Preh KeyTec GmbH
Technical Support Team
An der Stadthalle
97616 Bad Neustadt a. d. Saale
Germany
email: support@preh-keytec.de
Web: http://www.preh-keytec.com

Quickstart manual "Programming using the Preh WinProgrammer"

This quickstart manual shall show you the usage of the WinProgrammer and the basics of programming your Preh keyboard using a simple example keytable layout.

First of all, install the WinProgrammer and also the keyboard drivers, if necessary. Please carefully read the important notes in the readme file.

Special themes about "advanced" programming you can find in the annex and also in the WinProgrammer's online help function. If you have further problems when creating your keytable, our support team will certainly be able to help you. The best is to describe your problem in a short email – and please send your keytable (MWF file) along with this email.
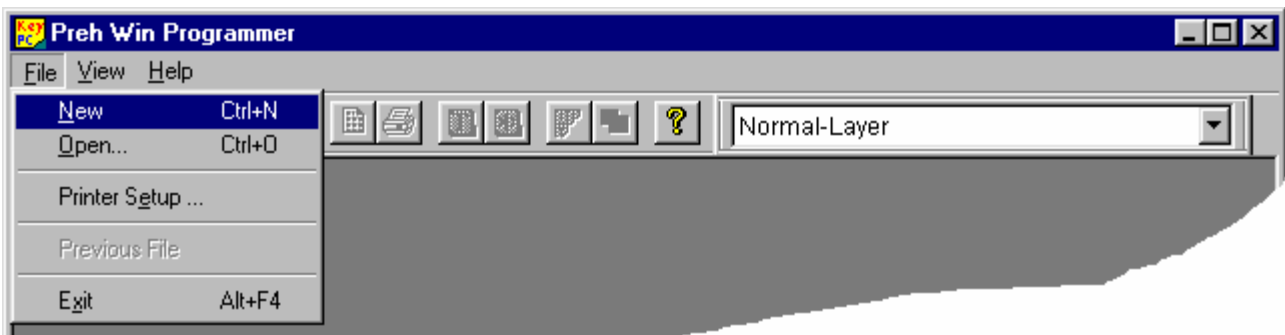
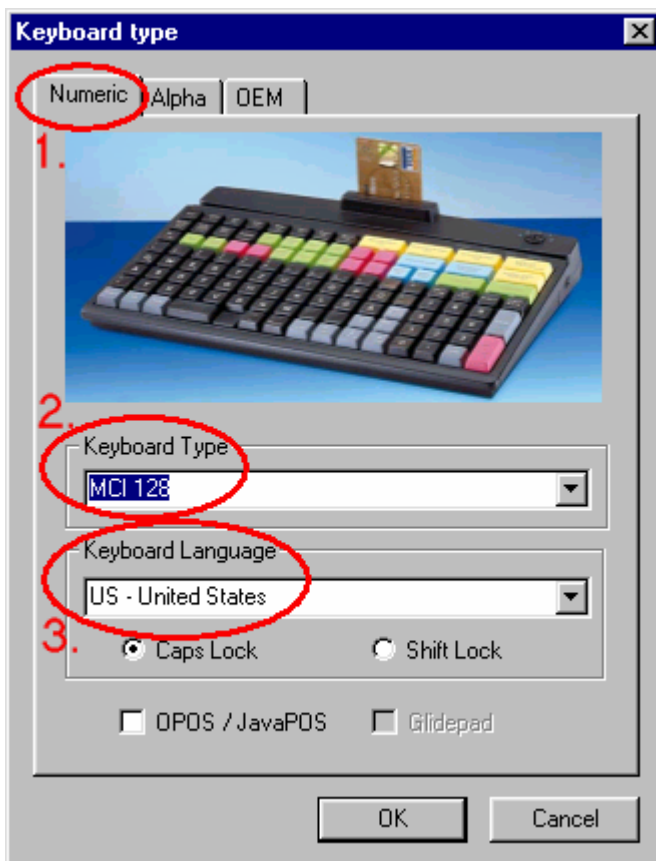## We're starting just here...

Figure 1

Figure 2

Then this dialogue appears. Here you first of all configure the basic keyboard settings:

1. Select keyboard group:
   The keyboard layouts are grouped on the register tabs Alpha, Numeric and OEM

2. Select your keyboard type:
   In our example we use a MCI 128, other layouts as appropriate.

3. Keyboard language and CapsLock behaviour:
   This setting must match the operating system's configuration on the target computer.

Continue by pressing OK.

Additional Informations:
For each type you will see an example picture. Additionally the selected keytable template will be displayed on the Desktop as a preview.

Activate option *OPOS / JavaPOS*, if you want to use the Preh OPOS or JavaPOS drivers. This will cause all module settings to be configured correctly.

Checkmark *Gliedepad* if you're keyboard is equipped with a Glidepad (Touchpad) pointing device. This setting is only useful for alpha layouts, to load a specially adapted keytable template.

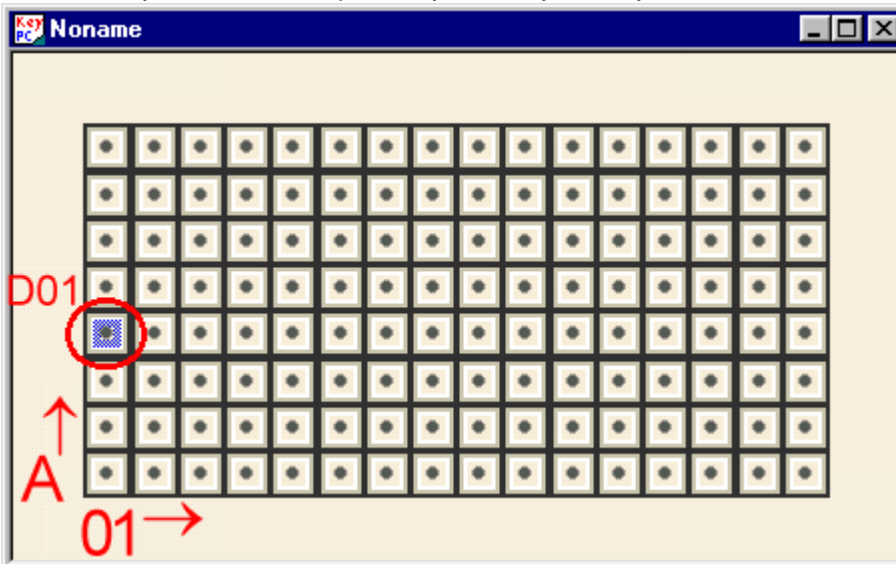Afterwards you see the complete keyboard layout on your screen:

