

SAC Controller

Version 2.1

User's Guide

Issue 1.1

November, 2006

Windows 95 / 98 / ME / NT4 / 2000

Purpose

The purpose of this document is to explain how to use the SAC Controller. It describes how to use the application, its functionality, how to install and configure it, plus how to distribute configurations with your software

www.ScreenKeys.com



SAC Controller

Information in this document is subject to change without notice.

The latest revisions of the ScreenKey documentation and software are placed on the SKI Web site.

Web: www.ScreenKeys.com

Technical Support is available

via Email support@ScreenKeys.com

via Web: www.ScreenKeys.com

© 2006 SK Interfaces Ltd.

All rights reserved.

DISCLAIMER:

SKI reserves the right to revise program interfaces, data file formats and functionality, at any time.

Foreword

The SAC Controller is a standalone application whose purpose is to control a ScreenKey console and form an interface between this and your Windows applications. It uses the information in a SAC file to control a ScreenKey console, and feeds keystrokes into a Windows application. The SAC file contains ScreenKey menus and keystroke information that will be interpreted and handled by the SAC Controller, so the target application (your application) does not need to worry about the ScreenKey console.

SAC Controller, version 1.0 was released with the SAC Engine Control. This is a sample application that comes with source code to show how to use the SAC Engine Control as a side application. From version 2.0 onwards, the SAC Controller is released as a standalone product.

This document will explain how the application works, what SAC files are, how to install and configure it and how to use the application. It also explains how to redistribute the application.

SKI – ScreenKey Interface

ScreenKey technology has been in use for a considerable number of years. In this time, SKI has developed a range of software tools to simplify the task of integrating a ScreenKey console into an application or hardware solution.

SKI Software Toolset (SKI) implements a new approach to controlling ScreenKey consoles and keyboards. SKI is a toolset designed specifically for Windows 32-bit platforms and utilizes Microsoft COM technology.

SKI is particularly targeted at Application Developers, offering a host of developer tools to allow tight and dynamic integration of ScreenKey™ technology.

The SAC Engine interacts with a ScreenKey™ console according to a pre-designed SAC file but also offers the developer the ability to trap Windows events and change individual ScreenKeys, jump to a different SAC menu, display application specific data, etc.

When the end-user application cannot be modified to incorporate the SAC Engine, the SAC Controller may be employed. This provides a mechanism to statically feed keystrokes to an application based on a pre-designed SAC file. *Note that the SAC Controller does **not** control the large LCD display integrated into the SK-7000 console.*

Alternatively, developers may write a user-specific application (using the SAC Engine) to provide a highly customized SAC Controller equivalent. SKI publishes sample source code for this opportunity.

Table of Contents

SAC CONTROLLER OVERVIEW.....	6
<i>What is a ScreenKey Console ?</i>	6
<i>What is SAC ?</i>	6
<i>What is SAC Editor?</i>	7
<i>What is SAC Controller ?</i>	7
<i>What is SAC Engine Control ?</i>	7
<i>What is SKI ActiveX Control ?</i>	8
<i>Who should read this document ?</i>	8
<i>Requirements</i>	8
<i>Cross reference</i>	8
INSTALLATION PROCEDURE.....	9
Hardware Installation	9
Software Installation.....	9
Uninstallation of Software.....	10
TECHNICAL DESIGN.....	11
Functional overview	11
USAGE.....	12
Usage	12
<i>Designing a SAC file</i>	13
<i>Starting the SAC Controller application</i>	13
<i>SAC Controller Configuration</i>	13
<i>SAC AutoStart</i>	14
<i>Target Window / Application Specification</i>	14
<i>Target Window / Application Specification Options</i>	15
<i>Command Line Parameters</i>	17
User Interface	18
<i>SAC Controller User Interface</i>	18
<i>SAC Controller Menus</i>	20
<i>SAC AutoStart Dialog</i>	25
<i>Options Dialog</i>	27
DISTRIBUTION OF SAC CONFIGURATIONS.....	30
File formats, Locations and Registry Entries.....	30
<i>SAC Files</i>	30
<i>KBD Files</i>	30
<i>SAC Configurations</i>	32
<i>SAC Controller Options</i>	33
Distributed Files and Components.....	34
<i>SAC File Icons</i>	35
<i>DCOM & Windows 95</i>	35
<i>PSAPI.DLL</i>	35
APPENDICES	
DOCUMENTATION CONTROL.....	36
A.1 Change Control	36
A.2 Abbreviations Used/Terms of Reference	36

A.3	Historical Change Reference	36
A.4	Change Summary	36

SAC Controller Overview

What is a ScreenKey Console ?

The ScreenKey console is an Input/Output device for use with specialized applications, such as process control, dealer desks, call centers, point-of-sale (POS), etc. The ScreenKey console combines the best features of a standard keyboard and a TouchScreen. It is made up of conventional keys and special ScreenKeys. A ScreenKey acts like a conventional key but each ScreenKey has a built-in LCD display panel. This enables the ScreenKey's legend to be changed dynamically. ScreenKeys are available in several configurations:

- Resolution: LC16 (32x16 pixels) or LC24 (36x24 pixels)
- Backlight Colors: RG (red-green) or RGB (red-green-blue) LED backlighting

What is SAC ?

SAC is short for ScreenKey Active Control. SAC files are the cornerstones of the ScreenKey Windows developer toolset. The SAC system facilitates quick and easy integration of ScreenKey technology into Windows applications, with minimal or no application modification. SAC is a self-contained data-file containing a set of drill-down menus for the ScreenKeys, the relationship between these menus, and other applicable rules and data. SAC files are generated and maintained using a special purpose Windows file editor, the SAC Editor. Each ScreenKey, in each menu, has properties such as a caption (text or graphics) to display on the key's LCD display, background and flashing colors, a menu to display when the key is pressed, a security level etc. In addition, each key can have one or more keystrokes assigned to it, which are sent to the application when the key is pressed. Other devices, such as the MSR unit and KeyLock, also have properties attached to them.

The SAC system is a software system driven by the keyboard itself on the basis of the contents of the SAC file, producing keystrokes for the application(s).

What is SAC Editor?

The SAC Editor is a software package that allows the design and maintenance of a ScreenKey menu-based navigation system in the form of a self-contained data file (SAC). The data file contains all ScreenKey text, graphics, backlight colors, flash colors and keypress return codes as well as navigation rules to move between different ScreenKey menus.

The SAC file can be used by other ScreenKey runtime components to control a ScreenKey console and to interact with a new or existing application.

Some of the benefits of ScreenKey consoles are:

- Interface the ScreenKey console to an existing application with **no changes** to the application itself
- Tailor the “Look and Feel” of the ScreenKey console to suit the application
- Design user interface offline from the runtime system
- Add security feature to the application using the Mode Lock with no application changes (available on some console types)
- Enhance application with dynamic content-sensitive keys

A ScreenKey menu is a description of a set of ScreenKeys; normally a number ScreenKey panels that each consists of 12 ScreenKeys, and their properties. The properties include what to display on the ScreenKey, in which color and flash-color, if the key is currently active and its security level, and what to feed to the target application when pressed.

What is SAC Controller ?

The SAC Controller is a standalone application that is meant to control a ScreenKey console and form an interface between this and your applications. It uses the information in a SAC file to control a ScreenKey console, and will feed keystrokes into your application. The SAC file contains ScreenKey menus and keystroke information that will be interpreted and handled by the SAC Controller, so the target application (your application) does not need to worry about the ScreenKey console at all.

