#include "sys.h"

#include "delay.h"

#include "usart.h"

#include "lcd.h"

#include "key.h"

#include "malloc.h"

#include "sdio\_sdcard.h"

#include "w25qxx.h"

#include "ff.h"

#include "exfuns.h"

#include "text.h"

#include "suocun.h"

#include "tts.h"

#include "timer.h"

#include<stdlib.h>

#include<string.h>

#define MAXSIZE 10//数组大小基数

#define MAXBK 10//书签数目

#define MAXDR 10//单页显示最多的目录数目

const char ext[]=".c";//书签的扩展名

void k3\_last(u8 n);//目录端和文本端的上一个

void move\_Bookmarkcursor\_Last(u16 n);//选择下一个书签

void selectSort(char(\*pt)[50], int n);//书签排序函数

void select\_Bookmark(u16 index);//选择要打开的书签，和确认键的功能类似

void move\_Bookmarkcursor\_Next(u16 n);//选择下一个书签

FRESULT bookmark\_Read(const u8\*path,u8\*buf\_Read);

int display\_Bookmark();//遍历出所有的.c文件并且按照书签1~10的顺序排序

u32 path\_Change(u32 cnt1,u16 cnt2);//得到要替换书签名的字串，例如要替换书签1 XXX，得到字串书签1

FRESULT create\_New(const u8\*path,const u8\*path2,UINT nums);//path是文件的路径，path2是写入的字符数组,先写text再写data最后写文本存在的路径

FRESULT times\_Init(const u8\*path);//初始化 创建次数.txt(nums.txt)，初始值为0，写入成功后返回值为0

void Int\_To\_Str(int x, char\* Str);//十进制转换成字符串

u32 times\_Read(const u8\*path);//读 创建次数.txt(nums.txt)获取创建书签的次数

FRESULT times\_Write(const u8\*path);//写 创建次数.txt(nums.txt)，增加创建书签的次数，写入成功后返回值为0

u16 scan\_Bookmark (const u8\* path,int num);//获取书签的数目并且将要替换的书签名赋值给nameReplaced

void findExt( char\* file\_name, char\* name, char\* ext);//得出文件名的扩展名和文件名

u8 s\_files(u8\*);

void highlight( u8 );

int mouse=0;

int countdown;

int flag=0;

u8 k1\_event(u8);

u8 k2\_event();

void k3\_event(u8);

int curplace=0;

u16 bkplace=0;//书签的位置信息

u8 buf\_Read[100]="";//存放书签读到的地址

u8 path[200]={""};//全局变量，存放当前地址

u8 nameReplaced[100]="";//要替换的书签名

u8\*pshu=0;

void k4\_event();

void k5\_event();

void k6\_event();

char buffer1[38];

char buffer2[121]="";

FIL fil1,fil2;//fil3删了

u8\*pathtxt=0;

u8\*pathdata=0;

int demo=1;

void dis\_txt();

void play();

void reco();

DWORD P1,P2;//P1是text的位置，P2是data的位置

DWORD P3,P4;//P3是读到的text的位置，P4是读到的data的位置

u8 path\_c[6\*MAXSIZE]="";//创建的地方

FRESULT res;

BYTE buffer[20\*MAXSIZE]="";//按照gb2312编码，一个汉字两个字节

char filename[50][50]={{""}};//存放fn的二维数组

char midname[50]="";//需要发声的数组

char bookmarkname[10][50]={{""}};//存放书签名的二维数组

char bkmidname[50]="";//修改后需要发声的数组

int main()

{

FRESULT res1;

int key\_v;

u8 menu=0;

int n\_files;

int bk\_nums;

delay\_init();

NVIC\_PriorityGroupConfig(NVIC\_PriorityGroup\_2);

uart\_init(9600);

KEY\_Init();

LCD\_Init();//5510的width480 height800

W25QXX\_Init();

my\_mem\_init(SRAMIN);

exfuns\_init();

SUOCUN\_init();

font\_init();

f\_mount(fs[0],"0:",1);

POINT\_COLOR=RED;

TIM3\_Int\_Init(99,7199);//0.01s

times\_Init("0:/nums.txt");

n\_files=s\_files("0:");

strcpy((char\*)path,"0:");

while(1)

{

key\_v=KEY\_Scan(0);

if(key\_v==1&&demo==1)

{

mouse=0;

n\_files=k1\_event(curplace);

}

else if(key\_v==2&&demo==1)

k3\_event(n\_files);//目录端下一个

else if(key\_v==3&&demo==1)

k3\_last(n\_files);//目录端上一个

else if(key\_v==3&&demo==0)

k5\_event();//文本端上一页

else if(key\_v==2&&demo==0)

k4\_event();//文本端下一页

else if(key\_v==5)

n\_files=k2\_event();//上一级

else if(key\_v==6)

bk\_nums=display\_Bookmark();//显示书签

else if(key\_v==1&&demo==2)

select\_Bookmark(bkplace);//书签端确认

else if(key\_v==3&&demo==2)

move\_Bookmarkcursor\_Last(bk\_nums);//书签端上一个

else if(key\_v==2&&demo==2)

move\_Bookmarkcursor\_Next(bk\_nums);//书签端下一个

if(countdown==30000)

{

dis(0);

LCD\_LED=0;

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOE,DISABLE);

}

}

return 0;

}

u8 s\_files(u8\*path)

{

u8 res=0;

DIR\*dir=0;

FILINFO\*finfo=0;

u8\*fn=0;

u8\*pathp=0;

int i=0;

int j=0;//屏幕显示位置，行数变量

BYTE buffer[20\*MAXSIZE]="";

SPeech("[g0][h0]");

dir=(DIR\*)mymalloc(SRAMIN,sizeof(DIR));

finfo =(FILINFO\*)mymalloc(SRAMIN,sizeof(FILINFO));

finfo->lfsize=\_MAX\_LFN\*2+1;

finfo->lfname= mymalloc(SRAMIN,finfo->lfsize);

//注意，这个必须放在finfo.lfanme之前。先为finfo申请内存，再才会有为finfo.lfname申请内存

//dir=(DIR\*)mymalloc(SRAMIN,sizeof(DIR));