Figure 3

The way of naming the key positions is shown in red letters (Figure 3):
- Using letters, starting from the lower left side, towards the top.
- Using numbers towards the right

Programming of our example key D01 on several layers:

On the highlighted key position **D01** we would like to have the following programming:
- `Normal-Layer!!!{Return}`   when "nothing else is pressed ", i.e. when no special status is active
- `Shift-Layer!!!{Return}`   when "Shift active", i.e. D01 pressed together with <Shift>
- `Control-Layer!!!{Return}`   when "Control active", i.e. D01 pressed together with <Control>

To get the programming dialog displayed, just double-click the key position D01:
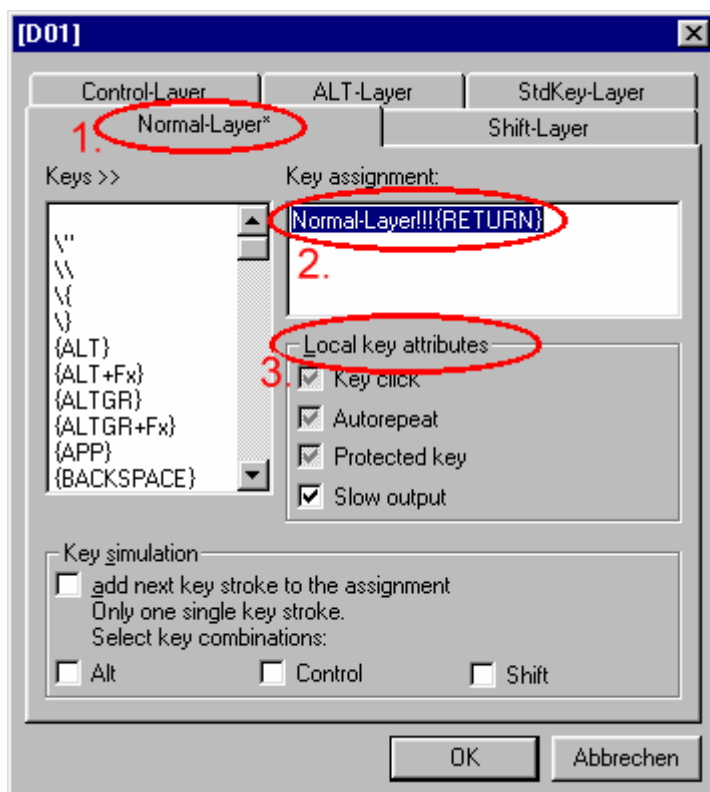

Figure 4

Normal Layer:
On the first step, select the "Normal" tab (see Figure 4, step 1). Here enter the sequence to be put out during "Normal layer active" into the field "key assignment" (step 2). – The key function {Return} can either be entered manually, or by selecting it from the "Keys>>" list on the left side.

Shift-Layer:
Now select the "Shift-Layer" tab. Repeat the steps of above and just enter:
`Shift-Layer!!!{Return}`

Control-Layer:
Now enter the corresponding sequence for the "Control-Layer":
`Control-Layer!!!{Return}`

Additionally you can program "Local key attributes" for each key programming (step 3). Please also see the annotations to the programming dialog. For example you can set a keyclick to be put out together with the programmed sequence as an acoustic feedback.

The **AlwaysActive** layer is described in topic ⇖Advanced Programming - Creating a new customized layer on page 7.

Use Drag&Drop to program the <Shift> and <Control> keys

Finally we need two keys for switching the keyboard status of "Shift" and "Control". The easiest way to program them is to copy those keys from an Preh Alpha keyboard template into your Keytable.
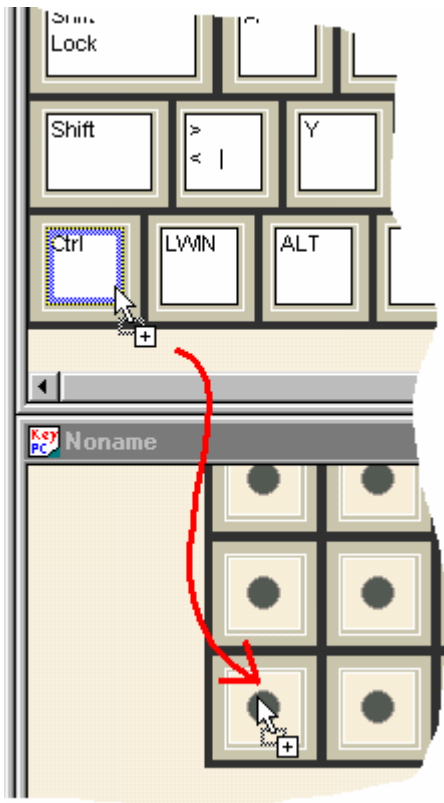

Figure 5

To quickly assign standard keys to your Preh keyboard, please use the following steps for an easy and secure way:

1. Open an additionall MC147 Alpha Keyboard as a template with *File → New*.
2. Select *Window → Tile* to have both layouts on the screen at the same time.
3. Look for the needed key on the MC147 Alpha layout and mark this key.
4. Now use the mouse to move it to the target position in your own layout. Drag&Drop is indicated by a mouse cursor with a small rectangle.
5. Finally adjust the key size using the right and lower key frame, if necessary.

When holding down Ctrl during moving the key, you will execute copying instead of moving. The mouse cursor then contains an additional + symbol.

The procedure described above always copies/moves the entire functionality of the key including the key assignment of all layers and the key labels.

