PrehKeyTec GmbH Technical Support
Scheinbergweg 10
97638 Mellrichstadt - Germany
email: support@prehkeytec.de
Web: http://support.prehkeytec.com

Quickstart manual "Programming with the WinProgrammer"

This Quickstart manual shall show you the usage of the WinProgrammer and the basics of programming your PrehKeyTec devices using a simple example.

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 of this manual and also in the WinProgrammer's online help. 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 an email – and please send your keytable (MWF file) along with this email.

We're starting here...

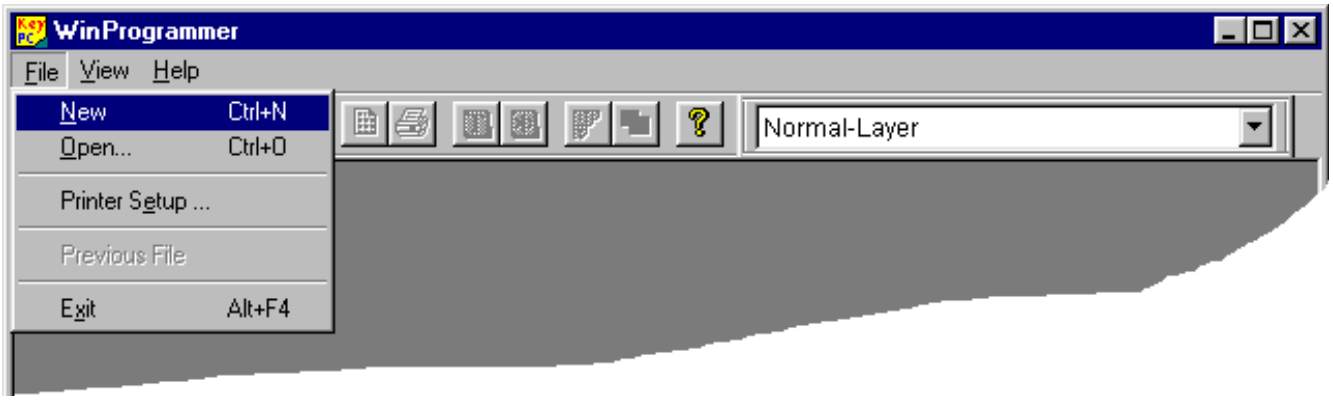


Figure 1

Figure 2

Then the keyboard type dialogue appears. Here you have to configure some basic keyboard settings:

1. Select keyboard group:
   The keyboard layouts are grouped on the register tabs Alpha, Numeric, Modules 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 Information:
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 intend to use our OPOS / JavaPOS services. This causes the modules MSR and keylock to be configured correctly.

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

Programming Standard Keys using Drag&Drop

Drag&Drop is the easiest way to program so-called standard keys like "Shift", "Control" and all the other alphanumeric keys. Simply copy such keys from a Default Layout template into your keyboard. This includes key code programming and also the key label.

To quickly assign standard keys to your PrehKeyTec devices in an easy and safe way:

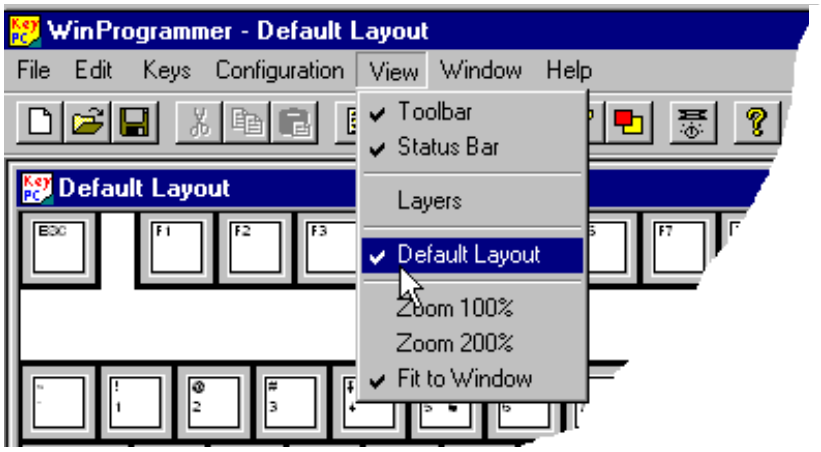


Figure 3

1. To show the template just enable *View* → *Default layout*.
2. This way you also can close the template keytable.

The WinProgrammer automatically opens the template using the same language as selected for the currently activated keytable.

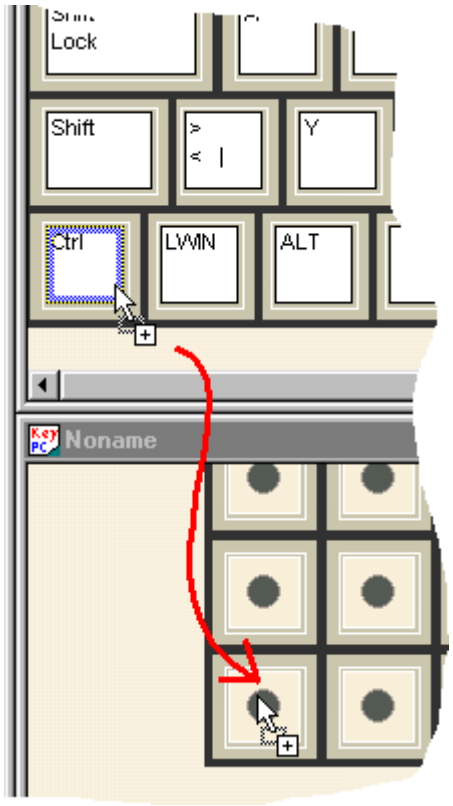


Figure 4

1. Select the source key on the Default layout by a mouse click.
2. Hold down the left mouse button and move the key to the target position into your own layout.
3. Finally adjust the key size using the right and lower key frame, if necessary.