//finfo =(FILINFO\*)mymalloc(SRAMIN,sizeof(FILINFO));

pathp=mymalloc(SRAMIN,511);

pathp[0]=0;

mymemset(pathp,0,511);

memset(&filename[0][0],0,sizeof(char)\*50\*50);

highlight(0);

strcat((char\*)pathp,(const char\*)path);

res=f\_opendir(dir,(const TCHAR\*)pathp);

if(res==0)

{

while(1)

{

res=f\_readdir(dir,finfo);

fn=(u8\*)(\*finfo->lfname?finfo->lfname:finfo->fname);

if(!finfo->fname[0]||res!=FR\_OK) break;

if(finfo->fattrib==AM\_DIR)

{

strcpy((u8\*)filename[i],(u8\*)fn);

i++;

}

}

if(i>MAXDR)

{

for(j=0;j<MAXDR;j++)

Show\_Str(60,32+42\*j,800,42,(u8\*)filename[j]+2,32,0);

}

else

{

for(j=0;j<i;j++)

Show\_Str(60,32+42\*j,800,42,(u8\*)filename[j]+2,32,0);

}

memset(midname,'\0',sizeof(midname));

strncpy(midname,filename[0]+2,strlen(filename[0])-2);

SPeech(midname);

}

f\_closedir(dir);

myfree(SRAMIN,finfo->lfname);

myfree(SRAMIN,dir);

myfree(SRAMIN,finfo);

myfree(SRAMIN,pathp);

return i;

}

u8 k1\_event(u8 curfiles)

{

u8 res=0;

DIR\*dir=0;

FILINFO\*finfo=0;

u8\*fn=0;

u8\*pathp=0;

int i=0;

int j=0;

u8 red=0;

u8 ret=0;

LCD\_Clear(WHITE);

SPeech("[g0][h0]");

dir=(DIR\*)mymalloc(SRAMIN,sizeof(DIR));

finfo =(FILINFO\*)mymalloc(SRAMIN,sizeof(FILINFO));

finfo->lfsize=\_MAX\_LFN\*2+1;

finfo->lfname= mymalloc(SRAMIN,finfo->lfsize);

pathp=mymalloc(SRAMIN,511);

pathp[0]=0;

strcat((char\*)pathp,(char\*)path);

memset(path,0,200);

memset(&filename[0][0],0,sizeof(char)\*50\*50);

res=f\_opendir(dir,(const TCHAR\*)pathp);

if(res==0)

{

while(1)

{

res=f\_readdir(dir,finfo);

fn=(u8\*)(\*finfo->lfname?finfo->lfname:finfo->fname);

if(!finfo->fname[0]||res!=FR\_OK) break;

if(finfo->fattrib==AM\_DIR)

{

if(i==curfiles) break;

i++;

}

}

}

strcat((char\*)pathp,"/");

strcat((char\*)pathp,(char\*)fn);

strcpy((char\*)path,(char\*)pathp);

f\_closedir(dir);

i=0;

res=f\_opendir(dir,(const TCHAR\*)pathp);

if(res==0)

{

while(1)

{

res=f\_readdir(dir,finfo);

fn=(u8\*)(\*finfo->lfname?finfo->lfname:finfo->fname);

if(!finfo->fname[0]||res!=FR\_OK) break;

if(finfo->fattrib==AM\_DIR)

{

strcpy((u8\*)filename[i],(u8\*)fn);

i++;

}

else

{

demo=0;

break;

}

}

if(demo==1)

{

if(i>MAXDR)

{

for(j=0;j<MAXDR;j++)

Show\_Str(60,32+42\*j,800,42,(u8\*)filename[j]+2,32,0);

}

else

{

for(j=0;j<i;j++)

Show\_Str(60,32+42\*j,800,42,(u8\*)filename[j]+2,32,0);

}

memset(midname,'\0',sizeof(midname));

strncpy(midname,filename[0]+2,strlen(filename[0])-2);

SPeech(midname);

}

}

f\_closedir(dir);

myfree(SRAMIN,finfo->lfname);

myfree(SRAMIN,dir);

myfree(SRAMIN,finfo);

myfree(SRAMIN,pathp);

curplace=0;

if(demo==1)

{

highlight(mouse);

}

else if(demo==0)

{

pathtxt=mymalloc(SRAMIN,511);

pathdata=mymalloc(SRAMIN,511);

pathtxt[0]=pathdata[0]=0;

mymemset(pathtxt,0,511);

mymemset(pathdata,0,511);

strcat((char\*)pathtxt,(char\*)path);

strcat((char\*)pathdata,(char\*)path);

strcat((char\*)pathtxt,"/text.txt");

strcat((char\*)pathdata,"/data.txt");

ret=f\_open(&fil1,(char\*)pathtxt,FA\_READ);

red=f\_open(&fil2,(char\*)pathdata,FA\_READ);

if(red==FR\_OK&&ret==FR\_OK)

dis\_txt();

}

return i;

}

u8 k2\_event()

{

FRESULT res=0;

DIR\*dir=0;

FILINFO\*finfo=0;

u8\*fn=0;

u8\*pathp=0;

u8\*patht=0;

char\*p=0;

char\*q=0;

int i=0;

int l=0;//显示行数

int j;//文件名数组的起始和终止位置

int k;//父目录标志

u8 fnt[100]={""};

u8 bkname[100]={""};

u32 cnt1;//创建书签的次数

u16 cnt2;//已有书签的个数

u32 cnt3;//要写入文本地址的长度

int index;//要新建的书签编号

LCD\_Clear(WHITE);

dir=(DIR\*)mymalloc(SRAMIN,sizeof(DIR));

finfo =(FILINFO\*)mymalloc(SRAMIN,sizeof(FILINFO));

finfo->lfsize=\_MAX\_LFN\*2+1;

finfo->lfname= mymalloc(SRAMIN,finfo->lfsize);

memset(&filename[0][0],0,sizeof(char)\*50\*50);

pathp=mymalloc(SRAMIN,511);

pathp[0]=0;

strcat((char\*)pathp,(char\*)path);

memset(fnt,0,100);

memset(bkname,0,100);//文档所属的名称

memset(path\_c,0,6\*MAXSIZE);

memset(buffer,0,20\*MAXSIZE);//要写入地址的路径

if(demo==0)

