

AM57x 平台调试串口修改说明

目 录

1 修改 U-boot 源码.....	3
2 U-Boot 编译.....	5
3 启动运行.....	6
更多帮助.....	9

创龙

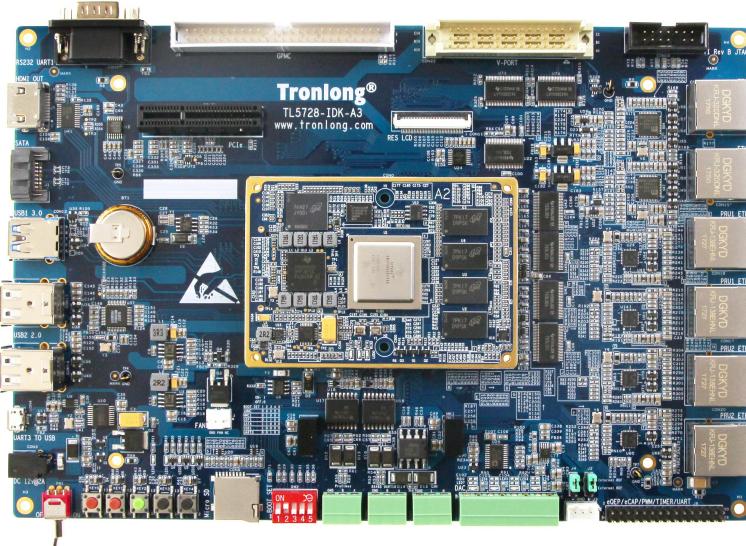
AM57x 平台开发板的默认调试串口为 UART3, 本例程以调试串口修改为 UART1 为例, 演示修改 AM57x 平台调试串口的方法。(本文是基于创龙 TL5728-IDK 开发套件进行测试)

平台简介:

AM5728 是 TI Sitara 系列高性能 SOC, 得益于异构多核处理架构, CPU 内集成了多核 DSP、多核 PRU、IVA-HD、GPU 等协处理单元, 通过硬件加速的方式极大增强 CPU 的数据、多媒体处理能力, 可满足工业协议支持、大数据计算、实时控制等应用需求, 同时采用先进的 28 纳米生产工艺, 极大降低处理器的功耗, 能耗比更加突出。

TL5728-IDK 是一款广州创龙基于 SOM-TL5728 核心板设计的开发板, 底板采用沉金无铅工艺的 4 层板设计, 它为用户提供了 SOM-TL5728 核心板的测试平台, 用于快速评估 SOM-TL5728 核心板的整体性能。

不仅提供丰富的 AM5728 入门教程和 Demo 程序, 还提供 DSP+ARM 多核通信开发教程, 全面的技术支持, 协助用户进行底板设计和调试以及 DSP+ARM 软件开发。



- 基于 TI AM5728 浮点双 DSP C66x + 双 ARM Cortex-A15 工业控制及高性能音视频处理器;
- 多核异构 CPU, 集成双核 Cortex-A15、双核 C66x 浮点 DSP、双核 PRU-ICSS、两个双核 Cortex-M4 IPU、双核 GPU 等处理单元, 支持 OpenCL、OpenMP、IPC 多核开发;
- 强劲的视频编解码能力, 支持 1 路 1080P60 或 2 路 720P60 或 4 路 720P30 视频硬件编解码, 支持 H.265 视频软解码;
- 高性能 GPU, 双核 SGX544 3D 加速器和 GC320 2D 图形加速引擎, 支持 OpenGL ES2.0;
- 支持 1 路 1080P60 HDMI 1.4a 输出或 1 路 LCD 输出;
- 开发板引出 V-PORT 视频输入接口, 可以灵活接入视频输入模块;

创龙

- 双核 PRU-ICSS 工业实时控制子系统，支持 EtherCAT、EtherNet/IP、PROFIBUS 等工业协议；
- 支持 2 路千兆网，用于网络调试、数据传输、工业以太网主站；
- 支持 4 路 PRU 百兆网，用于网络调试、数据传输、工业以太网从站；
- 外设接口丰富，GPMC、USB 2.0、UART、SPI、QSPI、I2C、DCAN 等工业控制总线和接口，支持高速接口 PCIe Gen2、USB 3.0、SATA 2.0；

