

## Especificação sk\_access V1.99.91 libsk\_access.so V1.99.3



### Descrição:

Sk\_access.dll / libsk\_access.so é uma biblioteca de funções desenvolvida pela Smak Teclados com o objetivo de facilitar o trabalho daqueles que desenvolvem softwares para os produtos SMAK. Este documento destina-se a desenvolvedores e contém as informações técnicas necessárias ao uso da biblioteca.

Rev. 1.98

## Índice

Histórico de alterações.....	3
Descrição sk_access.....	4
Princípio de funcionamento.....	4
Importação da lib.....	5
Plug and Play.....	6
Variáveis de ambiente.....	7
Demonstração e testes das funções da biblioteca Smak.....	9
Instalando a lib.....	9
Windows.....	9
Linux.....	10
Testes.....	10
Utilizando a biblioteca da SMAK em seu ambiente de desenvolvimento.....	10
Convenção de chamada.....	11
Funções exportadas pela dll / lib.....	11
Nota sobre a função GetOperatingSystem:.....	15
Nota sobre a função Redirect:.....	15
Nota sobre a função Send_Disp_Ctrl:.....	15
Notas sobre versões de Firmware.....	16

**Histórico de alterações:**Revisão 1.98 (24-02-2016) :

-Atualização info sobre Select\_Interface, Set\_Interface, Get\_Current\_interface.

Revisão 1.97 (3-2015) :

-SK8042 = 5 impede o uso de mouse PS2.  
-Acrescida variavel de ambiente SK\_ECO.  
-Acrescentado tópico "Importação da dll" e Convenção de chamada.  
-Revisão de textos.

Revisão 1.96 (12-05-2014) :

-Atualização para sk\_access 1.99.9 e 1.99.3 (HID + travamento COM)

Revisão 1.95 (12-12-2013) :

-Acrescidos texto sobre timers, InstallDriver.exe e Win7/8

Revisão 1.94 (26-08-2013) :

-Atualizado para sk\_access.dll 1.99.7  
-Acrescentado Notas sobre GetOperatingSystem e Redirect

Revisão 1.93 (10-05-2013) :

-Acrescentado Notas sobre versões de firmware

Revisão 1.92 (11-03-2013) :

-Atualizado para sk\_access.dll 1.99.4  
-Atualizado para libsk\_access.so 1.95  
-Colocadas funções da dll em ordem alfabética.  
-Acrescidos comandos diretos do display.

Revisão 1.91 (12-07-2012) :

-Atualizado para sk\_access.dll 1.97  
-Atualizado para libsk\_access.so 1.93  
-Completado texto relativo às variáveis de ambiente.

Revisão 1.9 (06-06-2012) :

-Revisada função Log\_On.  
-Acrescentada função Get\_Scan\_Id.

Revisão 1.8 (08-09-2011) :

-Revisão de textos.

Revisão 1.7 (27-07-2011) :

-Revisão de textos.  
-Documentação Linux.  
-Padronizada sintaxe C.

Revisão 1.6 (05-10-2010) :

- Revisão de textos.  
- Introdução de novas funcionalidades de sk\_access.dll versão 1.93.

Revisão 1.5 (27-02-2010) :

- Revisão de textos.  
- Introdução de novas funcionalidades.

Revisão 1.4 (21-07-2008) :

- Introdução de novas funcionalidades de sk\_access.dll versão 1.6.

Revisão 1.3 (11-09-2007) :

- Revisão de textos.  
- Introdução de novas funcionalidades de sk\_access.dll versão 1.5.

**Descrição sk\_access:**

Aqueles que desejam se comunicar diretamente com teclados da Smak, devem consultar a documentação de hardware de cada equipamento, como por exemplo: *Especificacao\_sko44\_PS2.pdf* ou *Especificacao\_sko44\_serial.pdf*, onde constam o protocolo de comunicação do equipamento com seus comandos disponíveis.

Para simplificar e agilizar o desenvolvimento de uma aplicação a SMAK desenvolveu a lib *"sk\_access.dll"/"libsk\_access.so"* que disponibiliza uma API com funções que permitem flexibilidade no acesso aos equipamentos, possibilitando principalmente o envio de mensagens para o display

**Princípio de funcionamento:**

A lib é única e pode ser usada para se comunicar com qualquer modelo de teclado Smak (com exceção a dispositivos de rede).

