#include <linux/init.h>

#include <linux/module.h>

#include <linux/kernel.h>

#include <linux/timer.h>

#include <asm/io.h>

#define GPIO\_BASE 0x3F200000

#define GPIO\_BLOCK\_SIZE 4096

// GPIO setup macros. Always use INP\_GPIO(x) before using OUT\_GPIO(x) or SET\_GPIO\_ALT(x,y)

#define INP\_GPIO(g) \*(gpio+((g)/10)) &= ~(7<<(((g)%10)\*3))

#define OUT\_GPIO(g) \*(gpio+((g)/10)) |= (1<<(((g)%10)\*3))

#define GPIO\_SET \*(gpio+7) // sets bits which are 1 ignores bits which are 0

#define GPIO\_CLR \*(gpio+10) // clears bits which are 1 ignores bits which are 0

#define GPIO\_SETB(g) GPIO\_SET |= 1 <<(g)

#define GPIO\_CLRB(g) GPIO\_CLR |= 1 <<(g)

#define LED\_PIN 21 // Maps to Pin 40 on the RasPi connector

volatile unsigned int \* gpio;

struct timer\_list led\_timer;

int status = 0;

/\* main function \*/

static void blink\_timer(unsigned long ptr)

{

printk("Toggling LED\n");

if (status == 0) {

GPIO\_SETB(LED\_PIN);

status = 1;

} else {

GPIO\_CLRB(LED\_PIN);

status = 0;

}

led\_timer.expires = jiffies + HZ;

add\_timer(&led\_timer);

}

static int blink\_init(void)

{

printk("Hello,LED world!\n");

gpio = ioremap(GPIO\_BASE,GPIO\_BLOCK\_SIZE);

INP\_GPIO(LED\_PIN);

OUT\_GPIO(LED\_PIN);

init\_timer(&led\_timer);

led\_timer.function = blink\_timer;

led\_timer.data = (unsigned long) status;

led\_timer.expires = jiffies + HZ;

add\_timer(&led\_timer);

return 0;

}

static void blink\_exit(void)

{

del\_timer(&led\_timer);

GPIO\_CLRB(LED\_PIN);

iounmap(gpio);

printk("Good bye!\n");

}

module\_init(blink\_init);

module\_exit(blink\_exit);

MODULE\_LICENSE("GPL");

MODULE\_AUTHOR("WWW");

MODULE\_DESCRIPTION("LED Blinker Driver");

MODULE\_VERSION("0.1");