1 修改 U-boot 源码

将光盘资料“U-Boot\U-Boot-2016.05\src”目录下的 U-boot 内核源码解压到 Ubuntu 的 AM57xx 工作目录。打开 U-boot 源码“board/ti/am57xx”目录下的 Kconfig 文件，按照下图方法将文件中的参数“3”修改为“1”，表示将 MLO 启动过程中的打印信息从 UART1 输出。

```
tronlong@tronlong-virtual-machine:~/AM57xx/U-Boot/U-Boot-2016.05/board/ti/am57xx$ pwd  
/home/tronlong/AM57xx/U-Boot/U-Boot-2016.05/board/ti/am57xx  
tronlong@tronlong-virtual-machine:~/AM57xx/U-Boot/U-Boot-2016.05/board/ti/am57xx$ ls  
board.c  Kconfig  MAINTAINERS  Makefile  mux_data.h  
tronlong@tronlong-virtual-machine:~/AM57xx/U-Boot/U-Boot-2016.05/board/ti/am57xx$ vi Kconfig
```

图 1

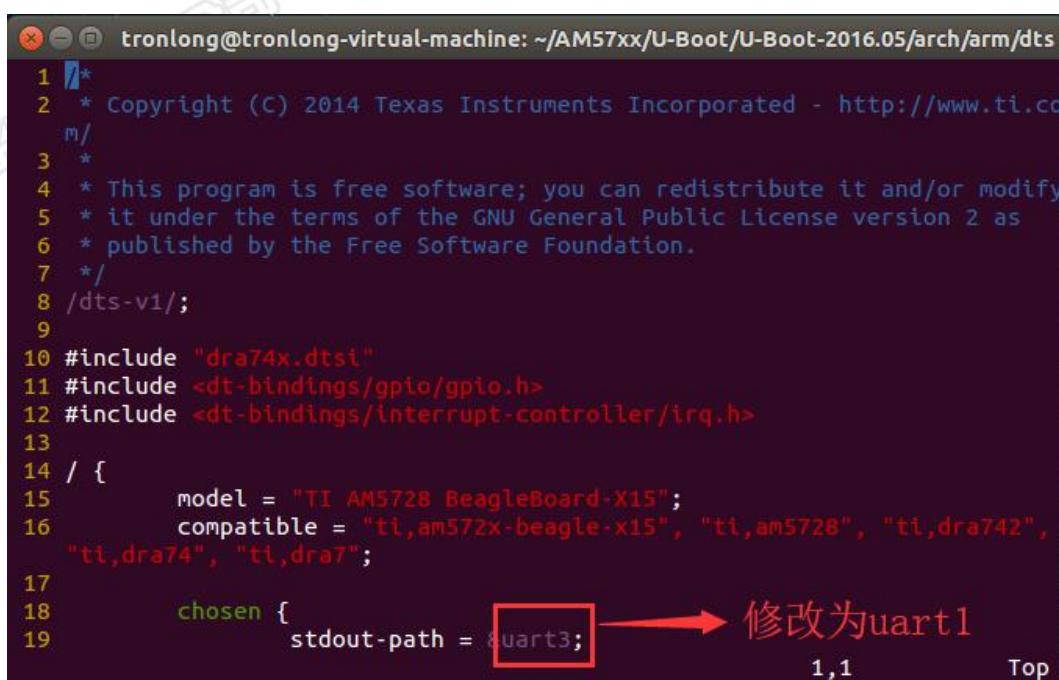
```
tronlong@tronlong-virtual-machine: ~/AM57xx/U-Boot/U-Boot-2016.05/board/ti/am57xx  
1 if TARGET_AM57XX_EVM  
2  
3 config SYS_BOARD  
4     default "am57xx"  
5  
6 config SYS_VENDOR  
7     default "ti"  
8  
9 config SYS_CONFIG_NAME  
10    default "am57xx_evm"  
11  
12 config AM572X_EVM  
13     bool "Select am572x-evm board"  
14  
15 config AM572X_IDK  
16     bool "Select am572x-idk board"  
17  
18 config CONS_INDEX  
19     int "UART used for console"  
20     range 1 6  
21     default 3 → 修改为参数“1”  
22     help  
23         The AM57x (and DRA7xx) SoC has a total of 6 UARTs available to it.  
1,1           Top
```

