ARM 嵌入式 Linux 设备树
简介及应用示例

Revision History

Date	Doc. Rev.	Linux BSP Version	Changes
2016-01-16	Rev. 0.1	V2.5	初始版本

1). 简介

设备树（Device Tree）是一种用来描述系统硬件的数据结构，一些硬件设备设计机制就是可被系统发现的（如 PCI Express 或者 USB 总线），而有一些则不是（尤其是内存映射外设）。对于后一种情况，不同于 X86 架构系统采用 BIOS 和操作系统沟通硬件拓扑信息，ARM Linux 通常情况是将硬件设备描述硬编码到系统内核（Linux Kernel）中，但由于 ARM 嵌入式设备的多样和离散性，即便如此也不能保证覆盖到所有设备，而且长久以来给 ARM Linux 内核代码维护造成了很大负担；基于这种情况，设备树的概念就被提出，将 ARM SOC 和板卡硬件平台描述信息从内核独立出来成为设备树文件，通过 bootloader 传递给内核来识别当前平台设备并加载相应的资源和驱动，这样就把 ARM 嵌入式 Linux 内核统一起来，更好的利于内核维护，而对于广泛的 ARM 嵌入式设备系统维护和迁移也更方便和有效率。

设备树机制从 Linux 内核 3.2 版本左右开始采用，其不仅可以定义 ARM SoC 内部内存映射外设，还可以定义整个板卡，下面就以 Toradex Colibri VF61 计算机模块搭配 Colibri Eva Board 为例来展示设备树的具体应用，另外关于设备树的更深入介绍，请参考这里。

2). 设备树文件说明

Toradex ARM 计算机模块工业产品级 Embedded Linux 源代码下载及编译指南请见这里，其中设备树文件位于 Kernel 源代码 arch/arm/boot/dts/ 目录下。

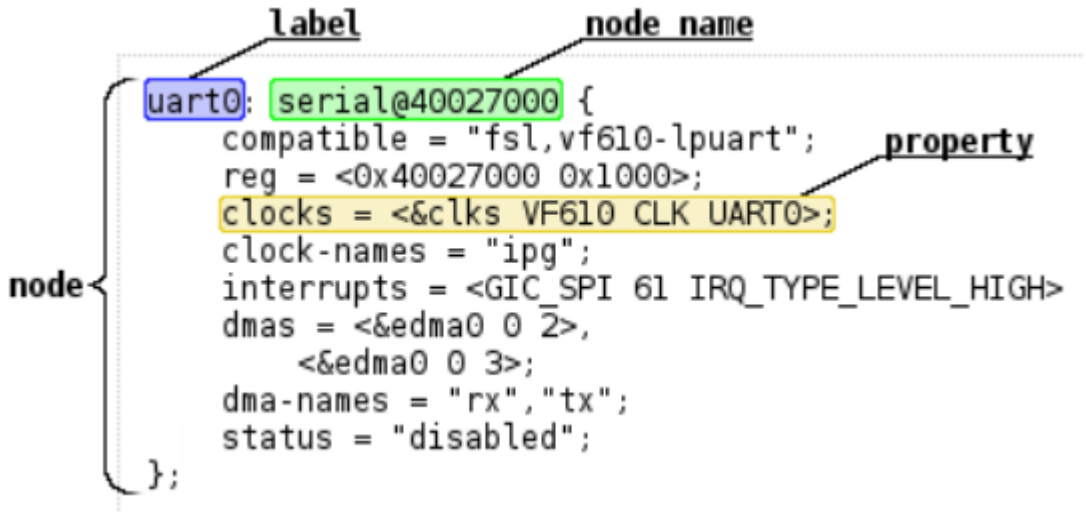
产品系列	SoC	Image 版本	SoC 级别	模块级别	Eva Board 级别
Colibri VF50	NXP/Freescale Vybrid	V2.3Beta5 onwards	vf500.dtsi	vf500-colibri.dtsi	vf500-colibri-eval-v3.dts
Colibri VF61	NXP/Freescale Vybrid	V2.3Beta5 onwards	vf610.dtsi	vf610-colibri.dtsi	vf610-colibri-eval-v3.dts
Colibri iMX6DL/S	NXP/Freescale i.MX6	all compatible	imx6q.dtsi	imx6qdl-colibri.dtsi	imx6dl-colibri-eval-v3.dts
Apalis iMX6Q/D	NXP/Freescale i.MX6	V2.3Beta3 onwards	imx6q.dtsi	imx6qdl-apalis.dtsi	imx6q-apalis-eval.dts