In our example we use this method to copy the Ctrl key to position A01 and the Shift key to position B01.

Alternative method:
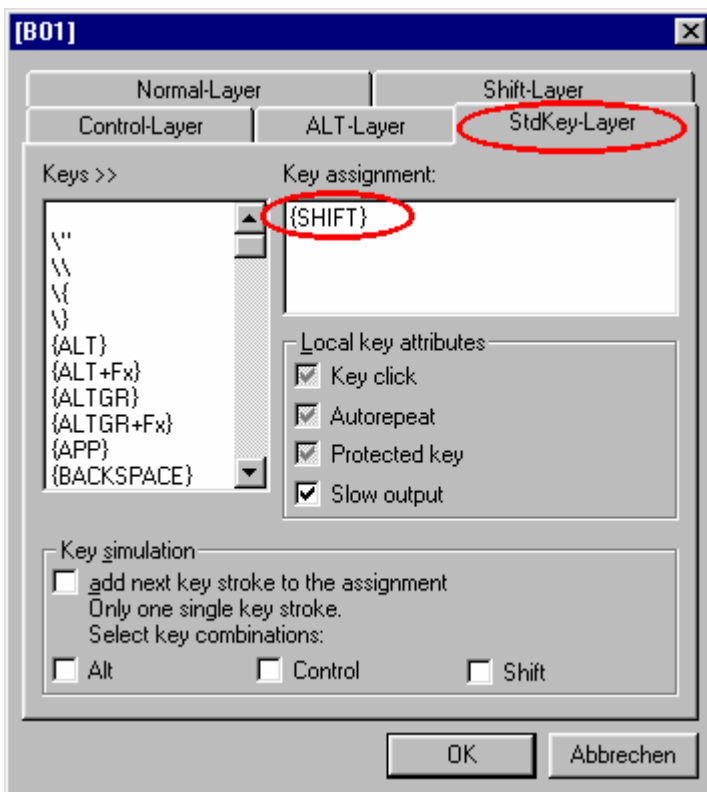**Programming the <Shift> key "by hand":**

Figure 6

Double-click the key position B01 to open the programming dialog.

- Select the **StdKey-Layer** tab
- Enter the key assignment as shown in Figure 6.

All special key functions are entered using the { }. You can select this assignments in the "Keys>>" list on the left side by double-clicking them.

To add the corresponding keyprint to your layout, select menu *Keys → Key label* –or use the context menu using the right mouse button.

The functionality of the special Layer StdKey is explained in the following chapter.

Very important annotations - The StdKey layer functionality:

For a better understanding, as this layer is something special:

- The layer StdKey generates a key assignments which behave identically to a standard key on a MF2 style keyboard.
- This means when an assignment is entered here, it completely maps this key functionality from a standard "MF2" style keyboard to a key position on the PREH keyboard.
- A key press on a standard keyboard normally sends the so-called *Make Code* to the computer and when releasing the key it sends the *Break Code*.
- For getting the normal function of status switching keys (Shift, Alt, Ctrl, etc.) these **must** be programmed on the StdKey layer to work "normally ".
- On the other side – key combinations and strings **must not** be programmed on StdKey layer. Just to remind, of course you also cannot find these on a "standard MF2 keyboard".
- Not getting confused, which assignment has to be sent, additional programming shall not be placed on the other layers.

Just an example, which you also can program, for example on key position A02:

Figure 7

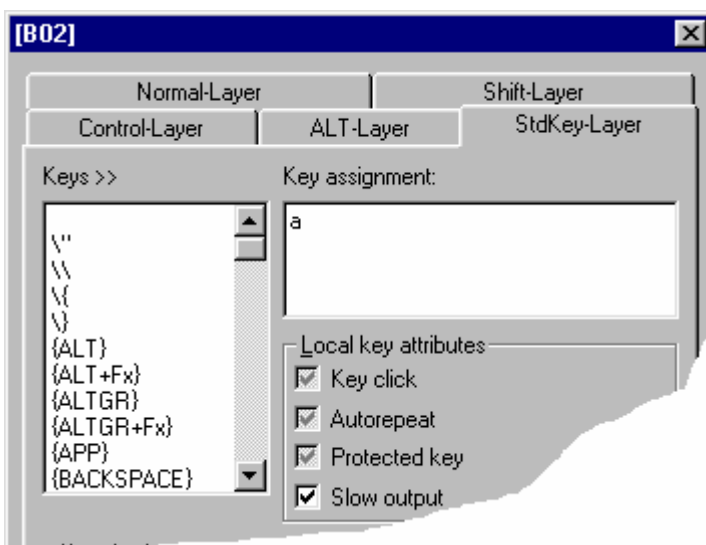When placing the following on the StdKey layer's key assignment: *a*

This will result in the following:
- *a* when being pressed during "normal" state
- *A* when being pressed during "shift" state
- And of course also all the other key combinations which are accessible on a "normal" keyboard.

Further layers do not need to be programmed, this key then works exactly the same way as on every standard keyboard.

Annotations to the programming dialog:

The list "Keys>>":
For entering special key functions (in our example the <Return> key) you also can select the appropriate entry in the list "Keys>>" on the left side by a double-click. The correct notation is then automatically transmitted to the *key assignment* field – in our example {Return}. A list of all supported macros and some notes on key combinations you can find in the Annex: ↳List of Supported Key Functions (Macros) on page 13.

Local key attributes:
In general, all small boxes here are provided with a gray checkmark, which signifies that the global layer settings apply. Nevertheless, you can also define local settings, i.e. settings only effective for this key position and only on this layer, depending on whether you switch the corresponding small box on or off:

☐ Function switched OFF

☑ Function switched ON

☑ The default settings defined in Menu *Configuration → Layer definition* are used

Key Simulation:
If the designation "Add Next Key Stroke to the Assignment" is active, your next key press will automatically be entered into the "Key Assignment" field. You can also select - as an option - one or more of the indicated small boxes for Alt, Control and Shift. For example, if you want to assign the key combination {ALT+Backspace}, you select " Add Next Key Stroke to the Assignment", as well as "Alt". You put the cursor in the "Key Assignment" field and then press only the backspace key.
Attention: Don't forget to switch off the "Add Next Keystroke..." function after use!

