设备树 of 常用 API

本文档罗列设备树中 device_node 结构体以及常用 of 函数。

结构体 device_node

```
struct device_node {
        const char *name;          //节点名称
        const char *type;          //设备类型
        phandle phandle;
        const char *full_name;    //全路径节点
        struct fwnode_handle fwnode;

        struct    property *properties;
        struct    property *deadprops;      /* removed properties */
        struct    device_node *parent;        //父节点指针
        struct    device_node *child;         //子节点指针
        struct    device_node *sibling;
        struct    kobject kobj;
        unsigned long _flags;
        void       *data;
#if defined(CONFIG_SPARC)
        const char *path_component_name;
        unsigned int unique_id;
        struct of_irq_controller *irq_trans;
#endif
};
```

结构体 property

```
struct property {
        char      *name;
        int       length;
        void      *value;
        struct property *next;
        unsigned long _flags;
        unsigned int unique_id;
        struct bin_attribute attr;
};
```

查找节点 API

函数：struct device_node *of_find_compatible_node(struct device_node *from,const char *type, const char *compat);

功能：of_find_compatible_node，通过 compatible 属性查找指定节点；

参数 from：指向开始路径的节点，如果为 NULL，则从根节点开始；

参数 type：device_type 设备类型，可以为 NULL；

参数 compat：指向节点的 compatible 属性的值（字符串）的首地址；

返回值：成功，得到节点的首地址；失败：NULL。

函数：struct device_node *of_find_matching_node(struct device_node *from,const struct of_device_id *matches);

功能：of_find_matching_node，通过 compatible 属性查找指定节点；

参数 from：指向开始路径的节点，如果为 NULL，则从根节点开始；

参数 matches：指向设备 ID 表，注意 ID 表必须以 NULL 结束；

* 范例：

```
#define DRIVER_NAME "leds_test"
static const struct of_device_id of_leds_dt_match[] = {
        {.compatible = DRIVER_NAME},
        {},
};
```

返回值：成功，得到节点的首地址；失败，NULL。

函数：struct device_node *of_find_node_by_path(const char *path);

功能：of_find_node_by_path，通过路径查找指定节点；

参数：带全路径的节点名，也可以是节点的别名；

返回值：成功，得到节点的首地址；失败，NULL。

函数：struct device_node *of_find_node_by_name(struct device_node *from,const char *name);

功能：of_find_node_by_name，通过节点名查找指定节点；

参数 from：开始查找节点，如果为 NULL，则从根节点开始；

参数 name：节点名；

返回值：成功，得到节点的首地址；失败，NULL。

提取通用属性 API

函数：struct property *of_find_property(const struct device_node *np, const char *name, int *lenp);

功能： of_find_property，提取指定属性的值；

参数 np：设备节点指针；

参数 name：属性名称；

参数 lenp：属性值的字节数；

返回值：成功，属性值的首地址；失败，NULL。

函数：int of_property_count_elems_of_size(const struct device_node *np,const char *propname, int elem_size);

功能：of_property_count_elems_of_size，得到属性值中数据的数量；

参数 np：设备节点指针；

参数 propname：属性名称；

参数 elem_size：每个数据的单位（字节数）；

返回值：成功，属性值的数据个数；失败，负数，绝对值是错误码。

函数：int of_property_read_u32_index(const struct device_node *np, const char *propname, u32 index, u32 *out_value);

功能：of_property_read_u32_index ，得到属性值中指定标号的 32 位数据值；

参数 np：设备节点指针；

参数 propname：属性名称；

参数 index：属性值中指定数据的标号；

参数 out_value：输出参数，得到指定数据的值；

返回值：成功，0；失败，负数，绝对值是错误码。

函数：int of_property_read_string(struct device_node *np, const char *propname, const char **out_string);

功能： of_property_read_string，提取字符串（属性值）；

参数 np：设备节点指针；

参数 propname：属性名称；

参数 out_string：输出参数，指向字符串（属性值）；

返回值： 成功，0；失败，负数，绝对值是错误码。

提取 addr 属性 API

函数：int of_n_addr_cells(struct device_node *np);

功能：of_n_addr_cells，提取默认属性“#address-cells”的值；

参数 np：设备节点指针；

返回值：成功，地址的数量；失败，负数，绝对值是错误码。

函数：int of_n_size_cells(struct device_node *np);

功能：of_n_size_cells，提取默认属性“#size-cells”的值；

参数 np：设备节点指针

返回值：成功，地址长度的数量；失败，负数，绝对值是错误码。

函数：__be32 *of_get_address(struct device_node *dev, int index, u64 *size, unsigned int *flags);

功能：of_get_address，提取 I/O 口地址；

参数 np：设备节点指针；

参数 index：地址的标号；

参数 size：输出参数，I/O 口地址的长度；

参数 flags：输出参数，类型（IORESOURCE_IO、IORESOURCE_MEM）；

返回值：成功，I/O 口地址的首地址；失败，NULL。

函数：u64 of_translate_address(struct device_node *dev, const __be32 *in_addr);

功能：of_translate_address，从设备树中提取 I/O 口地址转换成物理地址；

参数 np：设备节点指针；

参数 in_addr：设备树提取的 I/O 地址；

返回值：成功，物理地址；失败，OF_BAD_ADDR

函数：void __iomem *of_iomap(struct device_node *np, int index);

功能：of_iomap，提取 I/O 口地址并映射成虚拟地址；

参数 np：设备节点指针；

参数 index：I/O 地址的标号；

返回值：成功，映射好虚拟地址；失败，NULL。

函数：void __iomem *of_io_request_and_map(struct device_node *np, int index, const char *name);

功能：提取 I/O 口地址并申请 I/O 资源及映射成虚拟地址；

参数 np：设备节点指针；

参数 index：I/O 地址的标号；

参数 name：设备名，申请 I/O 地址时使用；

返回值：成功，映射好虚拟地址；失败，NULL。

提取 resource 属性 API

函数：int of_address_to_resource(struct device_node *dev, int index, struct resource *r);

功能：of_address_to_resource，从设备树中提取资源 resource（I/O 地址）；

参数 np：设备节点指针；

参数 index：I/O 地址资源的标号；

参数 r：输出参数，指向资源 resource（I/O 地址）；

返回值：成功，0；失败，负数，绝对值是错误码。

提取 GPIO 属性 API

头文件 include/of_gpio.h

函数：int of_get_named_gpio(struct device_node *np, const char *propname, int index);

功能：of_get_named_gpio，从设备树中提取 gpio 口；

参数 np：设备节点指针；

参数 propname：属性名；

参数 index：gpio 口引脚标号 ；

返回值： 成功，得到 GPIO 口编号；失败，负数，绝对值是错误码。

提取 irq 属性 API

函数：int of_irq_count(struct device_node *dev);

功能：of_irq_count 从设备树中提取中断的数量；

参数 np - 设备节点指针；

返回值：成功，大于等于 0，实际中断数量；0 则表示没有中断。

函数：int of_irq_get(struct device_node *dev, int index);

功能：of_irq_get，从设备树中提取中断号；

参数 np：设备节点指针；

参数 index：要提取的中断号的标号；

返回值：成功，中断号；失败，负数，其绝对值是错误码。

提取其他属性 API

函数：void *of_get_mac_address(struct device_node *np);

功能：of_get_mac_address ，从设备树中提取 MAC 地址；

参数 np：设备节点指针；

返回值：成功，MAC（6 字节）的首地址；失败，NULL。