The SAC Controller is merely an application offering a user interface to the SAC Engine Control, which is the software that is doing the actual SAC processing. These products are part of the SKI Toolset, which offers a wide range of ScreenKey solution tools. The SAC Controller also makes use of other ScreenKey software tools, the SAC Editor and the SKI ActiveX Control (used by SAC Engine Control). These are described below.

What is SAC Engine Control ?

The SAC Engine Control is a software package that forms the interface between the ScreenKey console/console and Windows applications. It is a COM (ActiveX) module, and can be used by modern programming languages like Visual C++, Visual Basic etc. A wrapper module for non-COM enabled programming languages is available.

The Control can be integrated fully into an application, using the properties, methods and events to exchange information with the ScreenKey console. It can be integrated with a minimum number of changes, feeding the keystrokes back via the keyboard message system. It can also be used as a separate side-application, sending the keystrokes to the currently active window.

What is SKI ActiveX Control ?

SKI ActiveX Control is a software package that allows the ScreenKey console to be easily integrated into Windows based applications. It can be used by modern programming languages, e.g. Visual C++, Visual Basic, etc that support COM technology. SKI ActiveX Control does not work with SAC files but offers control of individual ScreenKeys and their properties.

Who should read this document ?

This User's Guide is intended for persons who shall install, configure and operate the SAC Controller. It is also intended for persons who shall redistribute the SAC Controller with a third party application.

Reading this User's Guide will also benefit ScreenKey application developers, as well as ScreenKey users.

This guide assumes that the reader is familiar with all of the above, Microsoft Windows and Windows commands.

Users should have read the SAC Editor User's Guide.

Requirements

Most applications should be suitable for use in conjunction with the SAC Controller as long as they meet the following requirements...

Operating System:	The application must run on Windows 95/98/ME, Windows NT 4.0, or Windows 2000
PC Hardware:	The application must run on standard PC hardware. One COM port must be available per ScreenKey console.
Available PC Resources:	The PC must have sufficient available resources to run the ScreenKey console software. The SAC Engine Control and the SKI ActiveX Control can run on a PC with the amount of memory required for running the operating system.

Cross reference

- [1] ScreenKey Getting Started & Installation Guide
- [2] SAC Editor User's Guide
- [3] SAC Engine Control User's Guide
- [4] SKI ActiveX Control User's Guide
(formerly ScreenKey ActiveX Control User's Guide)
- [5] ScreenKey Console Technical Reference Manual
- [6] ScreenKey Low-Level Interface Programmer's Guide
- [7] SAC Engine Wrapper User's Guide

I N S T A L L A T I O N

Installation Procedure

Hardware Installation

See the **ScreenKey Getting Started & Installation Guide** document for details of hardware installation.

Software Installation

The SAC Controller is supplied on CDROM as part of the SKI Software Toolset. Alternatively it may be downloaded as an individual component from the web (www.ScreenKeys.com).

When distributed by CDROM, the CD usually incorporates an autorun facility that launches an installation helper utility (e.g. HTML file). Follow the on-screen instructions as described by this utility. If the utility is not present or the autorun feature is disabled on your computer, you can install the software directly from the CD. The ScreenKey software package typically includes several software applications that reside separately on the CD. Find and open the SAC Controller sub-folder using Windows explorer from the root of the CD and run Setup.exe from this folder.

The SKI ActiveX Control and SAC Engine Control, which are used by the SAC Controller, are installed automatically.

Because the SAC Editor should not be installed on all computers using the SAC Controller, this will not be installed by the SAC Controller Install. This is available as a separate installation program.

In all cases the installation can be started using the **Setup.exe** program, then follow the instructions.

- After the Welcome dialog, you are prompted if you wish to search for superceded ScreenKey applications. You may safely skip this step as it can be very time consuming. If the search is performed, a list of applications will be displayed and these should be uninstalled before installation is proceeded. The installation should be cancelled at this point and the outdated software can be uninstalled. It is possible to ignore this warning, but is not advised.
- Then the Software License Agreement has to be accepted before the installation can proceed.
- The default destination location for the software is **C:\Program Files\ScreenKey**, but this may be changed by clicking the Browse button. This only applies if a SKI software tool has not been installed onto the computer earlier. If it has, this folder will be selected automatically, and the user will be informed with a dialog. The existing ScreenKey software must be uninstalled to be able to install to a different destination. The installation will build a folder structure from the destination location for the different

parts of the software. The control will be put in .\Bin, the documentation in .\Doc, and the samples in .\Samples, etc.

- When the destination location is chosen, a “Start Copying Files” dialog is displayed and installation of the files will start when the Next button is clicked.
- When the files are installed, the installation program will search for existing SAC files. A list of these is displayed, and the user is advised to copy these to the folder displayed in the dialog.
- The next dialog allows the user to define the SAC Controller Options, which are the same options as can be defined in the Options dialog when the SAC Controller has been started.
- When the control has been installed, a number of files have to be registered in the registry. This is done automatically but if errors occur the user will be notified, and the Control will not function correctly. In this case, try the following:
 1. Uninstall earlier versions, and reinstall the control.
 2. Reboot the PC, stop all applications and install again.
 3. Log on as Administrator, and install again.
 4. View the Windows Event Log, correct any errors and install again.
 5. Contact support@ScreenKeys.com

If the control fails to register, a separate batch file is provided (RegSACCo.Bat), which can be run from a DOS shell when the errors have been corrected.

- Finally, if the computer needs to be rebooted in order for the software to function correctly, the user will be notified. The computer may be rebooted immediately or it can be done at a later stage.

Uninstallation of Software

To uninstall the SAC Controller:

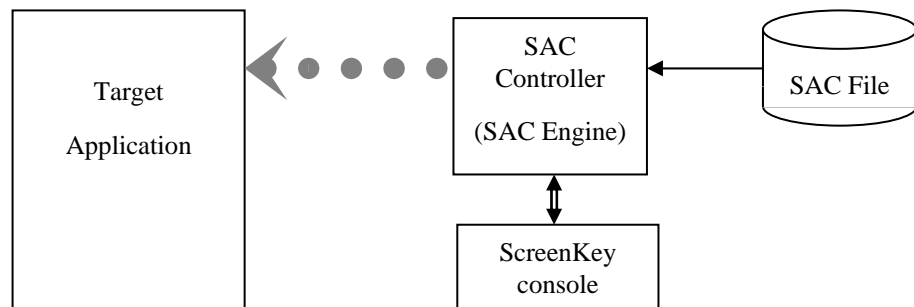
1. Terminate any application using the SAC Controller, the SAC Engine Control, and the SKI ActiveX Control, and make sure no other application is using any of the other files, like documentation and sample code.
2. Start **Add/Remove Programs**, from the **Control Panel**.
3. Locate the line **SAC Controller** in the list of installed programs and
4. Click the **Add/Remove** button.
5. Click **Yes** in the dialog if you want to completely remove the SAC Controller and all of its components.
6. The SAC Controller with all of its components and documentation will then be removed, all folders will be deleted if they are empty, and all registry entries will be removed. Since the control files may be shared, the uninstall program might ask the user if these shall be deleted, and that will normally be ok.
7. If parts of the application could not be removed, a button named Details... will be visible. Click on this to see what was not successfully removed. Terminate **Add/Remove Programs**. Any changed files, like KBD files etc., will be left, and if any of the controls was in use by an application it will not be possible to remove this.

TECHNICAL DESIGN

Technical design

Functional overview

The SAC Controller is, as mentioned above, a side-application controlling the ScreenKey console and feeding keystrokes specified in the SAC file to a user specified target application. The SAC Controller will therefore form an interface between the target application and a ScreenKey console, with no change to the target application.



