|  |
| --- |
| Intermec Technologies |
| Transmitting Functions Keys through RDM |
| How to guide to getting function keys to transmit through Windows Mobile Remote Desktop Mobile Application |

Table of Contents

[Introduction 3](#_Toc324952162)

[Background 3](#_Toc324952163)

[Getting RDM to Transmit Function Keys 3](#_Toc324952164)

[Getting the Keyboard to Transmit Function Keys through RDM 4](#_Toc324952165)

[RemapRDM 4](#_Toc324952166)

[Configuration 4](#_Toc324952167)

[Configuring with ConfigRRDM 5](#_Toc324952168)

[Configuring manually 7](#_Toc324952169)

[List of Supported Virtual Keys for Remapping 8](#_Toc324952170)

[List of Scan Codes for Function Keys 8](#_Toc324952171)

# Introduction

Users that have been using Windows CE terminal service client application may require transmitting function keys to their host application. As they migrate to Windows Mobile 6.x, they will have to use the Remote Desktop Mobile (RDM) application. However, one of the major limitations with RDM is that it does not support processing function keys (F1-F24). The following describes how to remap keys that are processed by RDM to cause function keys to be transmitted to the host application.

# Background

RDM does support some limited configuration abilities. With that, we can utilize it to help us transmit function keys. On the device, there exists a configuration file located at \Windows\tscscan.txt. This file controls how RDM transmit keys over RDP.

Example of contents in the tscscan.txt file:   
// Scan Code Expected unshifted char  
0x00 0x00 // 0x00  
0x00 0x00 // 0x01  
0x00 0x00 // 0x02  
0x00 0x00 // 0x03  
… (skipped virtual keys 0x04 – 0x40)[[1]](#footnote-1)  
0x1e 0x61 // 0x41 - VK\_A  
0x30 0x62 // 0x42 - VK\_B  
0x2e 0x63 // 0x43 - VK\_C  
0x20 0x64 // 0x44 - VK\_D  
0x12 0x65 // 0x45 - VK\_E  
0x21 0x66 // 0x46 - VK\_F  
0x22 0x67 // 0x47 - VK\_G   
… (skipped virtual keys 0x48 – 0xFF)

Each line in the file corresponds to a Virtual Key in order. Virtual Keys are symbolic names used in systems that are equivalent to keyboard and mouse inputs. The first column contains the keyboard scan code values and the second column contains the lower case ASCII values that correspond to the Virtual Key of that line. Anything after the “//” are comments and are ignored by RDM. In the above example, the Virtual Key 0x41 (VK\_A) is mapped to scan code 0x1e and ASCII value of 0x61 (‘a’).

## Getting RDM to Transmit Function Keys

To get RDM to transmit function keys, you will have to modify the tscscan.txt file. First you decide which key you want to remap to send the function key. For example, let’s decide that we want to remap the semicolon to send F1. In the tscscan.txt file, go to where semicolon is defined and you should see

0x27 0x3B // 0xba - VK\_SEMICOLON

You want to replace the first column value with the F1 scan code which is 0x3B and replace the second column with 0x00. To preserve the original mapping, you can use comment slashes and then add the new mapping values and any other comments. Now the line may look like this

0x3B 0x00 // F1 // 0x27 0x3B // 0xba - VK\_SEMICOLON

Save the tscscan.txt file and restart RDM. Now anytime the semicolon key is pressed, RDM will transmit F1 instead.

**Note that you will no longer be able to transmit any remapped key to the host application. It is important to be careful not to remap any keys you may need to transmit to the host application. This will affect your physical keyboard, SIP, and Virtual Wedge from sending the remapped key to the host.** As from the example above, if semicolon was remapped to F1, you will no longer be able to transmit a semicolon to the host.

Not all virtual keys are supported by RDM for remapping, please see the [List of Supported Virtual Keys for Remapping](#_List_of_Supported_1). Also, refer to [List of Scan Codes for Function Keys](#_List_of_Scan) for scan code values of the function keys.

## Getting the Keyboard to Transmit Function Keys through RDM

Some devices have a physical keyboard that has function keys. To use the physical keys to send function keys through RDM, you have to first remap all required function keys as explained in [Getting RDM to Transmit Function Keys](#_Getting_RDM_to). Next, you will have to use a key remapping utility to map the physical function key to the key that was remapped in RDM. For example, we decided that we want to remap the semicolon to send F1 in RDM. We would use the key remapping utility to map the F1 physical key to transmit a semicolon. When RDM receives the semicolon, it will then transmit F1 to the host application.