Notes:
Drag&Drop is indicated by a mouse cursor with a small rectangle.

When holding down the Ctrl-Key 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 label.

In our example we use this method to copy

- **Left Ctrl key**
- **Left Shift key**

For further information on standard keys and the StdKey Layer please also refer to Important annotations - The StdKey layer functionality on page 6

Numbering the key positions

In our example (MCI128) you will see the following blank layout:

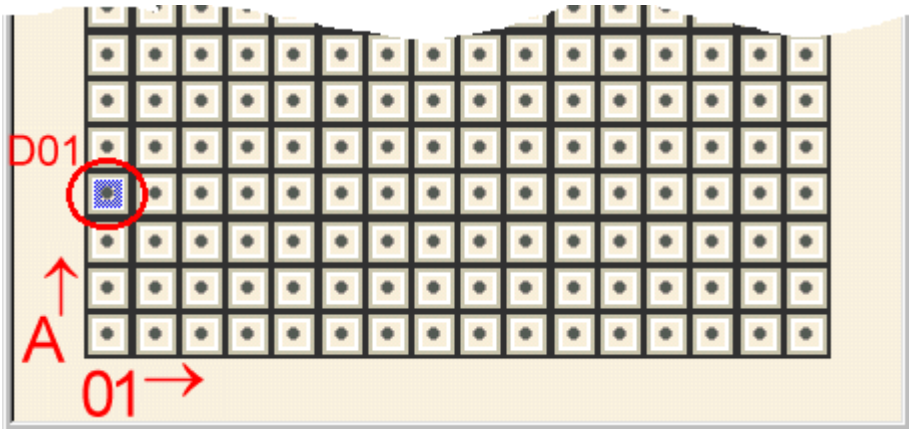


Figure 5

Numbering is done the same way for Numeric and also Alpha keyboards - as shown in Figure 5:

- Using letters (A, B, C…) starting from the lower left side, towards the top.
- Using numbers (01, 02, 03…) from left towards right.
- The key position is displayed in the title bar of the key assignment dialogue.

Language translation settings – MultiLanguage mode:

For having correct output of key sequences and especially module data, the PrehKeyTec keyboard and the operating system's keyboard driver must be set to identical translation. Since ever our programmable keyboards and the programming software support 8 basic languages: US, GR, FR, UK, SG, SF, IT, SP.

Starting with WinProgrammer 2.3 you can easily add additional languages.

Select *Configuration* → *Keyboard setup* to enter the following configuration dialogue:

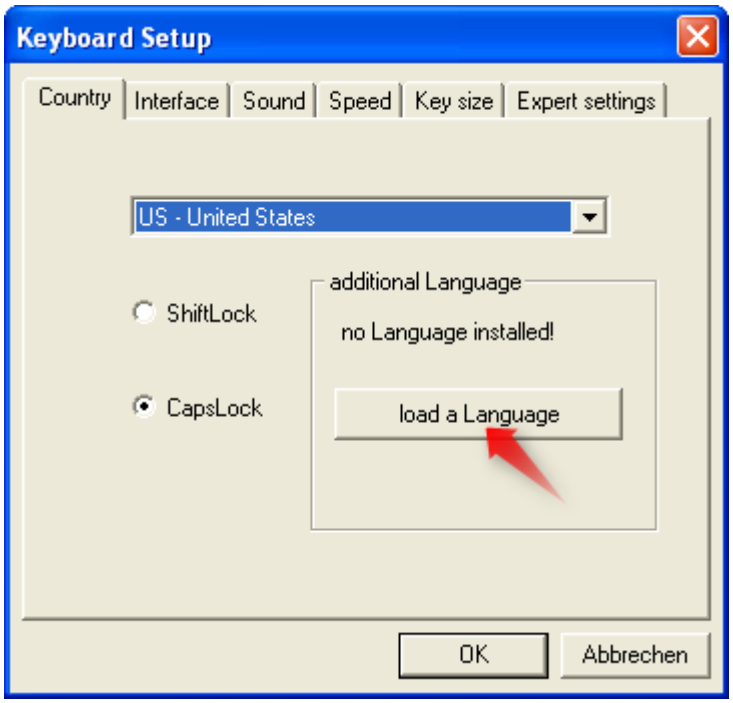


Figure 6

Important notes:

- The "MultiLanguage" feature is fully supported by latest MCI family keyboards having firmware 605/3090 or newer.
- Usually you have to cycle power to activate MultiLanguage mode.
- Active MultiLanguage mode is indicated by the addendum "ML" in the version string.
- Older keyboard generations (e.g. MC 128 W/X) do only partially support the Multi-Language feature:
  a) Key sequences are translated correctly by the WinProgrammer's compiler.
  b) Translating variable module data (e.g. MSR codes) is not supported. Such data can only be modified by the classic ASCII-Convert-Table method here.