设备树通常由多级别的多个设备树文件构成，一个设备树文件（dts 和 dtsi）可以包含另外一个可包含设备树文件（dtsi），如一个板卡级设备树文件（dts）一般会包含其所使用的 SoC 级别的设备树文件（dtsi）。如上图所示，为了支持 Toradex 产品，定义了三个级别的设备树文件：载板级别，模块基本以及 SoC 级别，这些区别也体现在了设备树文件的命名上面。

载板级别的设备树文件（如 vf610-colibri-eval-v3.dts）定义自 Colibri Eva Board 载板，但基于 Colibri 模块的标准定义，同样也兼容于其他 Colibri 载板（如 Iris 载板）；不过如果用户针对自己应用定制了载板，则需要对应定制化设备树文件以便使能非默认定义功能设备（如第二个网口）或者关闭一些无用的设备。

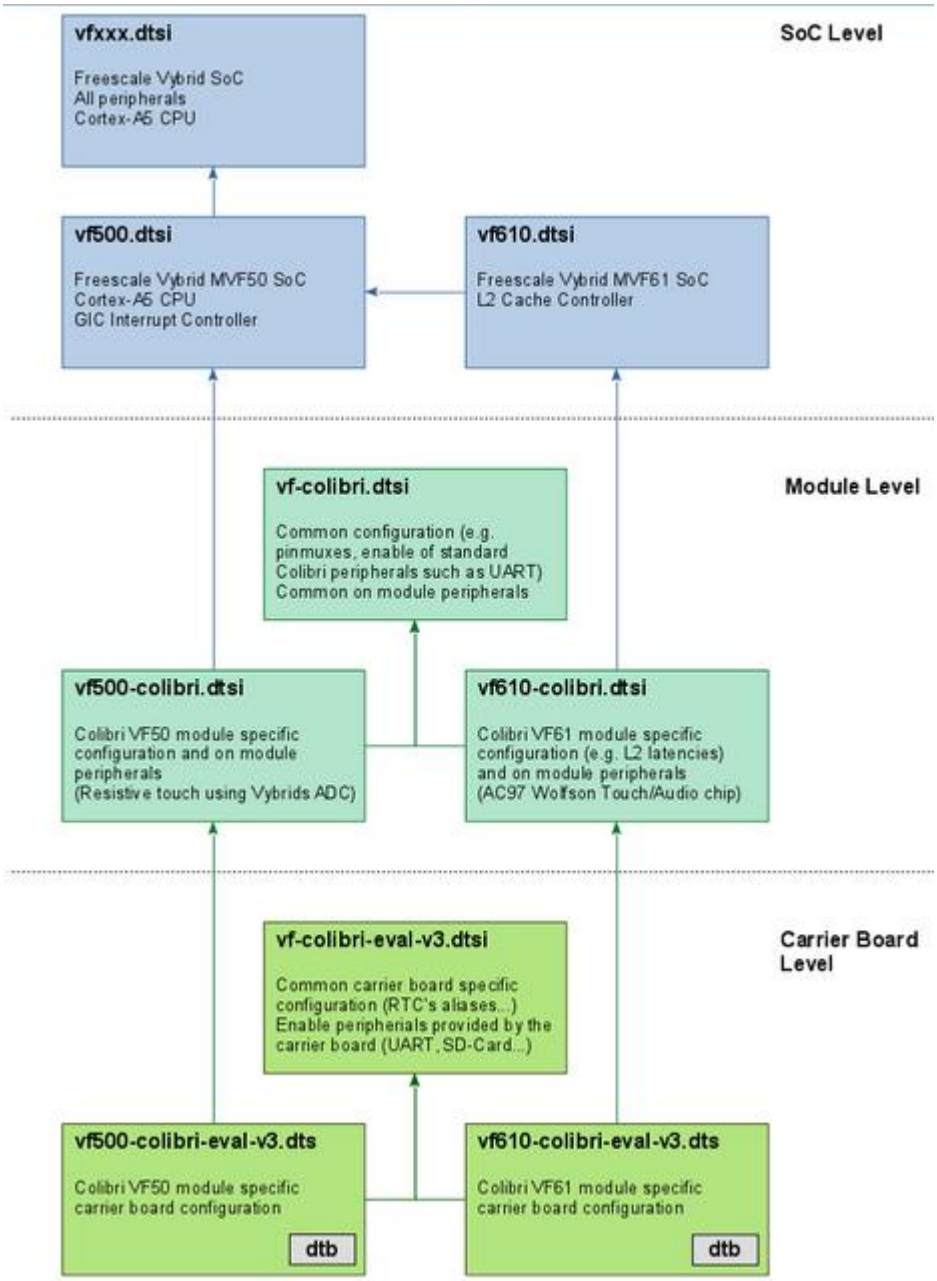
设备树文件（dts）最后要被编译成设备树二进制文件（dtb）以供 Linux 内核启动加载所使用，所需的编译器也都集成在 Linux 源文件里面可以直接调用，从后面的示例可以看到具体的编译方法。

