솔루션 상호운용성 인증 지원 [터치스크린과 가상키보드 호환성개발지원]

한국소프트웨어진흥원
공개SW기술지원센터

<Revision 정보>

일자	VERSION	변경내역	작성자
2007. 11. 17	0.1	초기 작성	김상운

	솔루션 상호운용성 인증지원	
	구분 : 기업 대 기관	단계: 완료
	작성자: 김상운	작성일: 2007년 11 월 17 일
	검토자: 허종윤	작성일: 2007년 11 월 17 일
	승인자: 박훈성	작성일: 2007년 11 월 17 일

1. 인증지원 기관 정보

구분	항목	내용	비고
기업/기관 정보	지역	서울시 광진구 구의동	
	기업/기관 명칭	(주)한글과컴퓨터	
	부서	리눅스컨설팅팀	
	직책	대리	
	담당자 이름	윤대종	
	전화번호 / 팩스번호	02-3424-3169	
	E-Mail	hjyoon@haansoft.com	

2. 인증 지원 사항

구분	항목	내용	비고
리눅스용 터치패드와 가상키보드 호환성 해결 요청	지원요청사항	자사가 개발하고 있는 UMPC용 리눅스 배포판에서 사용할 터치패드 기능과 가상키보드간의 호환성 검토	
	지원상세내역	- 터치패드 현 구현 기능 분석 - 터치패드 드라이버에서 현 지원 불가로 인해 가상키보드에서는 지원을 하나 사용불가능한 경우 분석 - 가상키보드의 가능들을 지원하기 위한 터치패드 드라이버 개선 및 관련 소스 제공	
	확인 사항	리눅스용 공개SW기반의 가상키보드에서 마우스 왼쪽 버튼 클릭시 사용하는 기능이 터치패드에서도 사용가능하도록 구현	

3. 지원 내역

- 개요

일반적으로 1세대 UMPC의 사용자 인터페이스는 대부분 화면을 터치하는 터치패드 방식이며 터치패드는 드래그, 클릭을 지원하나 마우스 왼쪽 버튼 또는 중간 버튼 클릭에 대한 입력방식이 리눅스용 드라이버에서는 없음.

그러나 현재 적용하고자 하는 리눅스용 가상키보드에서 펑션 설정을 하기 위해서는 마우스 오른버튼 이벤트가 필요하나 현재 구현된 터치패드 드라이버에서는 이에 대한 구현이 없어 이 기능에 대한 사용이 어려운 상황임

따라서 다음과 같은 소스를 개발 및 제공하여 마우스 오른버튼에 대한 클릭이 가능하도록 기능 구현

- 구현 대상 기능
* 마우스 왼쪽 버튼 클릭 : 화면 1회 터치 또는 압력
* 마우스 오른쪽 버튼 클릭 : 화면에 2초 이상 동일 자리에 터치 또는 압력
* 마우스 중간 버튼 클릭 : 차후 개발 예정

