**修订记录**

# 软件调试

## 搭建调试环境

现有的linux虚拟机工程在E:\Vmware\_workspace\Blackbox，可以用Vmware打开进入linux，用户名和密码都是user

linux虚拟机内所有环境都已经搭好，可以直接编译。

TI的SDK包目录为/media/DM8148/ti-ezsdk\_dm814x-evm\_5\_05\_02\_00

linux内核目录为/media/DM8148/linux-patched

SDK安装包目录：/media/software/ ezsdk\_dm814x-evm\_5\_05\_02\_00\_setuplinux

工具链安装包目录：/media/software/CodeSourcery.tar.gz

重新搭建调试环境以及SDK包的说明可以参考以下文档：

工具链以及SDK的在线资源参考链接：

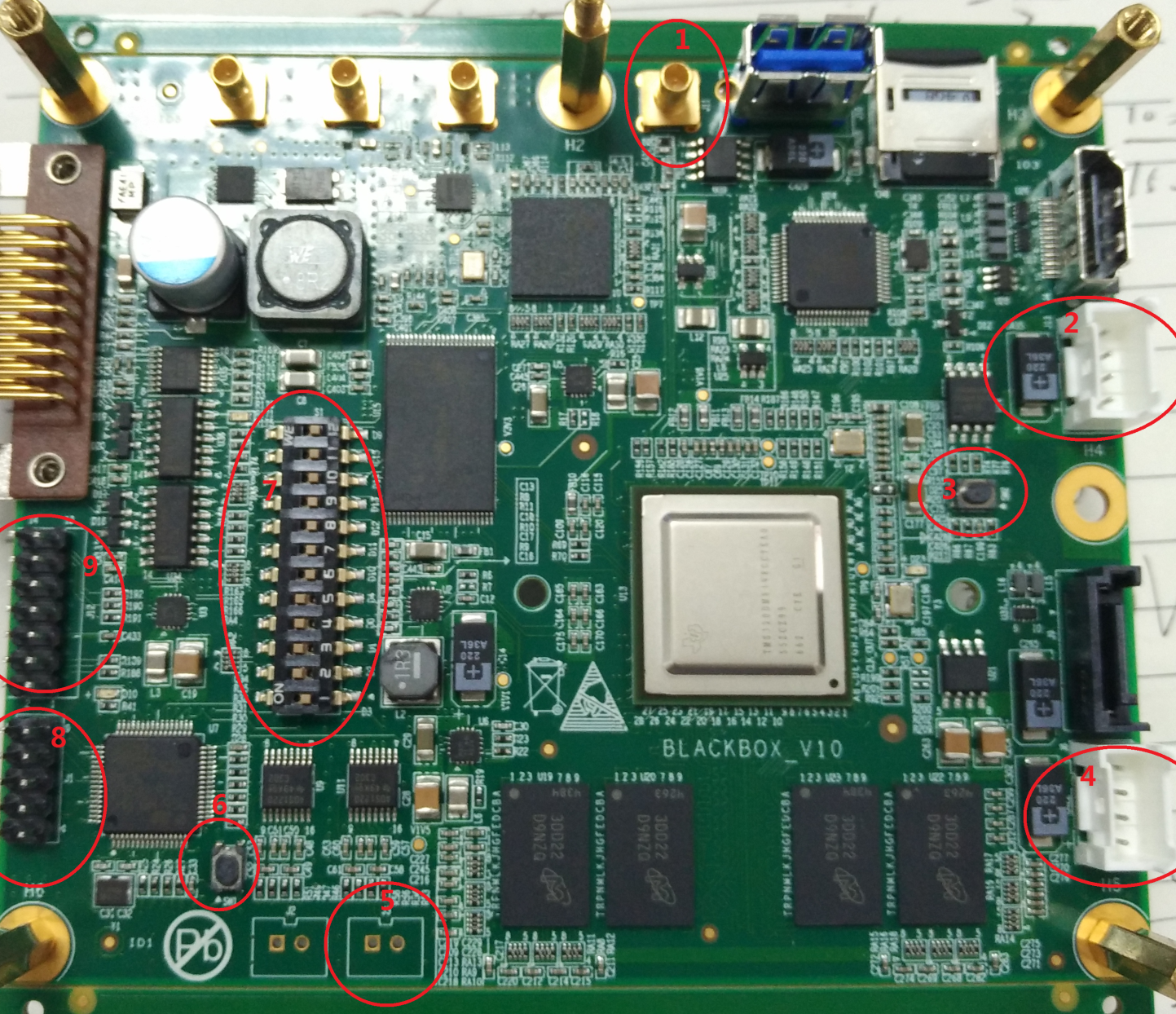
http://processors.wiki.ti.com/index.php/Category:EZSDK