设备树文件的基本单元是 node，一个设备树文件只能有一个 root node （/），其他 node 按照 parent/child node 以树状结构分布，每个 node 里面包含一些 property/value 来描述该 node 特性，如下面是一个 UART 设备的描述；另外低级别设备树文件的定义可以在更高级别的设备树文件中重新定义或者更改，最后生成的二进制文件以最后一次定义为准，因此我们定制化设备树文件时候通常只定制修改最高级别的载板级设备树文件即可；更详细的关于设备树文件语法的说明请见这里。



3). 定制设备树文件

本文以 Colibri VF61 计算机模块和 Eva board 载板为例，定制设备树文件以使能 GPIO 和 CAN bus。Colibri Vybird 系列产品设备树文件的架构如下图所示：



a). 创建新的载板级别设备树文件，这里为了方便直接复制 vf610-colibri-eval-v3.dts

$ cp arch/arm/boot/dts/vf610-colibri-eval-v3.dts arch/arm/boot/dts/vf610-colibri-my-carrier.dts

---

b). 编辑设备树文件 vf610-colibri-my-carrier.dts，将默认配置为 PWM 设备管脚配置为 GPIO

$ vi vf610-colibri-my-carrier.dts

//添加下面内容于设备树文件中

```
&pwm0 {
        status = "disabled";
};
&pwm1 {
        status = "disabled";
};
&iomuxc {
        vf610-colibri {
                pinctrl_additionalgpio: additionalgpios {
                        fsl,pins = <
                                VF610_PAD_PTB0__GPIO_22     0x22ed
                                VF610_PAD_PTB1__GPIO_23     0x22ed
                                VF610_PAD_PTB8__GPIO_30     0x22ed
                                VF610_PAD_PTB9__GPIO_31     0x22ed
                        >;
                };
        };
};
```

c). 配置编译环境并编译新的设备树文件

./ 安装交叉编译 Tool Chain，请从这里下载

$ tar xvf gcc-linaro-4.9-2014.11-x86_64_arm-linux-gnueabihf.tar.xz

$ ln -s gcc-linaro-4.9-2014.11-x86_64_arm-linux-gnueabihf gcc-linaro

---

./ 配置环境变量

$ export ARCH=arm

$ export PATH=~/gcc-linaro/bin/:$PATH

$ export CROSS_COMPILE=arm-linux-gnueabihf-

./ 修改 arch/arm/boot/dts/Makefile 文件， 插入 “vf610-colibri-my-carrier.dtb”