A lib não possui uma função Open para iniciar suas operações, qualquer função da lib quando é invocada, verifica o status de conexão com um teclado e se um teclado não está conectado/reconhecido a lib executa os procedimentos adequados para estabelecer esta conexão. Esta forma de conexão é automática e procura por um teclado em todas as interfaces disponíveis.

É possível definir uma conexão não automática, usando a função *Set\_interface*, com esta função podemos forçar a interface/porta que vai ser utilizada para se comunicar com o teclado. É possível com *Set\_Interface* controlar mais de um teclado de modelos/interfaces diferentes ao mesmo tempo.

É possível executar *Set\_interfaces* consecutivos para se comunicar com dispositivos diferentes, mas após uma conexão automática é necessário chamar *Reset\_interface*, para que a lib possa executar uma nova conexão automática.

**Importação da lib:**

Executando os procedimentos abaixo, todas as funções da lib estarão disponíveis para uso.

**Do VB:** Está disponibilizado o arquivo **sk\_access.vb6** com o “Declare” de todas as funções, podendo ser usado para Ctrl C + Ctrl V e facilitar o trabalho de digitação.

**Do Delphi/Lazarus:** Para auxiliar a importação estática, está disponibilizado o arquivo **sk\_access.inc**.

Somente acrescentar a diretiva de compilação **{ \$I sk\_access.inc }**

**Do C++:****Windows:**

Para auxiliar a importação implícita\*, está disponibilizado o arquivo de header **sk\_access.h** e o arquivo para linkagem implícita **sk\_access.lib**.

Acrescentar o arquivo **“sk\_access.lib”** na lista de linkagem.

Acrescentar o Header:

```
#include "sk_access_tcp.h"
```

e chamar as funções:

```
Loadsk_access(); //Para carregar os nomes das funções
```

e

```
Freesk_access(); //Para liberar os recursos da dll.
```

**Linux:**

Para auxiliar a importação implícita, está disponibilizado o arquivo de header **sk\_access.h**.

Acrescentar a lib **“libsk\_access.so”** na lista de linkagem.

Acrescentar o Header:

```
#include "sk_access_tcp.h".
```

E todas as funções da sk\_access.dll estarão disponíveis para uso.

*\*No Windows a linkagem implícita (estática) com sk\_access.lib aponta para uma dll Stub que faz o carregamento explícito da sk\_access.dll, a manobra é necessária porque a sk\_access.dll não está codificada em MSC++ e não gera o arquivo .lib necessário para a linkagem.*

## Plug and Play:

A biblioteca oferece uma plataforma unificada de desenvolvimento, de forma que um aplicativo feito utilizando suas funções, pode acessar indistintamente os teclados nas interfaces PS2, serial RS-232 / USB e HID sem modificação do código fonte do aplicativo do usuário.

Quando qualquer função da biblioteca é chamada, ele executa a função `Select_Interface`.

A função `Select_Interface` procura por um teclado na interface HID, última COM selecionada, PS2 e depois outras COM's (1..MAX\_COM).

Se o teclado é encontrado em alguma destas interfaces ele é travada e a partir daí todas as operações são formatadas para ela, por exemplo a função `DISP(string)` pode mandar uma string para o display de um teclado Hid, PS2 ou serial/USB sem que o usuário precise se preocupar com isso.

O SKO44 USB usa internamente um conversor USB-RS, sendo este conversor muito popular e também encontrado em celulares, GPSs e mesmo em impressoras fiscais.

A busca para localizar o teclado, apresentou conflito com alguns destes equipamentos, de forma que foi necessário minimizar a busca ao máximo.