Very useful to see what is programmed: *View → Layers*
When enabling this functionality the key positions programmed on the currently selected layer are colored pink. When moving the mouse pointer over the keys, the programmed sequence of this layer appears.
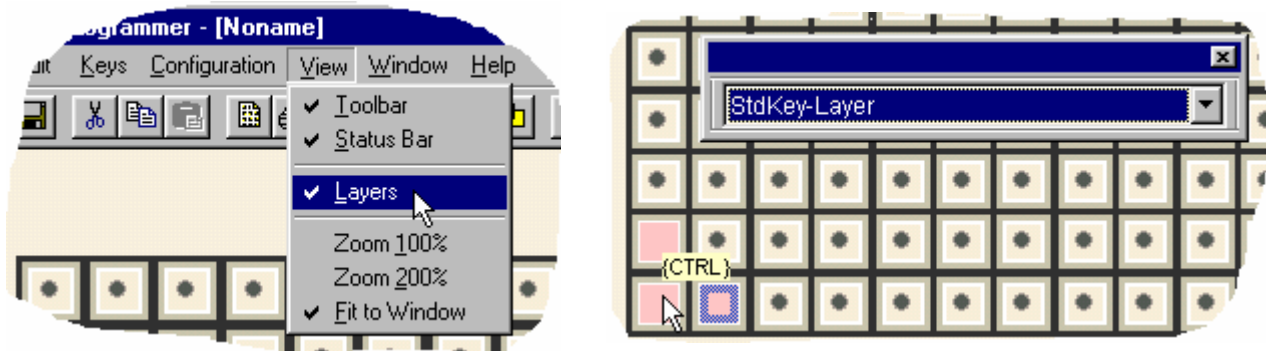

Figure 8

Changing the key size
- Each key cap only has *one mechanically active* position, no matter what size it has (1x1, 1x2, 2x2, etc.).
- Therefore it's recommended to program all covered key positions the same way, not to get confused by differently mounting the key caps afterwards.
- To change the key size, first mark the left upper key position. The actual key position is displayed with a grey-blue border. Then drag it's right and lower edge to the key size needed.
- When "expanding" the key size *now automatically* all covered key positions are assigned with the programming of the left upper position – this is used for Download and for saving the file.

Notes on this:
- Doing it this way, you just can remove for example a double-key, rotate it by 180° and go on without changing anything in kind of a mechanically damaged contact after a long intensive usage. So you get a two-times lifetime in this example.
- If different functions shall be placed on the covered key positions, the key size of course *must not* be enlarged.

Finally: Writing the keytable into the keyboard (Download)

Before starting the download, you should first of all save the new-created keytable using Menu File → Save / Save as... to avoid data loss.

Then start the download the following way:

1. Select menu *File → Update Keyboard*
2. Choose the correct keyboard interface (see notes below)
3. Press OK and follow the next steps to complete the download.

Interface selection depending on the type of your keyboard:
- PS/2 (AT) – if your keyboard is connected via the PC's "normal" keyboard connector.
- USB – if your Preh keyboard is connected via USB.
- COM – if your keyboard is connected to a COM port (RS232) – please note, a special hardware option is required for that. Important: Also the parameters for the normal operation MUST correctly be set in menu *Configuration → Keyboard Setup → Interface*. These parameters are then activated by cycling power. Further notes on that can be found in the Annex.

Please pay attention to the following points for a proper download:
- Do not move the mouse during downloading a keytable.
- During downloading keyboard inputs are not possible.
- If the download fails, follow the troubleshooting in the Annex – and see the WinProgrammer's Readme.
- In case of PS/2 the Preh keyboard always must be the first device, directly plugged into the computer.
- In case of PS/2 on Windows NT, 2000 or XP, the Preh PS/2 keyboard driver must be installed properly. The driver is installed by the Preh DriverPack.
- To enable the download for a USB-equipped PREH keyboard, the PREH DriverPack must be installed properly. Of course the operating system must support USB (minimum Win98SE, Windows 2000 or Windows XP) and the HID devices must be installed properly.

Functional test using some text editor

Afterwards, just start a text editor, such as the Windows *Notepad* or DOS *Edit* and try the things you programmed on key position D01 and B02. Additionally press the new-created Shift and Ctrl key to get the appropriate output.

Testing a PREH keyboard which is connected via **RS232** can easily be done in a terminal software, such as the Windows HyperTerminal: Select "Direct communication with COMx" and configure the RS232 communication parameters as previously done in *Keyboard Setup → Interface*.
Notes: The RS232 keyboard has to be restarted before interface parameter modifications take effect.

Check communication

If downloading the keytable is somehow not possible or the keyboard doesn't react as expected you should check the communication between PC and Preh keyboard. Please select menu *Help → About*, configure the correct interface and press button *Keyboard version*.
If all necessary drivers are installed correctly and the keyboard is connected properly the keyboard will transmit detailed informations about it's hardware configuration. If the keyboard does not report these informations, please follow the steps in chapter ↳Troubleshooting (annex, page 12).

Save and test keytable (Create MWX File)

Using this function you can check the current keytable for compile errors, without downloading the result into the keyboard connected.
Furthermore the downloadable binary MWX file is created after successful compilation. This MWX file format is especially useful to hand it down to your customers for download without changes. To download the MWX binary keytables you can use Copy2MWX (DOS) or C2K (Windows).

Advanced Programming

Advanced Programming - Creating a new customized layer

If you have the need to create a new or modified layer, just follow these steps below. In this example we create a layer which is active if both **<Ctrl> AND <Shift>** are pressed.

