AM335x 平台--在引导 SPL、Uboot、Kernel 期间 修改调试&打印串口

大家好，这篇文章主要基于 AM335x 的 linux SDK，讲述如何修改 UART 接口去打印调试信息。AM335x 一共有六个串口，分别是 UART0、UART1、UART2、UART3、UART4、UART5，六个串口一般能满足大部分的项目需求，如果还嫌少，可以使用带 PRU 的 AM335x 芯片，PRU 可以模拟四路串口，也即，板上可以跑 10 路串口。

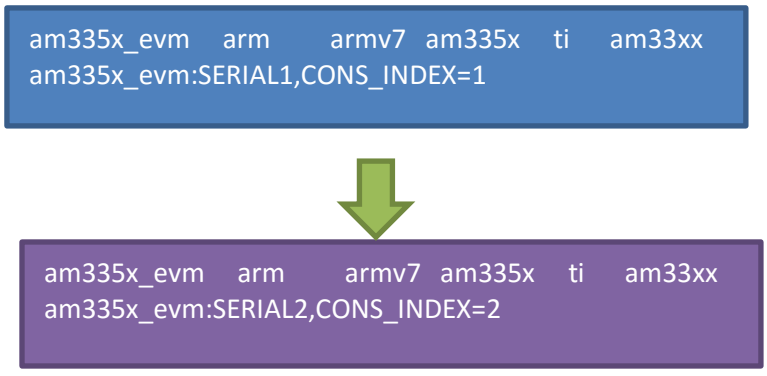
写这篇文章，主要是想帮助客户更加清晰的明白修改 UART 的流程。

由于 TI 发布的 SDK，大部分都是默认 UART0 作为调试串口。但在一些项目中，由于引脚的复用或冲突，很多客户希望灵活的改变 UART 口去打印调试信息，也即引导 SPL、Uboot、Kernel 这三个过程的信息。但是由于 Linux SDK 的 Uboot 过程较为复杂，文件较多，宏定义不明显等等原因，再加上 SDK 各个版本差异性，使得如此简单的一个修改串口的过程也让客户难以下手或者修改不成功。所以以下通过本人的测试和经验，对 processor SDK 3.0 和 EZSDK6.0 修改串口做出总结和对比。希望能够帮助大家。

EZSDK6.0 修改默认 UART 比较简单，processor SDK 3.0 修改起来可能复杂一点，那我先从简单出发，先介绍 EZSDK6.0 是如何修改，再说明 processor SDK 3.0 是如何修改。

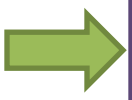
一. EZSDK6.0 修改 SPL 和 U-boot 的串口打印的步骤

1. uboot 在 make 时，makefile 会调用 mkconfig，mkconfig 会根据 boards.cfg 生成 config.h，config.h 包含相关的宏定义，宏定义决定了编译的方向。所以首先需要修改的是 boards.cfg，修改板卡类型对应的行，比如所需编译的板卡是 am335x_evm，将 uart0 修改为 uart1，操作如下：



生成的对应板卡的 config 区别：

```
/* Automatically generated - do not edit */
#define CONFIG_SERIAL1 1
#define CONFIG_CONS_INDEX 1
#define CONFIG_SYS_ARCH "arm"
#define CONFIG_SYS_CPU "armv7"
#define CONFIG_SYS_BOARD "am335x"
#define CONFIG_SYS_VENDOR "ti"
#define CONFIG_SYS_SOC "am33xx"
#define CONFIG_BOARDDIR board/ti/am335x
#include <config_cmd_defaults.h>
#include <config_defaults.h>
#include <configs/am335x_evm.h>
#include <asm/config.h>
#include <config_fallbacks.h>
```



```
/* Automatically generated - do not edit */
#define CONFIG_SERIAL2 1
#define CONFIG_CONS_INDEX 2
#define CONFIG_SYS_ARCH "arm"
#define CONFIG_SYS_CPU "armv7"
#define CONFIG_SYS_BOARD "am335x"
#define CONFIG_SYS_VENDOR "ti"
#define CONFIG_SYS_SOC "am33xx"
#define CONFIG_BOARDDIR board/ti/am335x
#include <config_cmd_defaults.h>
#include <config_defaults.h>
#include <configs/am335x_evm.h>
#include <asm/config.h>
#include <config_fallbacks.h>
```