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>

Just double-click the key position D01 – then you will see the following programming dialogue:

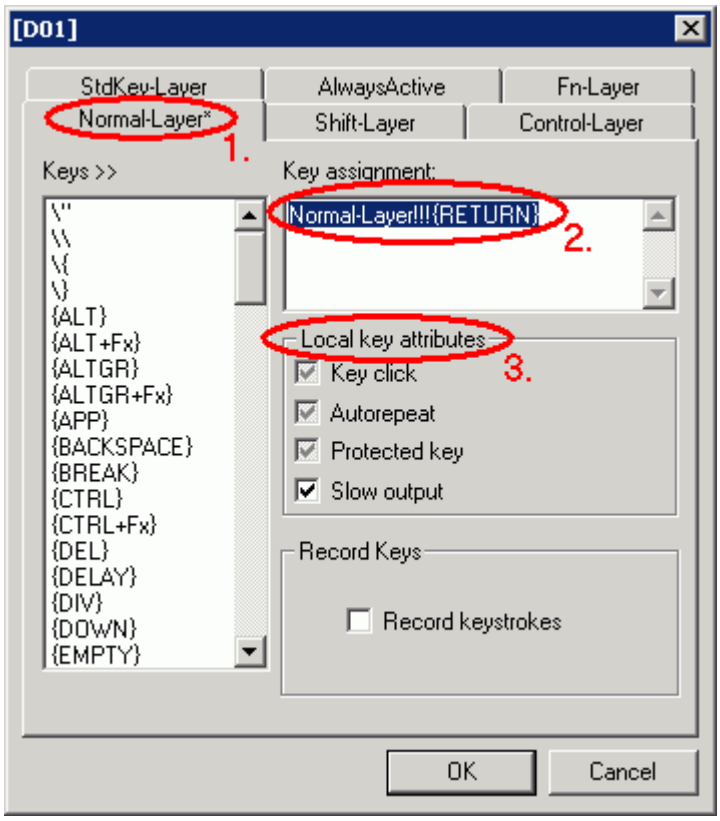


Figure 7

Normal Layer:
Step1: Select the "Normal" tab.
Step2: Enter the sequence to be output during "Normal layer active" into field "key assignment":
`Normal Layer!!!{Return}`

Shift-Layer:
Now select the "Shift-Layer" tab. Repeat step 2 of above and just enter:
`Shift-Layer!!!{Return}`

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

As a third step you can configure "Local key attributes" for each key programming. Please also see the annotations to the programming dialog. For example you can configure a key click as an acoustic feedback of the key press.

The key function {Return} can either be entered manually, or by selecting it from the "Keys>>" list on the left side.

Annotations on the programming dialogue

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 17.

Local key attributes:
You can 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 layer's default settings defined in Menu *Configuration* → *Layer definition* are used

As default, the checkboxes are provided with a gray checkmark, which signifies that WinProgrammer's global layer settings apply. To be independent of the WinProgrammer's layer configuration, we recommend to using local settings on/off only.

Record Keystrokes:
When activating option "Record Keystrokes", your next key presses will automatically be entered into "Key Assignment" of the currently selected layer. Example: 234234{F5}{TAB}{TAB}{RETURN}
Only single keystrokes are recorded - key combinations like {Alt+F4} have to be entered "manually".

Maximum 180 characters (macros and normal text) are allowed to be entered into field *key assignment*.

Customized Layers:
The layers **AlwaysActive** and **Fn-Layer** are described in topic ↳Customized layers AlwaysActive and Fn-Layer on page 10.

NEW: For latest MCI series keyboards a ↳Simplified Layer Concept "EasyLayer" was developed and integrated in the WinProgrammer since Version 2.3. Details are described on page 11. Please consider the system requirements.

Important annotations - The StdKey layer functionality

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

- The layer StdKey generates a key assignment which behaves 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 PrehKeyTec devices.
- 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*.

This results in the following consequences:

- 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 are **not allowed** on StdKey layer. Just to mention, of course you also cannot find such keys on a "standard MF2 keyboard".
- Not to get confused, which assignment will be sent, additional programming shall not be placed on any other layer of this key position.
- Especially when using multilayer macros like {KEY-UP} all other layers must be left blank. For placing such functions on different layers, use the alternative macros like {Up} instead.
- As for any standard USB keyboard the key repeating (Autorepeat) is basically done by the operating system. Thus for StdKey assignments an unchecked "Autorepeat" attribute will show any effect. If you need to have no key repeating for such key assignments, please move them to AlwaysActive layer.

Example for manually programming key position A02 using StdKey layer:

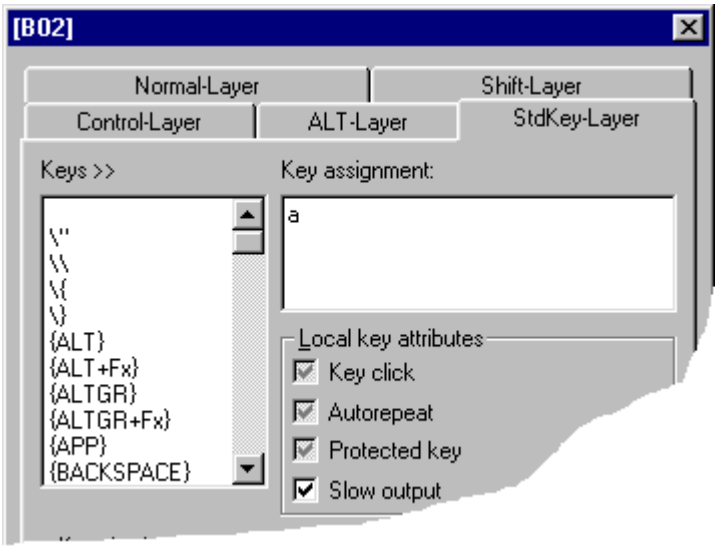


Figure 8

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 "standard" keyboard.