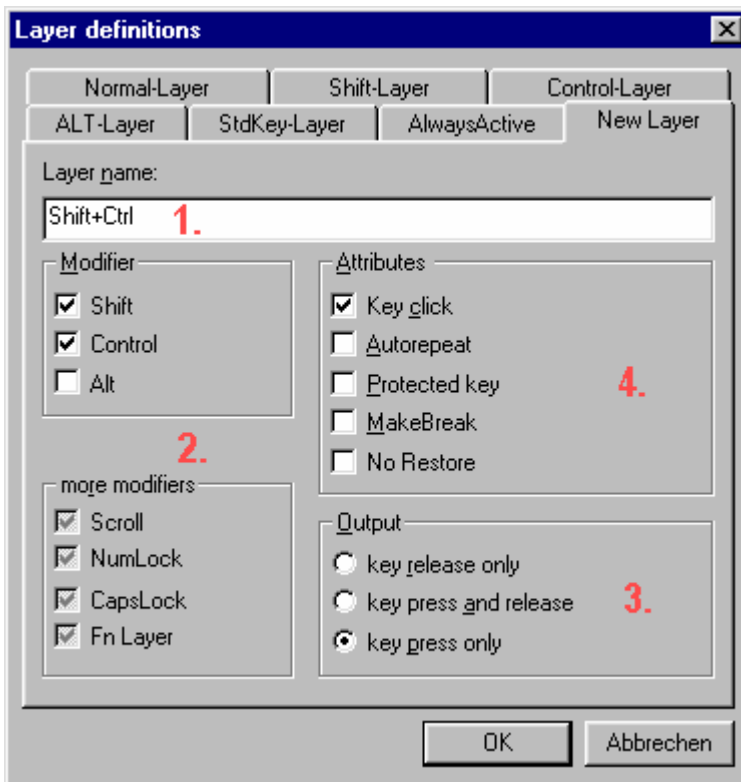Select menu *Configuration → Layer Definition*, and then you will get a dialog like this:

1. First select the tab *New Layer* and enter a name for the new layer

2. Configure the *Modifiers* / *More Modifiers* to program when this new layer should be active

3. Select when the programmed sequences should be output (usually: key press only)

4. Adjust the layer default attributes, for example *Key Click*

Figure 9

Now you can do assignments on this new layer the same way as for at the pre-defined layers Normal, Shift, etc.

Notes on the dialog Configuration → Layer Definition:
- The modifiers / more modifiers checkboxes have the following meaning:
  - ☐ MUST NOT be active / pressed
  - ☑ MUST be active / pressed
  - ☑ Don't care if this level is active or not

- To completely remove the user-defined layer, just open the *Layer definitions* dialog once again, delete the layer's name and press OK. Attention: All the assignments made on this layer also will be deleted!
- The attributes *MakeBreak* and *No Restore* usually should stay not marked.
- The detailed description can be found in the Online-Help index, topic *Layer Definitions*.

Customized layer AlwaysActive:

As an example for a customized layer we have pre-defined this with appropriate layer definitions. All codes placed on this layer will be output ignoring the status of Ctrl, Shift, etc. All modifier / more modifier conditions are set to "ignore" here. The codes is already output at key press, the same behavior as when using for example the Normal layer. Afterwards the previous status is restored.

Attention:

If code is also placed on other layers of this key position, the code on "AlwaysActive" will **not** be output. As the AlwaysActive layer is defined by all modifiers set to "ignore", this layer is evaluated after more exactly defined layers, such as "Normal". You can say, these layers have a higher priority.

Advanced Programming: Configuring the modules

Now you can go on configuring the keyboard's modules. All the modules are configured in the WinProgrammer menu *Configuration → Module setup*. The following keyboard modules can be configured here:
- MSR (magnetic stripe reader via the keyboard line) – which is described in the following points
- Key lock
- Barcode reader module
- Function card/pen
- KVK reader (German health card reader via keyboard line)
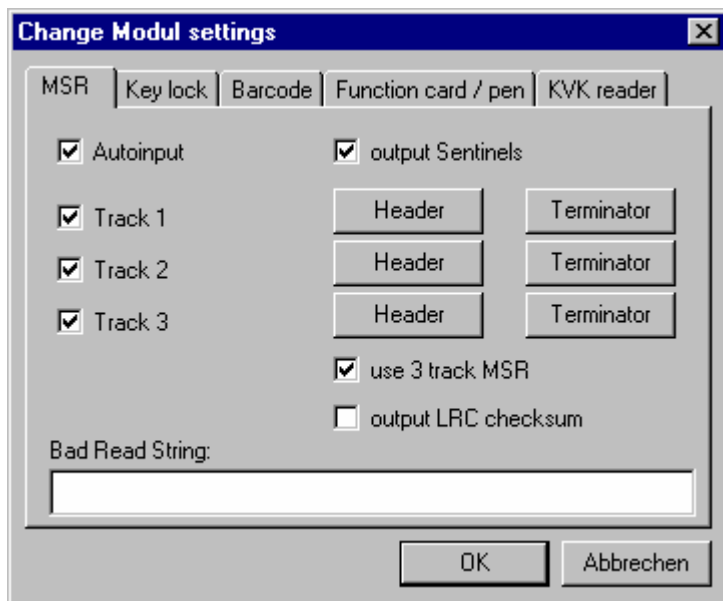
Magnetic Stripe Reader (MSR)

Figure 10

- **AutoInput**

If this checkbox is enabled, the complete data sequence is transferred automatically to the computer after a valid card is swiped. When switching OFF this option, the transmission must be invoked by a special command (see annex for the special commands). Transmission is done via the keyboard line.

- **Sentinels**

Each track on the magnetic stripe contains a so-called start- and an end sentinel. With this checkbox you can select, if these characters should be transmitted or not. See table below for the ISO 7811 definitions:

	Start sentinel (SS)	End sentinel (ES)
Track 1	%	?
Track 2 and 3	;	?

- **Track 1 / Track 2 / Track 3**

Select, which tracks should be transferred to the computer. Disabling the checkbox will suppress Header, card data (incl. the sentinels) and Terminator of this track.

- **Header / Terminator**

For each track you can define a sequence to be output as a *Header* and as a *Terminator.* These sequences are then output before and after each track data. The sequence is defined in the same way as a key assignment.

- **Output Checksum (LRC)**

The XOR-encoded checksum which is on the magnetic stripe can be transferred to the computer. If *output checksum* is activated, this byte is converted the same way as the other characters.