The SAC Controller reads the SAC file and initiates the ScreenKey console. When a key is pressed on the ScreenKey console, the KeyLock turned, or a magnetic card swiped, an event will be sent to the SAC Controller, or actually to the SAC Engine. This will then interpret the event, look up in the SAC file and decide on an action. The action can be either to update the ScreenKeys – display another ScreenKey menu, or to send the key sequence defined for that particular key to the target application, or a combination of the two. When sending keystrokes to the target application, the SAC Controller uses Windows Operating System functions like JournalPlayback or PostMessage.

When using JournalPlayback (recommended), the target application **MUST** be the currently active application because this function emulates the QWERTY keyboard. If it is not active, and a target application/window is specified, the SAC Controller will activate the application before feeding the keystrokes to it. If no target application is specified, the SAC Controller will feed the keystrokes to the currently active application. If the target application is not started, the SAC Controller will **NOT** send any keystrokes to any other application. This ensures that the SAC Controller does not upset other applications.

When using PostMessage (seldom used), the target application does not need to be the active one, but a target application should be specified.

The SAC Controller is merely an application offering a user interface to the SAC Engine Control, which is the software that is doing the actual SAC Processing. These products are part of the SKI Toolset, which offers a wide range of ScreenKey solution tools.

U S A G E

Usage

Usage

The SAC Controller is a side application whose purpose is to support the use of a ScreenKey console with a third-party main application.

The SAC Controller should be practically invisible to the user of the main application. It manages the operation of the ScreenKey console, i.e. menu navigation, ScreenKey colors, text and graphics and feeds keystrokes to the main (target) application based on a supplied SAC file. The SAC file is created with the SAC Editor, and contains ScreenKey menus and keystroke information that will be interpreted and handled by the SAC Controller

The SAC Controller is effectively a systems administrator tool. As such it has many diverse features and capabilities allowing it to be used and redistributed for use on a very wide range of varying types of Windows applications and implementations.

As the complexity of Windows applications grows, it becomes more difficult to 'track' an application for the purpose of feeding keystroke information to the 'correct' window and/or control. Typically Windows applications do not have a defined MAIN window and consist of multiple instances of different windows and sometimes even the same window. Background and hidden windows, toolbars, tooltip text, help windows must all be considered when trying to 'track' the target application.

A simple rule to apply is to always feed keystrokes to the top-level window. The SAC Controller can operate on this basis but it can also be fine-tuned to accurately track the target application even with the scenarios described above. In fact, the advanced tracking capability of the SAC Controller is one of its main features. A wide range of launch options are available to the developer to allow tight and accurate control over the target application tracking.

As a systems administrator tool, the SAC Controller is designed to be able to redistribute (for end-user operation) in different ways suitable to the type of target application and windows environment. Primarily, two different implementations for specifying target applications, SAC file and tracking parameters are offered. One uses the Windows registry and the other uses text-based control files. As a further support feature for developers, and to aid redistribution the SAC Controller may be launched by specifying command line options.

As the SAC Controller should be 'invisible' to the end-user, it may be configured to be launched automatically on start up of the computer and then appears as an icon in the System Tray.

The following sections describe how the different features and options offered by the SAC Controller may be used.

Note! Be aware that for the application to function properly, the user logged on must have sufficient privileges to read the register keys under `HKEY_LOCAL_MACHINE\SOFTWARE\ScreenKeys\SAC Controller`.

To be able to store configurations in the registry and to store changed SAC Controller options the user must have sufficient privileges to write to these keys.

Designing a SAC file

The SAC Controller is an application controlling a ScreenKey console from the contents of a SAC file, and feeding keystrokes to the target application. The first thing that has to be done is to design and create a SAC file, which is done using the SAC Editor. The design of the SAC file is crucial for a well working SAC solution. The SAC file should be designed to match your application(s) to get the best possible result and efficiency.

The “ScreenKey Getting Started & Installation Guide” should be read as an overview of how to create a SAC file.

Starting the SAC Controller application

The SAC Controller is meant to be started automatically, when the computer is started, and is not meant to be visible during processing. Despite this, the application has a user interface, where the parameters may be defined, stored and retrieved.

The application can be started with, or without command line parameters. If it is started without parameters, like from a Windows Shortcut or from the *Programs/ScreenKey/SAC Controller* from the *Start* menu, it will show its user interface from where the user can configure the application. If it is started with parameters, it will automatically try to start the SAC Processing and control the ScreenKey console.

The normal way of starting the SAC Controller will therefore be to start it with parameters, so it will automatically start the SAC Processing. This will be explained later under SAC AutoStart.

SAC Controller Configuration

For the SAC Controller to be able to control a ScreenKey console, it needs some information. This information is called a SAC Configuration, and can be defined and maintained with the SAC Controller application.

When the application is started, it shows a few input fields where the user can enter the configuration, this dialog is called the SAC Controller User Interface, or the Main dialog. The sections *SAC Controller User Interface* and *SAC Controller Menus* on page 18 describe how to configure the application in detail, but in brief, the user has to fill in the required information and store this as a configuration.

See *Target Window / Application Specification*, page 14 and *Target Window / Application Specification Options*, page 15 for information about how to specify a target window/application.

When this is done the SAC Processing, which is a term for actually controlling the ScreenKey console from the information in the SAC file, can start. Using the *Start SAC* from the *File* menu will start the SAC Processing.

When a configuration has been stored, the SAC Controller can be started as described in the chapter *Command Line Parameters*, page 17, in *SAC AutoStart*, page 14, or from the user interface as described above.

SAC AutoStart

As mentioned above, the SAC Controller would normally be a side-application to the target application. The SAC Controller user interface will therefore normally only be used to define configurations. When the configuration is defined and tested, the SAC processing can be started using the user interface, or from a Windows shortcut with the necessary parameters on the command line.

The SAC Controller can also start the SAC Processing automatically when the computer is started, which is called **AutoStart**.

When a configuration is saved, either as a Config or an SCC file, this configuration can be defined as an AutoStart configuration. The AutoStart configuration uses Windows functionality to start the application during the computer's boot sequence.

AutoStarting the SAC Controller is a preferred approach.

See *SAC AutoStart Dialog*, page 25 for more details on how to define AutoStarts.

Technical information:

The AutoStart configurations are defined in the registry under the key: HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run. Under this key a string value named SACControllerX, where X is a running number from 1 upwards. The SAC Controller AutoStart dialog should be used to maintain the AutoStarting configurations. Manual editing of the registry is not encouraged.

Note! For operating systems that use user names, the SAC Controller will only AutoStart when the user that was logged on when the AutoStart was defined is logging on. This is because the AutoStart definition is stored under HKEY_CURRENT_USER.

Target Window / Application Specification

The SAC Controller is a side-application controlling the ScreenKey console, from instructions in the SAC File, and feeding keystrokes to an application. The SAC Controller can either feed the keystrokes to the currently active application, or to a specified application or window.

If you want the SAC Controller to always work with the currently active application, you may leave the Target specification blank.

If you want it to work with one application only, this must be specified in the `_Target` field. The SAC Controller uses a feature in Windows operating systems, called `JournalPlayBack`, to feed the keystrokes to an application or window. This feature requires the application to be the active one, so the SAC Controller will make this application active before feeding keystrokes to it. `JournalPlayBack` simulates the QWERTY keyboard, and the keystrokes appear as if they were typed in using this keyboard. There is one exception to this, when the `PostMessage (/PM)` mode is in use, See *PostMessage Mode*, page 16

Using a Target specification ensures that the keystrokes will not be fed to another application by accident, e.g. if an application should be activated by accident.