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

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* or *Save as...* to avoid data loss.

Execute the download as described below:

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 PrehKeyTec device is connected via USB.

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 on these layers.

Download problems - 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 PrehKeyTec device. 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 report detailed information about its hardware configuration. If the keyboard does not report such information, please follow the steps below and also see chapter Troubleshooting (annex, page 16).

Please pay attention to the following points for a proper download:

- During downloading keyboard inputs are not possible.
- If the download fails, follow the troubleshooting in the Annex – and see the WinProgrammer's Readme.

USB Devices:

- Downloading to an USB-connected PrehKeyTec device requires no special hardware drivers.
- Of course the operating system must support the usual USB-HID devices - operating system standard drivers are installed automatically by Plug&Play. Check Windows' device manager for proper installation.
- Just the registration of "keyhook.dll" is required for various PrehKeyTec software packages. That is done automatically by WinProgrammer or DriverPack.

PS/2 Devices:

- PrehKeyTec PS/2 device must always be the first one, directly plugged into the computer.
- Please do not move the mouse during downloading a keytable.
- On Windows NT/2000/XP/or newer the PrehKeyTec PS/2 keyboard driver must be installed properly. The appropriate driver will be installed automatically by our DriverPack when selecting PS2.
- A reboot is mandatory. After rebooting, the device manager of Windows 2000/XP/etc. must finally list a "PrehKeyTec PS/2 Keyboard" at branch "keyboards".
- This PS2 driver does NOT support 64bit platform. Please use 32bit OS or move to our USB keyboards.

Useful functions

File → *Save and test keytable* - Create binary MWX File

This function checks the current keytable for compile errors, without downloading the result into the keyboard connected.

The downloadable binary MWX file is created after successful compilation. The content of the MWX file is identical to the “executable” configuration you would download into a connected keyboard.

This MWX file format is especially useful to hand it down to your customers for download without changes. To download the MWX binary keytables the customer then uses Copy2MWX (DOS) or the Download Utility C2K (Windows).

View → *Layers* - Very useful to see how the keys are programmed

When enabling this feature 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.

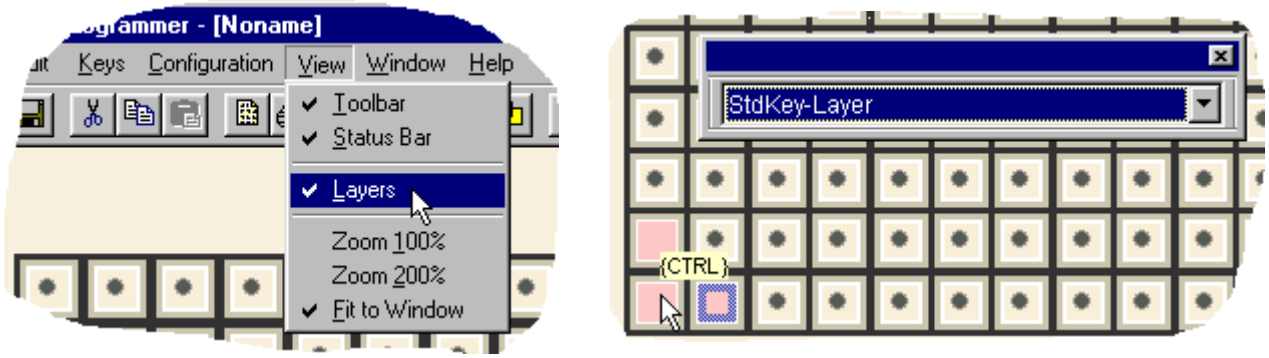


Figure 9

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 the right and lower edge to the key size needed.
- When "enlarging" 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 also for saving the file.

- As an advantage, you just can remove for example a double-key, rotate it by 180° and go on without changing anything in case 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 *must* remain 1x1.

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 as long as both **<Shift> AND <Ctrl>** are pressed.

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

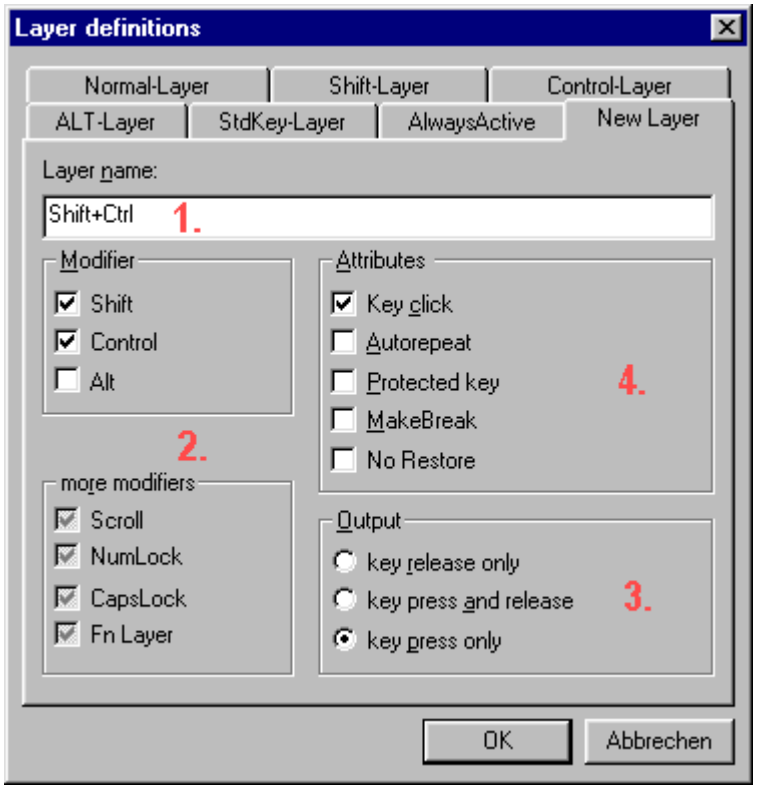


Figure 10

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*
5. After OK you will be asked if the settings should be stored for future use – please see notes below.