{

f\_close(&fil1);

f\_close(&fil2);

myfree(SRAMIN,pathtxt);

myfree(SRAMIN,pathdata);

strcpy((char\*)buffer,(const char\*)path);

cnt3=strlen((const char\*)buffer);

cnt1=times\_Read("0:/nums.txt");

cnt2=scan\_Bookmark("0:",-1);//已有书签的个数

index=path\_Change(cnt1,cnt2);

scan\_Bookmark("0:",index);

if(\*nameReplaced!=NULL)

{

f\_unlink((char\*)nameReplaced);

}

q=strrchr((const char\*)pathp,'/');

strcpy((char\*)bkname,q?q+3:pathp);//把数字和'/'去了

strcat(path\_c,bkname);

strcat(path\_c,ext);

res=create\_New((const u8\*)path\_c,buffer,cnt3);

if(res==FR\_OK)

times\_Write("0:/nums.txt");

q=NULL;

}

demo=1;

p=strrchr((const char\*)pathp,'/');

strcpy(fnt,p?p+1:pathp);

if(p!=NULL)

{

\*p='\0';

//pathp[(u8)p]='\0';//?

}

memset(path,0,200);

strcpy((char\*)path,(char\*)pathp);

res=f\_opendir(dir,(const TCHAR\*)pathp);//打开修改过的路径

if(res==0)

{

while(1)

{

res=f\_readdir(dir,finfo);

fn=(u8\*)(\*finfo->lfname?finfo->lfname:finfo->fname);

if(!finfo->fname[0]||res!=FR\_OK) break;

if(finfo->fattrib==AM\_DIR)

{

strcpy((u8\*)filename[i],(u8\*)fn);

if(!strcmp(fnt,fn))

k=i;

i++;

}

}

if(i>MAXDR)

{

for(j=(k/MAXDR)\*MAXDR;j<(k/MAXDR)\*MAXDR+MAXDR;j++,l++)

Show\_Str(60,32+42\*l,800,42,(u8\*)filename[j]+2,32,0);

}

else

{

for(j=0;j<i;j++)

Show\_Str(60,32+42\*j,800,42,(u8\*)filename[j]+2,32,0);

}

memset(midname,'\0',sizeof(midname));

strncpy(midname,filename[k]+2,strlen(filename[k])-2);

SPeech(midname);

curplace=k;

if(curplace>=MAXDR)

{

mouse=curplace%MAXDR;

highlight(mouse);

}

else

{

mouse=curplace;

highlight(mouse);

}

}

f\_closedir(dir);

myfree(SRAMIN,finfo->lfname);

myfree(SRAMIN,dir);

myfree(SRAMIN,finfo);

myfree(SRAMIN,pathp);

return i;

}

void k3\_event(u8 n)

{

int j=0;

int l=0;

int curplace2;

curplace=(curplace+1)%n;

if(n>MAXDR)

{

if(curplace%MAXDR==0&&n>(curplace/MAXDR)\*MAXDR+MAXDR)

{

LCD\_Clear(WHITE);

for(j=(curplace/MAXDR)\*MAXDR;j<(curplace/MAXDR)\*MAXDR+MAXDR;j++,l++)

Show\_Str(60,32+42\*l,800,42,(u8\*)filename[j]+2,32,0);

}

if(curplace%MAXDR==0&&n<=(curplace/MAXDR)\*MAXDR+MAXDR)

{

LCD\_Clear(WHITE);

for(j=(curplace/MAXDR)\*MAXDR;j<n;j++,l++)

Show\_Str(60,32+42\*l,800,42,(u8\*)filename[j]+2,32,0);

}

}

curplace2=curplace%MAXDR;

mouse=curplace2;

highlight(mouse);

memset(midname,'\0',sizeof(midname));

delay\_ms(1);

strncpy(midname,filename[curplace]+2,strlen(filename[curplace])-2);

SPeech(midname);

}

void k3\_last(u8 n)

{

int j=0;

int l=0;

int curplace2;

curplace=(curplace-1)%n;

if(n>MAXDR)

{

if(curplace<0)

{

LCD\_Clear(WHITE);

curplace=n-1;

for(j=(curplace/MAXDR)\*MAXDR;j<n;j++,l++)

Show\_Str(60,32+42\*l,800,42,(u8\*)filename[j]+2,32,0);

}

if((curplace+1)%MAXDR==0&&(curplace+1)!=0)

{

LCD\_Clear(WHITE);

for(j=(curplace/MAXDR)\*MAXDR;j<(curplace/MAXDR)\*MAXDR+MAXDR;j++,l++)

Show\_Str(60,32+42\*l,800,42,(u8\*)filename[j]+2,32,0);

}

}

else

{

if(curplace<0)

curplace=n-1;

}

curplace2=curplace%MAXDR;

mouse=curplace2;

highlight(mouse);

memset(midname,'\0',sizeof(midname));

delay\_ms(1);

strncpy(midname,filename[curplace]+2,strlen(filename[curplace])-2);

SPeech(midname);

}

void k4\_event()

{

if(f\_tell(&fil1)==f\_size(&fil1))

{

f\_lseek(&fil2,0);

f\_lseek(&fil1,0);

}

P1=f\_tell(&fil1);

P2=f\_tell(&fil2);

LCD\_Clear(WHITE);

f\_gets((char\*)buffer2,121,&fil2);

f\_gets((char\*)buffer1,38,&fil1);

play();

}

void k5\_event()

{

if(f\_tell(&fil1)==f\_size(&fil1))

{

if(f\_size(&fil1)%37==0)

{

f\_lseek(&fil1,f\_size(&fil1)-74);

f\_lseek(&fil2,f\_size(&fil2)-240);

}

else

{

f\_lseek(&fil1,f\_size(&fil1)-f\_size(&fil1)%37-37);

f\_lseek(&fil2,f\_size(&fil2)-f\_size(&fil2)%120-120);

}

}

else if(f\_tell(&fil1)==37)

{

if(f\_size(&fil1)%37==0)

{

f\_lseek(&fil1,f\_size(&fil1)-37);

f\_lseek(&fil2,f\_size(&fil2)-120);

}

else

{

f\_lseek(&fil1,f\_size(&fil1)-f\_size(&fil1)%37);

f\_lseek(&fil2,f\_size(&fil2)-f\_size(&fil2)%120);

}

}

else

{

f\_lseek(&fil1,f\_tell(&fil1)-74);

f\_lseek(&fil2,f\_tell(&fil2)-240);

