Auxiliary brake system

Morio Kikuchi

Abstract：

To brake system, we add an auxiliary brake system in which paddle at the front of steering wheel is an input device.

1. Paddle type brake system

Today, pedal type brake system is adopted in almost all of cars. Paddle type brake sytem is restricted within cars for disabled persons. Lately, automatic driving system is being put to practical use. Especially, automatic driving system has becomed popular in sudden stop. How is braking except sudden stop? For example, there is an event that the control of car boby becomes impossible by hard braking on snow-covered road, icy road. Generally, it seems that the situation is occurred by excessive stepping on a brake pedal by human's judgement.

So, we try considering introducing of technology of automatic driving system into braking except sudden stop as well. Figure 1 shows an arrangement of brake paddle L, R.

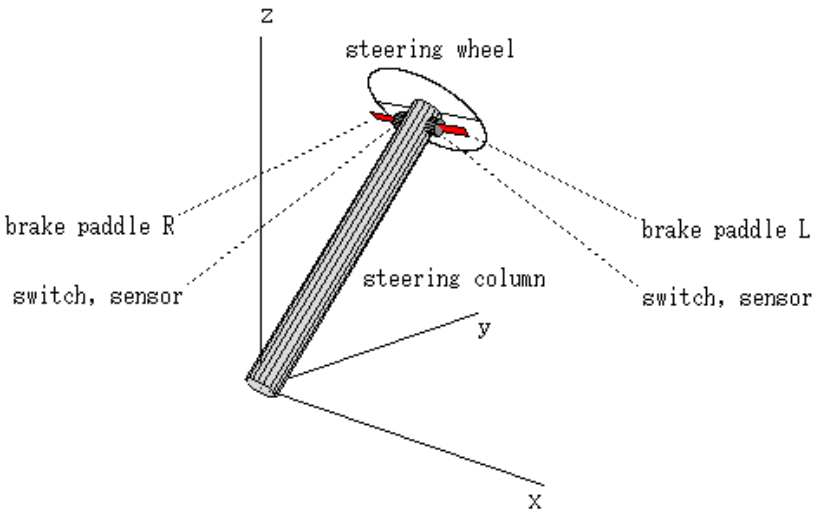

Figure 1

If brake paddle is operated, switch or sensor functions and brake works. Switch and sensor are used properly like the following:

・switch：most suitable braking by computer
・sensor：braking in which computer is not used

In the former, if switch is turned on operating brake paddle, most suitable braking according to speed and inter-vehicular distance for example is done. Brake paddle L, R is used properly for high speed and low speed or for dry road face and wet road face, snow-covered road, icy road. This is a kind of semi-automatic brake system. In the latter, the operational amount of brake paddle is reflected to the working amount of brake not as on and off but as an amount which varies smoothly.

2. Hard braking

We try considering doing hard braking in paddle type brake sytem. We devise a way of being able to

hard braking shifting the system to sensor braking if brake paddle L, R are operated at the same time. According to whether brake paddle can rotate with steering wheel, it becomes like the following:

- Rotation is possible : Hard braking is possible if the rotational angle of steering wheel is up to medium extent
- Rotation is impossible : Hard braking is possible if the rotational angle of steering wheel is small

However, because holding of steering wheel becomes incomplete if brake paddle L, R are operated at the same time, the driver must be proficient at driving to some extent.

****

補助ブレーキシステム

菊池盛雄

アブストラクト：
　ステアリングホイールの前のパドルを入力装置とする補助ブレーキシステムをブレーキシステムに追加します。

1. パドル式ブレーキシステム

　現在、自動車のほとんどはペダル式ブレーキシステムを採用しています。パドル式ブレーキシステムは身障者用の自動車などに限られています。最近は自動運転システムが実用化されつつあります。特に自動ブレーキシステムは急停止がポピュラーになっています。急停止以外の制動に関してはどうでしょうか。たとえば、雪道、アイスバーンにおいて急ブレーキによって車体のコントロールが不能になる、という事象があります。一般には人間の判断でブレーキペダルを踏みすぎることによってこのようになると思われます。

