١ مشغلات الاجهزة Device Driver

١.١ برمجة مشغل لوحة المفاتيح Keyboard Driver

Example ١.١: Keybaord Driver Interface

```
١
٢ #ifndef KEYBOARD_H
٣ #define KEYBOARD_H
٤
٥ #include <stdint.h>
٦
٧ enum KEY_CODE{
٨   KEY_SPACE             = ' ',
٩   KEY_0                 = '0',
١٠  KEY_1                 = '1',
١١  KEY_2                 = '2',
١٢  KEY_3                 = '3',
١٣  KEY_4                 = '4',
١٤  KEY_5                 = '5',
١٥  KEY_6                 = '6',
١٦  KEY_7                 = '7',
١٧  KEY_8                 = '8',
١٨  KEY_9                 = '9',
١٩  KEY_A                 = 'a',
٢٠  KEY_B                 = 'b',
٢١  KEY_C                 = 'c',
٢٢  KEY_D                 = 'd',
٢٣  KEY_E                 = 'e',
٢٤  KEY_F                 = 'f',
٢٥  KEY_G                 = 'g',
٢٦  KEY_H                 = 'h',
٢٧  KEY_I                 = 'i',
٢٨  KEY_J                 = 'j',
٢٩  KEY_K                 = 'k',
```

```
٣٠   KEY_L                = 'l',
٣١   KEY_M                = 'm',
٣٢   KEY_N                = 'n',
٣٣   KEY_O                = 'o',
٣٤   KEY_P                = 'p',
٣٥   KEY_Q                = 'q',
٣٦   KEY_R                = 'r',
٣٧   KEY_S                = 's',
٣٨   KEY_T                = 't',
٣٩   KEY_U                = 'u',
٤٠   KEY_V                = 'v',
٤١   KEY_W                = 'w',
٤٢   KEY_X                = 'x',
٤٣   KEY_Y                = 'y',
٤٤   KEY_Z                = 'z',
٤٥   KEY_RETURN           = '\r',
٤٦   KEY_ESCAPE           = 0x1001,
٤٧   KEY_BACKSPACE        = '\b',
٤٨   KEY_UP               = 0x1100,
٤٩   KEY_DOWN             = 0x1101,
٥٠   KEY_LEFT             = 0x1102,
٥١   KEY_RIGHT            = 0x1103,
٥٢   KEY_F1               = 0x1201,
٥٣   KEY_F2               = 0x1202,
٥٤   KEY_F3               = 0x1203,
٥٥   KEY_F4               = 0x1204,
٥٦   KEY_F5               = 0x1205,
٥٧   KEY_F6               = 0x1206,
٥٨   KEY_F7               = 0x1207,
٥٩   KEY_F8               = 0x1208,
٦٠   KEY_F9               = 0x1209,
٦١   KEY_F10              = 0x120a,
٦٢   KEY_F11              = 0x120b,
٦٣   KEY_F12              = 0x120b,
٦٤   KEY_F13              = 0x120c,
٦٥   KEY_F14              = 0x120d,
٦٦   KEY_F15              = 0x120e,
٦٧   KEY_DOT              = '.',
٦٨   KEY_COMMA            = ',',
٦٩   KEY_COLON            = ':',
٧٠   KEY_SEMICOLON        = ';',
```

```
٧١   KEY_SLASH             = '/',
٧٢   KEY_BACKSLASH         = '\\',
٧٣   KEY_PLUS              = '+',
٧٤   KEY_MINUS             = '—',
٧٥   KEY_ASTERISK          = '*',
٧٦   KEY_EXCLAMATION       = '!',
٧٧   KEY_QUESTION          = '?',
٧٨   KEY_QUOTEDOUBLE       = '\"',
٧٩   KEY_QUOTE             = '\'',
٨٠   KEY_EQUAL             = '=',
٨١   KEY_HASH              = '#',
٨٢   KEY_PERCENT           = '%',
٨٣   KEY_AMPERSAND         = '&',
٨٤   KEY_UNDERSCORE        = '_',
٨٥   KEY_LEFTPARENTHESIS   = '(',
٨٦   KEY_RIGHTPARENTHESIS  = ')',
٨٧   KEY_LEFTBRACKET       = '[',
٨٨   KEY_RIGHTBRACKET      = ']',
٨٩   KEY_LEFTCURL          = '{',
٩٠   KEY_RIGHTCURL         = '}',
٩١   KEY_DOLLAR            = '$',
٩٢   KEY_POUND             = '£',
٩٣   KEY_EURO              = '$',
٩٤   KEY_LESS              = '<',