}

P1=f\_tell(&fil1);

P2=f\_tell(&fil2);

LCD\_Clear(WHITE);

f\_gets(buffer2,121,&fil2);

f\_gets(buffer1,38,&fil1);

play();

}

void highlight(const u8 line)

{

LCD\_Fill(35,45,45,800,WHITE);//清除原先位置，白色填充

LCD\_Fill(35,45+42\*line,45,55+42\*line,BLUE);

}

void dis\_txt()

{

f\_lseek(&fil1,0);

f\_lseek(&fil2,0);

f\_gets(buffer2,121,&fil2);

f\_gets(buffer1,38,&fil1);

play();

}

void play()

{

int m;

int l;

char btmp[20][7]={""};

LCD\_Clear(WHITE);

Show\_Str(18,80,800,32,(u8\*)buffer1,32,1);

SPeech("[g0][h0]");

SPeech(buffer1);

delay\_ms(1);

for(l=0;l<strlen(buffer2);l++)

buffer2[l]=buffer2[l];

buffer2[l]='\0';

m=strlen(buffer2)/6;

for(l=0;l<m;l++)

memcpy(btmp[l],buffer2+6\*l,6);

LCD\_WShowBraille(m,30,240,btmp);

DX\_FCBraille(btmp);

memset(btmp,'\0',sizeof(btmp));

}

//void reco()

//{

// f\_write(&fil3,&P1, sizeof(P1), &bw);

// f\_write(&fil3,&P2, sizeof(P2), &bw);

//}

u16 scan\_Bookmark (const u8\* path,int num)//获取书签的数目并且将要替换的书签名赋值给nameReplaced

{

u16 cnt=0;

FRESULT res;

DIR \*dir;

FILINFO \*fno;

u8\*fn;

char buf[10];//存储fn

char numtostr[5]="";

memset(nameReplaced,0,100);

memset(nameReplaced,0,50);

dir=(DIR\*)mymalloc(SRAMIN,sizeof(DIR));

fno =(FILINFO\*)mymalloc(SRAMIN,sizeof(FILINFO));

fno->lfsize=\_MAX\_LFN\*2+1;

fno->lfname= mymalloc(SRAMIN,fno->lfsize);

res = f\_opendir(dir, (const TCHAR\*)path);

if(res==FR\_OK&&fno->lfname!=NULL)

{

while(1)//查询总的有效文件数

{

res=f\_readdir(dir,fno); //读取目录下的一个文件

fn=(u8\*)(\*fno->lfname?fno->lfname:fno->fname);

if(res!=FR\_OK||fno->fname[0]==0)break; //错误了/到末尾了,退出

res=f\_typetell(fn);

if((res&0XFF)==0X31)//取高四位,看看是不是.C书签文件

{

strncpy(buf,fn,6);

if(num==MAXBK)

{

sprintf(numtostr, "%d", num);

if(strstr(buf,"10"))

{

strcpy(nameReplaced,"0:/");

strcat(nameReplaced,fn);

}

}

else if(num>0)

{

sprintf(numtostr, "%d", num);

if(strstr(buf,numtostr)!=NULL&&strstr(buf,"0")==NULL)

{

strcpy(nameReplaced,"0:/");

strcat(nameReplaced,fn);

}

}

cnt++;//有效文件数增加1

}

}

}

f\_closedir(dir);

myfree(SRAMIN,fno->lfname);

myfree(SRAMIN,dir);

myfree(SRAMIN,fno);

return cnt;

}

void findExt( char\* file\_name, char\* name, char\* ext)

{

char\* end;

char\* ptr = file\_name;

while (\*ptr != '\0')

{

ptr++;//移到文件名末尾

}

end = ptr;//存一下

while(\*ptr!='.')

{

ptr--;//从后往前移动到.的位置

}

memcpy(name, file\_name, (ptr-file\_name)\*sizeof(char));

memcpy(ext, ptr+1, (end - ptr ) \* sizeof(char));

}

u32 times\_Read(const u8\*path)//读 创建次数.txt(nums.txt)获取创建书签的次数

{

DIR \*dir;

FIL \*fil;

FRESULT res;

u32 cnt=0;

dir=(DIR\*)mymalloc(SRAMIN,sizeof(DIR));

fil=(FIL\*)mymalloc(SRAMIN,sizeof(FIL));

res=f\_opendir(dir,"0:");

res=f\_open(fil,(const TCHAR\*)path,FA\_READ);

if(!res)

{

while(!res)

{

f\_read(fil,&cnt,4,&br);

if (res || br < 4)

break;

}

}

f\_close(fil);

f\_closedir(dir);

myfree(SRAMIN,dir);

myfree(SRAMIN,fil);

return cnt;

}

FRESULT times\_Write(const u8\*path)//写 创建次数.txt(nums.txt)，增加创建书签的次数，写入成功后返回值为0

{

DIR \*dir;

FIL \*fil;

FRESULT res;

u32 cnt=times\_Read(path);

dir=(DIR\*)mymalloc(SRAMIN,sizeof(DIR));

fil=(FIL\*)mymalloc(SRAMIN,sizeof(FIL));

res=f\_opendir(dir,"0:");

if(!res)

{

res=f\_open(fil,(const TCHAR\*)path,FA\_READ|FA\_WRITE);

if(!res)

{

cnt=cnt+1;

do

{

res = f\_write(fil,&cnt,4,&bw);

if(res)

break;

}

while (bw <sizeof(cnt));

}

}

f\_close(fil);

f\_closedir(dir);

myfree(SRAMIN,fil);

myfree(SRAMIN,dir);

return res;

}

FRESULT times\_Init(const u8\*path)//初始化 创建次数.txt(nums.txt)，初始值为0，写入成功后返回值为0

{

DIR \*dir;

FIL \*fil;

FRESULT res;

u32 cnt=0;

dir=(DIR\*)mymalloc(SRAMIN,sizeof(DIR));

fil=(FIL\*)mymalloc(SRAMIN,sizeof(FIL));

res=f\_opendir(dir,"0:");

if(!res)

{

res=f\_open(fil,(const TCHAR\*)path,FA\_CREATE\_NEW|FA\_WRITE);

if(!res)

res=f\_write(fil,&cnt,4,&bw);

