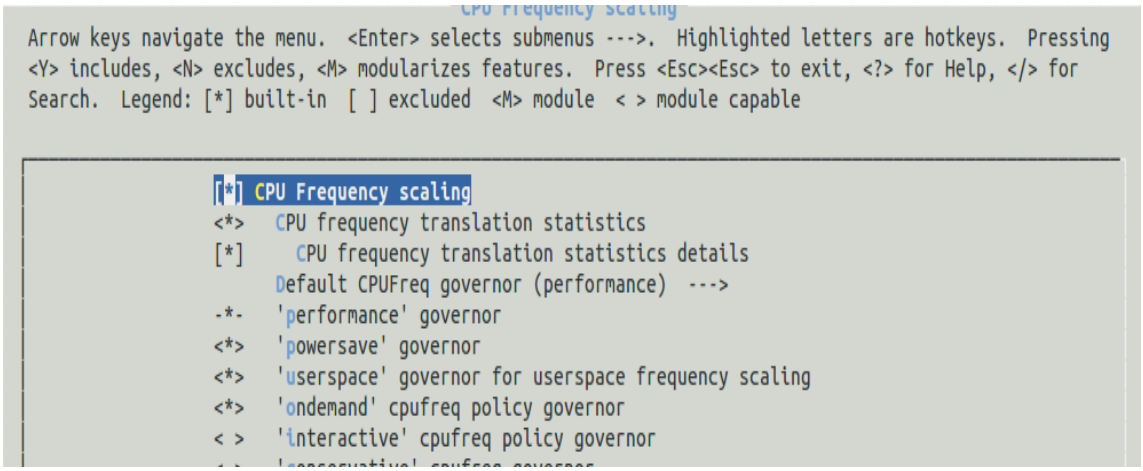


进入 Kernel Features 把 Use local timer interrupts 选上，如下图：



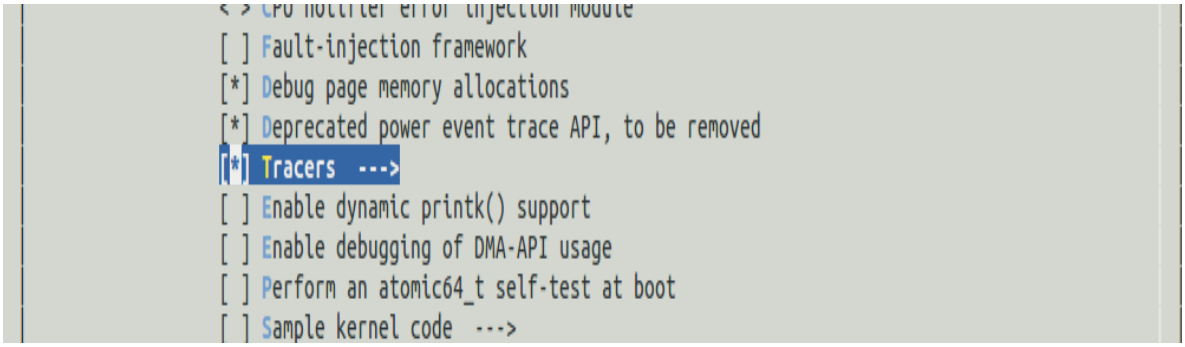
5.3.CPU Power Management

进入 CPU Power Management -> CPU Frequency scaling 把 CPU Frequency scaling 选上，如下图：

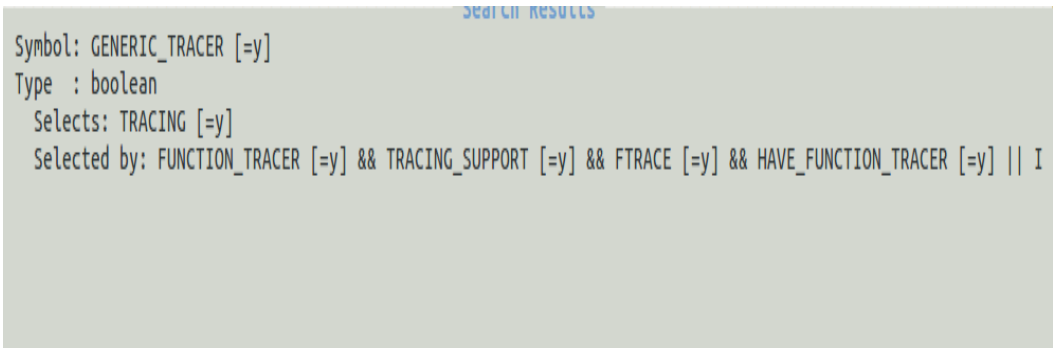


5.4.Kernel hacking

进入 Kernel hacking 把 Tracers 选上，如下图：



最后确认 CONFIG_GENERIC_TRACER 和 CONFIG_TRACING 都被选上，如下图：



```
Symbol: TRACING [=y]
Type  : boolean
  Selects: DEBUG_FS [=y] && RING_BUFFER [=y] && STACKTRACE [=y] && TRACEPOINTS [=y] && NOP_TRACER [=y] && BI
  Selected by: GENERIC_TRACER [=y] || ENABLE_DEFAULT_TRACERS [=n] && TRACING_SUPPORT [=y] && FTRACE [=y] &&
```