　そこで、急停止以外の制動にも自動運転システムの技術を取り入れることを考えてみます。図1はブレーキパドルL、Rの配置を示しています。

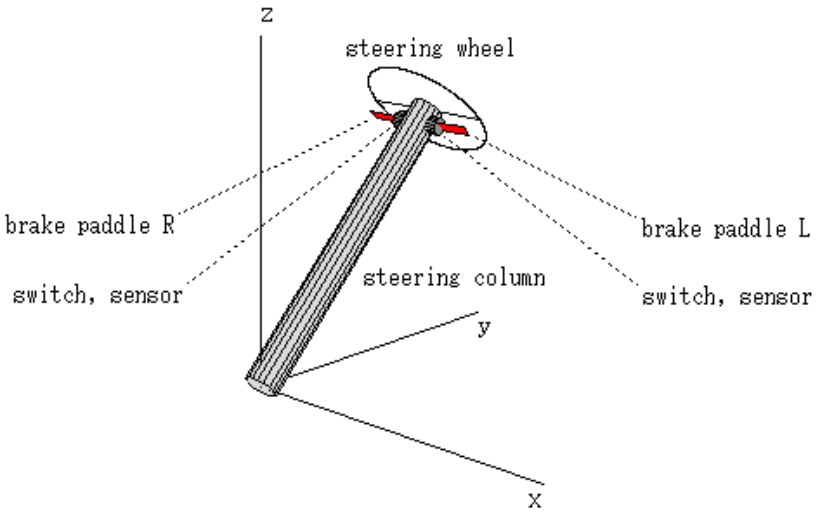

図1

ブレーキパドルを操作することによって、スイッチ、センサーが機能し、ブレーキが作動します。スイッチ、センサーは以下のような使い分けをします。

・スイッチ：コンピューターによる最適制動
・センサー：コンピューターを介さない制動

前者では、ブレーキパドルを操作してスイッチがonになれば、たとえば速度、車間距離に応じた最適な制動を行います。ブレーキパドルL、Rはたとえば高速用、低速用または乾燥した路面用、濡れた路面・雪道・アイスバーン用に使い分けます。これは半自動ブレーキシステムの一種です。後者では、ブレーキパドルの操作量はon、offではなく滑らかに変化する量としてブレーキの作動量に反映されます。

2. 急ブレーキ

パドル式ブレーキシステムで急ブレーキを行うことを考えてみます。ブレーキパドルL、Rを同時に操作すればコンピューターを介さないセンサー制動に移行させ、急ブレーキを行うことができるようにします。ブレーキパドルがステアリングホイールと共に回転できるかできないかで以下のようになります。

