增加一个组合键的过程

例：KEY\_UNLOCK & KEY\_OFFSET

驱动层：

文件:keyboard.c

添加按键是否按下的判断函数

static uint8\_t IsKeyDown\_key\_lock\_offset(void) {return (cur\_key\_value == (KEY\_UNLOCK & KEY\_OFFSET));}

添加组合键使用的按键管理结构

static KEY\_STRUCT s\_Key\_lock\_offset;/\* KEY\_UNLOCK & KEY\_OFFSET \*/

在函数InitKeyStr中添加按键信息初始化语句

init\_key\_info(&s\_Key\_lock\_offset , 0, KEY\_UNLOCK & KEY\_OFFSET, 0, COMBINATION\_KEY, IsKeyDown\_key\_lock\_offset);

至此驱动层修改完毕

应用层：

文件：key\_fun\_manage.h

在结构KEY\_FUNCATION中增加成员

KEY\_DISPOSE\_FUN key\_clock\_offset;///< 组合键 KEY\_CLOCK + KEY\_OFFSET

文件：key\_fun\_manage.c

在函数get\_key\_inf中增加语句

case KEY\_UNLOCK & KEY\_OFFSET: temp\_fun = &key\_funcation.key\_clock\_offset;break;

至此应用层修改完毕

用户层：

使用方法同普通的系统功能按键一样:

举例:

static CONFIG\_FUNCTION\_KEY\_INFO\_T sys\_key\_pool[]={

{ KEY\_UNLOCK & KEY\_OFFSET , sys\_key\_unlock\_offset\_up\_cb },

};

其中的回调函数需要自己实现：其定义格式为：

static void sys\_key\_unlock\_offset\_up\_cb(KEY\_MESSAGE \*key\_msg)