The MSR data are transferred in the following format:
```
<Header1><SS1><Data1><ES1><LRC1><Terminator1>
<Header2><SS2><Data2><ES2><LRC2><Terminator2>
<Header3><SS3><Data3><ES3><LRC3><Terminator3>
```

- **Use 3-track MSR**

Some special 3-track MSR (3-track types manufactured by KDE) can't be identified by some keyboard's electronics types. In this case, this option enables the tracks to be output in the correct order.

- **Bad Read String**

Using the *BadReadString* optionn a string can be defined, which is sent as a data string upon a faulty reading (corrupted or dirty card, data file not according to standard, etc.). The token \# inside the BadReadString is replaced by the error number:

    0 -- No start sentinel recognized
    1 -- Parity error
    2 -- Checksum error

To enable extended error detection within Preh OPOS / JavaPOS, the following Bad Read String has to be entered here: Err\#

Further informations and additional mode switches can be found in chapter ⇘Special Keyboard Modes using BadReadString (annex, page 15).

Important Notes on the options:

Of course the tracks must be supported by the **reader hardware**. The M1 reader type just can read track 1&2, a M2 reader reads track 2&3. The M3 reader type can read all three tracks.

If the **Slow Output** attribute is activated in the header, it's also active for the following track data. The attribute *Slow Output* causes to send the data more slowly. A buffer overflow might occur on the computer when the application is not able to process these data being sent too fast. It is recommended to enable *Slow Output* for the MSR data. The speed for "Slow output activated" additionally can be adjusted here: Menu *Configuration → Keyboard setup → Speed → Slow output speed* and should be set to *medium* for most applications.

You have consider *Slow Output* attribute is not evaluated, if no assignment is made here. So you have at least to write {empty} into the header assignment, to activate slow output for this Track.

Especially for the MSR module it's very important, the keytable's **country setting** matches the keyboard driver (operating system) on the target computer. Otherwise the characters might not be displayed correctly!

Because of the many different kinds of modules, the parameters **Checksum** and **BadReadString** may not be supported by some older keyboard types and magnetic card reader modules!

To calculate the **Checksum** in your software, just XOR-combine all track data including the start and end sentinels and compare the four least significant bits (track1: 5 LSB) with the LRC value of this track.

Example for a MSR configuration

- AutoInput: ON, Sentinels: ON, 3-track MSR: OFF, Checksum OFF
- Track 1 activated, Track 2 and 3 deactivated
- Track1 - Header: msr1
- Track1 - Terminator: end_msr1{Return}

When swiping a card with data on track 1 (DATA1 with the sentinels % and ?) the following sequence will be output – with a line feed at the end:
msr1%*DATA1***?end_msr1**

Note:

The testing of your programmed headers, terminators, etc. should be done in a text editor, such as the Windows *Notepad* or DOS *Edit*. Swipe a card and the data string appears, as previously programmed in the MSR module settings. If wrong sentinel characters appear, you should check if the keytable language matches the driver language.

Annex

System Requirements / Short description of the programming methods

WinProgrammer (Version 1.8 or above)

For the WinProgrammer you need an IBM AT or PS/2 compatible system (80386 or higher). The WinProgrammer runs under Windows9x, WindowsNT and Windows 2000 / XP. To enable the download, the appropriate driver for Windows NT, 2000 and XP has to be installed properly. Please see the Readme file for details about installation and usage.

Preh Programmer (PREH-MWX.EXE)

To work with the Preh Programmer you need an IBM AT or PS/2 compatible system (80286 or higher). The Preh Programmer „PREH-MWX.EXE" (Version 4.1.x and higher) can run under MS-DOS as well as under Windows 3.1, Windows95 or Windows98 in a DOS-box. For newer Windows versions (e.g. Windows XP)

Online Programming

Online Programming is only available for the MC/MWX keyboard family equipped with a daisy chain connector. In addition you need a standard PS/2 (AT) keyboard. Online programming is possible under every Operating System, such as DOS, Windows 3.1, Windows9x, WindowsNT, OS/2 and UNIX.

Download Utilities

If you want to download a previously created keytable (MWF or MWX-file) into the keyboard without using Preh Programmer or Win Programmer, you have the choice of our download utilities:

C2K (Copy to keyboard) Download Utility

If you prefer to work under Windows 9x, Windows NT, 2000 and XP use our C2K utility (Copy to keyboard). This is able to download both the MWX and MWF files. Please see the Readme file for details about installation and usage.

Copy2mwx.exe

If you prefer to work under DOS you can use the COPY2MWX.EXE program. You can find this program in the Preh-MWX programmer package.

> Syntax: `copy2mwx filename.mwx <Return>`

Try adding the parameter `/w` if it doesn't work in a Windows 9x DOS box:

> Syntax: `copy2mwx /w filename.mwx <Return>`

Of course this is not possible in DOS box of Win2000 and WinXP.
A similar copy2mwx utility is also available for OS/2 and Linux OS on request.

Differences WinProgrammer – Preh-MWX (DOS) – Online Programming

	Win Programmer or C2K Utility	Preh-MWX.EXE or Copy2mwx.exe	Online Programming [1]
MS-DOS, Windows 3.x	No	Yes [2]	Yes
Windows95/98/Me	Yes	Yes	Yes
WindowsNT, 2000, XP	Yes [3]	No	Yes
OS/2	No	No [4]	Yes
Unix / Linux	No	No [4]	Yes
Read keytable	No [5]	Yes	Not necessary
Write keytable	Yes	Yes	Not necessary
Save keytable	Yes	Yes	Not necessary
Max. number of levels	128	128	16
Key label printing	Yes	No	No

[1] Online programming is only available for old MWX/MC keyboards equipped with daisy chain connector.
[2] Use new copy2mwx.exe (dated 2005) to program MCI keyboards (USB/PS2 interface) in PS2 mode. USB mode not possible for DOS.
[3] When using PS2 interface, installation of PREH PS2 keyboard driver is required for writing keytable into the keyboard.
[4] Utility for downloading the MWX keytable file is available, installation of a special keyboard driver is required.
[5] Function is available using the "Upload" option for the C2K utility: The binary MWX keytable can be read out in case of service.