当把以上选项都选好后，编译内核，制作固件，后续我们会用到这个固件。

6.编译 gator 驱动

把从第 1 步拿到的源码包解压，获得 gator-driver 驱动源码。
然后开始编译 gator.ko 驱动。

进入 gator-driver 目录

$ cd gator-driver

执行编译指令,注意编译的路径请替换成对应内核所在的路径，确保主机的开发环境已经安装了交叉编译工具连。

$ make -C /work/android4.2_tablet_A20/lichee/linux-3.4 M=`pwd` ARCH=arm

CROSS_COMPILE=arm-linux-gnueabi- modules

编译成功后就会在当前目录下生成 gator.ko，如下图：

```
parker@parker:/work/jtag/gator-driver$ make -C android4.2_tablet_A20/lichee/linux-3.4 M=`pwd` ARCH=arm CROSS_COMPILE=
arm-linux-gnueabi- modules
make: *** android4.2_tablet_A20/lichee/linux-3.4: No such file or directory.  Stop.
parker@parker:/work/jtag/gator-driver$ make -C /work/android4.2_tablet_A20/lichee/linux-3.4 M=`pwd` ARCH=arm CROSS_CO
MPILE=arm-linux-gnueabi- modules
make: Entering directory `/work/android4.2_tablet_A20/lichee/linux-3.4'
  CC [M]  /work/jtag/gator-driver/gator_main.o
  CC [M]  /work/jtag/gator-driver/gator_events_block.o
  CC [M]  /work/jtag/gator-driver/gator_events_irq.o
  CC [M]  /work/jtag/gator-driver/gator_events_meminfo.o
  CC [M]  /work/jtag/gator-driver/gator_events_mmapped.o
  CC [M]  /work/jtag/gator-driver/gator_events_net.o
  CC [M]  /work/jtag/gator-driver/gator_events_perf_pmu.o
  CC [M]  /work/jtag/gator-driver/gator_events_sched.o
  CC [M]  /work/jtag/gator-driver/gator_events_armv6.o
  CC [M]  /work/jtag/gator-driver/gator_events_armv7.o
  CC [M]  /work/jtag/gator-driver/gator_events_l2c-310.o
  CC [M]  /work/jtag/gator-driver/gator_events_scorpion.o
  LD [M]  /work/jtag/gator-driver/gator.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /work/jtag/gator-driver/gator.mod.o
  LD [M]  /work/jtag/gator-driver/gator.ko
make: Leaving directory `/work/android4.2_tablet_A20/lichee/linux-3.4'
```

注意：指令黄色部分要根据自己的实际情况填写，内核路径要选择你在第 2 步所使用的内核。选择的编译交叉工具链也要和内核的对应。

7.注册 arm 帐号

注册 arm 帐号可以获得 DS-5 30 天试用权，已经有 arm 帐号的可以跳过这一步。

直接进入 arm 注册页面，按照提示填写即可：
https://login.arm.com/register.php

8.在电脑上安装 DS-5

进入源码包找到 install.sh 脚本。

添加脚本可执行权限

$ chmod +x install.sh

执行脚本

$ ./install.sh

选择 B 进入下一步，如下图：

```
====================================
Welcome to the Installer for ARM DS-5
====================================
--- Host target check...[x86_64]
This installation has a post install step that requires root privileges
The post install stage performs the following functions:
- Set default toolkit selection
- Installation of USB drivers for RealView ICE and DSTREAM hardware units

 (A) Abort the installation and try again as a user with root privileges (Recommended)
 (B) Continue with the installation and skip stages that require root privileges
     (execute "run_post_install_for_ARM_DS-5.sh" afterwards as root to gain full functionality)

Please answer with one of: 'A/a' or 'B/b'
Enter option: [default: A] B
```