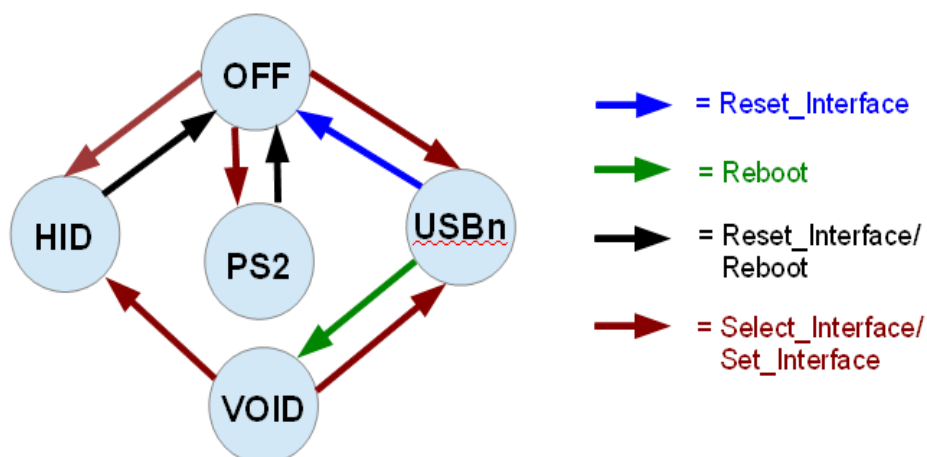
Então se um teclado USB-RS é encontrado após uma busca, esta interface é memorizada no registro do sistema.

Próximos acessos, mesmo após reboot ou recarga da lib/redirect, ficarão condicionadas à interface memorizada.

Para permitir uma nova busca (troca de porta USB) é necessário, ou chamar a função `reset_interface` da API(lib) da Smak ou chamar o programa `reset_itf` na linha de comando.

Devido a essa memorização, também existe um bloqueio de teclados PS2 até que um `reset_interface` seja acionado.