Now you can place key assignments on this new layer like on 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
  - IGNORE if this level is active or not
- To completely remove the user-defined layer, just open the *Layer definitions* dialog once again, mark and 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.
- A detailed description can be found in the Online-Help index, topic *Layer Definitions.*

Storing the layer settings

Layer settings are stored inside the start configuration file *preh.ini*. This will help to use customized layer settings for your different keyboard layouts.

After changing the layer settings you will be asked if the modifications also should be saved in the start configuration. Select yes, if this should be saved for future use. Otherwise changes just will apply to actual session and the current layout.

Menu *File* → *Default configuration* will reset the start configuration to the WinProgrammer's default configuration. To use this function you first have to close all opened layouts.

Customized layers AlwaysActive and Fn-Layer

We have predefined two customized layers with appropriate definitions. All codes placed on these layers will be output ignoring the status of Ctrl, Shift, etc. All modifier / more modifier conditions are set to "ignore" here.

Both layers provide similar behavior as for example the Normal layer: The assignments are output already when pressing the key. After finishing the output the previous keyboard status will be restored.

AlwaysActive:
This layer is always output, no matter if Shift, Ctrl, etc. are pressed. If you have to place *only one* function to this key position – AlwaysActive is therefore is mostly better than using Normal layer.

Attention:
When placing assignments also on other layers of this key position, the code on "AlwaysActive" might **not** be output. Because the AlwaysActive layer is defined by all modifiers set to "don't care", you can say it has very low priority. Therefore more exactly defined layers, such as "Normal", Shift, etc. are output first.

Fn-Layer:
The Fn Layer is ideal to create a configuration with additional assignments on a second layer. The Fn attribute is handled only inside the keyboard and therefore it's independent of the PC's keyboard attributes.

Simple macros are available for switching:

- {FN_ON} and {FN_OFF} are used for permanently switching the Fn-Layer on/off.
- {KEY-FN} placed on StdKey layer will result in a function key like on a notebook.

Example for usage of the Fn-Layer:

- Sequence on Normal layer: `Testing Normal Layer{Return}`
- Sequence on Fn-Layer: `Testing Fn Layer{Return}{FN_OFF`

Results:

- The above test key will usually output the demo sequence placed on normal layer.
- If Fn-Layer was activated by a second key using `{FN_ON}` - the testing key will then output the Fn demo sequence and finally reset the Fn status.

Notes:

- Your PC application can control the keyboard's Fn status during runtime using the commands `EF 0B` / `EF 0C`. For details please refer to chapter Special Commands for PrehKeyTec Devices on page 20.
- The predefined Fn-Layer can be used for single keys and also for key sequences. If you need a special Fn layer instead which has similar features as the StdKey-Layer, you must define an additional layer named "Fn-LayerStdKey" with matching settings.

NEW *Simplified Layer Concept "EasyLayer"*

A simplified layer concept was developed for the MCI keyboard family. This new layer concept is independent from the PCs attributes – like also the Fn-Layer. Now you can easily switch between those new layers.
Existing keytables using the "classic" layer concept via Shift, Control, etc. can still be used – even in parallel.

Advantages:

- Up to 16 different Layers.
- Switching between those layers using programming macro. Example: {EasyLayer2}
- Switching layers also possible from the outside by special PC commands.

Requirements:

- MCI Family Keyboard with Firmware 605/3090 or newer
- WinProgrammer V2.3 (MWXC32.DLL 4.0.41.4 or newer)

Example:

	Assignment	Output
AlwaysActive (EasyLayer0)	X	X
EasyLayer1	A	A
EasyLayer2		X
EasyLayer3	B	B
EasyLayer4		X

WinProgrammer

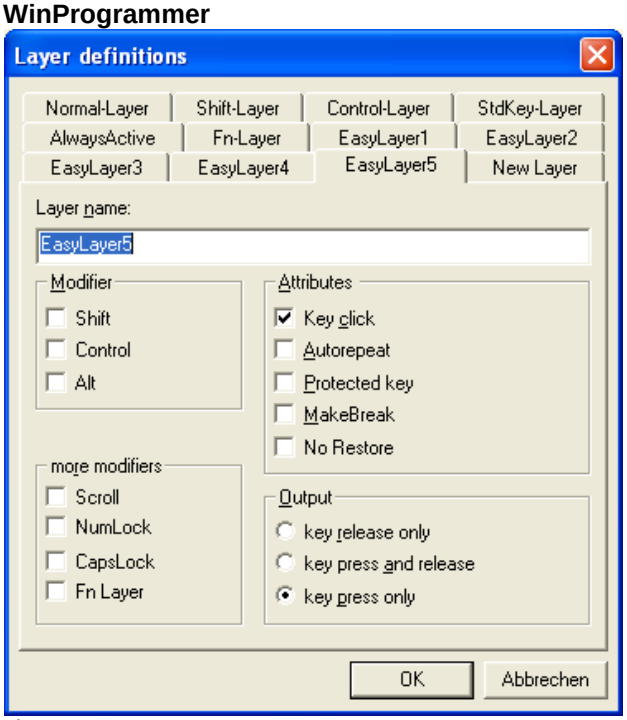


Figure 11

In the WinProgrammer these layers can be created by defining customized layers named "EasyLayer0" up to "EasyLayer15".