}

f\_close(fil);

f\_closedir(dir);

myfree(SRAMIN,fil);

myfree(SRAMIN,dir);

return res;

}

void Int\_To\_Str(int x, char\* Str)//整数转成字符串

{

int t;

char\* Ptr, Buf[5];

int i = 0;

Ptr = Str;

if (x < 10) // 当整数小于10，转换为0x格式

{

\*Ptr++ = x + 0x30;

}

else

{

while (x > 0)

{

t = x % 10;

x = x / 10;

Buf[i++] = t + 0x30; // 通过计算把数字编成ASCII码形式

}

i--;

for (; i >= 0; i--) // 将得到的字符串倒序

{

\*(Ptr++) = Buf[i];

}

}

\*Ptr = '\0';

}

FRESULT create\_New(const u8\*path,const u8\*path2,UINT nums)//path是文件的路径，path2是写入的字符数组,先写text再写data最后写文本存在的路径

{

DIR \*dir;

FIL \*fil;

FRESULT res;

dir=(DIR\*)mymalloc(SRAMIN,sizeof(DIR));

fil=(FIL\*)mymalloc(SRAMIN,sizeof(FIL));

res=f\_opendir(dir,"0:");

if(!res)

{

res=f\_open(fil,(const TCHAR\*)path,FA\_READ|FA\_WRITE|FA\_CREATE\_ALWAYS);//可以新建写入，也可以已有文件覆盖写入

res = f\_write(fil,&P1,sizeof(P1),&bw);

res = f\_write(fil,&P2,sizeof(P2),&bw);

if(!res)

{

do

{

res = f\_write(fil,path2,nums,&bw);

if(res)

break;

}

while (bw <nums);

}

}

f\_close(fil);

f\_closedir(dir);

myfree(SRAMIN,fil);

myfree(SRAMIN,dir);

return res;

}

u32 path\_Change(u32 cnt1,u16 cnt2)

{

char idx[MAXSIZE]="";

char bk[3\*MAXSIZE]="书签";

char \*str=idx;

memset(path\_c,0,6\*MAXSIZE);

strcpy((char\*)path\_c,"0:/");

if(cnt2<=MAXBK-1)

{

cnt2++;

Int\_To\_Str((int)cnt2, str);

strcat(bk,str);

strcat(bk," ");

strcat(path\_c,bk);

return 0;

}

else if(cnt2==MAXBK)

{

if ((cnt1 + 1) % (MAXBK) == 0)

{

Int\_To\_Str(MAXBK, str);

strcat(bk, str);

strcat(bk," ");

strcat(path\_c, bk);

return MAXBK;

}

else

{

cnt1 = (cnt1 + 1) % (MAXBK);

Int\_To\_Str((int)cnt1, str);

strcat(bk, str);

strcat(bk," ");

strcat(path\_c, bk);

return cnt1;

}

}

}

int display\_Bookmark()//遍历出所有的.c文件并且按照书签1~10的顺序排序

{

FRESULT res;

int i=0;

int line=0;

int num=0;//书签个数

DIR \*bkdir;

FILINFO \*bkfileinfo;//文件信息

u8 \*fn; //长文件名

char processedname[50]="";//文件名不带扩展名

char extension[3]="";

memset(processedname,0,50);

memset(&bookmarkname[0][0],0,sizeof(char)\*10\*50);

LCD\_Clear(WHITE);

bkdir=(DIR\*)mymalloc(SRAMIN,sizeof(DIR));

bkfileinfo =(FILINFO\*)mymalloc(SRAMIN,sizeof(FILINFO));

bkfileinfo->lfsize=\_MAX\_LFN\*2+1;

bkfileinfo->lfname= mymalloc(SRAMIN,bkfileinfo->lfsize);

demo=2;

bkplace=0;//记录读到书签所处目录的位置

highlight(0);

memset(path,0,200);

strcpy((char\*)path,"0:");

}

res=f\_opendir(bkdir,(const char\*)path);

if(res==FR\_OK)

{

while(1)//全部查询一遍

{

res=f\_readdir(bkdir,bkfileinfo);//读取目录下的一个文件

if(res!=FR\_OK||bkfileinfo->fname[0]==0)break; //错误了/到末尾了,退出

fn=(u8\*)(\*bkfileinfo->lfname?bkfileinfo->lfname:bkfileinfo->fname);

res=f\_typetell(fn);

if((res&0XFF)==0X31)//是否为书签文件

{

findExt( (char\*)fn, processedname, extension);

strcpy((u8\*)bookmarkname[line],processedname);

line++;

memset(processedname,0,50);

}

}

selectSort(bookmarkname, line);

for(i=0;i<line;i++)

Show\_Str(60,32+42\*i,500,42,(u8\*)bookmarkname[i],32,0);

memset(bkmidname,'\0',sizeof(bkmidname));

strcpy(bkmidname,bookmarkname[0]);

SPeech(bkmidname);

}

num=line;

f\_closedir(bkdir);

myfree(SRAMIN,bkfileinfo->lfname);

myfree(SRAMIN,bkdir);

myfree(SRAMIN,bkfileinfo);

return num;

}

FRESULT bookmark\_Read(const u8\*path,u8\*buf\_Read)//读书签内容，分为text的offset，data的offset，和书签记录的地址

{

DIR \*dir;

FIL \*fil;

FRESULT res;

dir=(DIR\*)mymalloc(SRAMIN,sizeof(DIR));

fil=(FIL\*)mymalloc(SRAMIN,sizeof(FIL));

memset(buf\_Read,0,100);

P3=0;

P4=0;

res=f\_opendir(dir,"0:");

if(!res)

{

res=f\_open(fil,(const TCHAR\*)path,FA\_READ);//可以新建写入，也可以已有文件覆盖写入

res = f\_read(fil,&P3,sizeof(P3),&br);

res = f\_read(fil,&P4,sizeof(P4),&br);

printf("第一个数字%d\r\n",(int)P3);

printf("第二个数字%d\r\n",(int)P4);

for(;;)

{

res = f\_read(fil,buf\_Read,100,&br);

printf("%s\r\n",buf\_Read);

if(res||br<100)break;

}

}

f\_close(fil);

f\_closedir(dir);

myfree(SRAMIN,fil);

myfree(SRAMIN,dir);

return res;

}

void select\_Bookmark(u16 index)