Diagrama de estados para troca de interfaces

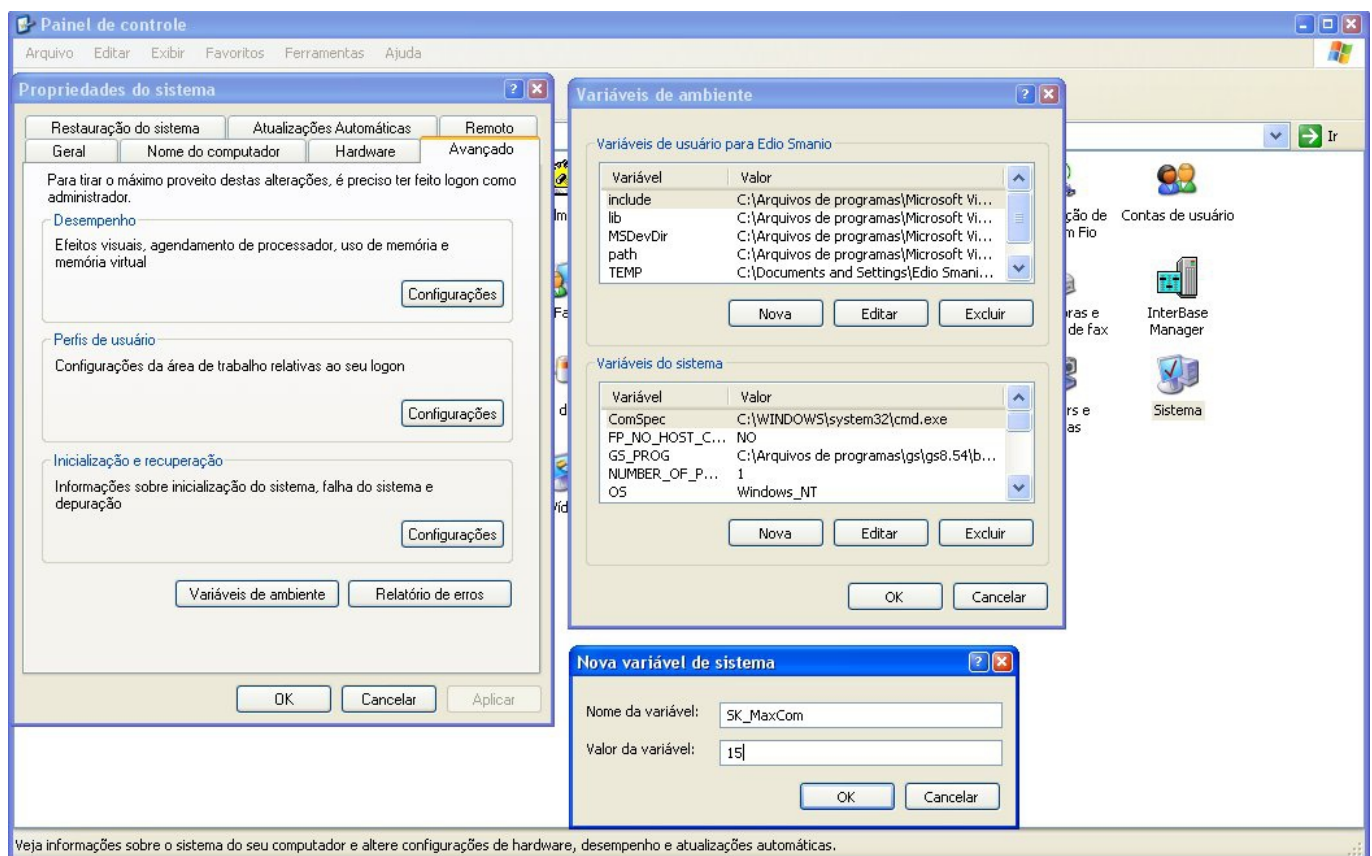


O valor de MAX\_COM por padrão é 8, significa que o teclado pode ser conectado nas COM's 1..8 que a biblioteca o encontrará, porém em alguns casos podem surgir COM's virtuais com números muito maiores, neste caso é necessário ampliar capacidade de busca.

### Variáveis de ambiente:

A variável de ambiente SK\_MaxCom redefine a maior porta a ser testada pela DLL, para criar/editar esta variável, no Windows, deve-se acessar:

Painel de controle -> Sistema -> Avançado -> Variáveis de ambiente -> Nova/Editar



A Função Set\_Interface não respeita o limite de Max\_Com, de forma que Set\_Interface('COM99'), sempre vai usar COM99 caso esteja disponível, independentemente da definição de SK\_MaxCom.

No Linux para criar a variável (por exemplo com o valor 15), colocar o comando `export SK_MaxCom=15` dentro arquivo `.profile` no diretório `$HOME`

Além da variável de ambiente SK\_MaxCom, existem também as variáveis: SK\_8042, SK\_LOG, SK\_USETSC, SK\_ECO e SK\_First\_USB.

Se a variável não foi criada, seu valor é lido como 0 pela dll/lib

SK\_8042 controla a forma com que a dll/lib interage com a controladora de teclado PS2. Pode assumir valores de 0 a 5 e 10, a dll (Windows) faz um teste da controladora, se for gerado pela dll SK\_8042 = 10, significa que a controladora foi testada e está OK. Se for gerado SK\_8042=5 a controladora tem problemas. No Linux não há teste automático

SK\_8042 pode ser alterado caso haja problemas de comunicação com o teclado ps2 (teclado não mostra Texto no display, teclado não programa, etc) devem ser testados os valores 1..5 (começando pelo valor 5) para descobrir qual melhor se adapta à controladora em questão.

Se SK\_8082=5 então um mouse PS2 não poderá ser usado em conjunto com o teclado neste computador.

SK\_LOG define o valor default do nível de registro do LOG. Pode assumir valores de 0..7, quanto maior o valor, mais detalhado é o log gerado, este valor sobrepõe o valor assumido pelo programa com a chamada a Log\_On, Log\_on no caso só consegue aumentar o valor do nível de registro.

SK\_USETSC permite que a dll use o TSC(Time Stamp Counter), como timer padrão, se essa opção não estiver ativada, a dll só vai usar HPET (High Performance Event Timer) ou PMTimer(Power Management Timer)

SK\_ECO Facilita o uso da lib com linguagens que usam UNICODE.

Se SK\_ECO = 0 → não UNICODE.

Se SK\_ECO =1 ou 2 → Verificar qual funciona com a sua linguagem UNICODE.

