PROTECTION PLUS PROFESSIONAL

Copy Protection and Total User Control

Version 2.0 for Clipper®
Copyright 1991
All Rights Reserved Worldwide

PROPLUS
Software

4260 Americana Drive Suite 126
Stow, Ohio 44224-4860

216-923-1768

TABLE OF CONTENTS

1. INTRODUCTION

1.1 How to use this manual

This manual is divided into several parts. Every section contains important information and should be read, although, of course, some chapters are more important than others. It would be in your best interest to read the manual at least once. The section names are listed below with a short description of their contents.

Keep in mind that online help is available for the library functions through the Norton Guides® database PROPLUS.NG; context-sensitive help is available in most areas of PROPLUS.EXE by pressing the F1 key.

The *"Introduction"* contains some important information you should be familiar with, such as how to get technical support, system requirements, and the licensing agreement. Be sure to read the section "Definitions" to understand some of the terminology used throughout this system.

If you want to get up and running as soon as possible, read the *"Quick Start"* section. After experimenting with Protection Plus for a little while, you should take time to read the rest of the manual in order to get a firm understanding of the entire system.

The first main part of the Protection Plus system is described in the *"PROPLUS.EXE"* section. PROPLUS.EXE is the program file for the client tracking and serial numbering of your applications. It creates the control file that is used by the library functions.

"Library Reference" gives explanations dealing with compiling and linking using the Protection Plus library as well as contains the syntax to each function in the library. For your convenience, a quick reference function list is included.

When you are ready to start coding, *"Sample Source"* gives you some examples that are ready to compile and run. This section gives a brief description on how to get started with the source code provided and general descriptions on accomplishing the various types of protection.

After completing the PROPLUS.EXE, library reference, and sample source sections, the section *"Commonly Asked Questions"* gives the last basis of information to get you started on the system. Look here before calling technical support.

If you are the type of person to have someone else do the dirty work, *"ProPlus Programming Service"* gives information about having a technical support person implement the library functions in your application and give thorough telephone support on how to modify it in the future.

1.2 Overview

Protection Plus is the most advanced copy protection and user control system currently available for Clipper®. It is written entirely in Clipper®, Microsoft® C, and Microsoft® Macro Assembler.

The Protection Plus system contains two parts. Part one is the PROPLUS.EXE program that stores all protection and version information for each customer. It is very similar to a client tracking program but is customized to managing software. Part two is the Clipper® library that contains many functions controlling the copy protection and user control.

No *special* programming knowledge or experience is required to use Protection Plus functions in Clipper® applications; simply include the desired function calls in the program source code and add the Protection Plus library to its link list.

One of the things that makes Protection Plus so great is its flexibility. Even though no special programming knowledge is required to use the Protection Plus functions, there is another requirement - *imagination!* The examples shown below only give an overview of what the system can do. Many Protection Plus users have called us up and said "what if?" So far, even though their questions have deviated from the original intent in the design of Protection Plus, **none** of them have been turned away with "It can't be done." *The moral of the story is that due to the nature of the Protection Plus system, almost any requirement dealing with protecting applications and user control can be met!*

Like Clipper® itself, applications using Protection Plus require no run-time licenses of any kind. One has the ability to protect an unlimited number of programs. Protection Plus does **not** change disk sector information or cause harm to any disk media. Protection Plus makes it possible to hard-code a unique serial number into applications automatically without recompiling each time. It is possible to completely protect and serialize an application in as little as 5 seconds.

Protection Plus Professional is easy to use!

1. Add a few lines of source code to the application and re-compile.

2. Run PROPLUS.EXE; add the new client to the customer file with the serial number and other parameters set correctly; press a function key with the application's distribution diskette inserted.

3. If copy protection is used, the first time the user runs the program, they will be prompted with a code entry number.

4. Simply load PROPLUS.EXE, bring this user's record up on the screen, press the F3 key and enter their code entry number. PROPLUS.EXE will then list the code to "add protection." When they enter the code, they will have a running program that *can't be executed on unauthorized computers*. They can never re-use the unlock code because the code entry numbers change every time the application is executed and the unlock codes vary every day as a function of the code entry numbers. All unauthorized copies of this program will not work and PROPLUS.EXE will even track if the user is trying to install the program again on a different computer once it has been installed.

Install an application as a DEMO!

1. If the demo expiration date in the client's record of the PROPLUS.EXE database is non-blank, the application will stop running after that day. This is an extremely effective way to generate sales. After the user has been using the application for 30 days free, they may have loaded so much data that they cannot live without it. When it stops working they will rush out a check to register the software.

2. Oh, so you ask, what happens when the user keeps backing up the date and time on their computer? Can they fool Protection Plus? The answer is simply *no way*! If the user tries to back up the date, the Protection Plus code is designed to catch them. The program knows when the program was started as well as when the last session ended. Each time the program is used, the window for usage gets smaller and smaller until they *must* purchase it.

3. If the user would like the demo version to keep running for a few extra days or a week or even a month, they can call you to get an unlock code. This way they can continue to use the program while they send in a payment.

4. When the payment is received, the application can be released from the demo mode. REMEMBER! It's no longer a demo, but it *can* be copy protected.

<u>Speaking of Payments!</u>

1. If any of the software distributed is leased by clients, pay attention! When adding a user to the PROPLUS.EXE database, simply make them a payment client and enter what day of the month payment is due. This will insure strict compliance with the payment agreements because the program will stop running on this date.

2. Each month when a check is received, an unlock code needs to be entered on the user's computer that will advance them to the next month. If by chance the payment day falls on a Saturday or Sunday, they will have full use of the program until the following Monday.

3. Once their payments have been completed, it is possible to give them a special code that will unlock them forever.

4. REMEMBER! It's no longer locking for payments monthly, but it *can* be copy protected.

Unlock Codes

And if that is not enough, Protection Plus allows one to customize 18 extra codes that may be used in applications. That is, one can have hidden procedures in the program that require an unlock code to access. If a user forgets their password, for example, they can call and receive a code to bypass the password-checking function. If it is undesirable for the user to have the ability to change certain information online, they can call for special access to that information.

List of Features

1. Copy protect Clipper® programs so they will not be handed out. This increases sales.

2. The unlock codes are different each time the program is loaded. This means that two users of the same product cannot exchange unlock numbers to gain access to the application software.

3. It is possible to know when a user is trying to install the system on a different computer than the one the software was originally installed on.

4. It is possible to remove protection from a computer in order to re-activate it on a new computer.

5. A hard-coded serial number can be written into the product's .EXE file without recompiling the source each time.

6. Each copy of Protection Plus is coded uniquely. No two libraries are the same.

7. There is no limit to the number of programs than can be protected. One can use it on as many programs as he wishes - even in the same directories!

8. PROPLUS.EXE is used as a client tracking program. It tracks who is using each application, what version they last purchased, when the product was shipped, when it was unlocked, how many executions they are allowed, how many workstations (on a Local Area Network, if any) can use the software, and much more. PROPLUS.EXE will automatically assign a new, unique 6 digit serial number to each client added. It can generate unlock codes easily as well as assist in creating distribution diskettes via a batch file at protect time.

9. Functions are provided in the library that allow one to add the special protection features that PROPLUS.EXE performs *to an existing client tracking program!* That is, one can protect diskettes, hard code serial numbers, and generate unlock codes from a custom written application!

10. It is possible to restrict an executable from only executing a certain number of times. This number is changeable from within the program, if desired, extending the program to a possible buyer.

11. Supports DOS and Local Area Networks.

12. There are two types of LAN protection functions - specific and non-specific. The specific type sets the application up to run on a particular number of workstations. The first "X" number of workstations (you set the "X") that execute the program are the *only* ones allowed to use it. The non-specific type of LAN protection limits the number of *simultaneous* users on the LAN to the same "X."

13. Soon to be available in other languages.

1.3 System Requirements

The following list of computer equipment is required in order to properly use the Protection Plus Professional system:

IBM PC/XT/AT or compatible
DOS 3.0 or higher
640 Kb RAM
Hard Drive with at least 1 Mb free
Monochrome / CGA / EGA / VGA Monitor
Any Parallel Text Printer (optional)
Clipper® Summer '87 or 5.x

The program, PROPLUS.EXE will operate on most Local Area Networks (LAN). This product is multi-user. You need to purchase unlocks for each workstation you intend to use with PROPLUS.EXE.