一直按回车键把信息读完后，输入“yes”，如下图：

```
R. Clang is licensed to you under the University of Illinois/NSCA Open Source License.

S. The Python interpreter and standard libraries, version 2.7.4 found in your installation at <install_dir>\sw\python
2.7\
and licensed to you under the Python Software Foundation Licence for Python 2.7.4. This package is also subject to ot
her third party licenses.

T. libstdc++ is licensed to you under the GNU General Public License version 3 plus runtime exception.

U. Portions of the software and firmware of ARM's Target Connection Products contain:
(i) an ARM Embedded Linux operating system together with patches developed by ARM to the Linux kernel, both of which
are licensed under the GNU General Public License version 2;
(ii) the GNU C library (glibc), licensed to you under the GNU Lesser General Public License versions 2.0 and 2.1;
(iii) Busybox licensed to you under the GNU General Public License version 2;
(iv) Python 2.5.2 is licensed to you under the Python Software Foundation License version 2; and
(v) other embedded software or data in the hardware unit, other than files in embedded directory /real-ice and subdir
ectories, is licensed to you under the GNU General Public License version 2.

V. Jython is licensed to you under the Python Software Foundation License version 2 and portions of the code are also
 subject to other terms and legal notices, including but not limited to the Apache Software License version 2.0, GNU
General Public License version 3, GNU Lesser General Public License version 3 and BSD.

W. The CoreSight Access Library is licensed to you under the Apache Licence version 2.0.

X. The Streamline Annotation Client is licensed to you under the terms of the BSD licence.

To the extent that ARM is obliged to do so, ARM hereby offers to supply the files which are subject to GNU licences (
identified above), in source code form, subject to the terms of the applicable GNU licence, upon request. This offer
is valid for three (3) years from the date of your acceptance of this Licence.

ARM Development Studio 5 v5.20
/end


Please answer with one of: 'yes' or 'no/quit'
Do you agree to the above terms and conditions? yes
```

一直输入“yes” 后，选择 DS-5 安装目录，如下图：

```
Please answer with one of: 'yes/y' or 'no/n'
Run installation platform requirement checks? [default: yes] yes

--- Running installation platform requirement checks

Running dependency check [succeeded]

Where would you like to install to? [default: /home/parker/DS-5]
```

等待安装，输入最后一次“yes”后，安装完成，如下图：

```
Please answer with one of: 'yes/y' or 'no/n'
Install desktop menu item additions? [default: yes] yes

--- Installing menu entries

--- Skipping post install setup scripts
    You can run these later by executing ./run_post_install_for_ARM_DS-5.sh with root privileges from inside the inst
allation.

---------------------------------
Installation completed successfully
---------------------------------

To start using ARM DS-5 either:
- Add /home/parker/DS-5/bin to your PATH
- Create a suite sub-shell using /home/parker/DS-5/bin/suite_exec <shell>
- Launch GUI tools via their desktop menu entries

The Release notes for the product can be found here: file:///home/parker/DS-5/sw/ARM_DS-5/readme.html

==================================
parker@parker:/work/jtag/DS-5/DS500-BN-00019-r5p0-20rel1$
```