SK\_First\_USB (somente Linux) define qual é o primeiro dispositivo COM: que deve ser considerado USB. Se SK\_First\_USB = 5 (valor default), uma tentativa de acesso à COM5: vai abrir /dev/ttyUSB0.

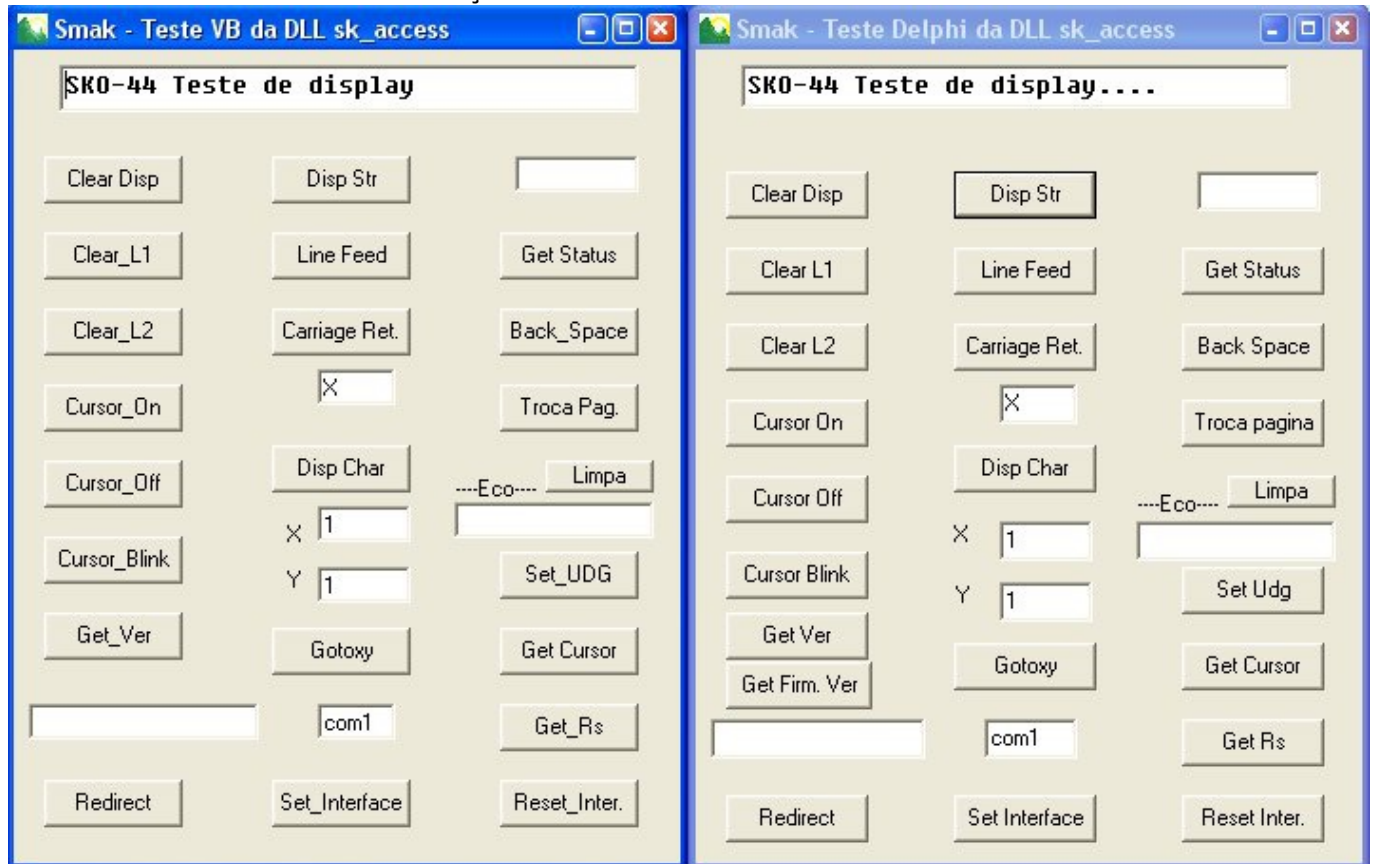
Particularidades de configuração da lib podem ser vistas no documento "Problemas\_conhecidos.txt"



### Demonstração e testes das funções da biblioteca Smak:

O utilitário de teste (Windows) disponibilizado em Delphi3 e VB6 (contendo inclusive o código fonte) foi desenvolvido especificamente como um exemplo para verificação da operação das funções exportadas pela biblioteca. A figura a seguir ilustra sua aparência no Windows.