1.4 Installation

Protection Plus is supplied on either 5.25" or 3.5" diskette(s). Please be sure to read the "README.TXT" file on disk #1. It contains any last minute changes, corrections and information since the manual was printed and should be read before continuing. It may be viewed using any ASCII text editor or displayed at the DOS prompt by typing:

```
C> TYPE A:README.TXT
```

If the text scrolls off the screen, press the [Pause] key or Control-S to pause the display and press any other key to continue. Information in the README.TXT file takes precedence over information in the manual, including the following installation procedure.

Installing the software

To install Protection Plus, make a working subdirectory on your hard drive where you would like to put the system files. Make that the active directory, place the distribution diskette #1 in a floppy drive, and type "A:UNCOMP" [enter].

UNCOMP.EXE is a self-extracting archive of compressed files. The files will be uncompressed in the active subdirectory. If you install the software in a directory that has files by the similar name, you will be prompted to overwrite those files individually.

For example, if your floppy drive is A:, and you would like to install the software onto C:\PROPLUS, type:

```
C:\> MD \PROPLUS [enter]
C:\> CD \PROPLUS [enter]
C:\PROPLUS> A:UNCOMP [enter]
```

If the software comes on more than one diskette, you must repeat the "A:\UNCOMP" command for the second diskette. If you wish, you may place the destination directory into your DOS path statement in your autoexec.bat file so that PROPLUS.EXE can be run from any drive and directory.

Be sure to read the following descriptions of the files installed, and move them to the appropriate directories. The final step is to read section 3.1 Initial Installation on page 45 to set up PROPLUS.EXE correctly. Once this is done, installation will be complete.

NOTE: To assist you in completely understanding the product, we strongly encourage you to read the section "Definitions" on page 33 and the section "Commonly Asked Questions" on page 139.

1.4.1 Filename descriptions

PROPLUS.EXE is the program file for the client tracking and serial numbering of your applications. Upon complete installation, this program will create the necessary database files and indexes it needs.

PPLUSS87.LIB and PPLUS50.LIB are the Clipper® libraries. One is for Clipper® Summer '87 and the other is for Clipper® 5.x.

PP_S87.OBJ and PP_50.OBJ are object files that must be linked into your application. Use the correct one for your compiler.

PROPLUS.NG is the Protection Plus Norton Guide® database which requires a Norton Guide® reader to use. This reader is distributed with Clipper® 5.x.

SAMPLE.TXT is a ASCII text file that provides a complete explanation of the sample source code that comes with Protection Plus. The sample source code shows where function calls to the Protection Plus library are logically placed most of the time. If you would like to utilize all of the protection features of this system, you should be able to take these example programs, change their cosmetics, and pop them into any of your existing applications.

README.TXT contains any last minute changes, corrections and information since the manual was printed and should be read before continuing, as noted above.

UNLOCK.EXE is used when an unlock code is needed for a user quickly. This avoids having to load the client tracking program just to find an unlock code. It can also be copied to a laptop or notebook computer for travelling purposes.

HCSERIAL.EXE allows you to hard code a serial number from the DOS prompt. This was written to be used in a protection batch file.

1.5 Changes since previous release

This section describes important information regarding system changes that have been made since the previous releases. There are several areas of the product that have changed! *If you are an old user or Protection Plus, please be sure to read this section in its* ***entirety*** *before attempting to use the software.* The material in this section is a subset of material found in the rest of the manual. Be sure to read the full details on the new functions in the appropriate manual sections.

1.5.1 Copying data files

The previous release required PROPLUS.EXE and the data to reside in the \PROT\ directory. This is no longer true. You should install this product (as described in the installation section) to a directory with a different name. You only need to copy the following files from your \PROT\ directory into the new one:

```
CLIENT.DB?        <- note the question mark
PRODUCT.DBF
SETUP.DBF
DISK.DBF
```

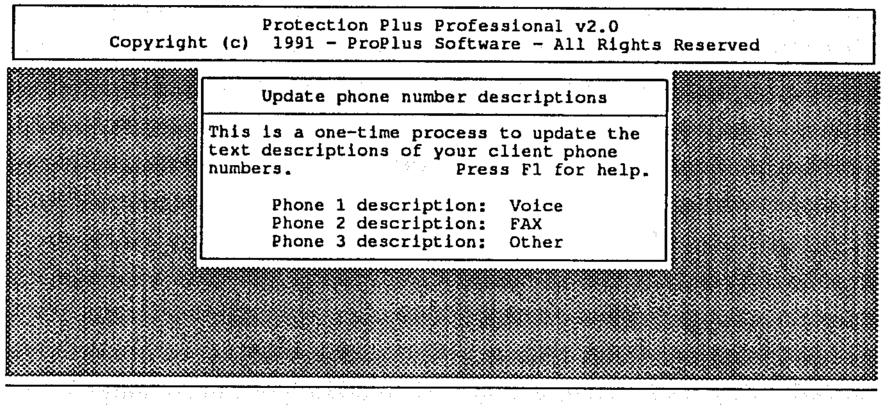
If you copy only these files, you will guarantee the removal of all unnecessary files. That is, some of the database files changed and not as many indexes and memo fields are present. If you decide to copy over your existing data as noted above, you must convert the data files by typing

```
PROPLUS [space] U
```

the first time you execute it. Once the files are converted, you can drop the "U" in the above line. All control files MUST be generated with this new PROPLUS.EXE. Old control files will NOT work and will generate a .F. from pp_chkpp().

If you are updating your data files from a previous release of Protection Plus, the first time you type "PROPLUS U", the following screen will appear. In the previous version of Protection Plus for Clipper®, there were two phone numbers allowed in each client database record. In the new version, three phone numbers can be entered each with a text field to define what the phone number is.

Since the old version had the name "Phone" and "FAX" hard-coded by the phone number, "Voice," "FAX," and "Other" are the defaults for the conversion process. If you would like to change the text description associated with your current data, enter the new text on the update screen when it appears.



```
                    Protection Plus Professional v2.0
         Copyright (c)  1991 - ProPlus Software - All Rights Reserved

                 Update phone number descriptions

                 This is a one-time process to update the
                 text descriptions of your client phone
                 numbers.                 Press F1 for help.

                       Phone 1 description:  Voice
                       Phone 2 description:  FAX
                       Phone 3 description:  Other
```

Descriptions

If you would like to change the text description associated with your current data, enter the new text in these fields.

1.5.2 Required changes on your part

You MUST link in a .OBJ file into the ROOT of your application whether or not you use overlays. This file is used for the serial numbering routines and copy protection schemes. There are two .OBJ files distributed. PP_S87.OBJ is for Clipper® Summer '87 and PP_50.OBJ is for Clipper® 5.x.

All functions are backward compatible with two exceptions - pp_cenum() used to create a public memvar called PROPLO which was then used by pp_ucode(). This public memvar was removed and in turn, you need to pass the code entry number to pp_ucode() via the second parameter. You will need to create a private, local, or public memvar to handle the result of pp_cenum(), print it on the screen for your client, and pass to pp_ucode() as the second parameter. This makes the nesting of hidden user codes easier.

Also, an optional third parameter to pp_ucode() is used to make the security even tighter. You may pass the pp_compno() as the third parameter (or any fixed number) and make the codes generated only work on a specific computer. If you do not pass the third parameter, the default fixed number is 1. This parameter is used if you would like to have the computer number as part of the unlocking sequence. If you wish to have maximum security, using this parameter would prohibit a user from telling you his hard drive crashed, when he only wanted to unlock your program on another computer. That is, he wants to unlock your program on another computer, so he calls you up while he is sitting at his second computer and tells you he needs a new unlock code for his computer (you think it is his first one). If you asked him what the pp_compno() was on his screen, and he lied and told you the number that was on his FIRST machine (which, according to your records is a legal copy), the unlocking sequence would ***not*** work on his second machine.