添加 DS-5 环境变量，DS-5 安装目录下有 bin 文件夹，我们要把他加到环境变量里，如下图：

$ vim ~/.bashrc

$ source ~/.bashrc

9.在 CT 上开启 Streamline 调试工具

9.1.加载驱动和 shell

把第 1 步编译出来的 android4.2 cubietrck 固件，烧写到板子上，用 usb 线连接电脑和板子。

把先前编译好的 gator.ko 和源码包里的 gatord ，推送到板子/data 目录

$ adb push gator.ko /data

$ adb push gatord /data

进入板子文件系统

$ adb shell

加载 gator.ko 驱动

$ insmod gator.ko

执行 gatord

$ chmod 777 gatord

$ ./gatord &

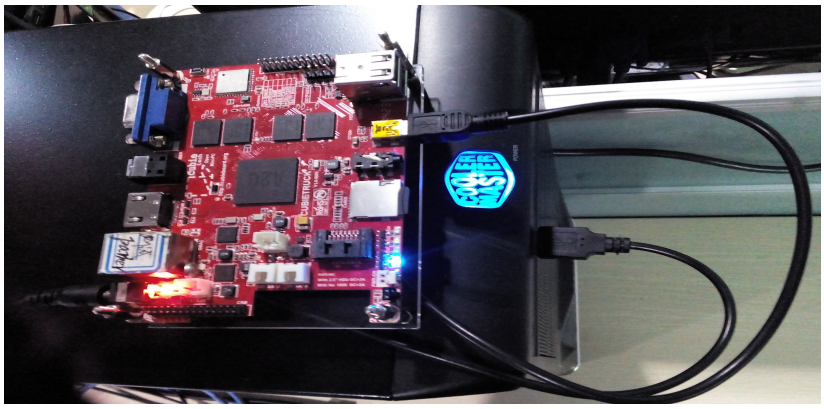
这样板子内部的环境就搭好了，如下图：

```
root@android:/data # insmod gator.ko
root@android:/data # lsmod
gator 57201 0 - Live 0x00000000 (O)
cdc_ether 3163 0 - Live 0x00000000
rtl8150 9023 0 - Live 0x00000000
mcs7830 5644 0 - Live 0x00000000
qf9700 5884 0 - Live 0x00000000
asix 13586 0 - Live 0x00000000
usbnet 13741 4 cdc_ether,mcs7830,qf9700,asix, Live 0x00000000
sunxi_csi0 30818 0 - Live 0x00000000
gc2035 13734 0 - Live 0x00000000
gc0308 11800 0 - Live 0x00000000
camera 36086 1 sunxi_csi0, Live 0x00000000
videobuf_dma_contig 4157 1 sunxi_csi0, Live 0x00000000
videobuf_core 16284 2 sunxi_csi0,videobuf_dma_contig, Live 0x00000000
sun7i_ir 5797 0 - Live 0x00000000
security_system 1067129 0 - Live 0x00000000
sw_device 11512 0 - Live 0x00000000
mali 151201 31 - Live 0x00000000 (O)
hdmi 25437 0 - Live 0x00000000 (O)
lcd 5155 0 - Live 0x00000000
disp 288683 13 mali,hdmi,lcd, Live 0x00000000
nand 142727 8 - Live 0x00000000 (O)
root@android:/data # chmod +x gatord
Bad mode
10|root@android:/data # chmod 777 gatord
root@android:/data # ./gatord &
[1] 3646
```

9.2.使用 adb 交互数据

android 使用 mini-usb 作为 CT 和 PC 主机传输数据的媒介，adb 调试开发选项已经默认打开，这里只需把硬件连线接，确保 PC 主机能识别到 CT 就可以了。

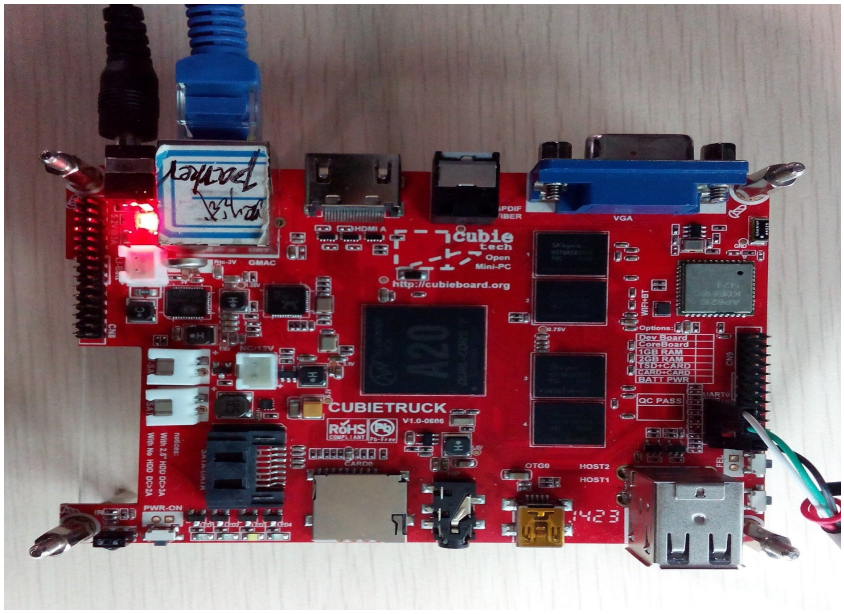
附一张 CT 实物接线图，接法很简单，只需上电和连接 mini-usb：



9.3.使用网络交互数据

这里使用 mini-usb 只是其中一种方法，如果没有 mini-usb 和 adb 工具，可以通过网络加串口调试工具的方式连接 PC 主机和 CT，但是 gator.ko 和 gatord 就要用 u 盘或者 TF 卡拷贝到板子内部。