"Easylayer0" is active when no other layers are "firing" – similar to the AlwaysActive Layer.
Therefore "Easylayer0" does not need to be defined.

Additional Comments:

- After starting the keyboard, "EasyLayer1" is active
- On "AlwaysActive" layer the default output can be programmed. Codes placed here are output if no assignments are placed on the currently active layer.
- In combination with the feature "programmable keylock" \P layers can now easily be switched using the keylock (see page ↳21 for details). An example can be found in the WinProg folder "Keytable".
- Restoring WinProgrammer's default layer settings: Close all keytables, then *File* → *Default Configuration*.
- With the command sequence `EF 5A XX` (0x00..0x0F) your PC application can switch between EasyLayers. For details please see ↳Special Commands for PrehKeyTec Devices on page 20.

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 interface) – 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)

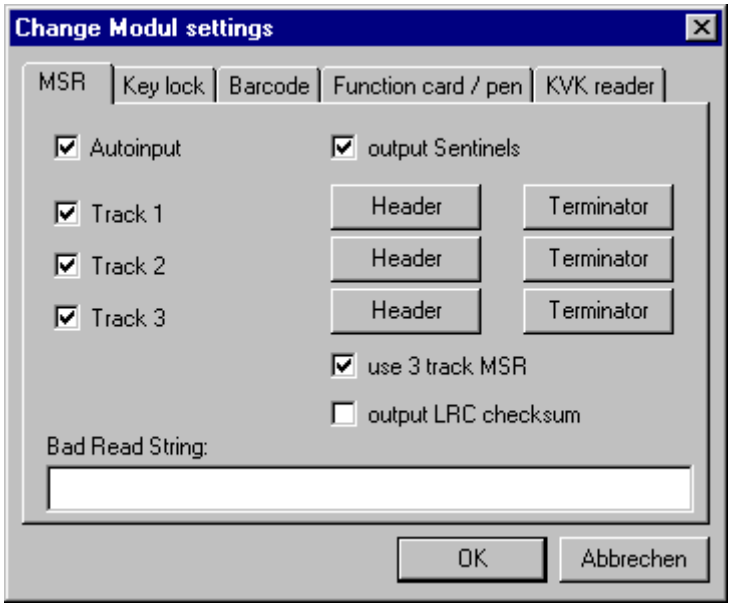


Figure 12

- **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 can't be identified automatically by the keyboard's encoder. Therefore this attribute should be enabled for all 3-track MSRs to enable the track output in correct order.

- **Bad Read String**

Using the *BadReadString* option 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 for our OPOS / JavaPOS services, the following BadReadString has to be entered here: Err\#

Further information and additional mode switches can be found in chapter Special Keyboard Modes using BadReadString (annex, page 21).

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 keyboard buffer overrun 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.

Please 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 of the 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**

Notes:
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.

When the keyboard was configured to OPOS/JavaPOS settings, please use such POS test application to verify your configuration. This is the factory default setting for MCI series keyboards.

Annex

System Requirements / Short description of the programming methods

WinProgrammer (Version 1.8 or newer)

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 and newer. To enable the download, the appropriate drivers (DriverPack) have to be installed properly. Please see the Readme file for details about installation and usage.

DOS Programmer (PREH-MWX.EXE)

To work with the DOS-Programmer you need an IBM AT or PS/2 compatible system (80286 or higher). The DOS-Programmer "PREH-MWX.EXE" (Version 4.1.x and higher) can run under MS-DOS as well as under Windows 3.1 and Windows9x in a DOS-box. Keyboards with maximum 128 key positions are supported.

For newer Windows versions (e.g. NT/2000/XP, etc) is *not* possible for any DOS tools to communicate with the keyboard hardware. Nevertheless the DOS programmer can be used to directly modify MWX files in a DOS box. For writing the files from/to the keyboard, please use our Windows Download Utility C2K.

Download Utilities

If you want to download a previously created keytable (MWF or MWX-file) into the keyboard without using the DOS-Programmer or WinProgrammer, 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. In addition it's able to read out the binary content of the keyboard in case of service. Please see the Readme file for details about usage.

Copy2mwx.exe

If have to program our PS2 keyboards directly in DOS, you can also use the utility COPY2MWX.EXE. This utility is included in the DOS-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 usage of the DOS utility copy2mwx.exe is also not possible in a DOS box of Win2000 and newer.
A similar copy2mwx utility is also available for other operating systems like Linux on request.

Differences WinProgrammer – DOS-Programmer

	Win Programmer or C2K Utility	Preh-MWX.EXE or Copy2mwx.exe
MS-DOS, Windows 3.x	No	Yes [2]
Windows9x	Yes	Yes [2]
WindowsNT, 2000, XP, Vista, 7	Yes [3]	No
OS/2	No	No [4]
Unix / Linux	No	No [4]
Read keytable	Yes [5]	Yes
Write keytable	Yes	Yes
Save keytable	Yes	Yes
Max. number of layers	128	128
Key label printing	Yes	No

[2] Use new copy2mwx.exe (dated 2005) to program MCI keyboards (USB/PS2 interface) in PS2 mode. USB mode is not possible for DOS.
[3] In order to communicate with PS2 keyboards, the installation of our PS2 hardware driver is required. Not available for 64bit OS.
[4] On request we provide an utility for downloading the MWX keytable. Installation of a special keyboard driver might be required.
[5] Function is available in the C2K download utility: The binary MWX keytable image can be read out – e.g. to copy into other keyboards. In addition conversion into MWF format is available by special tool "mwx2mwf".