This parameter is optional and defaults to 1. You can use the computer number as a basis for ANY of the 18 hidden user codes.

Copy protection functions and parameters

Even though the copy protection functions have a parameter now, the parameter is optional. If you do not send a parameter, the default of "single/external" type of copy protection is used - the same as previous releases.

Public variables reordered

The public variables PROPL* have changed order. If you utilize the public variables via pp_getvar(), you MUST look at the new list and re-code your application with the necessary changes.

1.5.3 Changes in the library

New security measures

ALL security measures have changed. The copy protection files on existing computers are not compatible with the old version. There is a function pp_oldcopy() that will test for the existence of the old copy protection. For the first update of your package, you might insert code like the example shown below.

```
if pp_oldcopy()
    pp_copywrite( 1 )
endif
```

Every copy of Protection Plus Professional system is serialized. The serial number is used in the libraries, PROPLUS.EXE, and in the UNLOCK.EXE program. Unlike previous versions, the whole unlocking system now generates codes that WILL ONLY UNLOCK YOUR APPLICATIONS.

If you have two licensed copies of the system with two different serial numbers, you *cannot* mix the two systems. ***You must use your UNLOCK.EXE with programs linked with your libraries only!***

Now VCPI compatible

The copy protection functions now adhere to 100% VCPI standards. This assures compatibility with all commercial 386 (and above) expanded memory managers. Refer to the definitions section on page 33 for an explanation of VCPI.

Control file can be named anything

The control file can be named anything you wish (except .DRV). That is, you can choose an alternate to the .PPP extension. Simply add the extension to your public PP_CFILE like:

```
PUBLIC pp_cfile
pp_cfile = "proplus.dat"
```

If you plan on using the single internal form of copy protection (described later), the first 8 characters of the control file MUST be the same as the .EXE program file.

The copy protection file (for both single user and multiple user) will have the same first 8 characters of your pp_cfile and will have a default extension of .DRV. You can set the attributes of this copy protection file.

Network functions added

The following is a list of network functions that were added from the previous released version:

pp_lanlim()	- returns the limited # of workstations
pp_lanusers()	- returns the current # of workstations
pp_lanplus()	- adds one user to the system
pp_lanminus()	- subtracts one user from the system
pp_nlanlim()	- sets a new limit to # of workstations
pp_lancheck()	- can we add another user?
pp_lanactive()	- is this application protected for LAN?
pp_lanincr()	- add 1 to limit of # of workstations
pp_landecr()	- subtract 1 from limit of # of workstations
pp_nlanct()	- set a new count value for current # workstations

Added/Changed functions

The following functions were changed or added. Full documentation is available via the Norton Guides® database or this manual. Please take a few moments and review that documentation.

pp_killprot()	- optional parameter added
pp_copywrite()	- optional parameter added
pp_copychk()	- optional parameter added
pp_oldcopy()	- added for backward compatibility
pp_isdrive()	- new function
pp_ctserial()	- new function
pp_ctconfile()	- new function
pp_ctcodes()	- new function
pp_crppp()	- changed for new control file layout
pp_chkpp()	- changed for new control file layout
pp_savepp()	- changed for new control file layout
pp_getvar()	- changed for new control file layout
pp_path()	- new function

pp_lastday()	- new function
pp_ucode()	- added two parameters, (one required)

Three types of copy protection now available

A deep explanation of the three types of copy protection can be found in the "Definitions" section on page 33. The list below gives an overview of the change.

1) *Single external* - (like previous versions)
The copy protection code is stored in an external file by the first eight characters (root) of the control filename with the extension of ".DRV" appended. Use this for most cases.

2) *Single internal* - (embedded in EXE file)
The copy protection computer snapshot is stored in the program .EXE file itself. This is great for applications that are not updated frequently or require an unlock with every update, or when a control file is not practical. Even though a control file is not used, the control file public variable is necessary. Normally, the control file can be named anything you wish (except .DRV). *If you plan on using the single internal form of copy protection (described next), the first 8 characters of the control file MUST be the same as the .EXE program file.*

3) *Multiple external* - (network)
The copy protection computer snapshots are stored in an external file by the first eight characters (root) of the control filename with the extension of ".DRV" appended. This is used when copy protection network functions are used (specific workstations unlocked). Refer to the "Definitions" section on page 33 for more information.

External copy protection file has selectable attributes

You can alter the attributes of the copy protection file created during unlock time. The old version uses the system attribute as the default. The default now is normal attribute. *Be sure not to use the R/O attribute unless you have special reason to do so!*

Version of Protection Plus Professional in control file

The version of the Protection Plus Professional package will be contained in the control file. This will allow for automatic updating to future versions of the package. Currently, all control files that were created with previous versions of Protection Plus will NOT work with this release.

New PUBLIC variables order

The following list has the updated public variable list. Your existing applications will need to be updated to include the changes.

PROPL1	-	Company name
PROPL2	-	Name
PROPL3	-	Address 1
PROPL4	-	Address 2
PROPL5	-	City
PROPL6	-	State
PROPL7	-	ZIP
PROPL8	-	Phone
PROPL9	-	Product name long form
PROPL10	-	Product serial number
PROPL11	-	Distributor name
PROPL12	-	Your company
PROPL13	-	Your address line 1
PROPL14	-	Your address line 2 (city, state, ZIP)

PROPL15	-	Your phone
PROPL16	-	Your FAX number
PROPL17	-	Last date system used
PROPL18	-	Last time system used
PROPL19	-	Expiration date

1.5.4 Changes in PROPLUS.EXE

Hotkey to shell to DOS

Use ALT-Z to shell to DOS from any point in the program. This will allow you to quickly run another program while within the PROPLUS.EXE application.

New PROPLUS.EXE organization

The whole organization of the application has changed. There used to be two separate entities; one was clients and one was products. The client database contained all users of all pieces of software jumbled together by company name. Now, the client database is separated by products. The highest level database is the products. The product database has several new fields:

hard-code?	do you want the serial number hard-coded?
control fn	you can now name the control file anything
create?	do you want to create the control file?
batch file	you can execute a batch file during protect time
when?	execute batch before/after control file creation

The first two options allow full control over the protect diskette operation. Some people only want the control file created and others only want the .EXE file to have the hard-coded serial number. Note that when you convert your old data over to this new format, both of these fields will be No (false). You will need to set them appropriately in all of your products first thing.

The library now allows you to name the control file anything you wish. This will let you hide the control file a bit, if you desire. The name must be entered here so that the control file is named correctly on the destination diskette. Normally, the control file can be named anything you wish (except .DRV). *If you plan on using the single internal form of copy protection (described later), the first 8 characters of the control file MUST be the same as the .EXE program file.*

You can now execute a batch file (or any executable program) during the disk protect process. The first parameter passed to the batch file is the serial number of the current client and the second is the destination drive. You could serialize a program, zip up all application files and automatically create distribution disks! You can get VERY creative with this feature. You have the choice of if the batch file is executed before the hard coding and creation of control file or afterwards. Be sure to read about the DOS utilities that allow for serial number hard-coding from the DOS prompt for use in this batch file. When the batch file is complete, PROPLUS.EXE looks at the DOS errorlevel. If the errorlevel is non-zero, an error is reported to the screen and the protection steps are aborted. The errorlevel number is printed on the screen. If the errorlevel is zero, everything continues fine.

Unlock code names are unique to each product

Now, each product has its own set of unlock codes description accessible from the product definition screen. To access either the batch protection, unlock codes, or customer database, a field "File" concept is used. Simply place the editing arrow next to the correct field and press the return key. These fields are not active during the add process of product definition.

Cannot access until current changes are accepted

This message appears when you change a field in the product database during the edit process and then attempt to access one of the three process-type fields (Process, Page, File). When any changes are made, you must press escape to return to the product browse screen and then press return to select the same product. You may then use the End or Uparrow to go to the bottom of the definition screen and select a process field.

Rewritten in Clipper® 5.01

PROPLUS.EXE is re-written in Clipper® 5.01. The menus now allow a hotkey and the system is a lot faster than the previous Summer '87 version. If you delete PROPLUS.CNF, the new default colors will take effect and the system looks nicer! You will have to re-enter your initial registration name if you delete this file.