Interface settings (AT, USB, RS232)

The Preh programmable keyboards basically can be configured to run these interfaces/protocols:
- **PS/2** (AT)
- **USB** (available if keyboard is equipped with USB interface)
- **RS232** (only for MWX/MC128 family with optional factory-fitted RS232 module)

Important notes:
Of course the individual capabilties of your keyboard depend on the hardware and the cabling the keyboard is equipped with.

The computer's bios usually will display a "keyboard error" message, if the keyboard's interface setting is mis-configured. In this case, please use one of the following key combinations to assign the correct interface.

Special Key Combinations

Below you find some helpful key combinations for configuring and troubleshooting Preh programmable keyboards. Press and hold down one of these key combinations during powering-on the computer/keyboard. You should hold the combination for about 5 seconds. Successful switch over is usually indicated by beep tone(s).

Please use the appropriate **key combinations** for the Preh keyboard family you're using:

Family Key Combination	MWX 84,128 MC 25,35,80,84,128 PC-POS	MCI MC147, MC140, MF112
A01 + B01	PS2 (AT) protocol	Autodetect: PS/2 or USB Protocol [3]
A01 + C01	XT (old 8086) protocol	Activate PS/2 Interface
A01 + D01	RS232 protocol with Preh default parameters [1]	Activate USB Interface [3]
A01 + A03 + A05	Activate a test keytable to check all key positions for electrical function. [2]	
A01 + A03 + D01	-	Restore the factory default keytable [4]

Notes:
[1] RS232 protocol is only available for MWX/MC with optional factory-fitted RS232 module (Default: 9600-8-O-1)
[2] Each key press and each key release should output a beep and some default key code. The stored keytable will not be changed. Cycle power to get the keyboard back in "normal" operation.
[3] USB and Autodetect is not available for MCI keyboards with "PS2 only" electronic boards. This boards are only capable PS2 protocol.
[4] The actually programmed keytable will be replaced by the factory default keytable. Also the module settings will be reconfigured to factory defaults.

Troubleshooting

Many problems are caused by loose or incorrectly connected cables. You should therefore first make sure that all cables have been properly connected. In addition you should also check any programming that you have carried out.

Problem	Possible cause	Remedy
Computer indicates "keyboard error" during start-up	• cable not correctly plugged in • cable defective • incorrect keyboard interface initialized • Timing problems between keyboard and computer	• check cable connections • replace keyboard cable • re-initialize keyboard interface • Switching off all modules which are not used via PREH WinProgrammer
Preh keyboard does not work, although the daisy-chained keyboard works	No keyboard assignment stored in the internal keyboard EEPROM	Generate keytable and download into your keyboard using the Preh WinProgrammer
Preh keyboard beeps at every key position, without displaying any characters	A fault has occurred in the transmission of the keyboard assignment table, or the contents of the EEPROM have been modified	Re-initialize keyboard interface (and download keyboard assignment table into the keyboard)
A keyboard buffer overflow occurs when transmitting long strings (e.g. MSR data)	Output speed of Preh keyboard too high.	Enable the „Slow output" attribute using the Preh WinProgrammer.
Modules do not function, or do not function correctly	Module is disabled.	Enable the module using the Preh WinProgrammer
Module data for MSR/Keylock is not output Notepad	Keyboard is configured to send module data via OPOS/JavaPOS channel (factory default for MCI family, especially in USB mode).	Use OPOS/JavaPOS demo application for testing – or disable using "OPOS Headers" using the Preh WinProgrammer.

Technical Support

- Please refer to your keyboard manual for additional informations.

- Consult the Keyboard FAQ pages on the Preh KeyTec website.

- Also please check the keytable and the module settings of your keyboard.

If all the steps above did not help to solve your problem:

- Contact your local Preh KeyTec distributor in order to get technical assistance

- Contact the Preh KeyTec technical support:
  techsupport@preh.com   for North and South America.
  support@preh-keytec.de   for worldwide support.

- Visit the Support Area on the Preh KeyTec Website:
  http://www.preh-keytec.com

List of Supported Key Functions (Macros)

The key functions (Macros) are usually entered by just double-clicking the entry in the "Keys>>" list on the left side. You also can type them manually – then pay attention to enter them in {} (curly brackets), i.e. {F1} for the F1 key.

Available Macros	Description + Annotations
\"	Quotation mark (sign itself is reserved code – also for the key label)
\\	Backslash (sign itself is reserved code – also for the key label)
\{	Curly brackets (sign itself is reserved code – also for the key label)
\}	Curly brackets (sign itself is reserved code – also for the key label)
\^	Caret (sign itself is reserved code)
{ALT}	(left) Alt key
{ALT+Fx}	Alt + Function key (x: number 1..12)
{ALTGR}	Right ALT (AltGr) key
{ALTGR+Fx}	AltGr + Function key (x: number 1..12)
{APP}	GUI (Win) application key
{BACKSPACE}	Backspace key - abbreviation: {BS}
{BREAK}	Break key (= CTRL + Pause)
{CTRL}	(left) Ctrl key
{CTRL+Fx}	Ctrl + Function key (x: number 1..12)
{DEL}	DEL key (numeric keypad)
{DELAY}	0.5 sec output delay
{DIV}	Division key on numeric keypad
{DOWN}	Moves cursor down
{EMPTY}	Empty string
{END}	End key
{ENTER}	ENTER key
{ESC}	ESC key
{F1}	Function key F1 ... F12
{FCx}	Abbreviation for {CTRL+Fx}
{FN_OFF}	Switches Function key modifier OFF (see also Key-FN)
{FN_ON}	Switches Function key modifier ON (see also Key-FN)
{FSx}	Abbreviation for {SHIFT+Fx}
{HOME}	Home key
{INS}	Insert key
{KEY-DEL}	DEL key (multi layer macro)
{KEY-DOWN}	Cursor down (multi layer macro)
{KEY-END}	END key (multi layer macro)
{KEY-FN}	Function key modifier on/off (press/release similar to Fn key of laptop)
{KEY-HOME}	Home key (multi layer macro)
{KEY-INS}	INS key (multi layer macro)
{KEY-LEFT}	Moves cursor to the left (multi layer macro)
{KEY-N00}	Numerical block 00 key (multi layer macro)
{KEY-PGDN}	PageDown key (multi layer macro)
{KEY-PGUP}	Page Up key (multi layer macro)
{KEY-PRTSC}	Print Screen key (multi layer macro)
{KEY-RIGHT}	Cursor right (multi layer macro)
{KEY-UP}	Cursor up (multi layer macro)
{LEFT}	Cursor left
{LWIN}	Left GUI (Win) key
{MAKENUM}	NumLock key (Make-Code only; to be placed on StdKey layer)
{MAKESCROLL}	ScrollLock key (Make-Code only; to be placed on StdKey layer)
{MAKESHIFTLOCK}	CapsLock key (Make-Code only; to be placed on StdKey layer)
{MINUS}	Minus key (Numeric block)
{MUL}	Multiplication key (Numeric block)
{N.}	Delete / Dot key (Numeric block)
{N0}	Numerical block keys 0 ... 9
{NO_DATA}	Suppress the data string (only for e.g. MSR Track Headers)
{NUL}	Null byte (only for RS232 version, equivalent to Ctrl+2)
{NUMLOCK}	NumLock key
{PAUSE}	Pause key