・回転できる　：ステアリングホイールの回転角が中程度までなら急ブレーキが可能
・回転できない：ステアリングホイールの回転角が小さければ急ブレーキが可能

ただし、ブレーキパドルL、Rを同時に操作すればステアリングホイールの保持が不完全になるので、ドライバーはある程度運転に習熟していなければなりません。

****

List 1:ustick.c
List 2:ustick.cfg

```
/* usb stick searcher                                                    */
/* ustick.c                                                              */
/* 2019 Morio Kikuchi                                                    */
/* WINDOW SYSTEM:Windows                                                 */

#include <windows.h>
#include <stdio.h>

#define INVERT SRCINVERT
#define GKS GetKeyState
#define GKS_ GetKeyState

#define VGACOLORS 16
#define ASIZE (MAX_PATH*2)
#define UNITDX (UDX+ds)
#define UNITDY (UDY+0)

char charflag=1,function=2,pflag=1;
int x_usb,y_usb,usb,f2code,ds;
int XRESO,YRESO,WB,UDX,UDY,DX_FRAME,DY_FRAME,DY_CAPTION,RSHIFT,DSHIFT;
long cfgmax,kmax;

unsigned char *pcfg,p[ASIZE];
int DS[][2]={{640,480}};

typedef struct {
/*int*/unsigned char red,green,blue;} srgb;
srgb irgb[VGACOLORS];
typedef struct {
unsigned long back_;int back,fore;} bf;
bf bfset[]={{WHITENESS,15,0},{BLACKNESS,0,15}};

HINSTANCE hinstance;
HWND hwnd;
HDC hdcdisplay,hdctmp1;
HBITMAP hbitmap1;
HFONT hfont;

unsigned char *search_cfg(unsigned char *),*while_puts_cfg(void);
void setup(char),printf_(double,double,double),keydowns_f2(void),initsysfont(int,int),
     bitblt(char,int,int,int,int,int,int),sentence(char,int,int,unsigned char *,int),
```

```
     setstccolor(int),use_subroop(void),restore_in_PAINT(void),
     cleardevice_(char,int,int,int,int),subroop(void),ustick(int,int),
     initpalette(void),closegraph_(void),kbhit_(void),beep(long);
int initgraph_(void);
double ff_fc(double);

COLORREF PALETTE(int color);
LRESULT CALLBACK wndproc_by_kbhit_(HWND,UINT,WPARAM,LPARAM);

int wndproc_filer(HWND,UINT,WPARAM,LPARAM);


/*int main(int argc,unsigned char **argv)*/int APIENTRY WinMain(HINSTANCE hinstance_,HINSTAN
{
int drivesmax,beepflag=1,len;
char home[ASIZE];
LPTSTR path;
MSG msg;
FILE *fp;
                                    /* main() */
/*hinstance=GetModuleHandle(NULL);*/
                                    /* WinMain */
hinstance=hinstance_;

WB=0;
setup(0);



path=GetEnvironmentStrings();

while(*path){
if(strnicmp("Path=",path,5)==0) break;

path+=lstrlen(path)+1;
}

len=strlen(path);
memcpy(&home[0],&path[5],len-5);
home[len-5]='\0';

if(access("\\dos\\ustick.cfg",0)==0)
strcpy(home,"\\dos\\ustick.cfg");
else if(access(".\\ustick.cfg",0)==0)
strcpy(home,".\\ustick.cfg");
else
```

```
strcpy(home,search_cfg(home));      /* (home):paths */

/*fp=fopen("txt","w+b");
fprintf(fp,"%s\n",home);
fprintf(fp,"%d\n",MAX_PATH);
fclose(fp);*/

if((fp=fopen(home,"rb"))==NULL){
start:

drivesmax=26-2;
beepflag=1;
x_usb=0;
y_usb=0;
UDX=8;
UDY=16;
RSHIFT=UDX/2;
DSHIFT=ff_fc(UDX/3.);
}
else{
fseek(fp,0L,SEEK_END);
cfgmax=ftell(fp)-1;
pcfg=(unsigned char *)malloc(/*sizeof(unsigned char)*(*/cfgmax+1/*)*/);
if(pcfg==NULL) {fclose(fp);goto start;}

fseek(fp,0L,SEEK_SET);
