# 一、INPUT子系统简介

 input子系统也是字符设备，input核心层会帮我们注册input字符设备驱动。既然内核已经帮我们写好了input驱动，那我们要干啥呢？需要我们去完善具体的输入设备，完善输入设备的时候就要按照inpu子系统驱动框架的要求来。

##  1、input\_dev

 申请并初始化并注册input\_dev。使用input\_allocate\_device申请，evbit表示输入事件，比如按键对应的事件就是EV\_KEY，如果要连按，那么还要加EV\_REP。

 设置按键对应的键值，也就是keybit。

 初始化完成input\_dev以后，需要向内核注册。使用input\_register\_device

##  2、事件上报

 按键按下以后上报事件，比如对于按键而言就是在按键中断服务函数，或者消抖定时器函数里面获取按键按下情况，并且上报，可以使用：

input\_event(struct input\_dev \*dev, unsigned int type, unsigned int code, int value);

 对于按键而言，也可以使用：

input\_report\_key

 使用上面两个函数上报完成输入事件以后，还需要使用input\_sync做同步，

##  3、input\_event

 应用程序可以通过input\_event来获取输入事件数据，比如按键值，input\_event是一个结构体：

struct input\_event {

 struct timeval time;

 \_\_u16 type;

 \_\_u16 code;

 \_\_s32 value;

};

struct timeval {

 \_\_kernel\_time\_t tv\_sec; /\* seconds \*/

 \_\_kernel\_suseconds\_t tv\_usec; /\* microseconds \*/

};

展开：

struct timeval {

 long tv\_sec; /\* seconds \*/

 long tv\_usec; /\* microseconds \*/

};

最终将input\_event展开以后：

struct input\_event {

 struct timeval {

 long tv\_sec; /\* seconds \*/ 32位表示秒

 long tv\_usec; /\* microseconds \*/ 32位表示微秒

};

 \_\_u16 type; 16位的事件类型

 \_\_u16 code; 16位的事件码，对于按键而言就是键码

 \_\_s32 value; 32位的值，对于按键就是按下或抬起

};

# 二、使用INPUT驱动框架

##  1、key设备树。

 /\* zozhongkai beep \*/

 beep{

 compatible = "alientek,beep";

 pinctrl-names = "default";

 pinctrl-0 = <&pinctrl\_beep>;

 beep-gpios = <&gpio5 1 GPIO\_ACTIVE\_HIGH>;

 status = "okay";

 };

## 三、运行测试

 查看原始数据：

0000000 005e 0000 4f71 0002 0001 000b 0001 0000

0000010 005e 0000 4f71 0002 0000 0000 0000 0000

0000020 005e 0000 5cab 0005 0001 000b 0000 0000

0000030 005e 0000 5cab 0005 0000 0000 0000 0000

0000040 005e 0000 f08c 0009 0001 000b 0001 0000

0000050 005e 0000 f08c 0009 0000 0000 0000 0000

0000060 005e 0000 ec55 000b 0001 000b 0000 0000

0000070 005e 0000 ec55 000b 0000 0000 0000 0000

0000080 005f 0000 7aa6 0000 0001 000b 0001 0000

0000090 005f 0000 7aa6 0000 0000 0000 0000 0000

# 四、编写应用程序

 按键驱动对应的文件就是/dev/input/eventX，(X=0,1,2,3。。。)，应用程序读取/dev/input/event1来得到按键信息，也就是按键有没有被按下。

 我们通过/dev/input/event1读到的信息是input\_event结构体形式的。

# 五、Linux内核自带按键驱动程序使用

 配置内核，选中内核自带的KEY驱动程序，然后会在.config里面产生：

 CONFIG\_KEYBOARD\_GPIO=y

 经过查找，内核自带的KEY驱动程序是gpio\_keys.c，这是一个标准的platform驱动，其中compatible = "gpio-keys"，因此我们在设备树中添加对应的节点的时候，其compatible属性必须是“gpio-keys“。