图 2

打开 U-boot 源码下的“arch/arm/dts/am57xx-beagle-x15.dts”文件，按照下图方法将文件中的参数“uart3”修改为“uart1”，表示将 u-boot.img 运行过程中的打印信息从 UART1 输出。

```
tronlong@tronlong-virtual-machine:~/AM57xx/U-Boot/U-Boot-2016.05/arch/arm/dts$ pwd  
/home/tronlong/AM57xx/U-Boot/U-Boot-2016.05/arch/arm/dts  
tronlong@tronlong-virtual-machine:~/AM57xx/U-Boot/U-Boot-2016.05/arch/arm/dts$ vi am57xx-  
beagle-x15.dts
```

图 3



```
tronlong@tronlong-virtual-machine: ~/AM57xx/U-Boot/U-Boot-2016.05/arch/arm/dts  
1 /*  
2  * Copyright (C) 2014 Texas Instruments Incorporated - http://www.ti.co  
3  */  
4  * This program is free software; you can redistribute it and/or modify  
5  * it under the terms of the GNU General Public License version 2 as  
6  * published by the Free Software Foundation.  
7  */  
8 /dts-v1/;  
9  
10 #include "dra74x.dtsci"  
11 #include <dt-bindings/gpio/gpio.h>  
12 #include <dt-bindings/interrupt-controller/irq.h>  
13  
14 / {  
15     model = "TI AM5728 BeagleBoard-X15";  
16     compatible = "ti,am572x-beagle-x15", "ti,am5728", "ti,dra742",  
"ti,dra74", "ti,dra7";  
17  
18     chosen {  
19         stdout-path = &uart3; → 修改为uart1
```

图 4

打开 U-boot 源码下的“include/configs/am57xx_evm.h”文件，按照下图方法将文件中环境变量 console 的参数“ttyO2”修改为“ttyO0”，表示将内核启动阶段的打印信息从 UART1 输出。

```
tronlong@tronlong-virtual-machine: ~/AM57xx/U-Boot/U-Boot-2016.05/include/config.h
13 #define __CONFIG_AM57XX_EVM_H
14
15 #define CONFIG_AM57XX
16
17 #define CONFIG_IODELAY recalibration
18
19 #define CONFIG_BOARD_EARLY_INIT_F
20
21 #define CONFIG_NR_DRAM_BANKS 2
22 #define CONFIG_MAX_RAM_BANK_SIZE (2 << 30) /* 2GB */
23
24 #define CONFIG_ENV_SIZE (64 << 10)
25 #define CONFIG_ENV_IS_IN_FAT "mmc"
26 #define FAT_ENV_INTERFACE "1:1"
27 #define FAT_ENV_DEVICE_AND_PART "uboot.env"
28 #define FAT_ENV_FILE "tty02"
29
30 #define CONSOLEDEV UART1_BASE /* Base EVM h
31 #define CONFIG_SYS_NS16550_COM1 as UART0 */
32 #define CONFIG_SYS_NS16550_COM2 UART2_BASE /* UART2 */
13,1 16%
```

图 5

2 U-Boot 编译

修改完成后，参照《U-Boot 编译方法》文档依次执行 U-Boot 清理、编译指令。将在 U-Boot 源码 am572x_evm 目录下新编译生成的 MLO 和 u-boot.img 文件，拷贝到 SD 系统启动卡的 boot 目录下。

```
tronlong@tronlong-virtual-machine:~/AM57xx/U-Boot/U-Boot-2016.05$ make CROSS_COMPILE=arm-linux-gnueabihf- ARCH=arm distclean
tronlong@tronlong-virtual-machine:~/AM57xx/U-Boot/U-Boot-2016.05$
```