Interface settings (AT, USB, RS232)

The PrehKeyTec programmable devices basically can be configured to run these interfaces/protocols:

- **PS/2** (AT)
- **USB** (available if device is equipped with USB interface)
- **RS232** (only for MWX and MC/WX family with optional factory-fitted RS232 module)

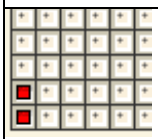
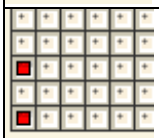
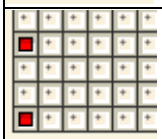
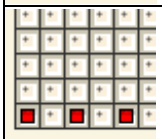
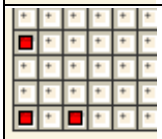
Important notes:
Of course the individual capabilities of your keyboard depend on the hardware and the cabling the keyboard is equipped with.

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

Special Key Combination

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

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

Keyboard Family / **Key combination**	**M 84/128 WX MC/WX (25, 35, 80, 84, 128) PC-POS**	**MCI Family latest MC147, MC140, MF112 MC 80 U**
 A01 + B01	Reset interface: PS2 (AT) protocol	Reset interface: Autodetect PS/2 or USB Protocol [3]
 A01 + C01	Reset interface: XT (old 8086) protocol	Reset interface: Fixed to PS/2 Interface
 A01 + D01	RS232 protocol with default parameters [1]	Reset interface: Fixed to USB Interface [3]
 A01 + A03 + A05	Activate a test keytable to check all key positions for electrical function. [2]	
 A01 + A03 + D01	Not supported here - Do **NOT** use! [5]	Restore the firmware 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. Please cycle power to get the keyboard back into "normal" operation.
[3] USB and Autodetect are not available for MCI keyboards with "PS2 only" electronic boards. These boards are only capable PS2 protocol.
[4] The actually programmed keytable will be replaced by the firmware default keytable. Also the module settings will be reconfigured to factory defaults.
[5] This key combination is not supported for older keyboard families. It will cause them to go into RS232 mode (like A01+D01). To get back to PS2 protocol, use key combination A01+B01 instead.

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
During booting the computer indicates a "keyboard error"	• 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 • Switch off all unused modules with our WinProgrammer
MC/WX keyboard does not work, although the daisy-chained keyboard works	No keyboard assignment stored in the internal keyboard EEPROM	Create and download a keytable into your keyboard using the WinProgrammer
Keyboard beeps at every key position, without displaying any characters	A fault has occurred in the transmission of the keytable, or the contents of the EEPROM have been modified incorrectly	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 too high for module data / key codes.	Enable the *Slow output* attribute using the WinProgrammer.
Modules do not function, or do not function correctly	Module is disabled.	Enable AutoInput for the module using the WinProgrammer
Module data for MSR/Keylock is not visible in Notepad	USB devices might be configured to send module data via the hidden OPOS/JavaPOS USB channel (due to security issues this is the factory default for all USB devices, like the MCI family).	• Temporarily activate "Test Mode" to check module function.via keystrokes. • Permanently disable "OPOS Settings" in your keytable with our WinProgrammer • Try our sample applications for OPOS/JavaPOS/MWXUSB to.
MC/WX keyboard does not work, but NumLock LED is on	Interface incorrectly set to RS232 using MCI's fw-default key comb. (don't use on MC/WX)	Reset to PS/2 interface by using A01+B01 key combination
PS2 download not working on 64bit Windows OS	PS2 driver does NOT support 64bit platforms.	Please use C2K on 32bit OS or move to our USB keyboards.

Technical Support

- Please refer to your keyboard manual for additional information.
- Consult the Keyboard FAQ pages on the PrehKeyTec 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 PrehKeyTec distributor in order to get technical assistance
- Contact the PrehKeyTec technical support:
  support@prehkeytec.de For worldwide support.
  techsupport@prehkeytecusa.com Support for North and South America.
- Visit the Support Area on the PrehKeyTec Website:
  http://support.prehkeytec.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.

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

Important notes:

1. Key combinations using uppercase letters
For key combinations usually lowercase letters have to be used. Using uppercase letters would result to a key combination with a shifted character. See example below:

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

2. Multi-Layer Macros
Multi-Layer macros like {KEY-UP} automatically define codes on several layers. Please only place them on StdKey-Layer - all other layers must be left empty then.
Technical note: Multi layer macros like {KEY-UP} exactly support the PS2 specified scancode sequences for such extended keys. For USB extended keys do not need multi layer macros any more. Here you can use the equivalent macros like {Up} instead.

3. Programming ASCII / ANSI Codes using macro {ALTxxx}

To achieve special ASCII/ANSI characters in DOS/Windows you have to press LeftAlt key, type the character's decimal code on numeric pad, Release Alt key.

Our keyboard does the same – if key assignment is done as described below:

{Alt###} ### indicating the decimal ASCII character code.

{Alt0###} ### indicating the decimal ANSI character code.

Examples:

{Alt65} will cause the keyboard to do the Alt-Combination for a capital A (ASCII/ANSI decimal 65).
{Alt0128} will output the Windows XP Alt-Combination for the Euro sign € (ANSI decimal 128).