/\*i从0开始，和数组元素下标的起始位置相同，找到下标为index的元素，打开并且调用打开书签的函数，

将路径存到path中，方便打开path中的text.txt和data.txt\*/

{

u8\*pathp=0;

int i=0;

u8 red=0;

u8 ret=0;

LCD\_Clear(WHITE);

pathp=mymalloc(SRAMIN,200);

pathp[0]=0;

strcat((char\*)pathp,"0:");

memset(path,0,200);

i=index;

strcat((char\*)pathp,"/");

strcat((char\*)pathp,(char\*)bookmarkname[i]);

strcat((char\*)pathp,".c");

// f\_closedir(dir);

////printf("%s\r\n",pathp);

i=0;

res=bookmark\_Read(pathp,buf\_Read);

if(res==0)

strcpy((char\*)path,(char\*)buf\_Read);

myfree(SRAMIN,pathp);

if(demo==2)

{

demo=0;

pathtxt=mymalloc(SRAMIN,511);

pathdata=mymalloc(SRAMIN,511);

pathtxt[0]=pathdata[0]=0;

mymemset(pathtxt,0,511);

mymemset(pathdata,0,511);

strcat((char\*)pathtxt,(char\*)path);

strcat((char\*)pathdata,(char\*)path);

strcat((char\*)pathtxt,"/text.txt");

strcat((char\*)pathdata,"/data.txt");

ret=f\_open(&fil1,(char\*)pathtxt,FA\_READ);

red=f\_open(&fil2,(char\*)pathdata,FA\_READ);

if(red==FR\_OK&&ret==FR\_OK)

{

f\_lseek(&fil1,P3);

f\_lseek(&fil2,P4);

f\_gets(buffer2,121,&fil2);

f\_gets(buffer1,38,&fil1);

play();

P3=0;

P4=0;

}

}

}

void move\_Bookmarkcursor\_Next(u16 n)//选择下一个书签

{

u16 cursor;

SPeech("[g0][h0]");

bkplace=(bkplace+1)%n;

cursor=bkplace;

highlight(cursor);

memset(bkmidname,'\0',sizeof(bkmidname));

delay\_ms(1);

strcpy(bkmidname,bookmarkname[bkplace]);

SPeech(bkmidname);

}

void move\_Bookmarkcursor\_Last(u16 n)//选择上一个书签

{

u16 cursor;

SPeech("[g0][h0]");

if((bkplace-1)<0)

bkplace=n-1;

else

bkplace=(bkplace-1)%n;

cursor=bkplace;

highlight(cursor);

memset(bkmidname,'\0',sizeof(bkmidname));

delay\_ms(1);

strcpy(bkmidname,bookmarkname[bkplace]);

SPeech(bkmidname);

}

void selectSort(char(\*pt)[50], int n)

{

int i, j, k;

char tmp[50]; //临时字符数组

for (i = 0; i < n - 1; i++) //遍历所有未排序的字符串的趟数

{

k = i; //变量k存放当前最小字符串的下标

for (j = i + 1; j < n; j++) //遍历下标从(i+1)~(n-1)的所有字符串

if (strcmp(pt[k], pt[j]) > 0) //字符串进行两两大小比较,如果>0,则执行if语句体

k = j; //将当前最小的字符串的数组下标赋值给变量k

if (k != i) //如果k不等于i,则执行if语句体

{ //两个字符串进行交换操作

strcpy(tmp, pt[k]);

strcpy(pt[k], pt[i]);

strcpy(pt[i], tmp);

}

}

if(n==MAXBK)

{

strcpy(tmp, pt[0]);

strcpy(pt[0], pt[1]);

strcpy(pt[1], tmp);

for (i = 0; i < n-1; i++)

{

strcpy(tmp, pt[i]);

strcpy(pt[i], pt[i+1]);

strcpy(pt[i+1], tmp);

}

}

}

#include "suocun.h"

#define suo1 PFout(0) //独立外设

#define suo2 PFout(1)

#define suo3 PFout(2)

#define suo4 PFout(3)

#define suo5 PFout(4)

#define suo6 PFout(5)

#define suo7 PFout(6)

#define suo8 PFout(7)

#define suo9 PFout(12)

#define suo10 PFout(13)

#define suo11 PFout(14)

#define suo12 PFout(15)

#define suo13 PGout(1)

#define suo14 PGout(2)

#define suo15 PGout(3)

void DXdis(u16 dx[15]);

void SUOCUN\_init(void)

{

GPIO\_InitTypeDef GPIO\_InitStructure;

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOF|RCC\_APB2Periph\_GPIOC|RCC\_APB2Periph\_GPIOG, ENABLE);

GPIO\_InitStructure.GPIO\_Pin =GPIO\_Pin\_0|GPIO\_Pin\_1|GPIO\_Pin\_2|GPIO\_Pin\_3|GPIO\_Pin\_4|GPIO\_Pin\_5|GPIO\_Pin\_6|GPIO\_Pin\_7|GPIO\_Pin\_12|GPIO\_Pin\_13|GPIO\_Pin\_14|GPIO\_Pin\_15;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_Out\_PP; //推挽输出

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz; //IO口速度为50MHz

GPIO\_Init(GPIOF, &GPIO\_InitStructure); //根据设定参数初始化GPIOF 0 1 2 3 4 5 6 7 12 13 14 15共12个IO口

GPIO\_InitStructure.GPIO\_Pin=GPIO\_Pin\_1|GPIO\_Pin\_2|GPIO\_Pin\_3;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_Out\_PP; //推挽输出

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz; //IO口速度为50MHz

GPIO\_Init(GPIOG, &GPIO\_InitStructure); //根据设定参数初始化GPIOG 1 2 3共3个IO口

suo1=0;//置零关闭锁存器

suo2=0;

suo3=0;

suo4=0;

suo5=0;

suo6=0;

suo7=0;

suo8=0;

suo9=0;

suo10=0;

suo11=0;

suo12=0;

suo13=0;

suo14=0;

suo15=0;

GPIO\_InitStructure.GPIO\_Pin=GPIO\_Pin\_0|GPIO\_Pin\_1|GPIO\_Pin\_2|GPIO\_Pin\_3|GPIO\_Pin\_4|GPIO\_Pin\_5|GPIO\_Pin\_6|GPIO\_Pin\_7;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_Out\_PP; //推挽输出