{PGDN}	Page Down
{PGUP}	Page Up
{PLUS}	Plus key (Numeric block)
{POSBarcode}	OPOS Barcode header / terminator
{POSFC}	OPOS Functioncard/-pen header / terminator
{POSKey001} ...128}	OPOS Key001 ... 128 scancodes
{POSKeylock}	OPOS Keylock header / terminator
{POSMSR1}	OPOS MSR Track1 header / terminator
{POSMSR2}	OPOS MSR Track2 header / terminator
{POSMSR3}	OPOS MSR Track3 header / terminator
{PRTSC}	Prtint Screen key
{RCTRL}	Right Ctrl key
{RESET}	Ctr + Alt + Del Macro
{RESETSTATUS}	Macro sending the release codes of both Shift, Ctrl, Alt and GUI keys
{RETURN}	RETURN key
{RIGHT}	Cursor right
{RSHIFT}	Right Shift key
{RWIN}	Right GUI (Win) key
{SCROLL-LOCK}	ScrollLock key
{SHIFT}	(Left) Shift key
{SHIFT+Fx}	SHIFT + Function key F1 ... F12
{SPACE}	Space Bar (in a string, this macro must be used at the end of a line)
{STAR}	Multiplication key (Numeric block)
{SYS}	Switches on SysRq function
{SYSBREAK}	Switches off SysRq function
{TAB}	Tab key
{UP}	Cursor up

Some examples of key combinations: {Ctrl+F5}, {Ctrl+a}, {Delay}, {Alt+x}, {SHIFT+{ALT+F4}} ...

Important notes:
For key combinations the lowercase letters have to be used. Using uppercase letters would result to a keycombination with a shifted character. See example below.

{Ctrl+a} ≠ {Ctrl+A}

{Ctrl+A} = {Ctrl+{Shift+a}}

Special Commands for PREH Keyboards

With the following commands you can control the PREH-Keyboard and Extension Modules. For an easy implementation into your Windows application, you should use our MWX function DLL. This can be downloaded from our website. See the documentation of the MWX function DLL for details.

Command	Response	Parameter	Response	Function
EC	FA	<LCD data>		send data to LCD
ED	FA	<LED data>		set LED's
EF...	FA / FE			Special commands for PREH keyboards
EF 03				init default keytable
EF 05				init test table
EF 10	<ID string>			read ID string
EF 18				MSR autoinput on
EF 19				MSR autoinput off
EF 1A	<MSR data>			MSR read data
EF 1E				KL autoinput on
EF 1F				KL autoinput off
EF 20	<KL data>			KL read data
EF 21				BCR autoinput on
EF 22				BCR autoinput off
EF 23	<BCR data>			BCR read data
EF 2B	"Beep"			Keyboard Beep

Special Keyboard Modes using BadReadString

Basically the MSR module's *BadReadString* option is used to define a text which is sent instead of the module data in case of erraneous magnetic card reading.
In addition various switches can be entered in the *BadReadString* option to configure special keyboard modes. The special keyboard mode switches listed below are available for most of the new Preh keyboard families having firmware dated 2001 or newer.

Special keyboard mode switches for BadRead String:

\#   MSR: Output the Error number for faulty MSR swipe reading:
          0: No start sentinel recognized, 1: Parity error, 2: Checksum error

\A   MSR: Support AAMVA

\C   MSR: Support CADL

\D   MSR: Empty Tracks: Don't output anything

\N   MSR: Don't output Sentinels for BadReadString

\S   MSR: Output Sentinels for BadReadString (using default sentinels).

\O   MSR: Output Sentinel % on Track 3 in case of BadReadString

\U   Universal language: Module data are output as Sequence of ALT-Combinations (e.g. A = ALT65)

\X   MSR: Error on all three tracks: Don't output anything (no data, no header, no terminator)

\R   MSR: Use old-style Caps behaviour.

\B   MSR: Beep in case of BadRead.

\W   Wincor mode for MSR/Keylock.

\L   MSR: SlowOutput activated.

\G   MSR: Empty tracks: Mark them with an asterisk *

Attention:
- Not all the switches are available for every Preh keyboard model. Especially older keyboard models do not support those switches at all.
- Usually the switches can be combined by entering them "in a row". Example: Err\#\A\C
- All characters must be entered into the BadReadString textbox as listed above – in *capital* letters.
- Usually you have to cycle power to activate the new parameters.

Copyright

Trademarks

All trademarks or product names quoted in this user manual are the property of their respective owners.

Examples: Microsoft, MS-DOS, Windows, Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP are registered trademarks of Microsoft Corporation.