图 6

```
tronlong@tronlong-virtual-machine:~/AM57xx/U-Boot/U-Boot-2016.05$ make CROSS_COMPILE=arm-linux-gnueabihf- ARCH=arm O=am572x_evm am57xx_evm_defconfig all
make[1]: Entering directory `/home/tronlong/AM57xx/U-Boot/U-Boot-2016.05/am572x_evm'
  GEN      ./Makefile
 HOSTCC  scripts/kconfig/conf.o
 HOSTCC  scripts/kconfig/zconf.tab.o
 HOSTLD  scripts/kconfig/conf

#
# configuration written to .config
#
  GEN      ./Makefile
scripts/kconfig/conf --silentoldconfig Kconfig
```

图 7

```
OBJCOPY spl/u-boot-spl-nodtb.bin
COPY     spl/u-boot-spl.bin
MKIMAGE MLO
CFG      spl/u-boot-spl.cfg
COPY     u-boot.dtb
MKIMAGE u-boot-dtb.img
make[1]: Leaving directory `/home/tronlong/AM57xx/U-Boot/U-Boot-2016.05/am572x_evm'
tronlong@tronlong-virtual-machine:~/AM57xx/U-Boot/U-Boot-2016.05$ ls am572x_evm/
arch    dts      MLO          test      u-boot-dtb.bin   u-boot.srec
board   examples net          tools      u-boot-dtb.img  u-boot.sym
cmd     fs       scripts     u-boot    u-boot.img
common  include  source      u-boot.bin u-boot.lds
disk    lib      spl         u-boot.cfg u-boot.map
drivers Makefile System.map u-boot.dtb u-boot-nodtb.bin
tronlong@tronlong-virtual-machine:~/AM57xx/U-Boot/U-Boot-2016.05$
```

图 8

3 启动运行

分别使用 USB 转 RS232 线和 Micro USB 线将 UART3、UART1 串口连接到 PC 端 USB，拨码开关选择从 SD 卡启动。开发板上电，快速点击任意键进入 U-Boot 命令行，执行如下指令清除保存在 eMMC 的环境变量：

```
Target# env default -f -a
Target# saveenv
Target# reset
```

```
=> env default -f -a  
## Resetting to default environment  
=> saveenv  
Saving Environment to FAT...  
writing uboot.env  
FAT: Misaligned buffer address (bef1ff78)  
done  
=> reset■
```

图 9

U-Boot 引导内核并进入文件系统后，看到串口调试终端 UART1 有启动过程的打印信息，此时 UART3 只打印登陆 root 用户的信息，如下图所示 com22 对应 UART1 串口，com41 对应 UART3 串口。