---

dtb-$(CONFIG_SOC_VF610) += \

vf500-colibri-eval-v3.dtb \

vf610-colibri-eval-v3.dtb \

vf610-colibri-my-carrier.dtb \

vf500-colibri-dual-eth.dtb \

vf610-colibri-dual-eth.dtb \

vf610-cosmic.dtb \

vf610-twr.dtb

./ 编译设备树文件，源代码根目录 linux-toradex 下，生成的文件可以在 arch/arm/boot/dts/下找到

$ make colibri_vf_defconfig

$ make dtbs

---

4). 部署新的设备树文件并测试

a). 将新的设备树文件“vf610-colibri-my-carrier.dtb”放置到目标板 Colibri VF61 Linux 系统 /boot 目录下

b). 如下修改目标板 uboot 环境变量

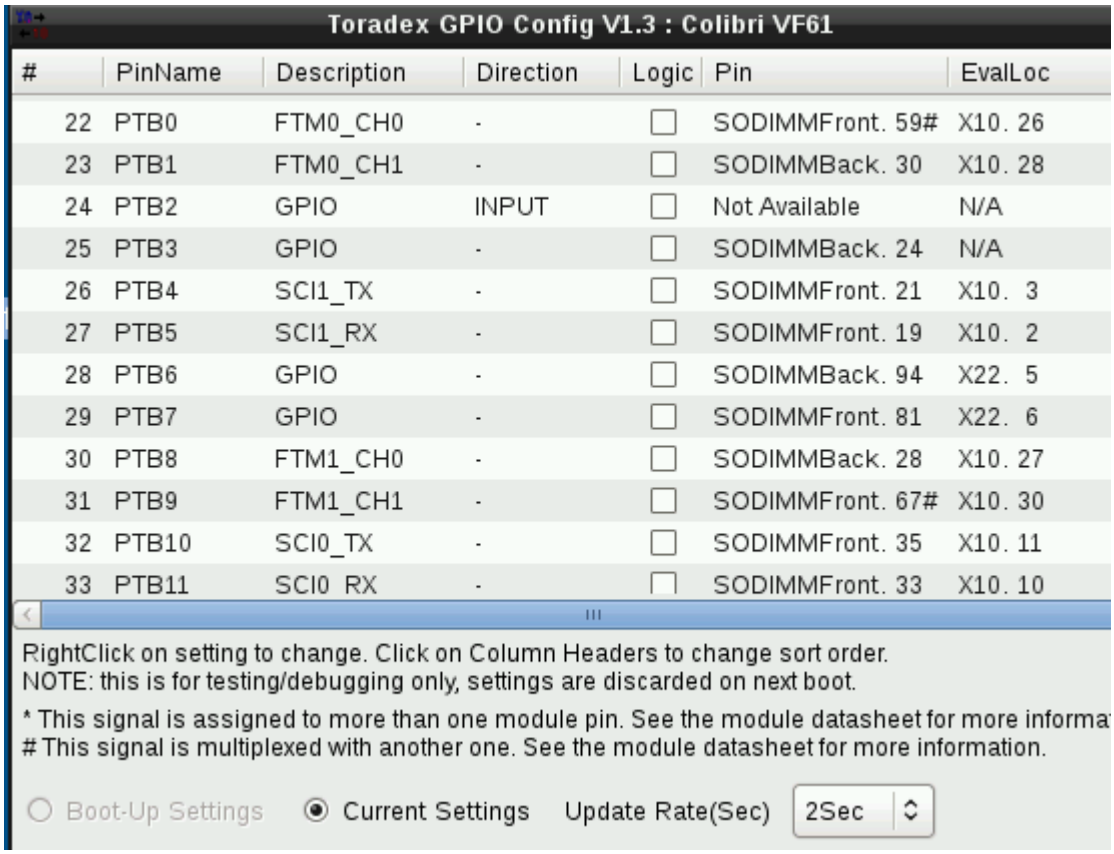
$ setenv fdt_board 'my-carrier'