附一张，网线和串口接线图，如果使用的是 wifi 则不需要接网线，只要确保获取到 IP 地址即可：



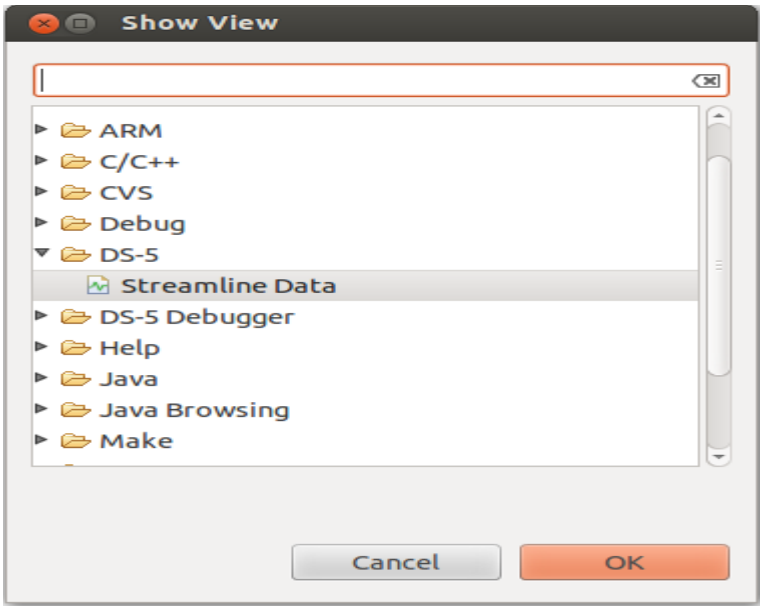
10.使用 DS-5

终端输入

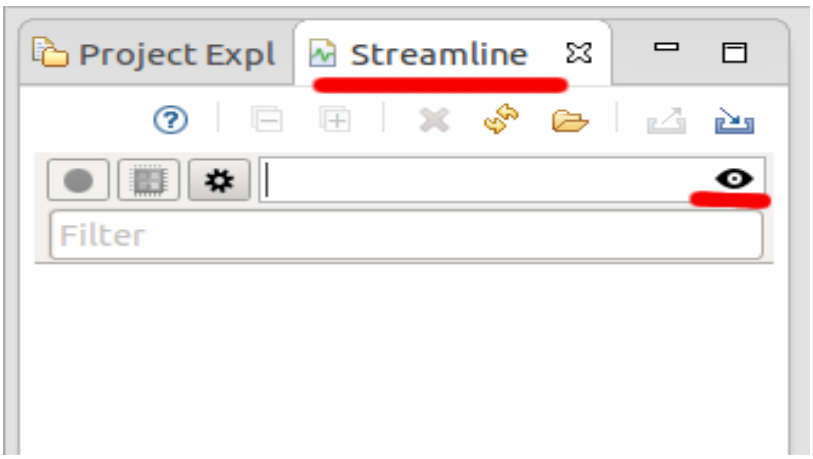
$ eclipse

10.1.创建 Streamline Data 工程

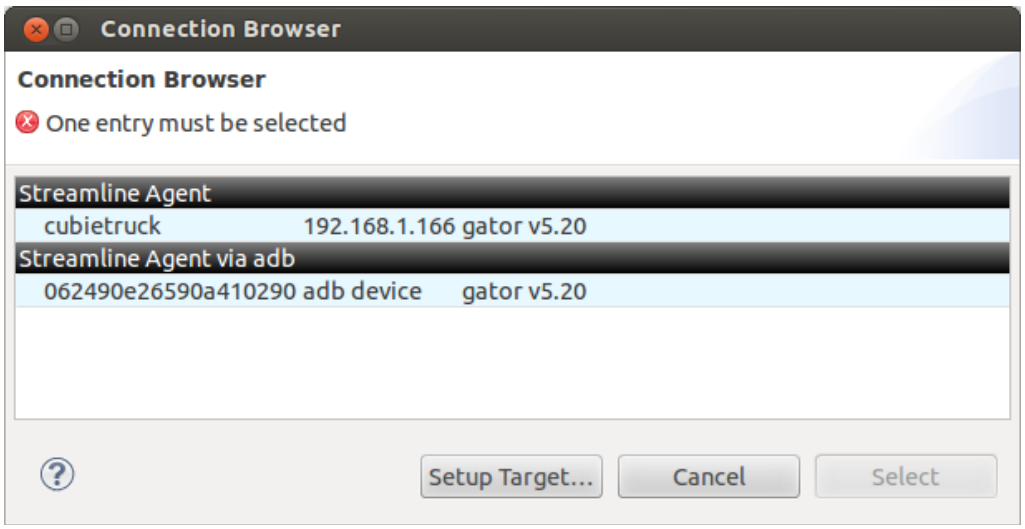
点击菜单栏的 Window > Show View > Other... ，选中 Streamline Data，点击 “OK”，如下图：