Specifying an application is very easy, you just specify the name of the application, e.g. *MyApp.exe*. The SAC Controller (or actually the SAC Engine) will then use default options to find the window that is most likely the window taking the keyboard input. If further refinement of the active window in an application is required, options in the Target field may be used.

The syntax for specifying an application or a window is as follows:

Application-name (Options...)

The application name may be the filename only, a filename with extension, or the full path of the file. Relative paths are not allowed because there is no working folder defined.

The specified window does not have to be created when the SAC Controller is started.

Target Window / Application Specification Options

The options are as follows:

Option syntax	Description
/CF:Caption	Caption Full. The target window must have a caption containing the specified text entirely, no more – no less.
/CS:Subcaption	Caption Substring. The target window must have a caption containing the specified text.
/CL:classname	Class name. The target window must have the specified class name (full name required)
/CA	Caption Any. The target window must have a caption.
/HP	Has parent. The target window must have a parent window.
/NP	No parent. The target window CANNOT have a parent window.
/HC	Has child. The target window must have at least one child window.
/NC	No child. The target window CANNOT have a child window.
/SO:n	Start Order. The target application shall be the one started as the n'th one. These are applications with the same name. Ex: If three instances of notepad is running, this option can specify to use the one started second as the target application. Note: The Start Order is always in the present, which means that if an application that has Start Order 1 is being tracked, is closed, the application that has start order 2 becomes start order 1 (since it is the oldest), and this will then be the target application.
/RM	Relaxed mode. This mode allows the SAC Engine to use the first window that matches the specification. Normally if more than one window matches (non-relaxed mode), an error will be issued.
/PM	PostMessage mode. See below.

As an aid to specify the right options the Spy++ tool from Microsoft can be used to view the process and window tree.

Because Windows applications do not have a defined MAIN window, it can be hard to find the right window handler. It can often take some trial and error before the right one is found. If there is such a thing as a typical Windows application, it will have a “main” window that has a caption, it has no parent, and it probably has one child window. For this reason default options are used when an application name is specified with no other options:

(/CA /NP /HC)

Except for options not specifying windows (/SO, /PM and /RM).

Typical examples would be:

Notepad.exe (/SO:2)

Feed keystrokes to the second instance of Notepad.

Wordpad.exe (/PM /CL:RICHEDIT)

Feeds keystrokes using PostMessage (application not the active one) to the edit window in WordPad

PostMessage Mode

The specified window does not have to be the active one in this mode, it can remain in the background. When using the PostMessage function the shift state is not handled properly (according to Windows documentation), so this approach is better suited for sending types other than strings, e.g. to simulate clicking of a button, the button can be specified and a "Space" can be sent to the window. This would then be the same as selecting the button and hitting the space bar to press it. In some cases the PostMessage would be chosen just because the window doesn't have to be the active one.

Note! This documentation is taken from the *SAC Engine Control User's Guide*, and is included to explain the PostMessage mode in the SAC Engine Control. If you run into a problem that will require this mode, you should probably use the SAC Engine Control instead of the SAC Controller.

Command Line Parameters

The SAC Controller can use two types of configurations, a *Config* or an *SCC-file*. A *Config* is a named set of parameters stored in the Windows registry. An *SCC-file* contains the same parameters, but stored as a file.

When the SAC Controller is started, as an ordinary application without any command line parameters, it will display a blank dialog to the user. This is how it will present itself the first time it is started. The next time it is started, again without any command line parameters, it will try to load the previously used parameters. Unless the flag *AutoStart Last Configuration* is set, SAC processing will not start automatically.

The SAC Controller can be used with command line parameters, e.g. from a program Shortcut. It will then try to start the SAC processing automatically with the supplied parameters. The parameters can be supplied in the following ways:

SACCont /C:<Config>

This command line syntax will start the SAC Controller, read the parameters from the configuration stored in the registry, and then attempt to start SAC processing with these parameters.

Example: SACCont /C:"My setup"
which will use the configuration with the name My setup

SACCont /F:<SCC-file>

This command line syntax will start the SAC Controller, read the parameters from an SCC file stored at the specified location, and then attempt to start SAC processing with these parameters.

Example: SACCont /F:"C:\Program Files\ScreenKey\SCC\My Setup.scc"
which will use the configuration stored in the SCC file named My Setup.scc in the folder C:\Program Files\ScreenKey\SCC\

SACCont <COM-port> <SAC-file> <KBD-file> <Target-spec...>

This command line syntax will start the SAC Controller, read the parameters in the following sequence: First the COM port, then the SAC file, then the KBD file, and the rest of the parameters will be interpreted as the Target specification (which can consist of multiple parameters). The previous two approaches are preferred to this.

Example: SACCont COM1 "C:\Program Files\ScreenKey\SacFiles\My SACFile.pkf"
"C:\Program Files\ScreenKey\Bin\KybgenUk.kbd"
MyApp.exe /CS:"Fido"
which will use the parameters read from the command line.

When *AutoStart* is in use, the parameters have to be defined and stored either in the registry or in an SCC-file, and then one of the first two syntaxes will be used. See *SAC AutoStart*, page 14.

Port	Input field	Mandatory for Storing and Running
<p>Specifies the communication port the ScreenKey console is attached to, e.g. "COM1".</p> <p>If the port does not exist, the error message Failed to open COM-port. will be given.</p>		
SAC File	Input field	Mandatory for Storing and Running
<p>Specifies the SAC file to be used.</p> <p>Specify its full path, or pick the file by using the Browse button to the right.</p> <p>The SAC File name may be up to 260 characters long</p>		
Browse	Button	
<p>The SAC File Browse button can be used to browse for a SAC file. A standard "Open file" dialog will be display to assist the user in finding the desired SAC File.</p>		
KBD File	Input field	Mandatory for Storing and Running
<p>Specifies the keyboard definition file to be used.</p> <p>This file should be found in the ScreenKey/Bin folder.</p> <p>Specify the KBD file with its full path, or pick the file by using the Browse button to the right.</p> <p>The KBD File name may be up to 260 characters long</p> <p>The KBD file is an ASCII file and may be altered, but extreme caution should be used. See <i>KeyBoard Definition file format</i>, page 31.</p>		
Browse	Button	
<p>The KBD File Browse Button can be used to browse for a KBD file. A standard "Open file" dialog will be display to assist the user in finding the desired KBD File.</p>		
Target	Input field	Optional
<p>Specifies the application or window to be used as the target window.</p> <p>The Target window is the window or application the keystrokes are to be sent to. If no target window is specified, the keystrokes will be sent to the currently active window.</p> <p>The Target specification may be up to 260 characters long.</p> <p>How to define a target application/window specification is described above.</p>		
Status	Info field	
<p>The Status field will show the status of the SAC Controller at all times.</p> <p>It will indicate if it is running the SAC Engine, if it is stopped, or if there is an error situation.</p>		

Error

Info field

The **Error** field will show any error message describing the error occurred during the latest operation. It will show errors occurring while operating the user interface, errors occurring as a result of controlling the ScreenKey console, and asynchronous errors occurring in the console or the console control software (mostly communication errors).

See the Troubleshooting section in *SAC Engine Control User's Guide* and *SKI ActiveX Control User's Guide* for information on error messages.

This information will also be added to the on-screen log, and to the Log file-if in use.

The format of the error message is:

“Error message (SAC Engine error number, Low-level error number)”

Example: **Failed to open COM-port. (0x4001,0x2067)**

Note: It is important to quote these error numbers when seeking support.

Log

Info field

The **Log** field will hold different types of information, like error messages and operation information, like Start and Stopping of the SAC Engine with the parameters used.

This information will also be written to the Log file if active.

SAC Controller Menus

There are three menus on the menu line; File, Config and Help.