Cada botão tem o nome da função sendo testada.



### Instalando a lib:

#### Windows:

A instalação da biblioteca consiste em obter o pacote de software SKO\_Tool\_BoxVxxx.exe disponível no site [www.smak.com.br](http://www.smak.com.br), executá-lo e responder as perguntas de instalação.

**Apenas para Windows-9x:** será preciso adicionar uma configuração na seção [386Enh] do arquivo "C:\WINDOWS\SYSTEM.INI". ( se desejar, você pode abrir o system.ini utilizando o sysedit do windows assim: Menu iniciar -> Executar -> sysedit ). Editando "SYSTEM.INI" localize a seção [386Enh] e adicione a seguinte linha:

```
DEVICE=C:\WINDOWS\SYSTEM32\sdrv9x.vxd
```

**Linux:**

Instalação em Linux Kernel 2.6.xx

Copiar libsk\_access.so para \usr\lib

**Testes:**

Conecte ao computador um teclado SKO-44 em seguida execute o utilitário de testes e em seguida teste alguns comandos.

Exemplos de testes:

- Teste da função DISP: Digite um texto na caixa de texto superior e aperte o botão "*Disp Str*". O texto digitado deverá aparecer no display do SKO-44.
- Teste da função CLEAR\_L1 e CLEAR\_L2: Aperte os botões "*Clear\_L1*" e "*Clear\_L2*" para limpar as linhas do display.

Para saber sobre todos os testes consulte a informação na tabela no final deste documento, a qual contém a descrição das funções exportadas pela biblioteca.

**Utilizando a biblioteca da SMAK em seu ambiente de desenvolvimento :**

Uma vez que você conseguiu verificar o correto funcionamento do utilitário de testes, isto significa que "*sk\_access.dll*" / "*libsk\_access.so*" está corretamente disponível para uso no sistema. É interessante lembrar que em caso de dúvidas sobre o uso das funções da biblioteca, o profissional desenvolvedor poderá consultar o código fonte do aplicativo de testes disponibilizado em VB e Delphi.

**Convenção de chamada:**

Nas funções abaixo, substituir `_mode` pela convenção de chamada (Calling Convention) apropriada.

A dll / lib segue a convenção padrão do sistema operacional sendo:

`_mode = _stdcall` (ou WINAPI) em Windows.

`_mode = _cdecl` em Linux.

**Funções exportadas pela dll / lib:**

Veja a seguir a tabela com as funções e procedimentos disponíveis na biblioteca Smak.