fread(pcfg,1,cfgmax+1,fp);
fclose(fp);

if((drivesmax=read_cfg(0))<0) drivesmax=26-2;
if((beepflag=read_cfg(1))<0) beepflag=1;
if((x_usb=read_cfg(2))<0) x_usb=0;
else                      x_usb=min(x_usb,DS[0][0]);
if((y_usb=read_cfg(3))<0) y_usb=0;
else                      y_usb=min(y_usb,DS[0][1]);
if((UDX=read_cfg(4))<0) UDX=8;
else                    UDX=max(UDX,4);
if((UDY=read_cfg(5))<0) UDY=16;
else                    UDY=max(UDY,8);
if((RSHIFT=read_cfg(6))<0) RSHIFT=UDX/2;
if((DSHIFT=read_cfg(7))<0) DSHIFT=ff_fc(UDX/3.);
}

reinfo:
ustick(drivesmax,beepflag);
```

```
charflag=1;
if(f2code==1) {f2code=-1;goto reinfo;}  /* R */

return 0;
}/** main **/


unsigned char *search_cfg(unsigned char *home)
{
strcpy(p,home);

kmax=strlen(p)-1;
strcpy(home,while_puts_cfg());

return home;
}/** search_cfg **/


double ff_fc(double val)
{
if(val>0){
if(val-(int)val>=0.5) val=(int)val+1;
}
else if(val<0){
if((int)val-val>=0.5) val=(int)val-1;
}

return val;
}/** ff_fc **/


void ustick(int drivesmax,int beepflag)
{
char i;
int drives,dtype,xsum,ysum;
char str[10],str_[10],old[MAX_PATH+1];

getcwd(old,MAX_PATH);

usb=0;
/*drives=0;*/

for(i=0;i<drivesmax;i++){
sprintf(str,"%c",i+'C');
strcat(str,":\\");
if(chdir(str)==0){
```

```
dtype=GetDriveType(str);
if(dtype==DRIVE_REMOVABLE){

usb++;
}

/*drives++;*/
}
}



if(initgraph_()==1) return;
cleardevice_(1,0,0,XRESO,YRESO);



setstccolor(bfset[WB].fore);
usb=0;
drives=0;
xsum=0;ysum=0;

for(i=0;i<drivesmax;i++){
sprintf(str,"%c",i+'C');
strcat(str,":\\");
if(chdir(str)==0){
dtype=GetDriveType(str);
if(dtype==DRIVE_REMOVABLE){
str[2]='\0';
if(usb>0){
sprintf(str_,",%s",str);
strcpy(str,str_);
}
strcpy(p,str);
kmax=strlen(p)-1;
while_puts_show_(xsum,ysum);

usb++;
xsum+=strlen(str)/**UNITDX*/;
}

drives++;
}
}

/*if(usb)
```

```
cleardevice_(1,usb*UNITDX*3-UNITDX,0,UNITDX,UNITDY*2);*/
chdir(old);

printf_(drives,0,1);



if(beepflag>0){
Beep(444,200);
Beep(666,200);
Beep(888,200);
}

use_subroop();                      /* breaks if R or Esc pressed. */

closegraph_();
}/** ustick **/


char gettype(long k)
{
char type;
unsigned char s[1],s_[1];

s[0]=p[k];
s_[0]=p[k+1];

/* code page:932(SJIS)-> */
if(s[0]>=0x20 && s[0]<=0x7e) type=0;                  /* word */
else if(s[0]>=0xa1 && s[0]<=0xdf) type=0;
else if(k==kmax) type=0;
else if(s[0]==0x0a) type=0;
else if(s[0]==0x09) type=1;
else if((s[0]>0x00 && s[0]<0x20) || s[0]==0x7f) type=2;
else if(s[0]<=0x7f) type=-1;                          /* others */
else type=3;                                        /* Double Byte Character */
/* <-code page:932(SJIS) */
/*else if(s1>=0x81 && s1<=0xfc && (s1<=0x9f || s1>=0xe0) &&
        s2>=0x40 && s2<=0xfc && s2!=0x7f) type=3;*/  /* Double Byte Character */

return type;
}/** gettype **/


unsigned char *while_puts_cfg(void)
{
```

```
char type;
long k,k1,k2;
unsigned char s[1];
static unsigned char jis[MAX_PATH/2];

k=0;k1=0;

while(1){
s[0]=p[k];
type=gettype(k);

if(type<=2){
if(s[0]==';' || k==kmax){           /* k==kmax:'\0' */