- 개발 환경
* 메인 드라이버 : ltouch v0.5 기반
* 관련 개발 언어 : C/C++

- 관련 개발 소스

```
/*****************************************************************************
 *          Function Definitions
 ****************************************************************************/


static CARD32
emulate3Timer(OsTimerPtr timer, CARD32 now, pointer _local)
{
        int sigstate;

        LocalDevicePtr local = (LocalDevicePtr)_local;
        ltouchPrivatePtr priv = (ltouchPrivatePtr) local->private;
```

```
sigstate = xf86BlockSIGIO();

xf86PostMotionEvent(local->dev, TRUE, 0, 2,
                    priv->cur_x,
                    priv->cur_y);

/*
 * Emit a button press -- release is handled in ltouchLBRBEvent
 */
if ( ( priv->touch_flags & LB_STAT ) &&
     !( priv->touch_flags & RB_STAT ) ) {
        DBGOUT(2, "Left Press₩n");
        xf86PostButtonEvent (local->dev, TRUE,
                             1, 1, 0, 2,
                             priv->cur_x,
                             priv->cur_y);
}

if ( ( priv->touch_flags & RB_STAT ) &&
     !( priv->touch_flags & LB_STAT ) ) {
        DBGOUT(2, "Right Press₩n");
        xf86PostButtonEvent (local->dev, TRUE,
                             3, 1, 0, 2,
                             priv->cur_x,
                             priv->cur_y);
}

/*
  Handling "middle" button press
*/
if ( ( priv->touch_flags & RB_STAT ) &&
     ( priv->touch_flags & LB_STAT ) ) {
        DBGOUT(2, "Middle Press₩n");
        xf86PostButtonEvent (local->dev, TRUE,
                             2, 1, 0, 2,
                             priv->cur_x,
                             priv->cur_y);
}
```

```
        priv->emulate3_timer_expired = TRUE;
        xf86UnblockSIGIO(sigstate);

        return 0;
}

void ltouchProcessAbs(ltouchPrivatePtr priv)
{
        struct input_event *ev; /* packet being/just read */

        ev = &priv->ev;
        priv->old_x = priv->cur_x;
        priv->old_y = priv->cur_y;

        if (ev->code == ABS_X)
                priv->cur_x = ev->value;

        if (ev->code == ABS_Y)
                priv->cur_y = ev->value;

        trigger_sm(priv);
}

void ltouchProcessRel(ltouchPrivatePtr priv)
{
        struct input_event *ev; /* packet being/just read */

        ev = &priv->ev;
        priv->old_x = priv->cur_x;
        priv->old_y = priv->cur_y;
        if ( ev->code == ABS_X ) {
                priv->cur_x += ev->value;
                if (priv->cur_x > priv->max_x)
                        priv->cur_x = priv->max_x;
                if (priv->cur_x < priv->min_x)
                        priv->cur_x = priv->min_x;
                return;
```

```
        }
        if ( ev->code == ABS_Y ) {
                priv->cur_y += ev->value;
                if (priv->cur_y > priv->max_y)
                        priv->cur_y = priv->max_y;
                if (priv->cur_y < priv->min_y)
                        priv->cur_y = priv->min_y;
                return;
        }
}

void ltouchLBRBEvent(ltouchPrivatePtr priv)
{
        struct input_event *ev; /* packet being/just read */
        LocalDevicePtr local = priv->local;
        static OsTimerPtr emulate3_timer = NULL;
        int btn = 0;

        ev = &priv->ev;
        if (priv->emulate3) {
                if ( (ev->value==1) && (emulate3_timer==NULL) )
                        emulate3_timer = TimerSet(emulate3_timer, 0,
                                                  priv->emulate3_timeout,
                                                  emulate3Timer,
                                                  local);

                if ( (ev->value == 1) && (ev->code == BTN_LEFT) )
                        priv->touch_flags |= LB_STAT;
                if ( (ev->value == 1) && (ev->code == BTN_RIGHT) )
                        priv->touch_flags |= RB_STAT;

                if ( (ev->value == 0) &&
                     (priv->touch_flags & RB_STAT) &&
                     (priv->touch_flags & LB_STAT) ) {
                        DBGOUT(2, "Middle Release₩n");
                        priv->touch_flags &= ~LB_STAT;
                        priv->touch_flags &= ~RB_STAT;
                        btn = 2;
                }
```

```
                if ( (ev->value == 0) && (ev->code == BTN_LEFT) &&
                     (priv->touch_flags & LB_STAT) ) {
                        DBGOUT(2, "Left Release₩n");
                        priv->touch_flags &= ~LB_STAT;
                        btn = 1;
                }
                if ( (ev->value == 0) && (ev->code == BTN_RIGHT) &&
                     (priv->touch_flags & RB_STAT) ) {
                        DBGOUT(2, "Right Release₩n");
                        priv->touch_flags &= ~RB_STAT;
                        btn = 3;
                }
                if (ev->value==0) {
                        TimerFree(emulate3_timer);
                        emulate3_timer=NULL;
                        priv->emulate3_timer_expired = FALSE;
                        xf86PostButtonEvent(local->dev, TRUE,
                                            btn, ev->value, 0, 2,
                                            priv->cur_x,
                                            priv->cur_y);
                }

        } else {
                if (ev->code == BTN_LEFT) {
                        xf86PostButtonEvent(local->dev, TRUE,
                                            1, ev->value, 0, 2,
                                            priv->cur_x,
                                            priv->cur_y);
                }


                if (ev->code == BTN_RIGHT) {
                        xf86PostButtonEvent (local->dev, TRUE,
                                             3, ev->value, 0, 2,
                                             priv->cur_x,
                                             priv->cur_y);
                }
        }
}
```