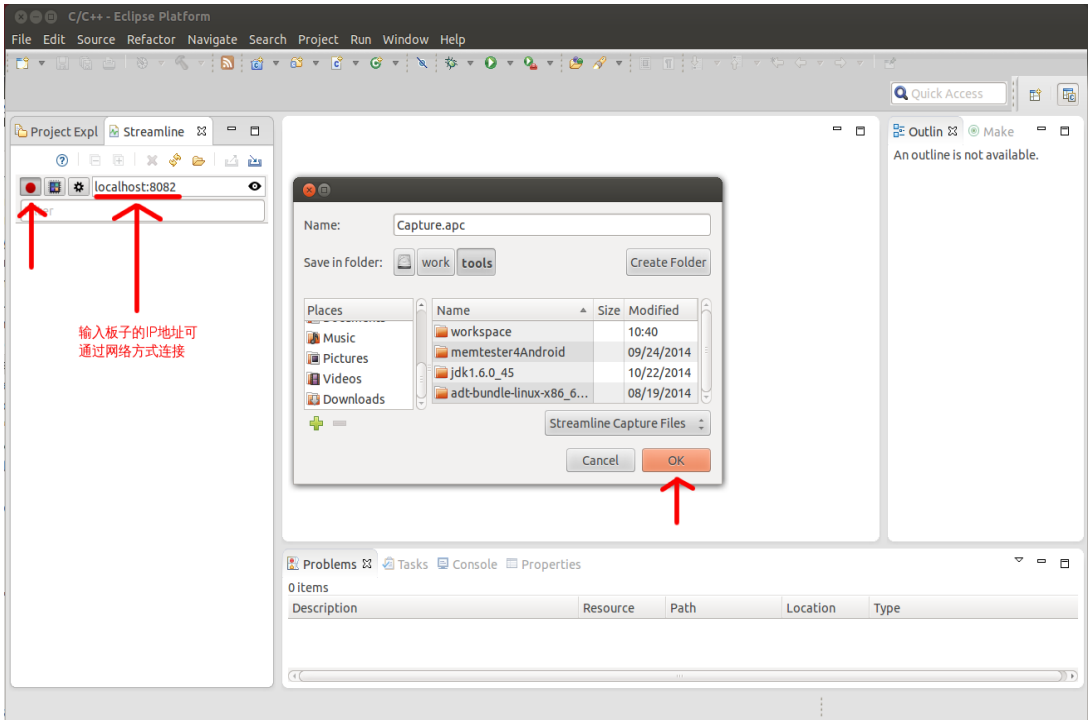
之前我们在第 6 步把板子的环境搭起来了，确保 usb 线已经连接了电脑和板子，选中 Streamline 工程，并点击眼睛一样的图标，如下图：