k2=k;
memcpy(&jis[0],&p[k1],k2-k1);
jis[k2-k1]='\0';
strcat(jis,"\\ustick.cfg");
if(access(jis,0)==0) break;
else strcpy(jis,"");

if(k==kmax) break;

k1=k+1;
}

k++;
}/**if(type)**/
else if(type==3){
if(/*k==kmax-1 || */k>=kmax-1) break;              /* ? */

k+=2;
}/**else if(type)**/
else{
}/**else(type)**/

}

return jis;
}/** *while_puts_cfg **/


int while_puts_show_(int xlast,int ylast)
{
char type;
int i,j,dx,dy;
long k;
```

```
unsigned char s[1];
unsigned char jis[2];

i=xlast;j=ylast;
k=0;

while(1){
s[0]=p[k];
type=gettype(k);

if(type<=2){
dx=(i+0)*UNITDX;dy=(j+0)*UNITDY;
sentence(1,dx,dy,s,1);

if(k==kmax) break;

k++;

i++;
}/**if(type)**/
else if(type==3){
jis[0]=p[k];
jis[1]=p[k+1];

dx=(i+0)*UNITDX;dy=(j+0)*UNITDY;
sentence(1,dx,dy,jis,2);

if(/*k==kmax-1 || */k==kmax) break;              /* ? */

k+=2;
}/**else if(type)**/
else{
}/**else(type)**/

}

return i;
}/** while_puts_show_ **/


void printf_(double val,double i,double j)
{
int dx,dy;
/* ustick */
unsigned char str[5],buf[20]="Drives=";
```

```
dx=i*UDX;dy=j*UDY;

itoa((int)val,/*buf*/str,10);/*gcvt(val,columns-1,buf);*/
strcat(buf,str);

setstccolor(bfset[WB].fore);
/*sentence(pflag,dx,dy,buf,strlen(buf));*/
strcpy(p,buf);
kmax=strlen(p)-1;
while_puts_show_(0,1);

if(pflag==1)
bitblt(pflag,0,0,XRESO,YRESO,0,0);
}/** printf_ **/


void initsysfont(int type,int ds_)
{
ds=ds_;

if(type==0){
/*9*/
hfont=CreateFont(UDY,UDX,0,0,
                 FW_NORMAL,FALSE,0,0,
                 DEFAULT_CHARSET,OUT_DEFAULT_PRECIS,
                 CLIP_DEFAULT_PRECIS,DEFAULT_QUALITY,
                 FIXED_PITCH | FF_ROMAN/*FF_MODERN*/,NULL);
SelectObject(hdcdisplay,hfont);
SelectObject(hdctmp1,hfont);
}
else{
hfont=CreateFont(UDY,UDX,0,0,
                 FW_NORMAL,0,0,0,
                 DEFAULT_CHARSET,OUT_DEFAULT_PRECIS,
                 CLIP_DEFAULT_PRECIS,DEFAULT_QUALITY,
                 FIXED_PITCH | /*FF_ROMAN*/FF_MODERN,NULL);
SelectObject(hdcdisplay,hfont);
SelectObject(hdctmp1,hfont);
}
}/** initsysfont **/


int read_cfg(int cfgnumber)
{
int i,j,k;
int length;
```

```
unsigned char buf[3],data[11];

itoa(cfgnumber+1,buf,10);           /* +1 */
length=strlen(buf);

if(cfgmax+1<length+2+1) return -1;

i=0;
while(1){
k=0;
while(1){
if(pcfg[i+k]==buf[k]) k++;
else break;
if(k==length) break;
}

if(k==length && pcfg[i+length]==':' && pcfg[i+length+1]==':'){
j=i+length+2;
while(1){
if(pcfg[j]>=0x30 && pcfg[j]<=0x39) j++;
else break;
if(j==cfgmax+1) break;
}

if(j==i+length+2) return -1;
break;                              /* detected */
}/**if(k,pcfg[i+length],pcfg[i+length+1])**/

i++;
if(i==cfgmax-2-(length-1)) {/*beep(50);*/return -1;}
}/**while**/

k=min(j-(i+length+2),10);
strncpy(data,&pcfg[i+length+2],k);
data[k]='\0';

return atoi(data);
}/** read_cfg **/


void setup(char flag)
{
OSVERSIONINFO osvi;

if(flag==0){
osvi.dwOSVersionInfoSize=sizeof(OSVERSIONINFO);
```

```
GetVersionEx(&osvi);

DS[0][0]=GetSystemMetrics(SM_CXSCREEN);