File menu

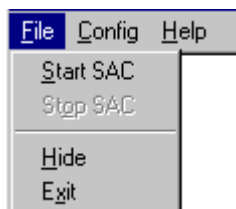


Figure 2: File menu

Start SAC

Disabled when running

Starts the SAC processing with the currently defined parameters. If the ScreenKey console is successfully initialized and no errors occur, the user interface will be disabled so the parameters cannot be change while running.

This command can be issued from the File menu when the application is visible, or from the local menu in the System Tray.

Stop SAC

Enabled when running

Stops the SAC processing. The user interface will again be enabled, and parameters can be changed.

This command can be issued from the File menu when the application is visible, or from the local menu in the System Tray.

Hide

Disabled when hidden

Hides or minimizes the SAC Controller application.

If the Use System Tray in the Options dialog is ON, the application will disappear, and can only be reached by clicking on its icon in the System Tray. If this is OFF, the application will only be minimized, and can be reached from the Taskbar.

This command can be issued from the File menu when the application is visible, or from the local menu in the System Tray.

Exit

Exits the SAC Controller.

If the SAC Controller is running (SAC processing started), you will be asked if you want to stop the processing.

If the configuration is changed, you will be asked if you want to abandon the changes or not.

This command can be issued from the File menu when the application is visible, or from the local menu in the System Tray.

Config menu

The configurations can be stored either in the registry, or in SCC files. Which method to use can be set in the Options dialog. The two modes are:

- **Config mode** - The configurations in this mode are stored in the Windows Registry. This is the preferred and default way to store the configurations.
- **SCC mode** - The configurations in this mode are stored in actual files, called SCC files, with the extension .SCC. SCC files are ASCII files, using a simple format, and are meant to be used for distribution of configurations only.

Config Mode

The menus under the Config menu will adapt to this selection.

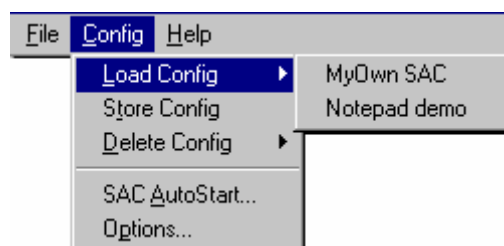


Figure 3: Config menu (Config mode)

Load Config

Disabled when running

Loads an existing configuration.

A Pop-up menu will appear to the right of the menu item, listing all stored configurations by their name. Select the configuration you want to load.

- Store Config** Disabled when running
Stores the current setting as a named configuration.
The configuration will be stored in the registry as the given Config name
The Config name is required to be able to store the configuration.
- Delete Config** Disabled when running
Deletes a configuration.
A Pop-up menu will appear to the right of the menu item, listing all stored configurations by their name. Select the configuration you want to delete.
- SAC AutoStart...** Disabled when running
Opens the SAC AutoStart dialog, where you can maintain which configurations to start automatically when the computer is started.
See *SAC AutoStart*, page 14 and *SAC AutoStart Dialog*, page 25.
- Options...** Disabled when running
Opens the Options dialog, where you can specify SAC Controller operating options and log file options.
See *Options Dialog*, page 27.
- MyOwn SAC Notepad demo** Disabled when running
These menu items are examples of stored configurations. Their names are the Config Names entered when the configurations were defined.
These are just examples of configurations.

SCC Mode

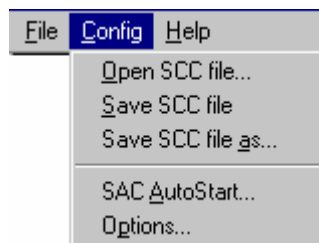


Figure 4: Config menu (SCC mode)

- Open SCC file...** Disabled when running
Opens a File Open dialog, from where you can browse to find and load the desired SCC file.
- Save SCC file** Disabled when running
Saves the current configuration back to the currently loaded SCC file.
If no SCC file has been loaded, a *Save as...* dialog will be presented with the Config name as the default SCC file name. If the SCC file has been loaded and changed, the configuration will be save to that very file.

Save SCC file as...	Disabled when running
Opens a <i>Save as...</i> dialog from where you can specify a name and location for the SCC file.	
The current Config name will be suggested as the default SCC file name.	
SAC AutoStart...	Disabled when running
As in Config mode.	
Options...	Disabled when running
As in Config mode	

Help menu

The SAC Controller on-line help offers extensive help:

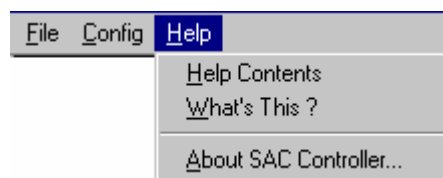


Figure 5: Help menu

Help Contents

Opens the main help page for SAC Controller Help.

What's This

When the cursor changes to an arrow and a question mark, click on the item you want help for. The same feature will also be activated if F1 is pressed from within the SAC Controller application.

About SAC Controller

Shows the application name, software version, copyright information, web-address and email address for support for the SAC Controller.

This command can be issued from the File menu when the application is visible, or from the local menu in the System Tray.

System Tray - Local menu

The SAC Controller will place an icon in the System Tray (bottom right of the screen), as long as this feature is not disabled in the Options dialog. When the SAC Controller is processing a SAC file (running), the application (user interface) will be hidden, and it will disappear from the Taskbar (bottom of screen). The icon will be green if the application is processing a SAC file, and if not it will be red. The figure below shows one instance of the SAC Controller running a SAC file, and one instance that is stopped. The only way to interact with the SAC Controller then is through this local menu. This menu can be activated by right-clicking the mouse on the SAC Controller icon.

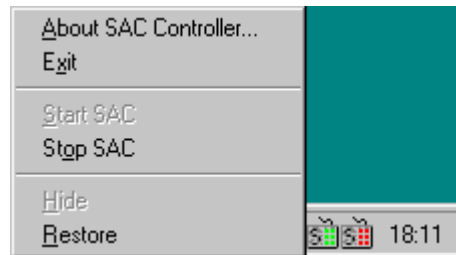


Figure 6: System Tray – Local menu

About SAC Controller

Shows the About box, described above.

Exit

Exits the SAC Controller, described above.

Start SAC

Starts the SAC processing, described above.

Disabled when running

Stop SAC

Stops the SAC processing, described above.

Enabled when running

Hide

Hides or minimizes the SAC Controller application, described above.

Disabled when hidden

Restore

Restores the SAC Controller application. The Main dialog will be shown in the current state. It can be invoked while processing a SAC file or not. Double clicking the icon will have the same effect as this command.

Enabled when hidden

Note that there is no way to activate the SAC Controller with the keyboard if it is hidden. If the computer is running without a mouse or other pointing device, the checkbox “Use System Tray” in the Options dialog should be unchecked.

SAC AutoStart Dialog

The SAC Controller can also start SAC Processing automatically when the computer is started, which is called AutoStart. In the SAC Controller AutoStart dialog you can define and maintain the configurations to be AutoStarted.

When a configuration is saved, either as a Config or an SCC file, this configuration can be defined as an AutoStart configuration. The AutoStarts may be a mix of Config and SCC files. If the current configuration has not been stored since the last change, the user will be asked to abandon the changes before showing the AutoStart dialog.