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz; //IO口速度为50MHz

GPIO\_Init(GPIOC,&GPIO\_InitStructure); //根据设定参数初始化GPIOC 0 1 2 3 4 5 6 7共8个数据口

GPIO\_ResetBits(GPIOC,GPIO\_Pin\_0|GPIO\_Pin\_1|GPIO\_Pin\_2|GPIO\_Pin\_3|GPIO\_Pin\_4|GPIO\_Pin\_5|GPIO\_Pin\_6|GPIO\_Pin\_7);

}

void DX\_ShowBraille(char tbtmp[10][13]){

u8 count\_s;//计数器

u16 dx[15]; //

char gdtmp[9];

//char stbtmp[120];//过渡数组gd

char ts[2][6];

int i,j,a,b,m,n,x1,x2,y,l,k;

for(i=0;i<5;i++)

for(j=0;j<12;j++)

{

m=tbtmp[i][j];

tbtmp[i][j]=tbtmp[9-i][11-j];

tbtmp[9-i][11-j]=m;

//stbtmp[i\*12+j]=tbtmp[i][j];

}

for(count\_s=0;count\_s<10;count\_s+=2){ //2个字24位为一个操作周期，5个周期10个字

for(i=count\_s;i<count\_s+2;i++)

{

n=tbtmp[i][2];

tbtmp[i][2]=tbtmp[i][0];

tbtmp[i][0]=n;

x1=tbtmp[i][5];

tbtmp[i][5]=tbtmp[i][3];

tbtmp[i][3]=x1;

y=tbtmp[i][8];

tbtmp[i][8]=tbtmp[i][6];

tbtmp[i][6]=y;

k=tbtmp[i][11];

tbtmp[i][11]=tbtmp[i][9];

tbtmp[i][9]=k;

strncpy(ts[0],tbtmp[i],6);

for(j=0;j<3;j++)

{

a=ts[0][j];

ts[0][j]=ts[0][5-j];

ts[0][5-j]=a;

}

strncpy(tbtmp[i],ts[0],6);

strncpy(ts[1],tbtmp[i]+6,6);

for(j=0;j<3;j++)

{

b=ts[1][j];

ts[1][j]=ts[1][5-j];

ts[1][5-j]=b;

}

//strncpy(tbtmp[i],ts[0],6);

strncpy(tbtmp[i]+6,ts[1],6);//以前错在strncat(tbtmp[i],ts[1],6)此步相当于没有，

//tbtmp[i][13]='\0';

//strncat(tbtmp[i],ts[1],6);

tbtmp[i]==tbtmp[i];

}

strncpy(gdtmp,tbtmp[count\_s],8);//一个周期内的第一组转换，复制btmp[count\_s]的前8位,完成第一组数据输出点采集 //据说memcpy效率高，可复制中间包含'\0'的数据，strncpy好了吗只能复制字符串，strncpy，memcpy有可能出现内存重叠情况，程序出错可以考虑修改为memmove

gdtmp[8]='\0';//手动补\0做字符串结束标记符号,保证正确识别 //memcpy(gdtmp,btmp[count\_s],8);

reverse(gdtmp);//字符串倒转，用于计算机存储正确点位序列 盲文点列中1~8（7，8为下一方的1，2）排列书写的顺序(低位在前高位在后)与对应D0~D7的赋值在数位上（从高到底）相反，所以倒转序列适应锁存器输出

dx[3\*count\_s/2]=binary(gdtmp);//二进制字符串转十进制整型数，存于点显数组第一个元素中，用于I/O口赋值

memset(gdtmp,'\0',sizeof(gdtmp));//完成一次存储操作后，将过渡数组清零。以采集第二组

strncpy(gdtmp,tbtmp[count\_s]+8,5);//第二组数据输出点采集，先复制第一组盲文点列的后5位已包含'\0'

strncat(gdtmp,tbtmp[count\_s+1],4);//再添加第二组盲文点列的前4个点序。完成第二组数据输出点的采集。

gdtmp[8]='\0';//手动补'\0',虽然一般情况下strncat会在目标字符串末尾补\0

reverse(gdtmp);//2

dx[3\*count\_s/2+1]=binary(gdtmp);//转十进制整数，存于点显数组的第二个元素中。

memset(gdtmp,'\0',sizeof(gdtmp));//完成第二次存储操作，过渡数组清零以采集第三组

strcpy(gdtmp,tbtmp[count\_s+1]+4);//复制第二组盲文点列后9位(包含\0)点列。 灵活运用strcpy中的所要拷贝的字符串的首地址(地址btmp[count\_s+1]+4为第二组盲文点列第4位(从0位开始计数)的地址，即后9位(包含\0)点列的起始地址)，并已补上\0。

reverse(gdtmp);//3

dx[3\*count\_s/2+2]=binary(gdtmp);//转十进制整数，存于点显数组的第三个元素中。

//至此一个周期内的三个数据输出点阵转换采集完毕，并转为十进制数值存于点显数组中。

}

//2.开始输出

//逐个选通，选通下一个前先关闭当前锁存器，

//更新前清空数据输出口，置0。

DXdis(dx);

}

void DX\_FCBraille(char tbtmp[20][7]){

u8 count\_s,i=0;

u16 dx[15];

char gdtmp[9], ts[2][6];

int i1,i2 ,j,m,n,x,y,k,a,b,t;

for(i1=0;i1<10;i1++)

{for(i2=0;i2<6;i2++)

{x=tbtmp[i1][i2];

tbtmp[i1][i2]=tbtmp[19-i1][i2];

tbtmp[19-i1][i2]=x;

}

}

for(count\_s=0;count\_s<20;count\_s+=4){

for(k=count\_s;k<count\_s+4;k++)

{

n=tbtmp[k][2];

tbtmp[k][2]=tbtmp[k][0];

tbtmp[k][0]=n;

m=tbtmp[k][5];

tbtmp[k][5]=tbtmp[k][3];

tbtmp[k][3]=m;

// t=tbtmp[k][5];

// tbtmp[k][5]=tbtmp[k][0];

// tbtmp[k][0]=t;