# RemapRDM

RemapRDM.exe is an application to simplify configuring a device to allow transmission of function keys through Remote Desktop Mobile. RemapRDM will only run if it finds a configuration file called **rrdmconfig.txt**. If it doesn’t find the configuration file, it will exit immediately. The default installation directory is in the \Windows folder.

## Configuration

RemapRDM is configurable through a text configuration file called rrdmconfig.txt. The configuration file will only take affect if it is located in the same directory as the RemapRDM.exe executable or in the \Flash File Store. RemapRDM will first look in the current directory. If it is not found there, it will then look in the \Flash File Store. If the configuration file is located in both locations, it will use the one that is located in the same directory as RemapRDM.exe. If persistence over a clean boot is required, it is recommended that the rrdmconfig.txt file is stored at the \Flash File Store\rrdmconfig.txt.

### Configuring with ConfigRRDM

ConfigRRDM.exe is an application to help configure RemapRDM by generating a rrdmconfig.txt configuration file.

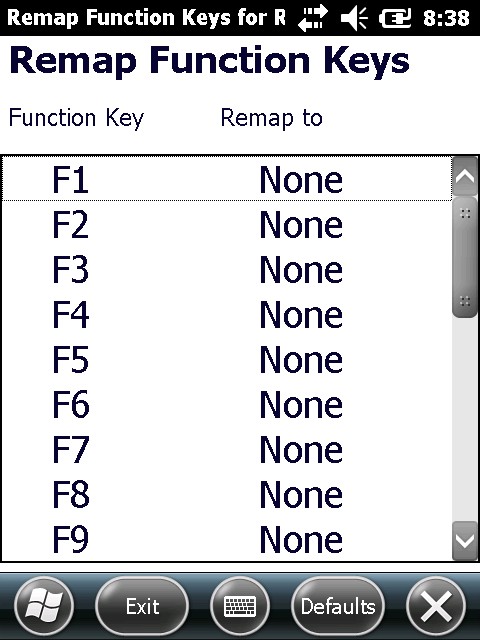
With this utility, you can select a function key item from a list to remap ([Figure 1: Main Screen](#_Figure_1:_Main)).

Once a function key is selected, you can select which key will remap to that function key  
 ([Figure 2: Remap Screen](#_Figure_2:_Remap)).

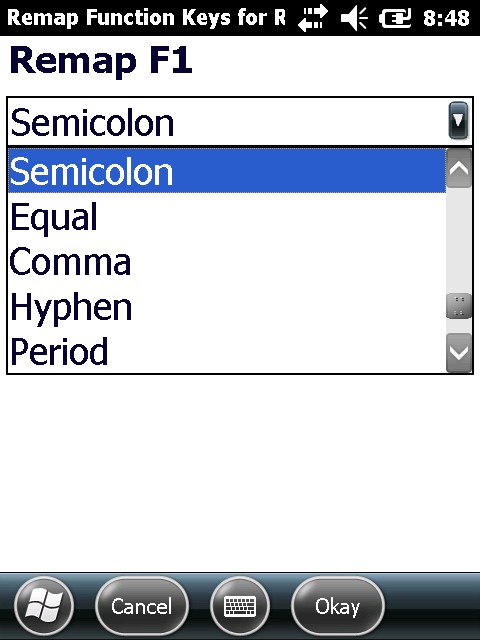
The **Defaults** button will set all function key to None.

The **Exit** button will prompt you whether or not you want to save the current configuration or cancel ([Figure 3: Exit Screen](#_Figure_3:_Exit)). Selecting yes will save the configuration file and then exit the application. Select no will exit the application without saving. Selecting cancel will not save and will return back to the main screen. When saving, this utility will save the configuration where it originally read the file from. It will first try to read the configuration file in the current directory. If it is not found there, it will look in the \Flash File Store. If no configuration file is found, it will create a new file and save the file in the current directory. The default installation directory is the \Windows folder so the rrdmconfig.txt file will most likely be there. If the rrdmconfig.txt file does not exist in the \Flash File Store, it will need to be regenerated after a clean boot because it will not be preserved.