PROTÓTIPO	DESCRIÇÃO	WIN	LINUX
void _mode Back_Space();	Back space do cursor	X	X
int _mode Buffer_In_Data();	Retorna número de dados no buffer de entrada		X
void _mode Carriage_Return();	Carriage return	X	X
void _mode Clear_Dis();	Apaga o display	X	X
void _mode Clear_L1();	Apaga a linha 1 do display	X	X
void _mode Clear_L2();	Apaga a linha 2 do display	X	X
void _mode Cursor_Blink();	Piscar o cursor	X	X
void _mode Cursor_Off();	Desligar o cursor no display	X	X
void _mode Cursor_On();	Ligar o cursor	X	X
void _mode Disable_Keyboard();	Desativa varredura do teclado	X	X
void _mode Disp_Char(char data);	Exibe um caracter no display	X	X
void _mode Disp(char *data);	Exibe uma string de até 80 caracteres.	X	X
void _mode Enable_Keyboard();	Ativa varredura do Teclado	X	X
void _mode Free_Sk_Access();	Remove hooks e libera memória	X	X
bool _mode Get_Com_Type();	Retorna tipo da porta COM: TRUE = Virtual, FALSE = Real	X	X
void _mode Get_Coms(char *Coms);	Retorna uma string com todas as COM's disponíveis	X	
int _mode Get_Current_Interface();	-1 = nenhuma interface 0 = PS2 , 1..n = COM1..COMn, 100=HID	X	X
char _mode Get_Cursor(char *x_pos, char *y_pos);	Lê coordenadas em x_pos e y_pos (home = 1,1), Função retorna posição absoluta (0 a 79) ou Time-out = 255.	X	X
void _mode Get_Data	'= Get_Data_Rs	X	X
int _mode Get_Data_Rs (char *buffer, int size);	Transfere para "buffer" um máximo de "size" caracteres lidos da porta serial. Retorna o número de caracteres efetivamente lidos.	X	X
void _mode Get_DLL_Version(char *ver);	Devolve na variável ver, a versão da DLL. "char *ver" deve ser maior que 20 caracteres.	X	X
void _mode Get_Firmware_Version (char *ver);	Retorna string no format (PS2)=v.w.w ou (RS)=v.w.w, "char *ver" deve ser maior que 15 caracteres.	X	X
void _mode Get_Scan_Id(char *id);	Le Scan_Id gravado com diretiva \$Scan	X	X
void _mode Get_Serial_Number (char *ser);	Retorna string com número de série. "char *ser" deve ser maior que 20 caracteres.	X	X
char _mode Get_Status();	Status da última operação, Retorna: 0 = OK ; 1 = erro e FFh = time-out.	X	X

PROTÓTIPO	DESCRIÇÃO	WIN	LINUX
int _mode GetOperatingSystem();	Identifica sistema operacional, Ver Nota	X	X
void _mode Gotoxy(char x , char y);	Posiciona em Linha Y, Coluna X, Home = (1,1).	X	X
void _mode Keyb_Reset();	Reseta o teclado.	X	X
int _mode Kill_Task(char *ExeFileName);	Mata tarefa em execução	X	
void _mode Line_Feed();	Line-feed no display	X	X
void _mode Log_Off();	Desliga registro de LOG	X	X
void _mode Log_On(int dbg_level);	Liga Registro de log. Será criado o arquivo sk_access_log com registro das atividades da DLL, dbg_level (1..7) controla detalhamento do log.	X	X
void _mode Redirect();	Captura a porta serial e redireciona para o buffer do teclado. Ver nota	X	X
void _mode Reset_Interface();	Esquece as informações de Interface.	X	X
void _mode Scroll_Enable_Off();	Tecla Scroll-lock Ativa/Desativa teclado serial	X	
void _mode Scroll_Enable_On();		X	
int _mode Select_Interface();	Procura por um teclado SKO44 conectado, Retorna: -1=ñ encontrado, 0= PS2, 1..99=COM(1..99), 100=HID	X	X
void _mode Send_Data(char data);	Envia um byte para o teclado	X	X
void _mode Send_Disp_Ctrl(char data);	Envia comando direto para o display	X	X
void _mode Set_Beep_On()	Liga Beep do teclado	X	
void _mode Set_Beep_Off()	Desliga Beep do teclado	X	
void _mode Set_Controller(char data)	Programa controladora 8042	X	X
void _mode Set_Eco_Off();	Desliga eco	X	X
void _mode Set_Eco_On();	Liga eco	X	X
int _mode Set_Interface(char *interf);	Parâmetro ='HID', 'PS2', 'COM1' a 'COMn', Fixa Interface. Retorna: -1=ñ encontrado, 0= PS2, 1..99=COM(1..99), 100=HID	X	X
void _mode Set_Page0();	Ativa pagina 0 de scancode.	X	X
void _mode Set_Page1();	Ativa pagina 1 de scancode.	X	X
void _mode Set_Psw_Off();	Desliga eco de senha	X	X
void _mode Set_Psw_On();	Liga eco de senha = *	X	X