2. 根据板卡的 pin mux 分配功能引脚 I/O，修改 mux.c (board\ti\am335x)，主要修改结构体 module_pin_mux 里面的 OFFSET(x)和 mode(x)，如 UART2，除了这里表示的引脚，你还可以选择其他引脚。

```
static struct module_pin_mux uart2_pin_mux[] = {
        {OFFSET(spi0_sclk), (MODE(1) | PULLUP_EN | RXACTIVE)},      /* UART2_RXD */
        {OFFSET(spi0_d0), (MODE(1) | PULLUDEN)},                    /* UART2_TXD */
        {-1},
};
```

可以下载 TI 的 PIN MUX 工具，它非常方便帮你生成相关引脚的程序配置，还可以在画原理图之前，非常可视化的帮你规划引脚分配问题，下载和教程的 LINK 如下：

http://processors.wiki.ti.com/index.php/Pin_Mux_Utility_for_ARM_MPU_Processors

3.修改 Hardware.h (arch\arm\include\asm\arch-am33xx)中的串口基地址，如将 UART0 改为 UART1，操作如下：

#define DEFAULT_UART_BASE UART1_BASE

在 Hardware_am33xx.h (arch\arm\include\asm\arch-am33xx) 增加宏定义：

#define UART1_BASE 0x481AA000

4. 如果想 linux image 也是用这个串口显示的话。修改 Am335x_evm.h (include\configs)下的引导变量**"console=ttyO0,115200n8\0" **，比如将 UART0 改为 UART1，操作如下：

**"console=ttyO1,115200n8\0" **

5.完成前面三部后，Make 编译成功后，并在板子上进行 bootload，可以发现：通过你配置的 UART 口，可以在终端上可以看到 SPL、Uboot、kernel 引导时的信息，但是 kernel 结束后，进入文件系统，这时文件系统还是默认 UART0 作为调试串口，所以若想在文件系统也采用直自己定义的串口进行调试打印，修改如下：

在文件系统中，打开文件/etc/inittab ，可以看到以下信息：

S:2345:respawn:/sbin/getty 115200 ttyO0

修改 ttyO0 即可，如想要 UART1 作为串口打印，将 ttyO0 改为 ttyO1，当然 115200 波特率也可以改。完成这个步骤后，将文件系统重新导入到板子上，运行，即可成功。

接下来分析 Processor SDK 3.0 的修改流程

二． Processor SDK 3.0 修改 SPL 串口打印的步骤：

1. Make uboot 时，makefile 文件会根据 Kconfig (board\ti\am335x)生成 autoconf.h 文件，该文件主要包含一些宏定义，查看发现其中一句：#define CONFIG_CONS_INDEX 1 ——其表示 UART0 会被编译成为 SPL 串口打印的设备。
   所以要修改串口设备，首先需要修改 Kconfig，可以看到以下蓝框的内容。修改 default 的值，该值决定 spl 初始化哪一个串口的引脚。 UART0-UART5 对应 default1-6，比如要使用 UART1 作为调试信息的串口打印，操作为：将 default 1 为 default 2

```
config CONS_INDEX
        int "UART used for console"
        range 1 6
        default 1
        help
          The AM335x SoC has a total of 6 UARTs (UART0 to UART5 as referenced
          in documentation, etc) available to it.  Depending on your specific
          board you may want something other than UART0 as for example the IDK
          uses UART3 so enter 4 here.
```