٩٥   KEY_GREATER           = '>',
٩٦   KEY_BAR               = '|',
٩٧   KEY_GRAVE             = '`',
٩٨   KEY_TILDE             = '~',
٩٩   KEY_AT                = '@',
١٠٠  KEY_CARRET            = '^',
١٠١  KEY_KP_0              = '0',
١٠٢  KEY_KP_1              = '1',
١٠٣  KEY_KP_2              = '2',
١٠٤  KEY_KP_3              = '3',
١٠٥  KEY_KP_4              = '4',
١٠٦  KEY_KP_5              = '5',
١٠٧  KEY_KP_6              = '6',
١٠٨  KEY_KP_7              = '7',
١٠٩  KEY_KP_8              = '8',
١١٠  KEY_KP_9              = '9',
١١١  KEY_KP_PLUS           = '+',
```

```
١١٢   KEY_KP_MINUS          = '—',
١١٣   KEY_KP_DECIMAL        = '.',
١١٤   KEY_KP_DIVIDE         = '/',
١١٥   KEY_KP_ASTERISK       = '*',
١١٦   KEY_KP_NUMLOCK        = 0x300f,
١١٧   KEY_KP_ENTER          = 0x3010,
١١٨   KEY_TAB               = 0x4000,
١١٩   KEY_CAPSLOCK          = 0x4001,
١٢٠   KEY_LSHIFT            = 0x4002,
١٢١   KEY_LCTRL             = 0x4003,
١٢٢   KEY_LALT              = 0x4004,
١٢٣   KEY_LWIN              = 0x4005,
١٢٤   KEY_RSHIFT            = 0x4006,
١٢٥   KEY_RCTRL             = 0x4007,
١٢٦   KEY_RALT              = 0x4008,
١٢٧   KEY_RWIN              = 0x4009,
١٢٨   KEY_INSERT            = 0x400a,
١٢٩   KEY_DELETE            = 0x400b,
١٣٠   KEY_HOME              = 0x400c,
١٣١   KEY_END               = 0x400d,
١٣٢   KEY_PAGEUP            = 0x400e,
١٣٣   KEY_PAGEDOWN          = 0x400f,
١٣٤   KEY_SCROLLLOCK        = 0x4010,
١٣٥   KEY_PAUSE             = 0x4011,
١٣٦   KEY_UNKNOWN,
١٣٧   KEY_NUMKEYCODES
١٣٨ };
١٣٩
١٤٠ extern bool keyboard_get_scroll_lock();
١٤١ extern bool keyboard_get_caps_lock();
١٤٢ extern bool keyboard_get_num_lock();
١٤٣ extern bool keyboard_get_alt();
١٤٤ extern bool keyboard_get_ctrl();
١٤٥ extern bool keyboard_get_shift();
١٤٦ extern void keyboard_ignore_resend();
١٤٧ extern bool keyboard_check_resend();
١٤٨ extern bool keyboard_get_diagnostic_res();
١٤٩ extern bool keyboard_get_bat_res();
١٥٠ extern bool keyboard_self_test();
١٥١ extern uint8_t keyboard_get_last_scan();
١٥٢ extern KEY_CODE keyboard_get_last_key();
```

```
١٥٣ extern void keyboard_discard_last_key();
١٥٤ extern void keyboard_set_leds(bool nums,bool caps,bool scroll);
١٥٥ extern char keyboard_key_to_ascii(KEY_CODE k);
١٥٦ extern void keyboard_enable();
١٥٧ extern void keyboard_disable();
١٥٨ extern bool keyboard_is_disabled();
١٥٩ extern void keyboard_reset_system();
١٦٠ extern void keyboard_install(int);
١٦١
١٦٢ #endif // KEYBOARD_H
```

Example ١.٢: Keybaord Driver Implemetation

```
١
٢ void keyboard_install(int irq) {
٣   // Install  interrupt handler (irq 1 uses interrupt 33)
٤   set_vector(irq, i386_keyboard_irq);
٥
٦   // assume BAT test is good. If there is a problem, the IRQ handler
        where catch the error
٧   _keyboard_bat_res = true;
٨   _scancode = 0;
٩
١٠  // set lock keys and led lights
١١  _numlock = _scrolllock = _capslock = false;
١٢  keyboard_set_leds (false, false, false);
١٣
١٤  // shift, ctrl, and alt keys
١٥  _shift = _alt = _ctrl = false;
١٦ }
١٧
١٨
١٩ //  keyboard interrupt handler
٢٠ void _cdecl i386_keyboard_irq () {
٢١
٢٢  _asm add esp, 12
٢٣  _asm pushad
٢٤  _asm cli
٢٥