Can be copied to and executed from any directory

Previous versions of the product were required to be run from the \PROT\ directory. Now, the application may be executed from ANY directory and it can even be placed in the DOS path to be executed anywhere. The system is smart enough to find its own data files!

Now tests destination diskette drive

When protecting diskettes, a test is performed on the floppy drives. Instead of just reporting an error, an intelligent message is reported to allow you to quickly correct the problem.

One computer number

The previous version had a page of 50 or so pp_compno() numbers. This page was in anticipation of the network functions. The new network functions do not require or allow the control of individual workstations and their computer numbers. Therefore, only one computer number slot is needed (for single users).

Unlock screen change

The unlock codes screen from within PROPLUS.EXE now displays the computer number that is associated with the current client record. If you are using maximum security (use the computer number as part of the pp_ucode() function), this number is a handy reference to verify the number your client tells you at unlock time and to enter into the optional computer number field on the unlock codes screen. If you are not using maximum security, leave the default "1" in the computer number field on the unlock codes screen. The "Definitions" section on page 33 gives more detail.

New client database indexes

Once a particular product is chosen and the customer list is executed, the list of clients is ordered by serial number now - the most logical way! Before, all clients of all products were listed in order of company name. Now, each product's clients are separated and by default they are ordered by serial number. The current index may be changed by pressing the F10 key at the browse screen.

Online help is here

A noble attempt was made to add context sensitive help to all prompts in the application. The text should be similar to that found in the user's manual.

Definable phone numbers

Instead of having a Phone and a FAX slot, three user-definable phone numbers are allowed. When a new client is added, a default "Voice" and "FAX" description is entered automatically.

Multiple distributors

There is now a database for multiple distributors. This feature allows for a quick production of distribution diskettes that are going to the same location. In order for the distributor field to be active, the distributor option must be enabled in the Master Settings database. That option is toggled on and off as needed.

Client tracking functions used

The new client tracking functions described in the library change section are used in PROPLUS.EXE where appropriate.

Added shareware type

The shareware type is just another descriptive name to restrict your applications. It goes along with <D>emo, <P>ayments, <N>one, and <E>xe count.

1.5.5 DOS utilities

HCSERIAL.EXE - to allow you to hard code a serial number from the DOS prompt. This was written to be used in a protection batch file. The parameters are shown below.

The parameters are:

<file>	file name to hard code serial number
<string>	serial number to be hard coded (6 char max)

If an error occurs, the DOS errorlevel will be set to one. This allows for perfect integration in a PROPLUS.EXE batch file during protection time.

UNLOCK.EXE - has been changed to allow you to enter an optional third parameter for the computer number. If you are going to use maximum security, you will want to add the third parameter to the pp_ucode() function. If you add the parameter to pp_ucode(), you must enter the third parameter in UNLOCK.EXE to get the correct unlock codes. Keep in mind that the third parameter is completely optional in both functions.

1.5.6 Sample source files

The sample source files are described in SAMPLES.TXT. There are sample Clipper® source files describing general use, LAN copy protection, regular copy protection, and use with Zachary® development.

1.6 License Agreement

ProPlus Software hereby grants you a non-exclusive, non-transferable license to use this product on one Central Processing Unit (CPU) only. The following paragraphs contain the agreement which you ("I" in the text) agree to be bound, by opening the sealed diskette packaging.

I understand that the Protection Plus logo, product names, software, manuals, documentation and other materials are either patented, copyrighted, trademarked or owned by ProPlus Software as trade secrets or proprietary information. I agree not to remove any product identification or notices of such proprietary restrictions from ProPlus Software's products.

I agree that I will only make copies for use as archival backup copies, and only if such copies are for my personal use. Any copying of materials or documentation is prohibited.

Government license

If you are acquiring the software on behalf of any unit or agency of the United States government, the following provisions apply:

The government acknowledges ProPlus Software's representation that the software and its documentation were developed at private expense, and no part of them is in the public domain.

The government acknowledges ProPlus Software's representation that the software is "restricted computer software" as that term is defined in clause 52.227-19 of the Federal Acquisition Regulations (FAR) and is "commercial computer software" as that term is defined in sub-part 227.471 of the Department of Defense Federal Acquisition Regulation Supplement (DFARS). The government agrees that:

(I) If the software is supplied to the Department of Defense (DOD), the software is classified as "commercial computer software" and government is acquiring only "restricted rights" in the software and its documentation as the term is defined in clause 252.227-7013 (C)(1) of the DFARS, and

(II) If the software is supplied to any unit or agency of the United States government other than DOD, the government rights in the software and its documentation will be as defined in clause 52.227-19 (C)(2) of the FAR.

Restricted rights legend

Use, duplication, or disclosure by the government is subject to restrictions as set forth in sub-paragraphs (C)(1)(ii) of the rights in technical data and computer software clause at DFARS 252.227-7013. ProPlus Software, 4260 Americana Drive Suite 126, Stow, OH 44224-4860.

ProPlus Software warrants that the magnetic media on which the enclosed computer program is recorded is free from defects in materials and workmanship under normal use. ProPlus Software warrants that the computer program will perform substantially in accordance with the user manual. The warranty covering the magnetic media and computer program is made only for ninety (90) days from the date of delivery of the program to me or my company as a user.

I agree to return this defective item shipping prepaid during the warranty period, and ProPlus Software must receive it within thirty (30) days of the end of this warranty period.

I agree to either insure the defective item being returned or assume the risk of loss or damage in transit. All warranty claims must be addressed to:

ProPlus Software
4260 Americana Drive Suite 126
Stow, Ohio 44224-4860

Any claim under the above warranty must include a dated proof of the date of delivery, such as a copy of my receipt or invoice. I agree to return the defective item shipping prepaid to ProPlus Software during the warranty period. I agree that ProPlus Software's liability for damages to the user or others resulting from use of the computer program shall not exceed the amount of the license fee payable to ProPlus Software under this Agreement.

I AGREE THAT PROPLUS SOFTWARE SHALL NOT IN ANY CASE BE LIABLE FOR SPECIAL, INCIDENTAL, CONSEQUENTIAL, INDIRECT OR OTHER SIMILAR DAMAGES ARISING FROM ANY BREACH OF THESE WARRANTIES, EVEN IF PROPLUS SOFTWARE OR ITS AGENT HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This means that ProPlus Software is not responsible for any costs, including, but not limited to, those incurred as a result of lost profits or revenue, loss of use of the computer program, loss of data, costs of re-creating lost data, the cost of any substitute program, claims by any party other than me, or for other similar costs.

I understand that some states do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to me.

I agree that any breach of one or more of the provisions of this agreement by me shall constitute an immediate termination of this agreement.

Nevertheless, I agree that in the event of such termination, all provisions of this agreement which protect the rights of ProPlus Software will remain in force.

This agreement shall be governed by the laws of the State of Ohio.

1.7 Getting Help

We thank you for purchasing Protection Plus! This manual contains a description of the features and capabilities of the system. However, if you have any questions left unanswered, or if you discover problems with any part of the program or this documentation, please feel free to call our Technical Support at (216) 923-1768, 10am-5pm, EST Monday through Friday.

You may leave a message on CompuServe in the Nantucket Forum, Databased Advisor Forum or via electronic mail to the address of 71441,3045. Messages left there will be answered within 48 hours.

1.8 Trademark Acknowledgements

Protection Plus	ProPlus Software (M.J. Consulting)
Clipper	Nantucket Corporation
MS Link	Microsoft Corporation
PLINK86plus	Phoenix Technologies Ltd.
Turbo Link	Borland International
Blinker	Blink, Inc.
Norton Guides	Peter Norton Computing
CompuServe	CompuServe Information Service

Zachary	Zachary Software, Inc.

Other brand and product names used in this documentation may be registered trademarks of their respective owners. Their use in this manual are for identification purposes only.