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) - please also see {FCx}
{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 above {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 above {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
{MAKECAPS}	CapsLock key (Make-Code only; to be placed on StdKey layer)
{MAKESHIFTLOCK}	Same code as above – only alternative name
{MAKENUM}	NumLock key (Make-Code only; to be placed on StdKey layer)
{MAKESCROLL}	ScrollLock 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, equivaltent 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 0x68 …
{POSKeylock}	OPOS Keylock header / terminator 0x65
{POSMSR1}	OPOS MSR Track1 header / terminator 0x62
{POSMSR2}	OPOS MSR Track2 header / terminator 0x63
{POSMSR3}	OPOS MSR Track3 header / terminator 0x64
{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 - please also see {FSx}
{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

New "Multimedia macros":
Available since WinProgrammer 2.3 (MWXC32.DLL V4.0.41.3):

{MEDIA_PREV}	Scan previous Track
{MEDIA_NEXT}	Scan next Track
{VOLUME_MUTE}	Mute
{LAUNCH_CALCULATOR}	Calculator
{MEDIA_PLAY_PAUSE}	Media Play/Pause
{MEDIA_STOP}	Media Stop
{VOLUME_DOWN}	Volume Down
{VOLUME_UP}	Volume Up
{BROWSER_HOME}	WWW Home
{BROWSER_SEARCH}	WWW Search
{BROWSER_FAVORITES}	WWW Favorites
{BROWSER_REFRESH}	WWW Refresh
{BROWSER_STOP}	WWW Stop
{BROWSER_FORWARD}	WWW Forward
{BROWSER_BACK}	WWW Back
{LAUNCH_EXPLORER}	My Computer
{LAUNCH_MAIL}	Mail
{LAUNCH_MEDIA}	Media Select
{POWER}	System Power
{SLEEP}	System Sleep
{WAKE}	System Wake

Special Notebook-like Macros:
Special Multi-Layer macros like {K-Fn4} the notebook-like output of an alpha area with integrated numeric pad. The codes are automatically sent - depending on the status of NumLock and *Fn*.

{K-Fn0}	M	/	{N0}
{K-Fn.}	. >	/	{N.}
{K-FnDiv}	/ ?	/	{Div}
{K-Fn1}	J	/	{N1}
{K-Fn2}	K	/	{N2}
{K-Fn3}	L	/	{N3}
{K-FnPlus}	; :	/	{Plus}
{K-Fn4}	U	/	{N4}
{K-Fn5}	I	/	{N5}
{K-Fn6}	O	/	{N6}
{K-FnMinus}	P	/	{Minus}
{K-Fn7}	7	/	{N7}
{K-Fn8}	8	/	{N8}
{K-Fn9}	9	/	{N9}
{K-FnMul}	0	/	{Mul}

Similar combined key function, but here only via Fn:

{K-FnDelIns}	{Del}	/	{Îns}
{K-FnNumScroll}	{MakeNum}	/	{MakeScroll}
{K-FnPrtScPause}	{PrtSc}	/	{Pause}

Special Commands for PrehKeyTec Devices

With the following commands you can control the PrehKeyTec devices and their internal modules. For an easy implementation into your Windows application, you should use our MWXUSB.DLL or OPOS/JavaPOS packages. These can be downloaded from our support website. See the included documentation for details.

Command	Response	Parameter	Response	Function
EC	FA	<LCD data>		send data to LCD
ED	FA	<LED data>		set LED's
FF	FA			Reset / reboot device
EF...	*FA (ack) / FE (nack)*			*Special commands for PrehKeyTec devices as listed below*
EF 03				init default keytable
EF 05				init test table
EF 0B				LED Attribute (Fn-Layer) ON
EF 0C				LED Attribute (Fn-Layer) OFF
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				Keyboard Beep
EF 42				Accept LED Off
EF 43				Accept LED Green
EF 44				Accept LED Red
EF 5A XX		XX		Switch to EasyLayer XX (XX=0x00..0x0F, please see EasyLayer requirements)

Important note:

- Above internal low-level commands should not be used unless we advise to do so.
- You should always prefer to use high-level functions of our MWXUSB.DLL or OPOS/JavaPOS.
- Above commands might not be available for every PrehKeyTec device. Especially older keyboard models do not support newer commands at all.
- Using such low-level commands on old keyboards will result in unpredictable behaviour.

Special Keyboard Modes using BadReadString

Basically the MSR module's *BadReadString* option is used to define a text which is sent instead of the track data in case of erroneous 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 PrehKeyTec devices having firmware dated 2001 or newer.

Special keyboard mode switches for BadReadString:

\# 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).

\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 MSR/Keylock: Special OEM data protocol (PS/2 interface only).

\L MSR: SlowOutput activated (regardless of setting for header/terminator).

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

\Q Keylock: Keyboard output is disabled in position 0.

\P Keylock: Keylock codes programmable (see parameters below)

Important note:

- Not all switches are available for every PrehKeyTec device 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.

Parameters for switch \P – Programmable keylock:

- Feature only available for MCI keyboards having firmware 605/3018 or newer
- When using switch \P, you always have to enter the *complete* scancode sequence into each of the module assignments.
- Programming of the keylock is done via keylock header/terminator and other unused modules:
  0: KLH (Keylock Header)
  1: KLT (Keylock Terminator)
  2: FIH (Function card / Insert header)
  3: FIT (Function card / Insert terminator)
  4: FRH (Function card / Remove header)
- On latest MCI keyboards it's very easy to switch between layouts – using the new "EasyLayer" concept (see page 11 for details)

Copyright

Published by PrehKeyTec GmbH

Trademarks