Linux设备树介绍及加载过程分析

PLCT实验室 实习生 王萌 2020.8.12

目录

Linux设备树介绍

- 设备树：是一个特殊的文件，后缀为*.dtb，linux在启动时加载，用于提供硬件平台的信息，主要是板级的硬件信息。
- 设备树的产生是为了替代linux内核中的硬件细节相关代码。
- 在2.x的内核中，这些信息以特定开发板的设备文件形式存储在arch目录中。
- 设备树出现后，内核中只需要加载相应设备的驱动，而一些板级的细节通过设备树提供，实现了驱动代码与设备信息的分离。

Linux设备树介绍

- 设备树的dtb文件为一个二进制文件，由dts文件编译而成
- dts文件有特定的语法(类似json，用键值对表示信息)，同时，dts文件中可以引用dtsi文件
- 通常，soc厂商会将soc的硬件信息或多个开发板公用的硬件部分写成dtsi文件，将特定于某块开发板的信息写成dts文件，这样，dts文件+dtsi文件就构成了完整的设备树

dts文件语法介绍

- 一个无任何意义的空设备树:
- 设备树中的节点以树形结构表示，根节点始终用/ 表示
- 每个节点有任意数量的属性，同时节点中也可以包含其他子节点
- 属性的值可以为多种形式，包括字符串，整数，二进制数据等

```
/ {
    node1 {
        a-string-property = "A string";
        a-string-list-
property = "first string", "second string";
        a-byte-data-
property = [0x01 0x23 0x34 0x56];
        child-node1 {
            first-child-property;
            second-child-property = <1>;
            a-string-property = "Hello, world";
        };
        child-node2 {
        };
    };
    node2 {
        an-empty-property;
        a-cell-
property = <1 2 3 4>; /* each number (cell) is
a uint32 */
        child-node1 {
        };
    };
};
```

dts文件语法介绍

- 不同属性用key-value对的形式来表示
- 尖括号<>中的数值被识别为32位无符号整型
- 方括号[]中的数值被识别为二进制数据
- 逗号可以用来连接多个相同类型或不同类型的value

```
/ {
    node1 {
        a-string-property = "A string";
        a-string-list-
property = "first string", "second string";
        a-byte-data-
property = [0x01 0x23 0x34 0x56];
        child-node1 {
            first-child-property;
            second-child-property = <1>;
            a-string-property = "Hello, world";
        };
        child-node2 {
        };
    };
    node2 {
        an-empty-property;
        a-cell-
property = <1 2 3 4>; /* each number (cell) is
a uint32 */
        child-node1 {
            mixed-
property = "a string", [0x01 0x23 0x45 0x67], <
0x12345678>;
        };
    };
};
```

dts文件语法介绍

- dts文件中节点的命名规则:
- 节点命名的格式为<name>[@<unit-address>]
- 其中name是必须的， unit-address是可选的，当设备有地址的时候，通常用@unit-address描述设备的主地址
- name是一个ascii字符串，长度最长为31个字符
- name通常为设备的类型，如serial@101F0000，spi@10115000
- 一个设备树中可以存在多个名字相同地址不同的节点

dts文件语法介绍

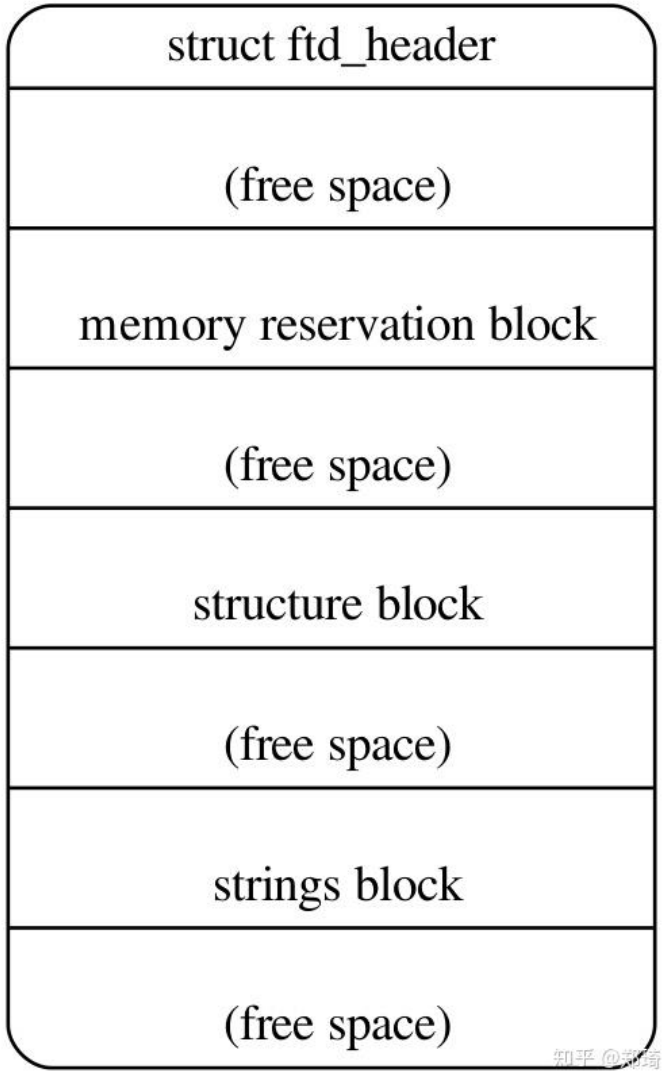
- 在dts文件中，有一些关键字具有特定含义
- compatible属性：用于描述设备型号，格式一般为“<manufacturer>,<model>”，如“sifive,fu540-c000“，compatible属性可以有多个字符串，表示与该设备兼容的其他设备
- compatible属性用于设备树节点与设备驱动程序的绑定
- reg属性：用于描述设备地址，形式一般为<address length [address2 length2] …>，其中address表示基地址，length表示大小