### 内核编译

make ARCH=arm CROSS\_COMPILE=/home/user/ToolChain/CodeSourcery/Sourcery\_G++\_Lite/bin/arm-none-linux-gnueabi- tisdk\_dm814x-evm\_defconfig

make ARCH=arm CROSS\_COMPILE=/home/user/ToolChain/CodeSourcery/Sourcery\_G++\_Lite/bin/arm-none-linux-gnueabi- uImage

## 板卡设置



1.模拟视频输出 2.gopro电源和控制 3.CPU复位 4.sata电源

5.测温电阻接口 6.STM32复位 7.启动设置开关 8.stm32调试口

9.CPU调试口

* 拨动开关S1的2、3、4、5拨到ON的位置，设置为SD卡启动
* SD卡插拔

按下图箭头方向（朝板卡内侧）拉SD卡槽盖，卡槽金属盖会自动弹起，将SD卡放入，盖上金属盖，然后朝板卡外侧拉金属盖，即可锁死



* 如果使用USB口传输数据，对于V10版本，需要将TP10接地
* sata电源插座和gopro插座的线材制作使用已采购的工具（编号638118100）

，工具使用说明可以参考工具盒内说明，或者参考链接

http://www.molex.com/molex/products/datasheet.jsp?part=active/0638118100\_APPLICATION\_TOOLIN.xml

## 板卡开机运行

开机进入登录界面，输入密码root

依次运行以下命令进行系统配置(或者运行./tv-out.sh):

/etc/init.d/matrix-gui-e stop

/etc/init.d/pvr-init stop

/usr/share/ti/ti-media-controller-utils/load-hd-v4l2-firmware.sh start

echo 0 > /sys/devices/platform/vpss/display2/enabled

echo ntsc > /sys/devices/platform/vpss/display2/mode

echo composite > /sys/devices/platform/vpss/display2/output

echo 1 > /sys/devices/platform/vpss/display2/enabled/

运行完以上命令后，就可以运行gst来实现视频相关功能，例如运行以下命令就可以采集hdmi视频，压缩存储在当前目录，并从插座J11输出模拟视频

gst-launch v4l2src always-copy=false queue-size=12 ! 'video/x-raw-yuv-strided,format=(fourcc)NV12,width=1920,height=1080,framerate=(fraction)30/1' ! omxbufferalloc numBuffers=12 ! omx\_mdeiscaler name=d d.src\_00 !'video/x-raw-yuv, width=(int)720, height=(int)480' !gstperf!v4l2sink sync=true device=/dev/video3 userpointer=true min-queued-bufs=2 d.src\_01 ! queue ! omx\_h264enc bitrate=5000000 ! filesink location=v4l2cap\_mdei.h264

以上命令中，load-hd-v4l2-firmware.sh实现驱动和固件（视频处理固件）加载功能，可以运行命令

/usr/share/ti/ti-media-controller-utils/load-hd-v4l2-firmware.sh stop来卸载驱动和固件

## GPIO使用

GPIO使用说明参考链接：http://processors.wiki.ti.com/index.php/Linux\_PSP\_GPIO\_Driver\_Guide

在文件/arch/arm/mach-omap2/board-ti8148evm.c中，函数ti814x\_gpio\_init(void)里可以将想要的gpio导出到用户空间使用，目前原理图中gopro\_pwr\_en\_n和gopro\_open\_n两个信号已经导出，分别对应gpio41和gpio40。