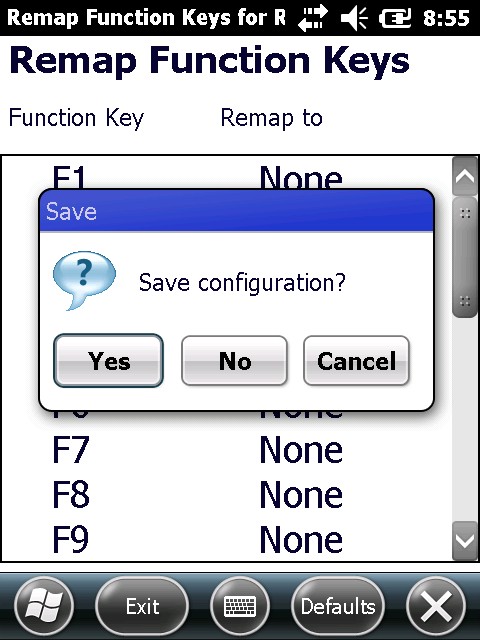
#### Figure 1: Main Screen



#### Figure 2: Remap Screen



#### Figure 3: Exit Screen



### Configuring manually

Create text file called rrdmconfig.txt or modify an existing one.

Example of contents in the rrdmconfig.txt file:  
// RDMFnRemap Configuration File  
0x00 // F1  
0x00 // F2  
0x00 // F3  
0x00 // F4  
0x00 // F5  
0x00 // F6  
0x00 // F7  
0x00 // F8  
0x00 // F9  
0x00 // F10  
0x00 // F11  
0x00 // F12  
0x00 // F13  
0x00 // F14  
0x00 // F15  
0x00 // F16  
0x00 // F17  
0x00 // F18  
0x00 // F19  
0x00 // F20  
0x00 // F21  
0x00 // F22  
0x00 // F23  
0x00 // F24

Each line corresponds to a function key in order. The first and only column is the hex value of the virtual key that will be used to remap the function key to so that it can be transmitted through RDM. Anything after the “//” are comments and ignored. For example, if we decided we want to remap semicolon to transmit F5 key, we would alter the line “0x00 // F5” to “0xBA // F5”. After you’re done, save your file where the RemapRDM.exe executable is located or in the \Flash File Store. For the changes to take effect, you must reboot the device. Please refer to [List of Supported Virtual Keys for Remapping](#_List_of_Supported_1) for Virtual Key values that can be used in this configuration file. It is possible to remap the function keys with Virtual Key values that are not in the supported list. They may or may not work however.

# List of Supported Virtual Keys for Remapping

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0x09 - Tab  0x0D - Return  0x1B - Escape  0x20 - Space  0x25 - Left  0x26 - Up  0x27 - Right  0x28 - Down  0x2E - Delete | 0x30 - 0  0x31 - 1  0x32 - 2  0x33 - 3  0x34 - 4  0x35 - 5  0x36 - 6  0x37 - 7  0x38 - 8  0x39 - 9 | 0x41 - A  0x42 - B  0x43 - C  0x44 - D  0x45 - E  0x46 - F  0x47 - G  0x48 - H  0x49 - I  0x4A - J  0x4B - K  0x4C - L  0x4D - M | 0x4E - N  0x4F - O  0x50 - P  0x51 - Q  0x52 - R  0x53 - S  0x54 - T  0x55 - U  0x56 - V  0x57 - W  0x58 - X  0x59 - Y  0x5A - Z | 0xBA - Semicolon  0xBB - Equal  0xBC - Comma  0xBD - Hyphen  0xBE - Period  0xBF - Forward Slash  0xC0 - Accent  0xDB - Left Bracket  0xDC - Back Slash  0xDD - Right Bracket  0xDE - Apostrophe |

# List of Scan Codes for Function Keys

|  |  |  |  |
| --- | --- | --- | --- |
| F1 - 0x3B  F2 - 0x3C  F3 - 0x3D  F4 - 0x3E  F5 - 0x3F  F6 - 0x40 | F7 - 0x41  F8 - 0x42  F9 - 0x43  F10 - 0x44  F11 - 0x57  F12 - 0x58 | F13 - 0x64  F14 - 0x65  F15 - 0x66  F16 - 0x67  F17 - 0x68  F18 - 0x69 | F19 - 0x6A  F20 - 0x6B  F21 - 0x6C  F22 - 0x6D  F23 - 0x6E  F24 - 0x76 |

1. The actual tscscan.txt will list all Virtual Keys from 0x00 to 0xFF. Virtual Keys 0x04 through 0x40 is removed in this example to show some more interesting Virtual Keys mappings like VK\_A through VK\_G. [↑](#footnote-ref-1)