PROTÓTIPO	DESCRIÇÃO	WIN	LINUX
void _mode Set_Udg (char caracter, char def0, char def1, char def2, char def3, char def4, char def5, char def6, char def7);	UDG (User Defined Graphics) Permite o desenho de até oito caracteres especiais que ficam armazenados nas posições 0 a 7 da tabela interna do display. Na procedure, o parâmetro “ <i>caracter</i> ” representa o código a ser atribuído ao desenho. “ <i>def 0</i> ” a “ <i>def 7</i> ” definem os elementos ativos da matriz 5x7 do caracter no display. Veja abaixo um exemplo dos valores de “ <i>caracter</i> ” e de “ <i>def 0 a 7</i> ” para formar a letra 'R' na posição 01h da tabela interna	X	X
	caracter=01h		
	def1 = x x x 1 1 1 1 0 = 1Eh		
	def2 = x x x 1 0 0 0 1 = 11h		
	def3 = x x x 1 0 0 0 1 = 11h		
	def4 = x x x 1 1 1 1 0 = 1Eh		
	def5 = x x x 1 0 1 0 0 = 14h		
	def6 = x x x 1 0 0 1 0 = 12h		
	def7 = x x x 1 0 0 0 1 = 11h		
void _mode Update_Leds_Off();	Não atualiza LED's	X	
void _mode Update_Leds_On();	Envia atualização dos LED's para a porta auxiliar do teclado serial	X	

**Nota sobre a função GetOperatingSystem:**

Retorno da função:

WINDOWS:

-1=Desconhecido;	0=Win95;
1=Win98;	2=Win98se;
3=WinME;	4=WinNT;
5=Win2000;	6=XP;
7=XPpro64;	8=Server2003;
9=HomeServer;	10 = Server2003R2;
11= Vista;	12 = Server2008;
13 = Server2008R2;	14 = Win7;
15 = Win8;	

LINUX:

22, 24 ,26 para Kernel 2.2, 2.4, 2.6.

**Nota sobre a função Redirect:**

A função Redirect serve como driver para os teclados Smak(sem display) seriais e USB(serial virtual), possibilitando que os teclados sejam usados normalmente como se fossem teclados HID.

Devido à memorização de interface mencionada anteriormente, para que seja possível trocar o teclado de porta USB, é necessário um reset\_interface.

**Nota sobre a função Send\_Disp\_Ctrl:**

O controle do display do teclado pode ser acessado diretamente através do comando "Send\_Disp\_Ctrl", a tabela abaixo mostra como devem ser composto um comando.

INSTRUÇÃO	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	DESCRIÇÃO
Clear Disp	0	0	0	0	0	0	0	1	Apaga Display e coloca cursor em (1,1)
Home	0	0	0	0	0	0	1	*	Posiciona cursor em (1,1)
									Retorna mensagem deslocada
Mode	0	0	0	0	0	1	I/D	S	I/D = sentido de deslocamento do cursor
									S = Desloca ou não a mensagem
Control	0	0	0	0	1	D	C	B	D = Ativa/desativa display
									C = Ativa/desativa cursor
									B = Ativa/desativa intermitência do cursor
Shift	0	0	0	1	S/C	R/L	*	*	S/C = Movimento do cursor
									R/L = Sentido de deslocamento da mensagem
INTERFACE	0	0	1	DL	N	F	*	*	DL = Número de bits da interface
									N = Número de linhas do display
									F = Tipo de formatação do caracter
CG Ram	0	1	Endereço						Aponta endereço corrente da memória gráfica
DD Ram	1	Endereço							Aponta endereço corrente da memória de dados

Ex.: Para apagar o display enviar 0x01

Para colocar o cursor em home enviar 0x02

Aqui estão apresentados todos os comandos aceitos pelo display, alguns destes comandos podem desconfigurar o display tornando-se necessário um reboot (liga/desliga) do teclado.

**Notas sobre versões de Firmware:**

Teclados PS2:

Versões < 1.93d

Não respondem a Get\_Status. E não são reconhecidos pela busca automática.

Deve ser usado explicitamente Set\_Interface("PS2");

Versões < 1.96

Não respondem a Get\_Firmware\_Version

Versões entre 1.95 e 1.96 travam se for usada a função Get\_Firmware\_Version

Teclados Seriais/USB:

Versões ente 1.92 e 1.93 travam se for usada a função Get\_Firmware\_Version

Teclados HID:

Sem ressalvas.