1.9 Definitions

Control file - created by PROPLUS.EXE and used by certain library functions to limit the execution of your applications. Both you and your client's address information are stored in this file. The file is created either by PROPLUS.EXE during "F2-Protect diskette" on a client record or by your own client tracking program via the client tracking functions. A majority of the Protection Plus library functions use the data kept in this file to determine appropriate actions. The control file must be named in your application by a public variable called "PP_CFILE." This public variable is used by most of the library functions. If this variable is not defined, the library functions will not work. More information on this topic can be found in the "Library Reference" section.

Copy protection - a method of protecting applications that prohibit use of them on unauthorized computers. When the software is sent to a client, **it must be unlocked over the phone**, using pp_copywrite(). This function places a picture of the user's computer on their hard drive. The pp_copychk() function is then used at the beginning of your source to compare the current computer to the image of the authorized computer. REMEMBER that ***everyone who buys your software must call you to be unlocked.*** This is how the copy protection features work in Protection Plus. There is no way to use the copy protection functionality without having each client call you to be unlocked. This method is, in turn, foolproof!

There are several sections in this manual that explain the proper use of copy protection. There is a disclaimer that needs to be presented, however! *You need to be aware of what you are getting into when you use copy protection.* That is, after the user is unlocked, technical support must still be available to handle the occasions when their hard drive crashes, etc. If you go out of business or are out of town while your client's software quits working, no one else will be able to unlock them! Some courtesy needs to be given to people whom you have doomed with copy protection.

There are three types of copy protection available in Protection Plus. It is through the copy protection functions that Local Area Network restrictions are enforced. There are two types of network functions and are both described in this definitions section. The different types of copy protection are used by the same set of functions. A numeric parameter is passed to the function to determine which type is being used (default=1). The three types of copy protection are as follows.

1) *Single external* - (like previous versions)
Use this for most cases. A snapshot of the computer is taken during protect time (execution of pp_copywrite()) and the image is placed in a file named by the first eight characters (root) of the control filename with the extension of ".DRV" appended. When updates are given to your client, there will not be a need to re-unlock them. Once the image file is on their hard drive, you only need to re-unlock them if the file gets corrupted. This type of protection cannot be used in LAN environments. It is intended for a single-user application.

2) *Single internal* - (embedded in EXE file)
The snapshot image of the computer is stored in the program EXE file itself. There are no external files needed. This is great for applications that are not updated frequently or require an unlock with every update, or when a control file is not convenient. Even though a control file is not used, the control file public variable is necessary. Normally, the control file can be named anything you wish (except .DRV). *If you plan on using the single internal form of copy protection, the first 8 characters of the control file MUST be the same as the .EXE program file.* Keep in mind that if you implement this type of copy protection, every update to the software will have to be re-unlocked on the client's computer. This type of protection cannot be used in LAN environments. It is intended for a single-user application.

3) *Multiple external* - (network)
The snapshot image of the computers are stored in an external file by the first eight characters (root) of the control filename with the extension of ".DRV" appended. This is used when copy protection network functions are used (specific workstations unlocked). You distribute an application with a control file that has the "LAN users" field set to the number of workstations that are permitted to use the software, "X". When the software is installed, the user must call you to unlock it. The product then becomes "armed" and ready. The first "X" workstations to run the application become the registered workstations. If any other workstations beside the "X" ones attempt to use the software, pp_copychk() will return a .F.. Refer to the function descriptions in "Library Reference" for more information.

<u>Computer number</u> - a number generated by taking a snapshot of the computer. It is exactly the checksum of the ROM BIOS chips located in upper memory. This number is unique for every different computer made. The only repeat of this number is on two *identical* computers.

This number can be used to catch a clients' attempts to tricking you into installing the software on multiple computers. Refer to "maximum security" for more information. There is a space in the client record for this number. It is used as part of the basis of the copy protection scheme.

Distributor - a company or person that distributes products for you. Using the distributor feature, you may add bulk number of clients (that have the same address information) to the PROPLUS.EXE databases. At a glance, you can tell which serial numbers are reserved to distributor copies.

Expiration types - When you plan to distribute some form of limited application, there are two possible routes. If you intend to use copy protection, consider the copy protection test for demo testing. That is, in your application, check for copy protection. If the check fails (using library functions), set a flag signifying a demo mode. Otherwise, have the program function normally. You can use the demo mode flag to limit the number of records allowed in a particular database or prohibit the use of a menu function. This allows your client to copy the software freely and distribute. When the copy protection test fails, there is an instant demo. Once someone else uses the software in a demo mode, they might send you a check to purchase their own copy.

The other route is to use one of the expiration types listed below. Keep in mind that if you only use a Date expiration, there is nothing to stop one of your clients from copying an unlocked piece of software to multiple computers. When these two routes (demo expiration and copy protection) are put together, you can achieve the same protection scheme as PROPLUS.EXE. That program gives a new user 14 days to use the system. During that time, only 1 product can be entered. When the 14 days have gone by, access to the data entered is prohibited until the software is purchased.

If a valid copy protection file is found, the demo mode is not enabled, independent of the setting in pp_exptype(). This is a perfect example of how copy protection and demo expiration can be used together. The list below describes each type of execution limiting available in the Protection Plus system.

Exe Count	Will cause the program to stop after "X" number of program executions. The "X" is set in the execution counter limit field. Note that this type of expiration is not limited to program executions.
Payments	Allows you to have the product stop working on a particular day of the month. If the expiration day lies on a weekend, the next business day will be used instead. That way, the user can use the product over the weekend and get re-unlocked during a business day.
Demo	You set the date of expiration. Unlike payments, the date to expire is absolute-independent of the day of the week.
Shareware	Used if you would like to distinguish between a demo and shareware. The mechanics of protection are the exact same as demo.
None	Will add no expiration to the program.

<u>Hard code serial number</u> - When you wish to have the 6 digit serial number placed directly into your .EXE file, use the hard coding functions. This allows you to customize an application without changing the serial number in the code and re-compiling. Hard coding can be accomplished using PROPLUS.EXE during "F2-Protect diskette" on a client record or by your own client tracking program via the client tracking functions.

The DOS program HCSERIAL.EXE may also be used to achieve this result. The DOS utility was designed to be used in a protection batch file (described below).

<u>LAN functions</u> - There are two types of LAN protection concepts - specific workstation protection and non-specific workstation protection.

SPECIFIC - The specific type sets the application up to run on a particular number of workstations. The first "X" number of workstations (you set the "X") that execute the program are the *only* ones allowed to use it. This is actually one of the three forms of copy protection available in the Protection Plus system. Read about "copy protection" defined in this manual section.

NON-SPECIFIC - Only "X" number of workstations can use the product at the same time. That is, it limits the number of simultaneous users. **<u>This does NOT allow for copy protection or for restricting certain workstations from using the product.</u>** This uses a semaphore approach to allow any workstation to use your software (provided the total number of simultaneous users is not exceeded). Be aware that this type of protection has its disadvantages. Not only ***<u>must</u>*** you add the appropriate Protection Plus functions to your ERRORSYS.PRG in case a run-time error occurred on a workstation, but you must handle the possibility of a power outage. You must be available to re-unlock a client's computer in the event the power went out while the workstations were online with your application. The following functions are used for this type of network protection:

pp_lanlim()	- returns the limited # of workstations
pp_lanusers()	- returns the current # of workstations
pp_lanplus()	- adds one user to the system
pp_lanminus()	- subtracts one user from the system
pp_nlanlim()	- sets a new limit to # of workstations

pp_lancheck()	- can we add another user?
pp_lanactive()	- is this application protected for LAN?
pp_lanincr()	- add 1 to limit of # of workstations
pp_landecr()	- subtract 1 from limit of # of workstations
pp_nlanct()	- set a new count value for current # workstations

Maximum security - gives you the ability to watch any clients you have that you do not trust. It involves using an optional third parameter in the unlocking functions (pp_ucode(), pp_ctcodes(), UNLOCK.EXE). This parameter is used if you would like to have the computer number (or any other fixed number) as part of the unlocking sequence. This concept is best illustrated with an example. If you wish to have maximum security, using this parameter would prohibit a user from telling you his hard drive crashed, when he only wanted to unlock your program on another computer. That is, he wants to unlock your program on another computer, so he calls you up while he is sitting at his second computer and tells you he needs a new unlock code for his computer (you think it is his first one). If you asked him what the pp_compno() was on his screen, and he lied and told you the number that was on his FIRST machine (which, according to your records is a legal copy), the unlocking sequence would not work on his second machine. This parameter is optional and defaults to 1 if not used.