这时，选择 “Streamline Agent via adb”， 如下图：



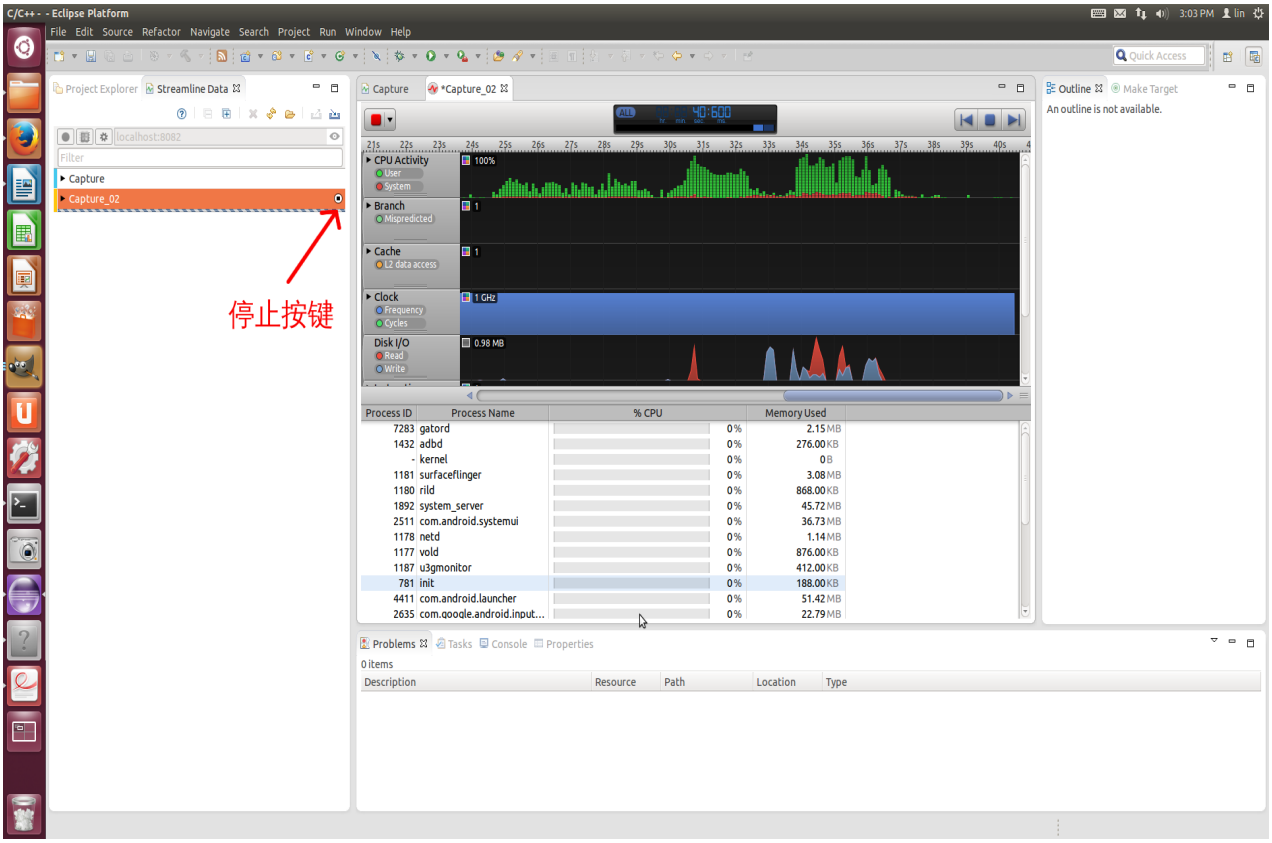
最后点击红色图标，选择保存路径，就可以开始调试了，如下图：

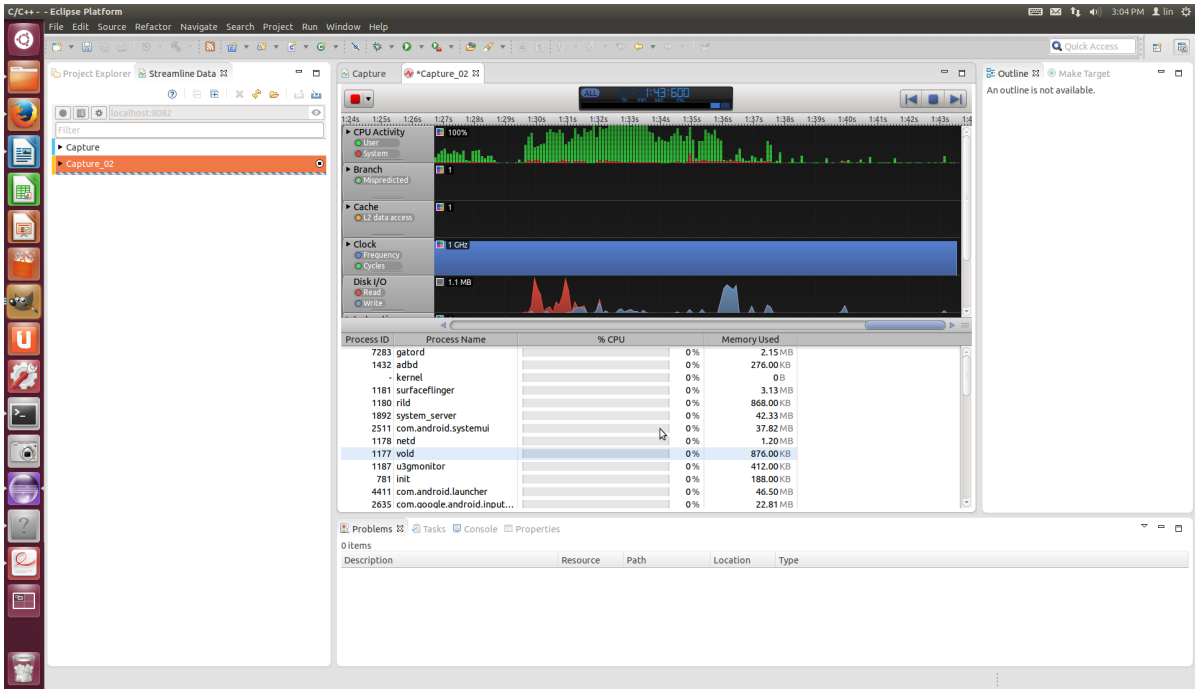


注意：linux 系统的板子无法通过 adb 连接，我们只要把 “localhost：8082” 替换成板子的 IP 地址 (例如输入：192.168.1.174) 即可通过网络的方式连接。