DS[0][1]=GetSystemMetrics(SM_CYSCREEN);

DX_FRAME=GetSystemMetrics(SM_CXSIZEFRAME);
DY_FRAME=GetSystemMetrics(SM_CYSIZEFRAME);
DY_CAPTION=GetSystemMetrics(SM_CYCAPTION);
}
else{
/* ustick */
if(usb<=3) XRESO=UNITDX*12;
else XRESO=UNITDX*12+UNITDX*3*(usb-3);
YRESO=UNITDY*3+UDX;
}
}/** setup **/


int initgraph_(void)
{
WNDCLASS wndclass;

initpalette();

setup(1);

wndclass.hInstance    =hinstance;
wndclass.lpszClassName="SSCLASS";
wndclass.lpszMenuName =NULL;
wndclass.lpfnWndProc  =(WNDPROC)wndproc_by_kbhit_;
wndclass.style        =CS_HREDRAW | CS_VREDRAW;
wndclass.hIcon        =LoadIcon(hinstance,"MYICON");
wndclass.hCursor      =LoadCursor(NULL,IDC_ARROW);
wndclass.cbClsExtra   =0;
wndclass.cbWndExtra   =0;
if(WB==0)
wndclass.hbrBackground=GetStockObject(WHITE_BRUSH);
else
wndclass.hbrBackground=GetStockObject(BLACK_BRUSH);

if(RegisterClass(&wndclass)==0) return 1;

hwnd=CreateWindow("SSCLASS","mini",
                  /*WS_POPUP,*/
                  WS_OVERLAPPEDWINDOW,
/* ustick */
```

```
                    x_usb,y_usb,
                    XRESO+DX_FRAME,YRESO+DY_CAPTION+DY_FRAME,
                    NULL,NULL,hinstance,NULL);
if(hwnd==NULL) {MessageBox(NULL,"Memory not enough.","SS",MB_OK);return 1;}

SetWindowPos(hwnd,HWND_TOP,0,0,0,0,SWP_NOMOVE | SWP_NOSIZE);
ShowWindow(hwnd,SW_SHOWDEFAULT);

hdcdisplay=GetDC(hwnd);

hbitmap1=CreateCompatibleBitmap(hdcdisplay,XRESO,YRESO);  /* visual page */

hdctmp1=CreateCompatibleDC(hdcdisplay);

SelectObject(hdctmp1,hbitmap1);

SetBkMode(hdcdisplay,TRANSPARENT);
SetBkMode(hdctmp1,TRANSPARENT);

SetBkColor(hdcdisplay,PALETTE(bfset[WB].back));
SetBkColor(hdctmp1,PALETTE(bfset[WB].back));

initsysfont(1,1);                   /* ROW */

return 0;
}/** initgraph_ **/


void closegraph_(void)
{
DeleteObject(hfont);
DeleteObject(hbitmap1);
DeleteDC(hdctmp1);

ReleaseDC(hwnd,hdcdisplay);
DestroyWindow(hwnd);

UnregisterClass("SSCLASS",hinstance);
}/** closegraph_ **/


COLORREF PALETTE(int color)
{
return RGB(irgb[color].red,irgb[color].green,irgb[color].blue);
}/** PALETTE **/
```

```
void cleardevice_(char flag,int x,int y,int xsize,int ysize)
{
if(flag==0)
PatBlt(hdcdisplay,x,y,xsize,ysize,bfset[WB].back_);
else if(flag==1)
PatBlt(hdctmp1,x,y,xsize,ysize,bfset[WB].back_);
}/** cleardevice_ **/


void bitblt(char flag,int x,int y,int xsize,int ysize,int x_,int y_)
{
if(flag==0) {}
else if(flag==1)
BitBlt(hdcdisplay,x_,y_,xsize,ysize,
       hdctmp1,x,y,SRCCOPY);
}/** bitblt **/


void restore_in_PAINT(void)
{
ValidateRect(hwnd,NULL);

/*restore_3();*/
bitblt(1,0,0,XRESO,YRESO,0,0);
}/** restore_in_PAINT **/


void initpalette(void)
{
int i;

irgb[0].red=0;irgb[0].green=0;irgb[0].blue=0;

irgb[9].red=0;irgb[9].green=0;irgb[9].blue=255;  /* blue */
irgb[10].red=0;irgb[10].green=255;irgb[10].blue=0;  /* green */
irgb[11].red=0;irgb[11].green=255;irgb[11].blue=255;  /* cyan */
irgb[12].red=255;irgb[12].green=0;irgb[12].blue=0;  /* red */
irgb[13].red=255;irgb[13].green=0;irgb[13].blue=255;  /* magenta */