Protection batch file - allows you to execute a DOS batch file (or any executable program) during the disk protect process ("F2-Protect diskette"). That is, when you are protecting a distribution diskette, any work you do to prepare the disk for shipment can be done in a batch file. Simply write the batch file in DOS, and enter the name of the file in the product screen. Every time you protect a diskette, the batch file will be executed. The first parameter passed to the batch file is the serial number of the current client and the second is the destination drive.

You could serialize a program, zip up all application files and automatically create distribution disks! You can get VERY creative with this feature. You have the choice of if the batch file is executed before the hard coding and creation of control file or afterwards. Be sure to read about the DOS utilities that allow for serial number hard-coding from the DOS prompt for use in this batch file. When the batch file is complete, PROPLUS.EXE looks at the DOS errorlevel. If the errorlevel is non-zero, an error is reported to the screen and the protection steps are aborted. The errorlevel number is printed on the screen. If the errorlevel is zero, everything continues fine. There is a sample of a protection batch file described in the Sample Source section.

Unlock codes - At the heart of the Protection Plus system is the library function pp_ucode(). This function allows you to hide 18 different routines (unlimited, if nested levels are used) and allow access to them only when you are on the phone with the user. This concept is the basis for all copy protection and demo functions. It is used for almost every function the library. Access is given by 1) having a hidden function key pop up a box; 2) in the box display the code entry number and computer number (computer number optional); 3) request a numeric from the user; 4) provide a do case structure on the result of passing the number entered to pp_ucode. The user calls you and tells you the code entry number (and you may optionally verify the computer number, described elsewhere). You enter the code entry number (and maybe computer number) into UNLOCK.EXE or PROPLUS.EXE to find the appropriate unlock code to give him. When he types in the number, access to the appropriate routine will be granted. *Unlock codes are based on the system date. When the pp_ucode() function is used, both the unlocking computer and the recipient computer MUST have the same system date.*

VCPI compatible - This is important for people using 80386 (and above) machines along with commercial expanded memory managers. VCPI is a standard used in these memory managers to communicate with software applications. It is important to comply to the standards since many of these memory managers perform fancy footwork on the machine hardware and often times hide the necessary information needed by Protection Plus copy protection routines to take a snapshot of the computer. **Protection Plus is 100% compatible with the VCPI standard and should function properly on all commercially available expanded memory managers.**

2. QUICK START

There is no easy way to get started on Protection Plus quickly. This section will highlight the important information in this manual so that you may quickly locate and read it.

If you are unfamiliar with the Protection Plus system, read the overview in the introduction section. Be sure to read the definitions section (page 33) for a brief explanation of terminology used throughout the package. Read "Commonly Asked Questions."

The installation section contains a list of descriptions for each file copied to your hard drive by the installation program. It also tells you where to copy each of the files, if necessary.

After installing the distribution diskettes, read "Initial Installation" of PROPLUS.EXE. ***It is important to follow the directions step by step.*** Most areas of PROPLUS.EXE have context-sensitive help accessible though the F1 key.

Read the short descriptions of the functions in the library contained in the "Library Reference" function quick-reference section. Browse through SAMPLES.TXT to become familiar with what the sample source code has to offer. The sample source code shows where function calls to the Protection Plus library are logically placed most of the time. If you would like to utilize all of the protection features of this system, you should be able to take these example programs, change their cosmetics, and pop them into any of your existing applications.

If you insist on not reading the manual and still have questions or problems, please read the section of the manual in question before calling technical support. Technical support cannot properly provide answers to someone who is not familiar with the general use of the Protection Plus system.

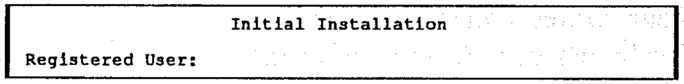
3. PROPLUS.EXE PROGRAM

3.1 Initial Installation

There are quite a few steps that need to be completed to get PROPLUS.EXE set up properly. Make sure the current DOS working directory is where the Protection Plus system was installed. Then type "PROPLUS" [enter]. For example, if the system was installed into C:\PROPLUS, type the following:

```
C:\>CD \PROPLUS         [enter]
C:\PROPLUS>PROPLUS  [enter]
```

When PROPLUS.EXE comes up running the first time, you will be asked to enter your company name. This name will be included in the header of all reports generated by the program.



```
                    Initial Installation
 Registered User:
```

Enter Name to be PERMANENTLY imbedded in all report titles.

After entering your company name, all database files will be created, and then you will then see the power-up banner screen. The power-up screen will contain one of two things. If PROPLUS.EXE has been unlocked, the registered user name and address will appear. If it has not, a DEMO message will appear. The power-up screen will disappear after 20 seconds; you may press any key before the 20 seconds to access the main menu faster.

When you install your purchased version of Protection Plus, the DEMO screen **will** appear. *Do not worry - this is normal.* PROPLUS.EXE is protected with the Protection Plus system.

Since the client tracking program is copy protected, there is one more step to follow. The copy protection file must be placed on your hard drive. If this is not done, the program will run in a limited mode allowing only 1 product and 10 clients. Before this limit is exceeded, you must call Technical Support. The phone number and hours are located in "Getting Help" on page 32. Someone will walk you through the online registration procedure, which will take no more than one minute.

Now that PROPLUS.EXE is running, here is a list of items that need to be completed before you are ready to protect diskettes. For specific instructions on how to accomplish each task, refer to the individual section in the manual for the items listed below:

- Choose the option for "Disk picklist" and add the disk sizes available on your system and their DOS drive letter.
- In the "Master settings," be sure to add your address information as well as your desired protection defaults.
- Add any distributors you are frequently in contact with in the "Distributor" database.
- Add the names and information for each product you wish to track with Protection Plus.
- Add the clients currently using each product.
- Don't forget to read the section "Definitions" on page 33.

3.2 Main Menu

The first time that you run PROPLUS.EXE, you will notice that it is in the DEMO mode. You will have to call Technical Support. You will be given a code that will unlock your PROPLUS.EXE. Refer to the "Initial Installation" section for PROPLUS.EXE for more information.

The main menu has several selections to choose from. Use the arrow keys to position the menu bar over the desired option and press [enter]. The rest of the manual dedicated to PROPLUS.EXE has a separate section for each menu option and any sub-options. Be sure to read over each section to become familiar with the proper operation of the client tracking program. Screen snapshots are provided for your convenience.

3.3 Product Database

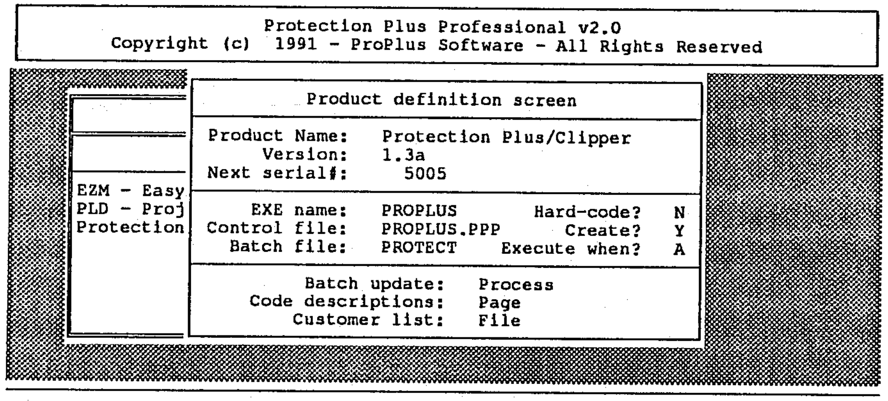
The product database option, available from the main menu, provides access to all currently active products being distributed. After selecting this menu function, a browse window appears displaying all products entered. If none have been entered, the insert mode is automatically invoked; Otherwise the following keys are active:

Enter	- edits current record
Insert	- adds a new record
Delete	- deletes the current record
F1	- help
Arrows	- moves record pointer and field display

During insert or if any changes are made during edit, you will be presented with a prompt that will ask to Accept, Retry, or Cancel. If all changes are correct, press [A]. If you wish to edit a field, press [R]. Should you decide that you do not want the changes to be saved, press [C].