dts文件语法介绍

- 与reg属性相关的还有两个特殊的属性
- #address-cells 和#size-cells
- #address-cells和#size-cells表示每个address和length分别包含多少个32位整数，在64位系统中，这个值可以设置为2以实现64位寻址

Linux启动时设备树加载过程

- 设备树文件在加载之前需要通过dtc工具将dts格式的代码编译为dtb格式的二进制文件
- 编译好的设备树文件由bootloader在启动时加载到内存中，并在启动内核时将设备树的地址一并传递给内核



Linux启动时设备树加载过程

- 当kernel启动后，kernel会首先根据设备树中的平台信息确认内核是否支持
- 接下来会解析出设备树中的运行时配置，主要包括
  - bootarg信息
  - 根节点的address-cells，size-cells的值
  - memory节点的reg信息
- 接下来将设备树中的节点device-nodes解析出来放到一个全局链表allnodes中

Linux启动时设备树加载过程

- 在初始化过程完成之后，内核还会将设备节点转换为platform-device用于匹配驱动
- 并不是所有节点都会转换为platform device，比如内存设备显然不需要
- 只有满足特定条件的节点会转化为平台设备
  - 该节点必须含有compatible属性
  - 根节点的子节点(节点必须含有compatible属性)
  - 含有特殊compatible属性的节点的子节点(子节点必须含有compatible属性):这些特殊的compatilbe属性为: "simple-bus","simple-mfd","isa","arm,amba-bus"

Linux设备驱动调用设备树中节点

- 一个设备树中节点的驱动程序结构体如右图所示
- 可以发现有一个.of_match_table对象，里面存储着一个compatible字符串，即驱动可以通过设备树中的compatible属性来匹配对应的驱动程序

```
static struct of_device_id
beep_table[] = {
{.compatible = "fs4412,beep"},
};
static struct platform_driver
beep_driver=
{
.probe = beep_probe,
.remove = beep_remove,
.driver={
.name = "bigbang",
.of_match_table = beep_table,
},
};
```

Linux设备驱动调用设备树中节点

- 驱动程序获取设备树中的资源，如终端，io内存等
- 这些资源存储在platform_device结构体中的resource对象中
- 如获取io内存资源
- struct resource *res_mem = platform_get_resource(pdev,IORESOURCE_MEM,0);
- 之后就可以进行申请内存，映射地址等操作

```
struct platform_device {
    const char   * name;     /* 名字 */
    int      id;
    struct device    dev;
    u32      num_resources;/* 资源总数 */
    struct resource * resource;/*资源 */
    struct platform_device_id    *id_entry
;
};
1.struct resource {
2.    resource_size_t start;
3.    resource_size_t end;
4.    const char *name;
5.    unsigned long flags;
6.    struct resource *parent, *sibling,
*child;
7.};
```

Linux设备驱动调用设备树中节点

- 驱动程序获取设备树中的资源，如终端，io内存等
- 这些资源存储在platform_device结构体中的resource对象中
- 如获取io内存资源
- struct resource *res_mem = platform_get_resource(pdev,IORESOURCE_MEM,0);
- 之后就可以进行申请内存，映射地址等操作

```
struct platform_device {
    const char   * name;     /* 名字 */
    int      id;
    struct device    dev;
    u32      num_resources;/* 资源总数 */
    struct resource * resource;/*资源 */
    struct platform_device_id    *id_entry
;
};
1.struct resource {
2.    resource_size_t start;
3.    resource_size_t end;
4.    const char *name;
5.    unsigned long flags;
6.    struct resource *parent, *sibling,
*child;
7.};
```

Linux设备驱动调用设备树中节点

- 常用的一些设备树读取属性的api
  - of_find_node_by_path();//按路径查找
  - of_find_compatible_node();//按属性查找
  - //读取属性
  - device_property_read_string();
  - evice_property_read_string_array();
  - device_property_read_u32();
  - device_property_read_u32_array();
  - device_property_read_u8_array();