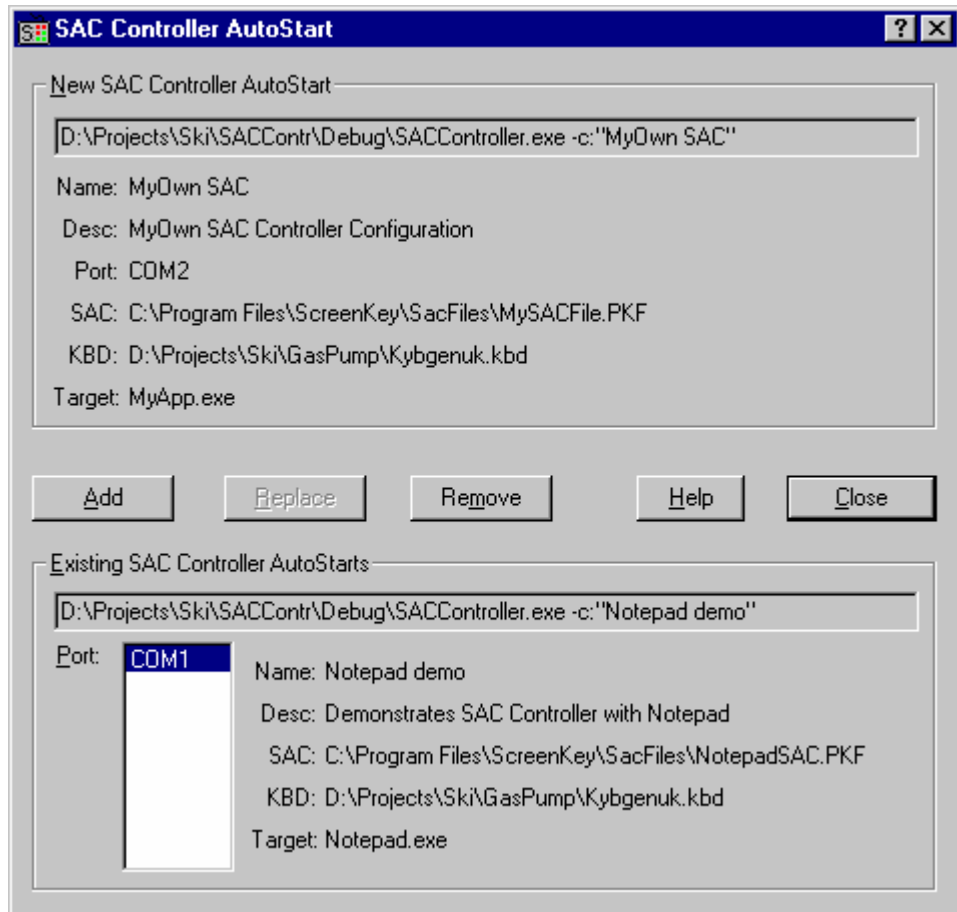


Figure 7: SAC Controller AutoStart dialog

At the bottom of the dialog, any existing AutoStart configurations will be shown. These are shown per communication port, and can be view in detail by selecting the communications ports from the Port list. The command line with parameters appears at the top of this section, and the details will be shown below this.

The new command line and the corresponding details is shown in the top section. As mentioned above, one AutoStart configuration can be defined per communication port. Some of the buttons will therefore be disabled depending if any existing configurations are using the same port as the new configuration.

Add

Button

Adds the new configuration to the list of AutoStarts. If a configuration using the same communication port is already defined, the Add button will be disabled. Instead the Replace button will be enabled so the existing configuration can be replaced with the new one.

Replace

Button

Replaces the existing AutoStart with the new one. This button will be enabled if the new AutoStart configuration is using a communication port that is used already by an existing AutoStart.

Remove

Button

Removes the current selected configuration. This button will only be enabled if there are existing AutoStarts defined and a port is selected.

Close

Button

When finished, the Close button will close the dialog. All maintenance of the AutoStart list is performed immediately, so there is no Cancel button in this dialog and the Close Window button will have the same effect as the Close button.

Port

List

By selecting a communication port from the Port list, you can view the AutoStart setting for the specific port. The Remove button will also be enabled, and the AutoStart item can be removed. If the new AutoStart about to be defined uses the same port, the Replace button will also be enabled..

To define a configuration as AutoStarting, the desired configuration has to be loaded in the main dialog (SAC Controller), before the AutoStart dialog can be used to add it to the list.

Note:

The SAC Controller user interface prevents the user from deleting a configuration defined as an AutoStarting configuration. This does not apply to SCC files, because they can be deleted from outside the SAC Controller.

Another danger is that a configuration that is defined as AutoStart, can be changed and then saved, probably resulting in a different behavior. This is allowed, as long as the Port is not changed to the same as an existing AutoStart configuration. You will be able to save such changes, but can then expect problems if the AutoStart conflict is not resolved before a restart of the computer.

If such a conflict occurs, the user will be warned when entering the SAC AutoStart dialog, and the conflict must be resolved before exiting the dialog. The user will also be warned before exiting the SAC Controller, and the conflict must be resolved before exiting the application.

Options Dialog

Options are grouped into two categories, SAC Controller Options that are generic operations options, and log file options. These settings apply to the SAC Controller application itself, and therefore all configurations.

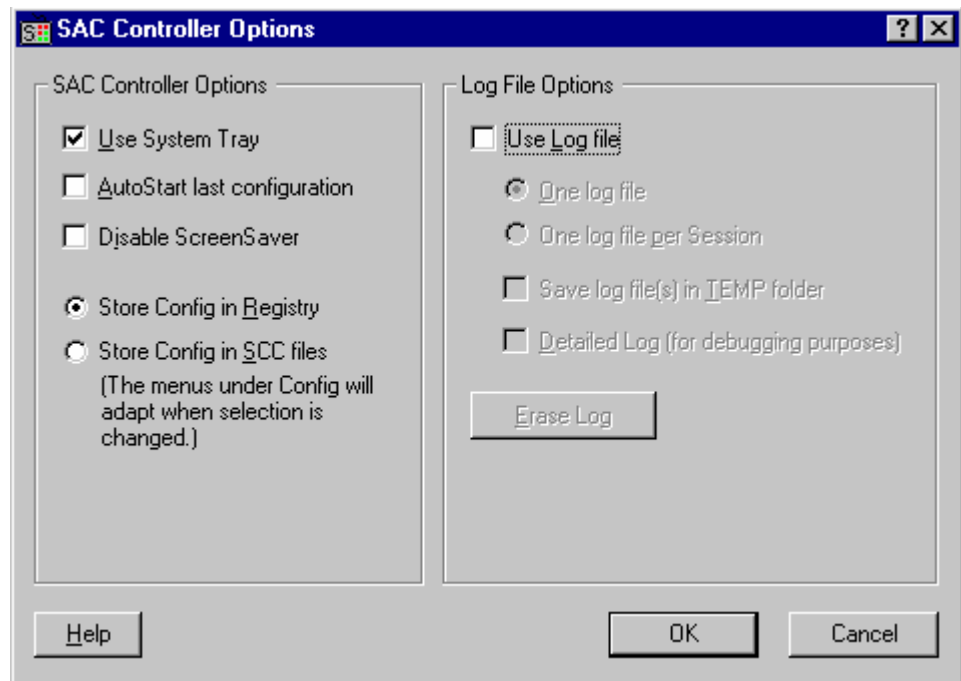


Figure 8: SAC Controller Options dialog

SAC Controller Options

Use System Tray

Default is **ON**

Decides if the program shall place an icon in the system tray when running.

When the SAC Controller starts, it will be shown as an ordinary application, but when it is running the SAC file, it will be minimized automatically. When this checkbox is ticked and the application is minimized, it will disappear from the taskbar. Instead it can be found as an icon in the system tray. If the user does not want the application to disappear, this checkbox can be unticked, and it will act as a normal application. One reason for not wanting it to go to the system tray can be if the mouse is not in use.