If multiple product names have been defined, you may quickly locate one by typing in a few characters of the name and hitting return. Once a record has been selected for editing, many options are available. Access is provided to product information, batch updating, and customer lists. This option is the most widely used to manage the products and their users.

The figure shown next is an example of the product screen.



```
                Protection Plus Professional v2.0
      Copyright (c)  1991 - ProPlus Software - All Rights Reserved

                         Product definition screen

           Product Name:   Protection Plus/Clipper
                Version:   1.3a
           Next serial#:     5005

EZM - Easy     EXE name:   PROPLUS       Hard-code?   N
PLD - Proj Control file:   PROPLUS.PPP      Create?   Y
Protection   Batch file:   PROTECT     Execute when?  A

                   Batch update:    Process
              Code descriptions:    Page
                  Customer list:    File
```

<u>Product name</u>

Enter the name of the product to be added. Note that the only key field that separates your clients from product to product is this field. This means that if you have two versions of the same product and you desire to have two separate client lists for both versions, you will need to place some sort of signifier to make the product names for the two versions unique.

<u>Version</u>

Enter the version string for the product here. Note that the only key field that separates your clients from product to product is the product name field. This means that if you have two versions of the same product and you desire to have two separate client lists for both versions, you will need to place some sort of signifier to make the product names for the two versions unique. This field is used to show which version a particular customer is on. It might be used also for batch updating in the future.

Serial number

If you are adding a product, this is the beginning serial number to which clients' disks will be labeled. If you are editing this field, change this number to the next desired serial number. That is, when a client is added, this number will be the next serial number assigned. When a client is successfully added, this number is automatically incremented by one.

EXE name

This is the name of the product's .EXE file. This file is basically only used for hard coding serial numbers into the distribution diskettes. The next option allows control over whether or not the serial number is hard-coded into this file. If hard-coding is not used, this field is only for looks!

Hard code serial number?

If you wish to hard code serial numbers into the .EXE file when you protect a diskette from the "F2-Protect" in the customer database, answer Yes to this question. If you answer No, the hard-code serial number routine will be skipped, but the serial number of the product will STILL be located in the control file.

Control file name

This is the control file name. This file will be created when a distribution diskette is protected. Note that you can now designate the exact name to be created. Note that this has to be the same name as the public PP_CFILE variable in your application. Do not forget to include the right extension.

Make control file?

This option allows you to disable the creation of control files. This question is asked since there are several ways to use the Protection Plus Professional package, and you may not want to create a control file. There is only one basic reason to answer No to this question. If you are using ProPlus to hard code serial numbers and track clients ONLY and do not use the control file features in your application, it is not necessary to create a control file on the destination diskette upon the "F2-Protect" command. Otherwise, "Just say Yes!" Note that there is not a utility to create a control file as there is to hard-code serial numbers into the .EXE file.

Batch file name

The DOS program name you enter here will be executed first when the F2-Protect diskette is pressed from a customer screen. This will allow you to protect an executable, compress it, create a control file, and totally automate the process of making distribution diskettes.

There is a little DOS utility for hard-coding serial numbers which can be used to hard-code the number, and then you can use your own compress utility or whatever you desire to do! This makes distribution diskette creation easy! The first parameter given to the batch file is the serial number, and the second is the destination DOS drive.

Run batch file Before/After?

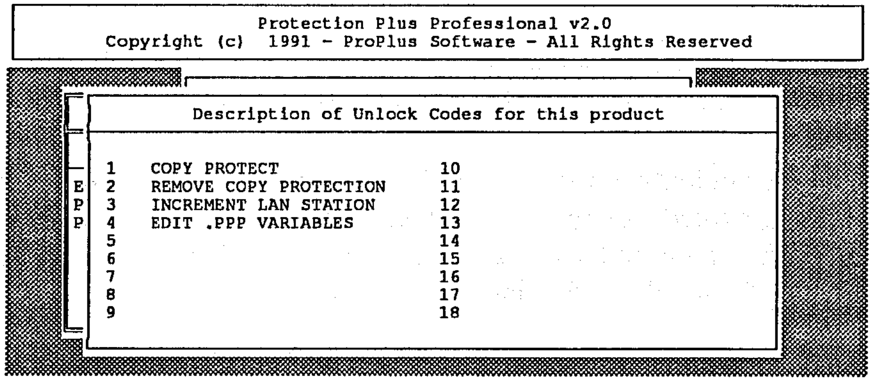
The Batch time field in the product database designates when the batch file listed is executed. This field is only accessible if the batch filename is not blank. There are two choices for this field:

B - before - execute the batch file before creating the control file and/or hard-coding the serial number in the .EXE file.

A - after - execute the batch file after creating the control file and/or hard-coding the serial number in the .EXE file.

3.3.1 Unlock codes description

There are 18 hidden user codes available for one to put into the target application. This screen is for entering the functional description of each code for future reference. Every product can have a different unlock function list. The following figure shows an example of the unlock codes description screen.



```
                    Protection Plus Professional v2.0
     Copyright (c)  1991 - ProPlus Software - All Rights Reserved

             Description of Unlock Codes for this product

   1   COPY PROTECT               10
 E 2   REMOVE COPY PROTECTION     11
 P 3   INCREMENT LAN STATION      12
 P 4   EDIT .PPP VARIABLES        13
   5                              14
   6                              15
   7                              16
   8                              17
   9                              18
```

<u>Description</u>

Simply move the editing arrow to the function number to be changed and press return to edit the field. When complete, press return. The descriptions entered here are for your reference only and have no functional significance.

Keep in mind that only fields that have a non-blank description will give an unlock code in the client database unlock code screen.

3.3.2 Customer database

The browse window that appears after selecting the Customer "File" field under a product record contains a list of the current users of the product. The default sorting order is by serial number. The sorting order may be changed by the F10 key. This key displays a list of available sorting keys.

If no customers are entered for a particular product, the insert mode is automatically entered. Otherwise, the following keys are active:

Enter	- edits current record
Insert	- adds a new record
Delete	- deletes the current record
F1	- help
F10	- changes active index
Arrows	- moves record pointer and field display

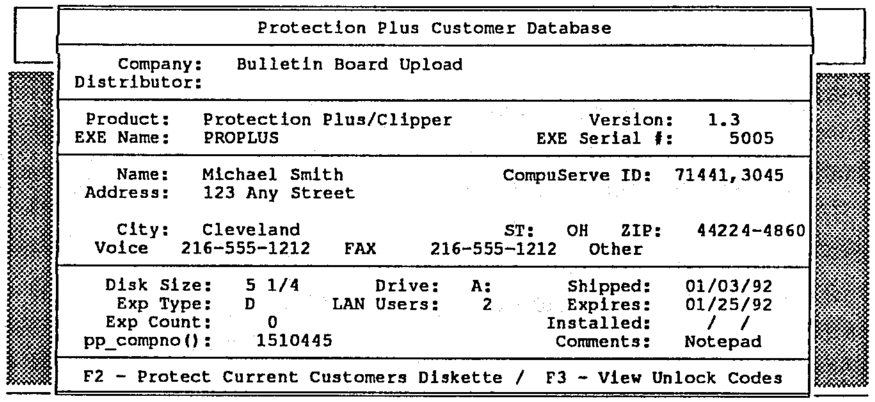
You may quickly locate a record based on the current key (default is serial number) by typing in the text to search for. A prompt for this information is located toward the bottom of the screen. Use F10 to change the active index.

From the editing screen:

F2	- protect diskette (see page 75 for instructions)
F3	- view unlock codes (see page 77 for instructions)

During insert or if any changes are made during edit, you will be presented with a prompt that will ask to Accept, Retry, or Cancel. If all changes are correct, press [A]. If you wish to edit a field, press [R]. Should you decide that you do not want the changes to be saved, press [C].