//

}

strncpy(gdtmp,tbtmp[count\_s],6);

strncat(gdtmp,tbtmp[count\_s+1],2);

gdtmp[8]='\0';

reverse(gdtmp);

dx[i]=binary(gdtmp);

memset(gdtmp,'\0',sizeof(gdtmp));

strncpy(gdtmp,tbtmp[count\_s+1]+2,4);

strncat(gdtmp,tbtmp[count\_s+2],4);

gdtmp[8]='\0';

reverse(gdtmp);

dx[i+1]=binary(gdtmp);

memset(gdtmp,'\0',sizeof(gdtmp));

strncpy(gdtmp,tbtmp[count\_s+2]+4,2);

strncat(gdtmp,tbtmp[count\_s+3],6);

gdtmp[8]='\0';

reverse(gdtmp);

dx[i+2]=binary(gdtmp);

memset(gdtmp,'\0',sizeof(gdtmp));

i+=3;

}

DXdis(dx);

}

void DXdis(u16 dx[15]){//点显器整屏显示

GPIO\_Write(GPIOC,0x00);

suo1=1;

GPIO\_Write(GPIOC,dx[0]);

suo1=0;

GPIO\_Write(GPIOC,0x00);

suo2=1;

GPIO\_Write(GPIOC,dx[1]);

suo2=0;

GPIO\_Write(GPIOC,0x00);

suo3=1;

GPIO\_Write(GPIOC,dx[2]);

suo3=0;

GPIO\_Write(GPIOC,0x00);

suo4=1;

GPIO\_Write(GPIOC,dx[3]);

suo4=0;

GPIO\_Write(GPIOC,0x00);

suo5=1;

GPIO\_Write(GPIOC,dx[4]);

suo5=0;

GPIO\_Write(GPIOC,0x00);

suo6=1;

GPIO\_Write(GPIOC,dx[5]);

suo6=0;

GPIO\_Write(GPIOC,0x00);

suo7=1;

GPIO\_Write(GPIOC,dx[6]);

suo7=0;

GPIO\_Write(GPIOC,0x00);

suo8=1;

GPIO\_Write(GPIOC,dx[7]);

suo8=0;

GPIO\_Write(GPIOC,0x00);

suo9=1;

GPIO\_Write(GPIOC,dx[8]);

suo9=0;

GPIO\_Write(GPIOC,0x00);

suo10=1;

GPIO\_Write(GPIOC,dx[9]);

suo10=0;

GPIO\_Write(GPIOC,0x00);

suo11=1;

GPIO\_Write(GPIOC,dx[10]);

suo11=0;

GPIO\_Write(GPIOC,0x00);

suo12=1;

GPIO\_Write(GPIOC,dx[11]);

suo12=0;

GPIO\_Write(GPIOC,0x00);

suo13=1;

GPIO\_Write(GPIOC,dx[12]);

suo13=0;

GPIO\_Write(GPIOC,0x00);

suo14=1;

GPIO\_Write(GPIOC,dx[13]);

suo14=0;

GPIO\_Write(GPIOC,0x00);

suo15=1;

GPIO\_Write(GPIOC,dx[14]);

suo15=0;

GPIO\_Write(GPIOC,0x00);

}

//满空方测试显示

void dis(char b){

GPIO\_Write(GPIOF,0xf0ff);//全打开锁存器

suo13=1;

suo14=1;

suo15=1;

if(b){//全上

GPIOC->BSRR=0xff;

delay\_ms(100);

}

else{ //全下

GPIOC->BRR=0xff;//置零

delay\_ms(100);

GPIOC->BRR=0xff;

}

GPIO\_Write(GPIOF,0x00);//关闭锁存器

suo13=0;

suo14=0;

suo15=0;

}

void LCD\_WShowBraille(int n,u16 x,u16 y,char braille[][7]){

char k,l;//K:字序。10个字，0-9；l：字的点序，0-12，12个点表示两方一个字

int x0,y0;

for(k=0;k<n;k++)

for(l=0;l<6;l++)

{

//x0=x+14\*(l/3)+2\*(l/6)+74\*k;

//y0=y+150\*(k/10)+25\*(l%3);

x0=x+16\*(l/3)+32\*k+5\*k;

y0=y+25\*(l%3)+150\*(k/n);

//x0=x+16\*(l/3)+32\*k+5\*k;

//y0=y+25\*(l%3)+150\*(k/20);

if(braille[k][l]=='1')

gui\_fill\_circle(x0,y0,3,BLUE);

else LCD\_Draw\_Circle(x0,y0,2);

}

void gui\_fill\_circle(u16 x0,u16 y0,u16 r,u16 color)

{

u32 i;

u32 imax = ((u32)r\*707)/1000+1;

u32 sqmax = (u32)r\*(u32)r+(u32)r/2;

u32 x=r;

gui\_draw\_hline(x0-r,y0,2\*r,color);

for (i=1;i<=imax;i++)

{

if ((i\*i+x\*x)>sqmax)// draw lines from outside

{

if (x>imax)

{

gui\_draw\_hline (x0-i+1,y0+x,2\*(i-1),color);

gui\_draw\_hline (x0-i+1,y0-x,2\*(i-1),color);

}

x--;

}

// draw lines from inside (center)

gui\_draw\_hline(x0-x,y0+i,2\*x,color);

gui\_draw\_hline(x0-x,y0-i,2\*x,color);

}

}

void gui\_draw\_hline(u16 x0,u16 y0,u16 len,u16 color)

{

if(len==0)return;

LCD\_Fill(x0,y0,x0+len-1,y0,color);

}

void PrintCom(uint8\_t \*DAT,uint8\_t len)

{

uint8\_t i;

for(i=0;i<len;i++)

{

USART\_SendData(USART3,\*DAT++);

delay\_ms(2);

while(USART\_GetFlagStatus(USART3,USART\_FLAG\_TC)==RESET);

}

}

void SPeech(char \*buf)

{

unsigned char len[100];

unsigned int buflen;

buflen=strlen(buf);

//固定配置信息,包头

len[0]=0xFD;

len[1]=0x00;

len[2]=buflen+2;

len[3]=0x01;

len[4]=0x01;

//发送数据

memcpy(&len[5],buf,buflen);

PrintCom(len,5+buflen);

// int i;

// for(i=0;i<5+buflen;i++)

// {

// USART\_SendData(USART3,\*len++);

// while(USART\_GetFlagStatus(USART3\_FLAG\_TC)==RESET);

}