设定PIT(可编程的间隔型定时器)，写在timer.c中，改Makefile

编写

void init\_pit(void); // 初始化PIT

增加计时器中断处理

void inthandler20(int \*esp);        // 计时器中断

在汇编中加入函数\_asm\_inthandler20和c函数\_inthandler20声明

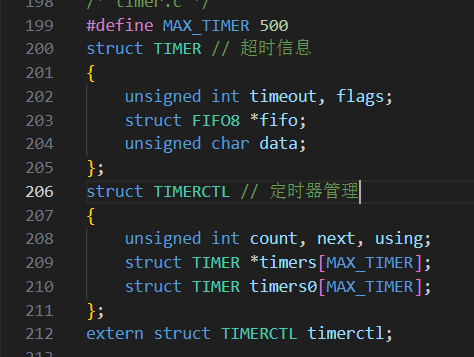
在.h中加入声明

void asm\_inthandler20(void); // 调用定时器中断处理函数

在dsctbl.c的初始化GDI,IDT中把这个中断注册到IDT

    set\_gatedesc(idt + 0x20, (int)asm\_inthandler20, 2 \* 8, AR\_INTGATE32);

设定多个计时器和加快中断处理，加两结构体和四个函数



struct TIMER \*timer\_alloc(void);  // 取得新生成的未使用定时器

void timer\_free(struct TIMER \*timer);    // 释放定时器的内存

void timer\_init(struct TIMER \*timer, struct FIFO8 \*fifo, unsigned char data); // 定时器初始化

void timer\_settime(struct TIMER \*timer, unsigned int timeout);                // 设定定时器

测试，3秒和10秒后显示字符串