AutoStart last configuration

Default is **OFF**

The SAC Controller can be started with or without command line parameters. If it is started without parameters, the user interface will be visible and the configuration can be defined. To automatically start SAC processing, parameters can be given on the command line (like with a shortcut). These can be the parameters needed to run, a configuration name or an SCC file.

When this flag is set, the SAC Controller will retrieve the parameters last used and start SAC processing. This flag must be used with caution, and preferably only while developing SAC files and testing them.

Disable ScreenSaver

Default is **OFF**

If the Disable ScreenSaver flag is set, the SAC Controller will disable the Windows Screen Saver when SAC processing is started. Some ScreenSavers may cause problems for the SAC Controller, and if the keys are pressed on the ScreenKey console while the ScreenSaver is active it might also cause problems. If problems are seen, try to check this checkbox, and set the monitor up to go to "sleep" after a time period instead, if the monitor has such a feature. When the SAC Controller is not doing SAC processing, the ScreenSaver will be turned back on again.

Store Config in Registry Store Config in SCC files

Default is **Store Config in
Registry**

The configuration information for the SAC Controller, which is information like COM port, SAC file etc., can be stored either in the registry, or in SCC files.

If the SAC Controller is started with a Registry Configuration as command line parameter while it is SCC mode, the mode will change to Config mode, and visa versa.

The menus under the Config menu in the main dialog will adapt to this selection.

The SCC files are ASCII files, using a simple format, and are meant to be used for distribution of configurations only.

Log File Options

The SAC Controller can be set up to create a Log file where important information during operation of the program will be stored.

Log File modes

There are two modes of operation, normal mode and detailed mode.

In **normal** mode the following information will be stored:

- Start and stop of the SAC Controller
- Start and stop of SAC processing, including parameters.
- Information about loading and storing of configurations.
- Error messages, SAC processing errors, console communication errors, console errors and errors occurring when operating the application.

In **detailed** mode the following additional information will be stored:

- KeyStoke events, including information about the key pressed and the corresponding scancodes.
- KeyLockTurn events, including the new KeyLock position.
- MSRSwipe events, including track data.
- MenuChange events, including new menu name.
- TrackChange events, if window is being tracked or tracking is lost.

The detailed log should be used for debugging purposes only because the file could grow very large.

Use Log fileDefault is **OFF**

If set the SAC Controller will generate a log file, and the level of details depends on the setting of *Detailed Log*.

One Log fileDefault is **One Log file****One Log file per Session**

The log files can be generated as one continuous file or as one file per session. A session is when the SAC Controller application is started.

If **One Log file** is used, the file name is *SACControllerX.log*, where *X* is a running number from 1 upwards (1 per instance running)

If **One Log file per Session** is used, the file name is *Scyymmddhhmmss.log*, where yy=year, mm=month, dd=day, hh=hour (24 hour), mm=minutes and ss=seconds.

Save log file(s) in TEMP folderDefault is **OFF**

The file will normally be stored in the ScreenKey/Bin folder, but it can be set up to be stored in the Windows TEMP folder. This folder is defined in the operating system, as the TEMP or TMP environment variable. By storing in the TEMP folder, it or they, can be deleted more often because this folder would normally be emptied every so often.

Detailed LogDefault is **OFF**

If ticked, the SAC Controller will generate a detailed log as described above.

Erase Log

Sticky Button

Default is **OFF**

The Log file, together with the contents of the Log window in the main user interface window, can be erased easily by clicking on the Erase Log button. If the button is clicked, the content of the log file will be erased when the user clicks OK and the dialog disappears. This function will NOT erase previous used log files, only the current.

C O N F I G U R A T I O N D I S T R I B U T I O N

Distribution of SAC Configurations

File formats, Locations and Registry Entries

If you plan to use the SAC Controller with your software solution, you might want to create SAC files and SAC configurations, and distribute these with your software. You might also want to set the SAC Controller options different from the default values.

You should then install these in the existing ScreenKey folders. By default the SAC Controller installation program will suggest you install it to a folder called C:\Program Files\ScreenKey, but users are allowed to install it elsewhere as long as other ScreenKey software is not already installed. In that case you are forced to install to the existing folder structure. Under the ScreenKey folder there is a fixed structure of subfolders:

ScreenKey

- Bin - Contains all program files, help files etc.
- Doc - Contains user documentation
- SacFiles - Initially empty, but will contain SAC files.
- etc. - Other folders might be created if other ScreenKey software is installed

The *ScreenKey/Bin* folder can be found in the registry under:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\  
App Paths\SACCont.exe\Path
```

SAC Files

Any SAC files should be installed to the **ScreenKey\SacFiles** folder. The SAC Controller defaults to this folder.

The SAC Editor might or might not be installed on the target computer, depending on if the operator is meant to be able to alter the SAC file. If the SAC Editor is installed, which has a separate installation program, this will create this folder and try to use it for SAC files created.

KBD Files

Remember that the KBD file is the link between the SAC Editor and the SAC Controller, and that the same KBD file **MUST** be used when creating the SAC file and running it with the SAC Controller.

If you have localized or altered the KBD file (KeyBoard Definition file), you should install this in the **ScreenKey\Bin** folder. The file should then be renamed so it describes the configuration. If the original KBD file will not be used anymore it should be removed.

There is one problem with using a KBD file different from KybgenUK.kbd, the SAC Editor uses this as default. There is though a way to change this default setting, changing the file named **paskeyed.ini** found in the **ScreenKey\Bin** folder when the SAC Editor is installed. This is a standard INI file and can be changed with Notepad. Change the ScanKeyFile under the PAS Editor section to the name of your new file, without the .kbd extension.

```
[PAS Editor]
ScanKeyFile=NewKBDFile
```

Since the SAC Editor does not have to be installed on all computer running the SAC Controller, this install will also distribute a KBD file; kybgenuk.kbd.

Keyboard Definition file format

The default .KBD file, KYBGENUK.KBD, supplied with SAC Editor and the SAC Controller is based on the 101 key UK keyboard layout. The .KBD file is a text file and as such can be edited using any text editor.

The following extract from KYBGENUK.KBD shows the various elements of a .KBD file:

```

          Key number
          /      UnShifted
         / /      Shift
        / / /      Ctrl
       / / / /      Alt
      / / / / /      Comment (Key name)
     / / / / / /
    / \ / \ / \ / \
   / \ / \ / \ / \
  0, "011B011B011B   Esc"
  1, "3B0054005E006800F1"
  2, "3C0055005F006900F2"
  3, "3D00560060006A00F3"
  4, "3E00570061006B00F4"
  5, "3F00580062006C00F5"
  6, "4000590063006D00F6"
  7, "41005A0064006E00F7"

```

For example, Shift-F7 has a scan code value of 5A00 where 5A is the position code and 00 for the ASCII value (indicates a special key).

NOTE: The Editor expects to find the scan codes for certain keystrokes in a fixed position in the .KBD file. The fixed position keystrokes are at lines 42, 57, 69, 70, 71, 73, 74, 85 and 89-100 in the default file KYBGENUK.KBD

NOTE: The KeybgenUK.kbd can normally be used when a standard 101 key keyboard with US layout is used.

SAC Configurations

The SAC Controller uses SAC Configurations to hold the information needed to run. These configurations contain the following information:

- A Config Name
- A Config Description (optional)
- A communication port specification
- A SAC file name
- A KBD file name (KeyBoard Definition file)
- A Target specification (Optional)

The Configurations can be stored in two different ways, either in the Windows Registry (Config mode), or in an SCC file (SCC mode).

Config in Registry

When the SAC Controller is installed, it creates a key structure in the registry.

HKEY_LOCAL_MACHINE\SOFTWARE\ScreenKeys\SAC Controller

Under this key, the application options are stored, but also the Configurations. Each configuration has a key with the same name as the Configuration. Under this key there are a number of sting values:

- Name - Name of the configuration, always the same as the Config key
- Description - Description of the configuration (optional)
- Port - Name of the communication port
- SACFile - Name of the SAC file with full path.
- KBDFile - Name of the KBD file with full path.
- Target - Target application/window specification

The string values are all of the type REG_SZ, and they all have to exist even if they are blank.

You are free to create these registry keys yourself, either by doing it programmatically, or by importing a registry file. This can be done from an installation program, a separate program or by using the Registry Editor.

If the SAC Controller has not been installed yet, the key **ScreenKeys\SAC Controller** might not exist, so it must be created, or even better, to add these registry keys after the SAC Controller has been installed.

SCC file

Another way to distribute a configuration is to use an SCC file. An SCC file is a **SAC Controller Configuration** file. This file holds the same information as the registry configurations. The file is a plain ASCII file with the same layout as .INI files. It can be created manually or by using the *PrivateProfile* functions in Windows.

The format of the SCC file is as follows:

```
[SAC Config]
Name=Config name
Description=Description of configuration
Port=Portname e.g. COM1
SACFile=SAC file name with full path, extension .PKF
KBDFile= KBD file name with full path, extension .KBD
Target=Target window specification (process name and options)
```

For the SAC Controller to locate these files, they must have the file extension .SCC.

The SCC files are meant to be used for distribution of configurations only.

SAC Controller Options

The SAC Controller stores its options in the registry under the key:

HKEY_LOCAL_MACHINE\SOFTWARE\ScreenKeys\SAC Controller

These options/flags will be set when the program is exited or when the options are changed. These flags can be set manually or from an installation program. They are all of the type REG_DWORD. They are:

- UseSysTray - 1 if to place icon in the System Tray, 0 if not
- AutoStartLast - 1 if to autostart the last configuration, 0 if not
- DisableSS - 1 if Disable ScreenSaver is active, 0 if not
- SCCMode - 1 if in SCC mode, 0 if in Config (registry) mode
- UseLogFile - 1 if we are going to generate a log, 0 if not
- OneLogFile - 1 if we are using one contiguous log, 0 if one per session
- LogToTEMP - 1 if we shall store log file in TEMP folder, 0 if in Bin folder
- DetailedLog - 1 if log is to be detailed, 0 if standard log

See the description of the *Options Dialog*, page 27 for further details, and default values.

You are free to create these registry keys yourself, either by doing it programmatically, or by importing a registry file. This can be done from an installation program, a separate program or by using the Registry Editor.

If the SAC Controller has not been installed yet, the key **ScreenKeys\SAC Controller** might not exist, so it must be created, or even better, to add these values after the SAC Controller has been installed.

Distributed Files and Components

The SAC Controller installation contains the following files:

File name	Description	Registering
SACCont.exe	SAC Controller application.	NO
SACCont.hlp	On-line help file for the SAC Controller	NO
SAC Controller User's Guide.pdf	SAC Controller User's Guide (this document)	NO
SACEngin.exe	SAC Engine Control.	YES
SACEnginPs.dll	Proxy Stub for SAC Engine Control.	YES
SACEngWr.dll	SAC Engine Control Wrapper DLL. Used by the SAC Controller.	NO 1)
SKPlayBack.dll	ScreenKey PlayBack DLL. Used by the SAC Controller when feeding keystrokes to target application.	NO 1)
Psapi.dll	This is a Microsoft DLL. It is used by the SAC Engine Control to track the Target window. It is though only necessary for Windows NT and 2000. When running on Windows 95/98 or ME, Windows built in functionality is used instead. This DLL is a part of the Windows SDK and can be freely distributed.	NO 1)
SkAxCtl.exe	SKI ActiveX Control. This is used by the SAC Engine Control	YES
SkAxCtlps.dll	Proxy Stub for SKI ActiveX Control.	YES
KybggenUK.kbd	KeyBoard Definition file for UK layout.	YES
*.skf	ScreenKey Font files. These files may be distributed for CodePages other than 1252. They should be distributed with the control, residing in the same ScreenKey/Bin folder.	N/A
pkficon.ico pkbicon.ico	SAC Icon files. See below.	YES

1) These files are installed to the Windows System folder (WINNT/SYSTEM32 or WINDOWS/SYSTEM)

Some of the EXE and DLL files have to be registered in the registry. SACEngin.exe and SkAxCtl.exe can do the registering by themselves, but the DLL files need to be registered with the program RegSvr32.exe. RegSvr32.exe is distributed from Microsoft Corp., and can be used freely. It can be found in the Windows System folder (WinNt\System32 or Windows\System).

To register the components one has to execute the following commands, in the following order:

```
RegSvr32 SkAxCtlPS.Dll
SkAxCtl.exe /RegServer
RegSvr32 SACEnginPS.Dll
SACEngin.exe /RegServer
```

This can be done by executing a batch file, or by including the commands in an installation program.

Be aware that if both files, Regsvr32.exe and the DLLs, are not in the current working folder or the RegSvr.exe program is not in the standard path, it might be necessary to specify a path for the either one or both files.

Note that the components can be installed in any suitable directories, as long as all supplementary files are installed together with EXE-files, and that they are correctly registered.

Note! The registration of the components is done automatically by the SAC Controller Installation program. The above details is for information purposes only.

SAC File Icons

To define the SAC file icons, the icon files has to be installed to the hard disk and the following keys must be defined in the registry:

```
HKEY_CLASSES_ROOT\pkf
  (default) = pkffile

HKEY_CLASSES_ROOT\pkb
  (default) = pkbfile

HKEY_CLASSES_ROOT\pkffile
  (default) = "SAC File"
  DefaultIcon = "<full path>\pkficon.ico

HKEY_CLASSES_ROOT\pkbfile
  (default) = "SAC Backup File"
  DefaultIcon = "<full path>\pkbicon.ico
```

DCOM & Windows 95

The controls are using COM to communicate with each other. Com is built in to all Windows operating systems, but there is an error in COM in Windows 95. Before the controls can be installed on a Windows 95, DCOM95 has to be updated. This installation is available, free of charge, from Microsoft Corp., and is also available on the SAC Engine Developer CD.

PSAPI.DLL

The dynamic link library psapi.dll contains functionality for retrieving process information in Windows NT and 2000. When installing to these operating systems, this dll must be present. If an existing version of this file is older that the one distributed with the SAC Engine Control, this must be upgraded.

Windows 95, 98 and ME has this functionality built in to the operating system, and the file is therefore not needed on these systems.

A P P E N D I X A

Documentation Control

A.1 Change Control

This document is the responsibility of the author and is subject to formal change control after the initial approved release (i.e. issue 1.0).

A.2 Abbreviations Used/Terms of Reference

SKI	ScreenKey Interface
SAC	ScreenKey Active Control
SCC	ScreenKey Controller Configuration
KBD	KeyBoard Definition (file)
POS	Point of Sale--the cash register in a shop.
PWD	Superceded ScreenKey console software toolset.
LCD	Liquid Crystal Display
MSR	Magnetic Stripe Reader
ActiveX	Name of technology used for communication between applications/modules.
COM	Component Object Model, As for ActiveX.
DLL	Dynamic Link Library

A.3 Historical Change Reference

Issue	Date	Author	Changes Made
1.0	04/10/01	Bjorn Liane	Initial release
1.1	30/11/06	M McDonnell	Update for new registry paths

A.4 Change Summary

Issue	Change description
1.1	Update for new registry paths under "ScreenKeys" instead of "RTI Ltd"