irgb[14].red=255;irgb[14].green=255;irgb[14].blue=0;  /* yellow */

irgb[15].red=255;irgb[15].green=255;irgb[15].blue=255;

for(i=1;i<7;i++){                    /* 1 -> 6 */
if(irgb[9+(i-1)].red==255)
irgb[i].red=127+64;
```

```
if(irgb[9+(i-1)].green==255)
irgb[i].green=127+64;
if(irgb[9+(i-1)].blue==255)
irgb[i].blue=127+64;
}

for(i=7;i<9;i++){                    /* 7, 8 */
irgb[i].red=127+/*64*/16*(8-i);
irgb[i].green=irgb[i].red;
irgb[i].blue=irgb[i].red;
}
}/** initpalette **/


void setstccolor(int color)
{
SetTextColor(hdcdisplay,PALETTE(color));
SetTextColor(hdctmp1,PALETTE(color));
}/** setstccolor **/


void sentence(char flag,int x,int y,unsigned char *str,int size)
{
x+=RSHIFT;
y+=DSHIFT;

if(flag==0)
TextOut(hdcdisplay,x,y,str,size);
else
TextOut(hdctmp1,x,y,str,size);
}/** sentence **/


void beep(long millisecond)
{
Beep(888,millisecond);
}/** beep **/


void subroop(void)
{
if(f2code!=0) charflag=1;

while(1){
kbhit_();
if(charflag==0) return;
```

```
}
}/** subroop **/


void use_subroop(void)
{
char function_old,charflag_old;

function_old=function;function=2;
/*charflag_old=charflag;*/

subroop();

function=function_old;
/*charflag=charflag_old;*/
}/** use_subroop **/


void keydowns_f2(void)
{
if(GKS(VK_ESCAPE)<0) {charflag=0;f2code=0;}
else if(GKS('R')<0) {charflag=0;f2code=1;}
}/** keydowns_f2 **/


void kbhit_(void)
{
MSG msg;

if(PeekMessage(&msg,NULL,0,0,PM_REMOVE)){
/*if(GetMessage(&msg,NULL,0,0)){*/
TranslateMessage(&msg);
DispatchMessage(&msg);
}
}/** kbhit_ */


LRESULT CALLBACK  wndproc_by_kbhit_(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
{
if(wndproc_filer(hwnd,umsg,wparam,lparam)!=0) return 1;

return DefWindowProc(hwnd,umsg,wparam,lparam);
}/** wndproc_by_kbhit_ **/


int wndproc_filer(HWND hwnd,UINT umsg,WPARAM wparam,LPARAM lparam)
```

```
{
if(umsg==WM_KEYDOWN){
/************************* menu keydowns -> ***************************/
/************************* <- menu keydowns ***************************/

/************************* dialog keydowns -> ***************************/
/************************* <- dialog keydowns ***************************/

keydowns_f2();

return 1;
}/**else if(umsg)**/
else if(umsg==WM_LBUTTONDOWN || umsg==WM_RBUTTONDOWN){
charflag=0;f2code=1;

return 1;
}/**if(umsg)**/
else if(umsg==WM_SYSKEYDOWN){
}/**else if(umsg)**/
else if(umsg==WM_CHAR){
}/**else if(umsg)**/
else if(umsg==WM_CLOSE){
charflag=0;f2code=0;

return 1;
}/**else if(umsg)**/
else if(umsg==WM_PAINT){
restore_in_PAINT();

return 1;
}/**else if(umsg)**/
else{}

return 0;
}/** wndproc_filer **/



/* usb stick searcher                                              */
/* \dos\ustick.cfg                                                 */

No.1::20    /* drivesmax */
No.2::1    /* beepflag */
No.3::0    /* x coordinate */
No.4::100    /* y coordinate */
No.5::8    /* width of font */
No.6::16    /* height of font */
```

```
No.7::4    /* right */
No.8::3    /* down */
No.9::5    /* beepon(sec) */
No.10::666    /* freq(hz) */
```