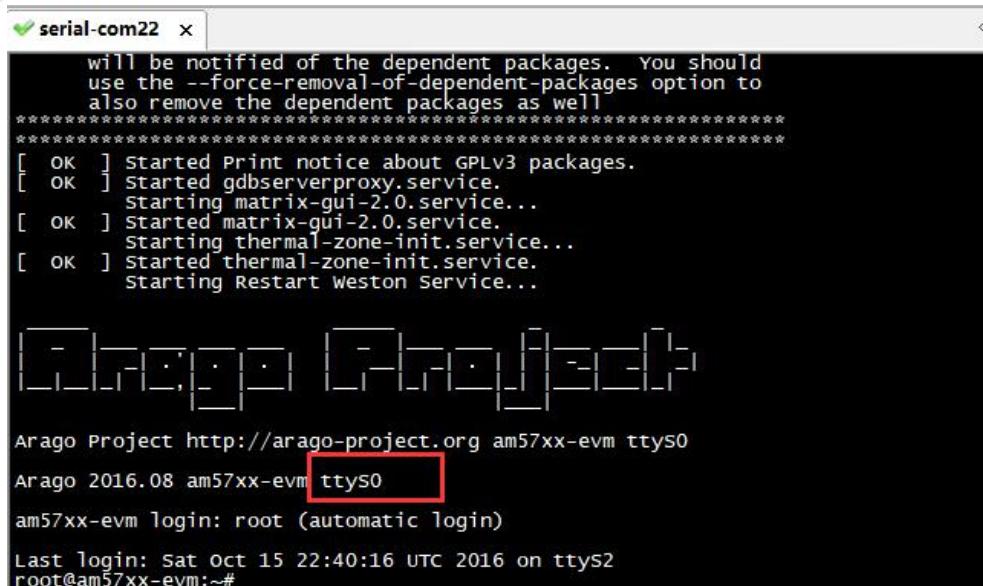


图 10

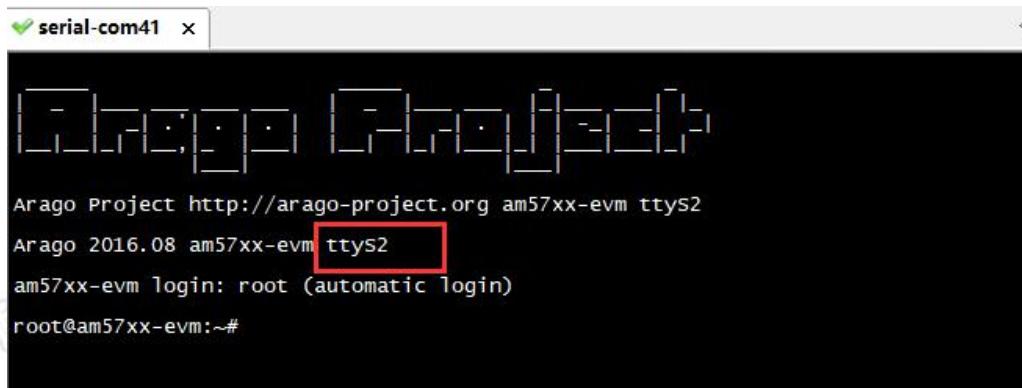
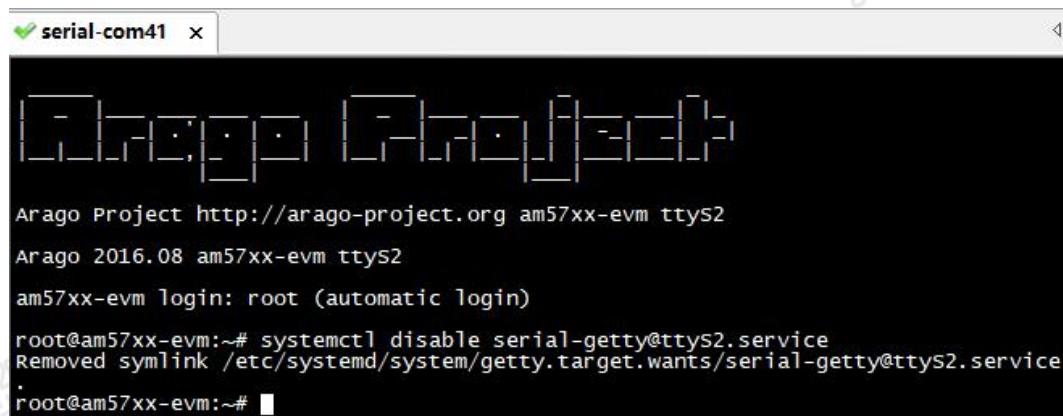


图 11

在 UART3 对应的 com41 调试口输入如下指令关闭 UART3 打印终端信息：

Target# systemctl disable serial-getty@ttyS2.service



```
serial-com41 x
[...]
Arago Project http://arago-project.org am57xx-evm ttys2
Arago 2016.08 am57xx-evm ttys2
am57xx-evm login: root (automatic login)
root@am57xx-evm:~# systemctl disable serial-getty@ttyS2.service
Removed symlink /etc/systemd/system/getty.target.wants/serial-getty@ttyS2.service
.
root@am57xx-evm:~#
```

图 12

重新启动系统后，可看到 UART1 有启动过程打印信息，而 UART3 再无打印信息。

更多帮助

销售邮箱: sales@tronlong.com

技术邮箱: support@tronlong.com

创龙总机: 020-8998-6280

技术热线: 020-3893-9734

创龙官网: www.tronlong.com

技术论坛: www.51ele.net

线上商城: <https://tronlong.taobao.com>