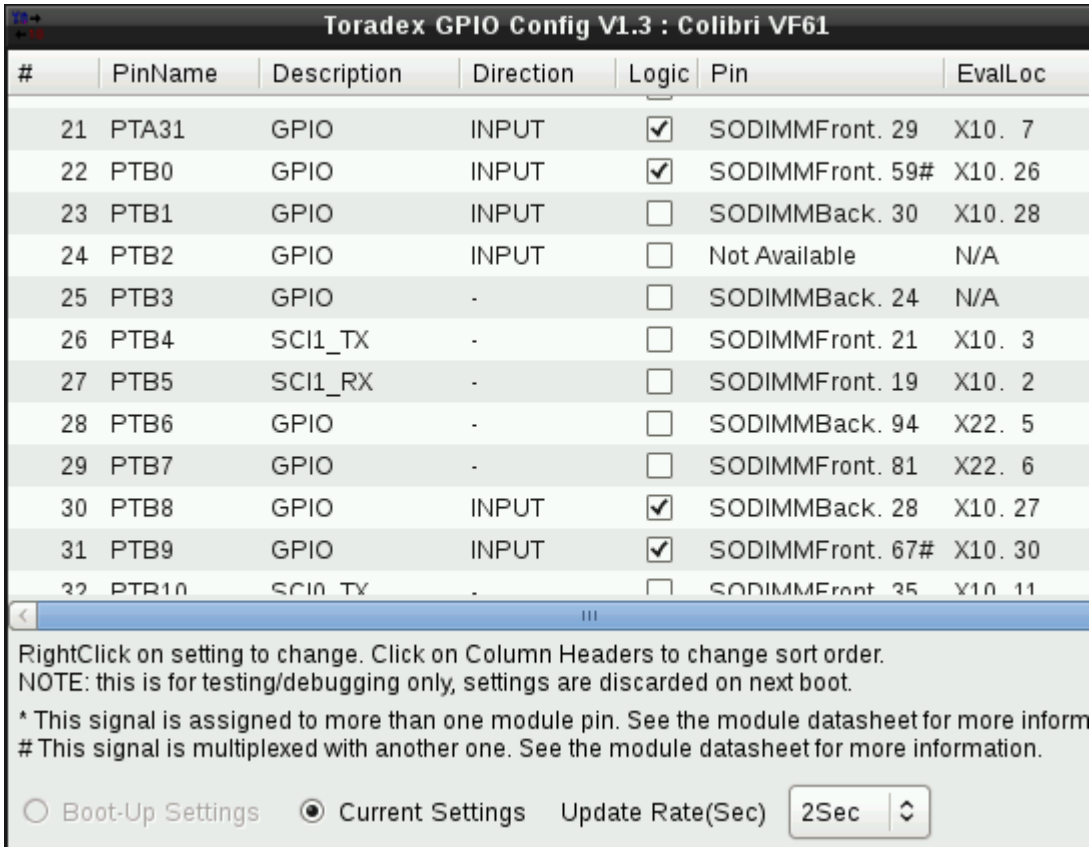
$ saveenv

---

c). 重启后则系统加载新的设备树文件

下面两个截图分别是更改前和更改后使用 Toradex 提供的 GPIOConfig 工具对 PWM 对应管脚进行查看，可以看到由原来的 PWM 属性变成了 GPIO，修改成功后则可以按照这里的说明直接调用 GPIO 使用。





d). 对于 CAN，Colibri VF61 支持两个 CAN 接口，CAN0 和 CAN1，在设备树中使能 CAN 设备示例如下

./ 编辑 vf610-colibri-my-carrier.dts，添加下面内容

```
&can0 {
   status = "Okay";
};
&can1 {
   status = "Okay";
};
&i2c0 {
   status = "disabled";
};
```

./ 和上述方法一样重新编译设备树文件后部署，然后就可以在系统中调用 CAN 了，更详细的说明可以参考[这里](#)。