٢٦  static bool _extended = false;
```

```
٢٧
٢٨   int code = 0;
٢٩
٣٠   // read scan code only if the keyboard controller output buffer is
        full (scan code is in it)
٣١   if (keyboard_ctrl_read_status () & KEYBOARD_CTRL_STATS_MASK_OUT_BUF
        ) {
٣٢
٣٣     // read the scan code
٣٤     code = keyboard_enc_read_buf ();
٣٥
٣٦     // is this an extended code? If so, set it and return
٣٧     if (code == 0xE0 || code == 0xE1)
٣٨       _extended = true;
٣٩     else {
٤٠
٤١       // either the second byte of an extended scan code or a single
            byte scan code
٤٢       _extended = false;
٤٣
٤٤       // test if this is a break code (Original XT Scan Code Set
            specific)
٤٥       if (code & 0x80) {  //test bit 7
٤٦
٤٧         // covert the break code into its make code equivelant
٤٨         code -= 0x80;
٤٩
٥٠         // grab the key
٥١         int key = _keyboard_scancode_std [code];
٥٢
٥٣         // test if a special key has been released & set it
٥٤         switch (key) {
٥٥
٥٦           case KEY_LCTRL:
٥٧           case KEY_RCTRL:
٥٨             _ctrl = false;
٥٩             break;
٦٠
٦١           case KEY_LSHIFT:
٦٢           case KEY_RSHIFT:
٦٣             _shift = false;
```

```
٦٤              break;
٦٥
٦٦            case KEY_LALT:
٦٧            case KEY_RALT:
٦٨              _alt = false;
٦٩              break;
٧٠          }
٧١        }
٧٢        else {
٧٣
٧٤          // this is a make code — set the scan code
٧٥          _scancode = code;
٧٦
٧٧          // grab the key
٧٨          int key = _keyboard_scancode_std [code];
٧٩
٨٠          // test if user is holding down any special keys & set it
٨١          switch (key) {
٨٢
٨٣            case KEY_LCTRL:
٨٤            case KEY_RCTRL:
٨٥              _ctrl = true;
٨٦              break;
٨٧
٨٨            case KEY_LSHIFT:
٨٩            case KEY_RSHIFT:
٩٠              _shift = true;
٩١              break;
٩٢
٩٣            case KEY_LALT:
٩٤            case KEY_RALT:
٩٥              _alt = true;
٩٦              break;
٩٧
٩٨            case KEY_CAPSLOCK:
٩٩              _capslock = (_capslock) ? false : true;
١٠٠             keyboard_set_leds (_numlock, _capslock, _scrolllock);
١٠١             break;
١٠٢
١٠٣           case KEY_KP_NUMLOCK:
١٠٤             _numlock = (_numlock) ? false : true;
```

```
١٠٥             keyboard_set_leds (_numlock, _capslock, _scrolllock);
١٠٦             break;
١٠٧
١٠٨           case KEY_SCROLLLOCK:
١٠٩             _scrolllock = (_scrolllock) ? false : true;
١١٠             keyboard_set_leds (_numlock, _capslock, _scrolllock);
١١١             break;
١١٢         }
١١٣       }
١١٤     }
١١٥
١١٦     // watch for errors
١١٧     switch (code) {
١١٨
١١٩       case KEYBOARD_ERR_BAT_FAILED:
١٢٠         _keyboard_bat_res = false;
١٢١         break;
١٢٢
١٢٣       case KEYBOARD_ERR_DIAG_FAILED:
١٢٤         _keyboard_diag_res = false;
١٢٥         break;
١٢٦
١٢٧       case KEYBOARD_ERR_RESEND_CMD:
١٢٨         _keyboard_resend_res = true;
١٢٩         break;
١٣٠     }
١٣١   }
١٣٢
١٣٣   // tell hal we are done
١٣٤   int_done(0);
١٣٥
١٣٦   // return from interrupt handler
١٣٧   _asm sti
١٣٨   _asm popad
١٣٩   _asm iretd
١٤٠ }
```