2. 在文件中 Hardware.h (arch\arm\include\asm\arch-am33xx)修改 DEFAULT_UART_BASE

   /* UART */

   #define DEFAULT_UART_BASE UART0_BASE

   在数据手册的 memory map 中可以查询

UART0	0x44E0_9000	0x44E0_9FFF	4KB	UART Registers
	0x44E0_A000	0x44E0_AFFF	4KB	Reserved
UART1	0x4802_2000	0x4802_2FFF	4KB	UART1 Registers
	0x4802_3000	0x4802_3FFF	4KB	Reserved
UART2	0x4802_4000	0x4802_4FFF	4KB	UART2 Registers
	0x4802_5000	0x4802_5FFF	4KB	Reserved
UART3	0x481A_6000	0x481A_6FFF	4KB	UART3 Registers
	0x481A_7000	0x481A_7FFF	4KB	Reserved
UART4	0x481A_8000	0x481A_8FFF	4KB	UART4 Registers
	0x481A_9000	0x481A_9FFF	4KB	Reserved
UART5	0x481A_A000	0x481A_AFFF	4KB	UART5 Registers
	0x481A_B000	0x481A_BFFF	4KB	Reserved

   举个例子：要使用 UART5 作为串口打印，则#define DEFAULT_UART_BASE UART5_BASE

   同时在 Hardware_am33xx.h (arch\arm\include\asm\arch-am33xx)

   增加 #define UART5_BASE 0x481AA000

3. 在文件 Board.c (board\ti\am335x)里的以下函数，修改 eserial1_device 的值，eserial1_device 代表初始化 UART0，eserial2_device 代表初始化 UART1…..以此类推。若板子是 ICEv2 则修改 eserial4_device。

```
struct serial_device *default_serial_console(void)
{
        if (board_is_icev2())
                return &eserial4_device;
        else
                return &eserial1_device;
}
```

上面这三个步骤若成功了，SPL 的调试信息会通过相对应被初始化的串口进行打印，但是 Uboot 还是从默认 UART，如 UART0 来打印信息。这是为什么呢？

因为新版的 SDK，也即 Processor SDK 3.0 和旧版的 EZSDK6.0 有很大区别，Processor SDK 3.0 在 Uboot 期间是使用 device tree 来初始化芯片外设的。所以我们还需要修改 Uboot 的默认的 UART 的配置。操作如下。

Processor SDK 3.0 修改 U-boot 的串口打印的步骤：

由于 processor SDK 的 uboot 是使用 device tree 来引导和初始化外设的。device tree 的文件是：Am335x-evm.dts (arch\arm\dts)。

1. 找到在 Am335x-evm.dts 文件里如下框框的部分，若要将打印串口从 uart0 修改为 uart1，在文件以下位置进行修改，操作如下：

```
chosen {
        stdout-path = &uart0;
        tick-timer = &timer2;
        };
```

→

```
chosen {
        stdout-path = &uart1;
        tick-timer = &timer2;
        };
```

由于 uart0 和 uart1 本 SDK 中已经在 Am335x-evm.dts 中有定义了，对该节点已经描述清楚，但是若要修改其他 uart 的，则请看第二个步骤，否，请跳过第二步骤。

2. 若想支持其他 uart，则需要在 **Am335x-evm.dts 文件里**增加 uart 的节点信息，比如增加 uart2，步骤如下：添加下面的语句，可以参考文件中 uart0 表示的方式，模范即可。

```
&uart2 {
    pinctrl-names = "default";
    pinctrl-0 = <&uart2_pins>;
    status = "okay";
};
```

```
Uart2_pins: pinmux_uart2_pins {
        pinctrl-single,pins = <
                0x950 (PIN_INPUT_PULLUP | MUX_MODE1)        /* uart0_rxd.uart2_rxd */
                0x954 (PIN_OUTPUT_PULLDOWN | MUX_MODE1) /* uart0_txd.uart2_txd */
                >;
        };
```

该 RX 和 TX 的引脚可以更改其他 IO，具体内容请看引脚分配或根据工具 PIN MUX 进行配置。

3. 如果想引导 kernel 的时候也打算用更改的串口显示调试信息的话。修改 Am335x_evm.h (include\configs)下的引导变量： "ttyO0,115200n8\0"，如要修改串口 UART1，则在文件以下位置修改为：

```
"init_console=" \
        "if test $board_name = A335_ICE; then "\
            "setenv console ttyO3,115200n8;" \
        "else " \
            "setenv console ttyO0,115200n8;" \
                "fi;\0" \
```

```
"init_console=" \
        "if test $board_name = A335_ICE; then "\
            "setenv console ttyO3,115200n8;" \
        "else " \
            "setenv console ttyO1,115200n8;" \
                "fi;\0" \
```