将任务状态段结构体搬到.h中,创建加了任务编号和活动状态的结构体

struct TASK

{

    int sel, flags;      /\* sel用来存放GDT的编号\*/

    struct TSS32 tss;

};

和任务管理

struct TASKCTL

{

    int running; /\*正在运行的任务数量\*/

    int now;     /\*这个变量用来记录当前正在运行的是哪个任务\*/

    struct TASK\* tasks[MAX\_TASKS];

    struct TASK tasks0[MAX\_TASKS];

};

。。。。。。

为加入任务唤醒，在缓冲结构体中加入任务的信息

struct FIFO32 // 缓冲

{

    int \*buf;

    int p, q, size, free, flags;

    struct TASK \*task;

};

修改缓冲区初始化

void fifo32\_init(struct FIFO32 \*fifo, int size, int \*buf, struct TASK \*task); // 初始化FIFO缓冲区

在写入向FIFO写入数据函数中增加唤醒任务功能

int fifo32\_put(struct FIFO32 \*fifo, int data);                                // 向FIFO传送数据并保存

为增加任务优先级增加任务等级结构体

struct TASKLEVEL

{

    int running; /\*正在运行的任务数量\*/

    int now;     /\*这个变量用来记录当前正在运行的是哪个任务\*/

    struct TASK \*tasks[MAX\_TASKS\_LV];

};

修改

struct TASK

和

struct TASKCTL

再修改

int fifo32\_put(struct FIFO32 \*fifo, int data);                                // 向FIFO传送数据并保存

增加和修改这些函数

struct TASK \*task\_init(struct MEMMAN \*memman);             // 初始化任务管理

struct TASK \*task\_alloc(void);                             // 初始化任务结构

void task\_run(struct TASK \*task, int level, int priority); // 增加正在运行任务数量和设置优先级

void task\_switch(void);                                    // 任务切换

void task\_sleep(struct TASK \*task);                        // 任务休眠

struct TASK \*task\_now(void);                               // 返回活动中任务状态段地址

void task\_add(struct TASK \*task);                          // 向不同的等级中添加任务

void task\_remove(struct TASK \*task);                       // 在不同的等级中删除任务

void task\_switchsub(void);                                 // 决定切换到那个等级

修改timer.c中

void inthandler20(int \*esp)

在主函数文件中修改

void make\_window8(unsigned char \*buf, int xsize, int ysize, char \*title, char act);    // 画窗口

测试

