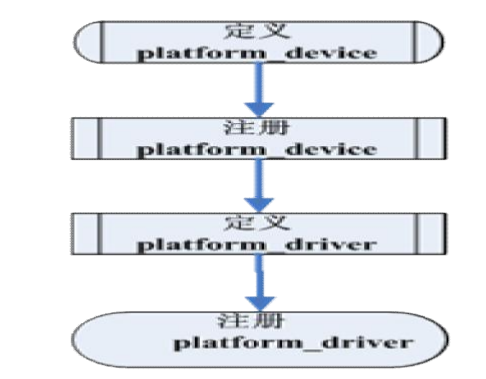
# Platform总线

Platform总线实在Linux2.6内核加入的一种虚拟总线，platform机智的本身使用并不复杂，有两部分组成：

platform\_device和platform\_driver。

Platform驱动与传统的设备驱动模型相比，优势在于platform机制将设备本身的资源注册进内核，由内核统一管理，在驱动程序使用这些资源时使用了统一的接口，这样提高了程序的可移植性。

platform 开发流程如下图所示



平台设备描述

struct platform\_device {

const char \*name; /\*设备名\*/

int id; /\*设备编号，配合设备名使用\*/

struct device dev;

u32 num\_resources;

struct resource \*resource; /\*设备资源\*/

}

struct Platform\_device的分配使用

struct platform\_device

\*platform\_device\_alloc(const char \*name, int id)、

平台设备注册

int platform\_device\_add(struct platform\_device \*pdev)

设备资源

struct resource {

resource\_size\_t start; //资源的起始物理地址

resource\_size\_t end; //资源的结束物理地址

const char \*name; //资源的名称

unsigned long flags; //资源的类型，比如MEM，IO，IRQ类型

struct resource \*parent, \*sibling, \*child; //资源链表指针

}

获取资源

struct resource \*platform\_get\_resource(struct platform\_device\*dev, unsigned int type, unsigned int num)

代码示例

#include <linux/module.h>

#include <linux/types.h>

#include <linux/fs.h>

#include <linux/init.h>

#include <linux/platform\_device.h>

#include <mach/regs-gpio.h>

#include <linux/interrupt.h>

#include <linux/device.h>

#include <linux/io.h>

/\*平台资源的定义\*/

static struct resource s3c\_buttons\_resource[] = {

[0]={

.start = S3C24XX\_PA\_GPIO,

.end = S3C24XX\_PA\_GPIO + S3C24XX\_SZ\_GPIO - 1,

.flags = IORESOURCE\_MEM,

},

[1]={

.start = IRQ\_EINT8,

.end = IRQ\_EINT8,

.flags = IORESOURCE\_IRQ,

},

[2]={

.start = IRQ\_EINT11,

.end = IRQ\_EINT11,

.flags = IORESOURCE\_IRQ,

},

[3]={

.start = IRQ\_EINT13,

.end = IRQ\_EINT13,

.flags = IORESOURCE\_IRQ,

},

[4]={

.start = IRQ\_EINT14,

.end = IRQ\_EINT14,

.flags = IORESOURCE\_IRQ,

},

[5]={

.start = IRQ\_EINT15,

.end = IRQ\_EINT15,

.flags = IORESOURCE\_IRQ,

},

[6]={

.start = IRQ\_EINT19,

.end = IRQ\_EINT19,

.flags = IORESOURCE\_IRQ,

}

};

static struct platform\_device \*s3c\_buttons;

static int \_\_init platform\_init(void)

{

s3c\_buttons = platform\_device\_alloc("mini2440-buttons",-1);

platform\_device\_add\_resources(s3c\_buttons,&s3c\_buttons\_resource,7);

/\*平台设备的注册\*/

platform\_device\_add(s3c\_buttons);

}

static void \_\_exit platform\_exit(void)

{

platform\_device\_unregister(s3c\_buttons);

}

module\_init(platform\_init);

module\_exit(platform\_exit);

MODULE\_AUTHOR("David Xie");

MODULE\_LICENSE("GPL");

MODULE\_ALIAS("platform:mini2440buttons");

平台驱动描述

struct platform\_driver {

int (\*probe)(struct platform\_device \*);

int (\*remove)(struct platform\_device \*);

void (\*shutdown)(struct platform\_device \*);

int (\*suspend)(struct platform\_device \*, pm\_message\_t state);

int (\*suspend\_late)(struct platform\_device \*, pm\_message\_t state);

int (\*resume\_early)(struct platform\_device \*);

int (\*resume)(struct platform\_device \*);

struct device\_driver driver;

}

平台驱动注册

int platform\_driver\_register(struct platform\_driver \*)