注：V10版本gopro\_pwr\_en\_n信号电路有问题，目前默认信号为低电平，gopro电源一直打开，必须改版之后才可以关闭，具体参考故障记录



## GSTreamer使用

### GST编程

请参考链接

https://gstreamer.freedesktop.org/data/doc/gstreamer/head/manual/html/index.html

http://docs.gstreamer.com/display/GstSDK/Home

### gst应用程序编译方法

1. 设置交叉编译环境： source ${EZSDK}/linux-devkit/environment-setup
2. cd <directory where your application is>
3. arm-none-linux-gnueabi-gcc -lz -o my-gst my-gst.c `pkg-config --libs --cflags gstreamer-0.10`

### gst需要做如下修改之后才能使用

SDK包目录下文件ti-ezsdk\_dm814x-evm\_5\_05\_02\_00/component-sources/gst-openmax\_GST\_DM81XX\_00\_07\_00\_00/omx/gstomxbufferalloc.c

--- gst-openmax-dm81xx.orig/src/omx/gstomxbufferalloc.c 2012-11-14 16:15:41.392143697 -0600

+++ gst-openmax-dm81xx/src/omx/gstomxbufferalloc.c 2012-11-14 16:17:40.760145714 -0600

@@ -190,6 +190,7 @@  
 gst\_element\_add\_pad (GST\_ELEMENT (filter), filter->sinkpad);

gst\_element\_add\_pad (GST\_ELEMENT (filter), filter->srcpad);

+ filter->out\_port.portptr = gst\_omxportptr\_new(&filter->out\_port);

filter->silent = FALSE;

filter->out\_port.num\_buffers = 10;

filter->out\_port.always\_copy = FALSE;

### gst 测试用 pipeline实例

* v4l2采集，omx 输出显示hdmi

gst-launch --gst-debug=3 v4l2src always-copy=false queue-size=12 ! 'video/x-raw-yuv-strided, format=(fourcc)NV12, width=1920, height=1080, framerate=(fraction)30/1' ! omxbufferalloc numBuffers=12 ! omx\_scaler ! omx\_ctrl display-mode=OMX\_DC\_MODE\_1080P\_30 ! gstperf ! omx\_videosink sync=false

* V4l2采集，V4l2输出显示hdmi

gst-launch v4l2src always-copy=false queue-size=12  ! 'video/x-raw-yuv-strided,format=(fourcc)NV12,width=1920,height=1080,framerate=(fraction)60/1' !  omxbufferalloc numBuffers=12 ! omx\_scaler ! gstperf  ! v4l2sink sync=false

* LCD显示测试图像

gst-launch -v videotestsrc ! 'video/x-raw-yuv,width=800,height=480' ! omx\_scaler ! 'video/x-raw-yuv,width=800,height=480' ! omx\_ctrl display-mode=OMX\_DC\_MODE\_1080P\_60 display-device=LCD ! gstperf ! omx\_videosink display-device=LCD  sync=false -v --gst-debug=3

* v4l2采集，omx输出显示lcd

gst-launch -v --gst-debug=3 v4l2src always-copy=false queue-size=12 ! 'video/x-raw-yuv-strided,format=(fourcc)NV12, width=1920, height=1080, framerate=(fraction)30/1' ! omxbufferalloc numBuffers=12 ! omx\_scaler ! 'video/x-raw-yuv,width=800,height=480'! omx\_ctrl display-mode=OMX\_DC\_MODE\_1080P\_60 display-device=LCD ! omx\_videosink top=0 left=0 display-device=LCD  sync=false

* LCD播放测试视频

gst-launch -v filesrc location=/usr/share/ti/data/videos/dm816x\_1080p\_demo.264 ! 'video/x-h264' ! h264parse access-unit=true ! omx\_h264dec ! omx\_scaler ! 'video/x-raw-yuv,width=800,height=480' ! omx\_ctrl display-mode=OMX\_DC\_MODE\_1080P\_60 display-device=LCD ! omx\_videosink sync=false top=0 left=0 display-device=LCD