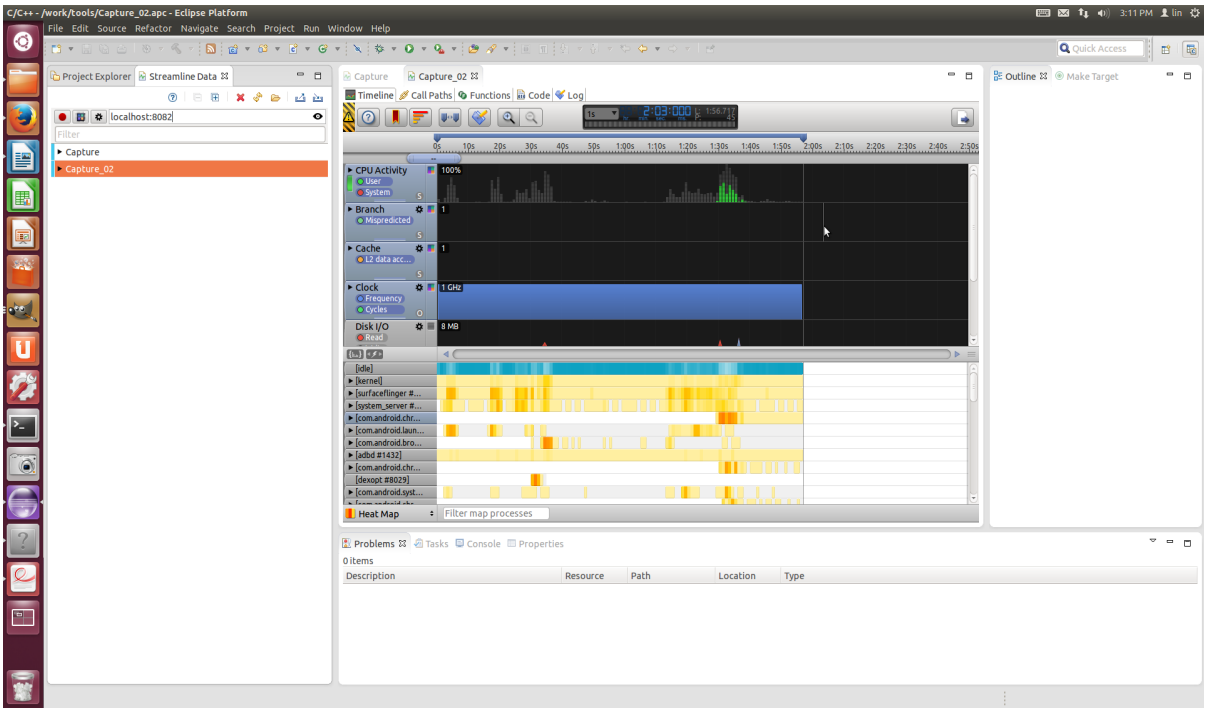
10.2.DS-5 工作效果图

DS-5 正在监测系统：



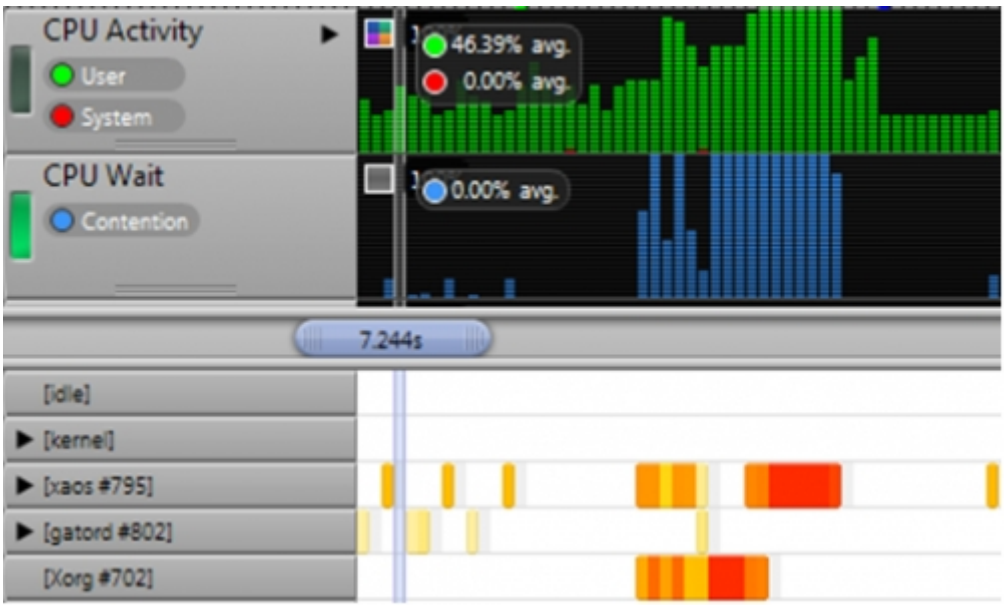


DS-5 停止工作，导出分析结果：



10.3.Streamline 简单分析

Streamline 进程选项卡可以配置显示不同信息，通过点击矩形旁边的 GPU 和 CPU 等活动图，可以知道每个 GPU 线程的时间和 CPU 等待时间。例如，在下面的例子中你可以看到 Xorg 花了很多时间在等待 xaos 释放处理器，另外 xaos 也在等待其他线程的活动结束。



更详细的说明请查阅 ARM 官网的文档：
http://ds.arm.com/zh-cn/developer-resources/ds-5-documentation/