* V4l2采集 ，h.264压缩

gst-launch v4l2src always-copy=false queue-size=12 num-buffers=2000 ! 'video/x-raw-yuv-strided,format=(fourcc)NV12,width=1920,height=1080,framerate=(fraction)30/1' ! omxbufferalloc numBuffers=12 ! omx\_h264enc bitrate=5000000 ! filesink location=v4l2cap.h264

* V4l2采集，V4l2输出显示TV

先设置输出为TV

echo 0 > /sys/devices/platform/vpss/display2/enabled

echo ntsc > /sys/devices/platform/vpss/display2/mode

echo composite > /sys/devices/platform/vpss/display2/output

echo 1 > /sys/devices/platform/vpss/display2/enabled

gst-launch v4l2src always-copy=false queue-size=12 ! 'video/x-raw-yuv-strided,format=(fourcc)NV12,width=1920,height=1080,framerate=(fraction)30/1' !  omxbufferalloc numBuffers=12 ! omx\_scaler !'video/x-raw-yuv, width=(int)720, height=(int)480' !v4l2sink sync=true device=/dev/video3 userpointer=true min-queued-bufs=2

* V4l2采集，同时编码存储和TV显示

gst-launch v4l2src always-copy=false queue-size=12 ! 'video/x-raw-yuv-strided,format=(fourcc)NV12,width=1920,height=1080,framerate=(fraction)30/1' ! omxbufferalloc numBuffers=12 ! omx\_mdeiscaler name=d d.src\_00 !'video/x-raw-yuv, width=(int)720, height=(int)480' !gstperf!v4l2sink sync=true device=/dev/video3 userpointer=true min-queued-bufs=2 d.src\_01 ! queue ! omx\_h264enc bitrate=5000000 ! filesink location=v4l2cap\_mdei.h264

gst-launch -v --gst-debug=3 v4l2src always-copy=false queue-size=12 ! 'video/x-raw-yuv-strided,format=(fourcc)NV12,width=1920,height=1080,framerate=(fraction)30/1' ! omxbufferalloc numBuffers=12 ! tee name=t  !  queue ! omx\_h264enc bitrate=5000000 ! filesink location=v4l2cap\_tee.h264 t.  ! queue ! omx\_scaler !  'video/x-raw-yuv, width=(int)720, height=(int)480' !v4l2sink sync=true device=/dev/video3 userpointer=true min-queued-bufs=2 （该命令还有问题，暂时不能用tee来实现）

* 多文件存储和显示

gst-launch v4l2src always-copy=false queue-size=12 ! 'video/x-raw-yuv-strided,format=(fourcc)NV12,width=1920,height=1080,framerate=(fraction)30/1' ! omxbufferalloc numBuffers=12 ! omx\_mdeiscaler name=d d.src\_00 !'video/x-raw-yuv, width=(int)720, height=(int)480' !gstperf!v4l2sink sync=true device=/dev/video3 userpointer=true min-queued-bufs=2 d.src\_01 ! queue ! omx\_h264enc bitrate=5000000 ! multifilesink location=v4l2cap\_mdei%d.h264 max-file-size=20000000

## 温度电压监控

温度电压监控采用STM32F303芯片来实现，软件工程路径如下：

E:\MCU\_workspace\Blackbox\_Monitor\Voltage-Temp-Monitor-STM32F303R8

运行该程序，串口1就会输出板卡电压和温度信息，用串口调试助手可以实时观看，串口设置为115200,8N1，如下图所示:



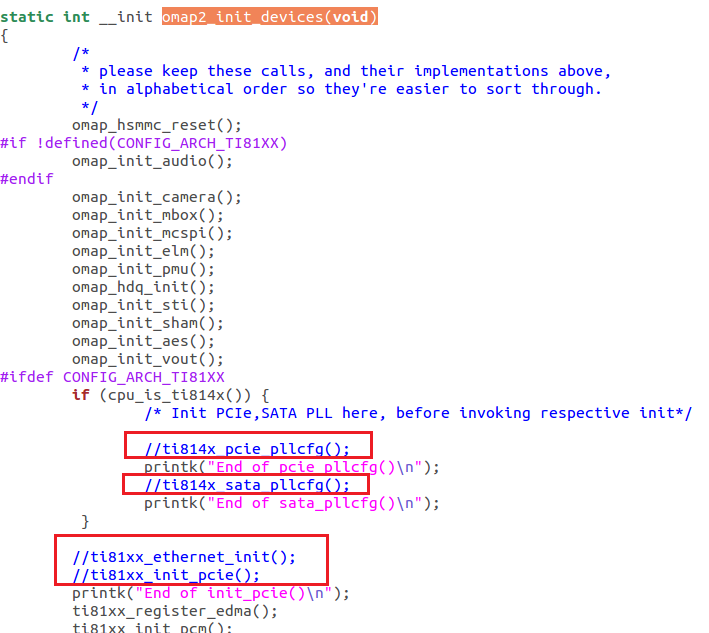
所有电压单位为mV，温度单位为度。其中T\_STM32为STM32芯片温度，Ex\_T为插座J3连接的外置Pt100热敏电阻所处位置的温度。

显示的数据log可以保存为txt格式，用excel直接打开分析数据

# 故障记录

## V10故障记录

1. SATA时钟源需要使用外部高精度差分时钟，如果使用内部20Mhz时钟，在3Gbps速率下容易出错。改版加入外部时钟后，需要更改内核文件来启用外部时钟。/arch/arm/mach-omap2/device.c 函数omap2\_init\_devices(void)里，启用如下几个函数：



1. GPIO1[9]管脚（Gopro\_pwr\_en\_n信号）电压为1.8V，如果要使用gopro电源关闭功能，需要更换gpio控制管脚，或者外加mos管
2. USB\_ID信号拉低，否则USB没法使用
3. SD\_CD信号（card detect）要拉低，可以更改BOM，把D25换成0603电阻；或者直接去掉
4. R204去掉，上拉到5V了，应该上拉到3.3V

# 其他参考信息

## 不同配置的DDR映射

http://processors.wiki.ti.com/index.php/EZSDK\_Memory\_Map#Changing\_Memory\_Map\_For\_512MB\_Board

## 手动修改bootargs文件boot.scr

boot.scr is created by a boot script file. If you have one of these you can use mkimage to create your boot.scr file. If you want to base your boot script off a boot.scr and don't have the original source, you should be able to create the file from boot.scr with:

#NOTE: the path to your boot.scr might be different (ie /media/boot)

dd bs=1 skip=72 if=/mnt/boot.scr of=/tmp/boot.script

#now backup the original boot.scr:

cp /mnt/boot.scr /mnt/boot.scr.bak

After editing boot.script to have the contents you want, you can generate your boot.scr with:

mkimage -A arm -T script -C none -n "My Boot.scr" -d /tmp/boot.script /mnt/boot.scr

## 添加外置视频解码芯片驱动

http://processors.wiki.ti.com/index.php/DM81xx\_AM38xx\_Adding\_External\_Decoders\_to\_V4L2\_Capture\_Driver

## Gstreamer 更新包

https://e2e.ti.com/support/dsp/davinci\_digital\_media\_processors/f/717/t/327198

## 制作启动SD卡

使用虚拟机内的shell文件/media/DM8148/mksdboot-tl/mksdboot-tl.sh

1. 虚拟机内的目录/media/DM8148/mksdboot-tl/已经包含可以使用的镜像文件和文件系统，可以直接跳过步骤2/3/4
2. 将文件系统压缩包放入/media/DM8148/mksdboot-tl/filesystem
3. 将uboot镜像文件MLO、u-boot.bin、u-boot.min.spi、uImage放入 /media/DM8148/mksdboot-tl/boot
4. 将模块驱动放入/media/DM8148/mksdboot-tl/modules（如果没有，或者驱动已经包含在文件系统内，可以忽略该步骤）
5. 将sd卡插入计算机，ubantu自动识别sd卡
6. 确认sd卡节点名，执行命令sudo fdisk -l
7. 确认需要制作的节点为sdd



1. 执行脚本文件sudo ./mksdboot-tl.sh --device /dev/sdd，按回车执行