The next figure shows an example of the customer database screen.



```
                 Protection Plus Customer Database

     Company:   Bulletin Board Upload
 Distributor:

  Product:   Protection Plus/Clipper           Version:   1.3
 EXE Name:   PROPLUS                       EXE Serial #:    5005

     Name:   Michael Smith               CompuServe ID:  71441,3045
  Address:   123 Any Street

     City:   Cleveland                   ST:   OH   ZIP:   44224-4860
   Voice   216-555-1212   FAX     216-555-1212   Other

   Disk Size:   5 1/4       Drive:   A:        Shipped:   01/03/92
    Exp Type:   D       LAN Users:   2         Expires:   01/25/92
   Exp Count:    0                           Installed:    /  /
 pp_compno():    1510445                      Comments:   Notepad

  F2 - Protect Current Customers Diskette /  F3 - View Unlock Codes
```

Company name

Enter the client's company name in this field. Keep in mind that this is a selectable key field and is the prompt for batch updating (along with serial number). That is, this field should be unique and should be able to clue you in quickly as to what customer this is.

When adding records with the distributor flag enabled, this field is filled in automatically with "Distributor" so that when looking at the client browse screen, you can tell at a glance which serial numbers are being held by a distributor. Just hit return on this field when the word "Distributor" appears.

You will be alerted if you are entering a company name that already exists. You will still be able to add the record even though a duplicate name is present.

In order to cancel an add, you must hit PGUP to move the cursor to this field and then hit Escape. You can ONLY abort an add when the cursor is on this field.

Distributor name

Enter the distributor that will be holding this serial number. For a list of active distributors, hit return while this field is blank. Using the distributor option allows you to copy name and address information quickly. Once a distributor is selected, the address information is copied automatically into this client's record. If you do not want the distributor option enabled, you must set the distributor flag to No in the master settings screen.

Product

Enter the name of the product this client is purchasing. When adding, by default it has the current product selected. Just hit return so that PROPLUS.EXE can fill in the correct next serial number, version number, and .EXE name. When editing this record, you may move this client to a different product by changing this field. The product to be moved to must exist already. You may not add a new product from this screen. The serial number name will NOT be updated. The version number and .EXE name will be changed. You will need to change the serial number manually.

Version

Enter the current version of the product this client is using. This is great for reports that display which clients need an update to your program. When a client receives a newer version of the product, but you do not need to protect the diskette, be sure to manually update this field so that your records are up to date. This field will be used in the future to generate mass mailing letters to inform users of software updates and a few other things.

EXE name

Enter the .EXE name of the product this user is purchasing. The name is automatically entered during a product selection and is not accessible during an insert. You may change it manually during edit time. Under normal circumstances, this field is never changed.

Serial number

Enter the serial number being assigned to this client. This field is automatically set during a client insert. Once the client has been added, this field is never changed automatically again. It can only be changed manually. When the client is added, the next serial number in the product database is automatically updated.

Client information

There are many fields that contain general data for this client. They are described as follows:

Name	- User name
CompuServe ID	- obvious, if applicable
Address 1	- mailing address of client
Address 2	- "
City	- "
State	- "
ZIP	- "
Phone descrips.	- three telephone descriptions are allowed
Phone numbers	- three telephone numbers are allowed
Comments	- memo field for general information

These fields help you keep accurate records of how to reach your clients. Some of the address fields are put into the control file during protect disk time so that you may display registered user information, as PROPLUS.EXE does.

Disk Size

Enter the diskette size this client requires. The diskette information for your computer system is entered from the "Disk picklist" option off the main menu. You may also add disk sizes from this screen.

If you define your picklist as 5 1/4 on A: and 3 1/2 on B:, you may just enter a 5 or a 3 and hit return. PROPLUS.EXE will be smart enough to fill in the rest of the information.

When the disk size is entered, the DOS drive to protect will be filled in automatically. Note that if you change your hardware and reverse the disk drive's logical DOS drive letter, you will have to manually update each client's disk size so that the drive letter will be changed.

Shipped date

Enter the date the product was shipped into this field. During the add client mode, this field is set to the current system date. This field helps track when each product was shipped.

Expiration type

Enter the expiration type of this product. The following types are allowed:

Exe Count	Will cause the program to stop after "X" number of program executions. The "X" is set in the execution counter limit field. Note that this type of expiration is not limited to program executions.

Payments	Allows you to have the product stop working on a particular day of the month. If the expiration day lies on a weekend, the next business day will be used instead. That way, the user can use the product over the weekend and get re-unlocked during a business day.
Demo	You set the date of expiration. Unlike payments, the date to expire is absolute-independent of the day of the week.
Shareware	Used if you would like to distinguish between a demo and shareware. The mechanics of protection are the exact same as demo.
None	Will add no expiration to the program.

The purpose of the different types is to tell PROPLUS.EXE how to fill in the expiration date or execution count limit fields for you. When N, P, D, or S is chosen, the expiration date is updated according to how you set the master settings. When E is chosen, the execution count limit is set to the default from the master settings screen.

LAN nodes

If this client will be using the application on a LAN network, set this number to the maximum number of allowed workstations. There are two types of network restriction. This field is used for both of those types. These types are summarized below:

SPECIFIC - The first "X" number of workstations accessed once unlocked are the only allowed workstations to use the product.

NON-SPECIFIC - Only "X" number of workstations can use the product at the same time. This does NOT allow for copy protection or for restricting certain workstations from using the product.

Refer to the definitions section of the manual (page 33) for more information.

Expiration date

Enter the expiration date for this product. The expiration type and date will be embedded into the control file for manipulation by the library functions. You must implement the library functions in your application to enforce the date expiration. This field has the default data copied into it based on the type of expiration chosen. If the "E" type expiration is used, this field is skipped.

Execution count

Enter the execution counter limit for this product. The counter limit value will be embedded into the control file for manipulation by the library functions. You must implement the library functions in your application to enforce the date expiration. This field has the default data copied into it based on the type of expiration chosen. If any of the other types of expiration are used ("PSD"), this field is skipped.

Installed date

Enter the date into this field when the product was unlocked or installed. This field helps track when each product was installed or unlocked.

Computer number

If copy protection is being used, when the client calls to be unlocked, enter the computer number from his unlock screen (if you choose to display it), into this field. It will allow you to catch a client from copying the product to another computer and asking you to re-unlock him.

This field is also displayed on the unlock codes screen. It is used there to help verify the user's computer when "maximum security" is used.

Comments

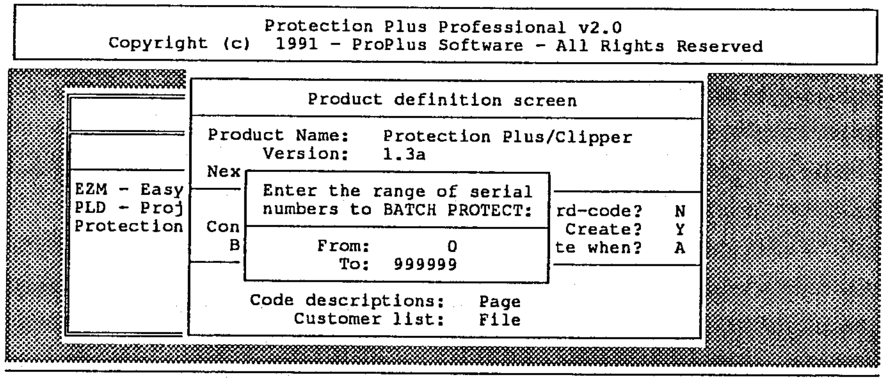
Enter any message having to do with this customer.

3.3.3 Batch protection

Batch updating allows you to protect a mass amount of diskettes automatically. This is handy when a new version of your product is released and a group of serial numbers upgraded, or a bulk number of people purchased your program and you want to create distribution diskettes for all of them.

Basically, you are prompted to enter a disk for a particular client and hit any key. The protect diskette routine is then executed the same as if you hit F2 from the client screen. The version number and the ship date will be put back into the source file to update your records.

You are permitted to enter a range of serial numbers to be updated during this procedure. The next figure shows an example of the batch updating screen.

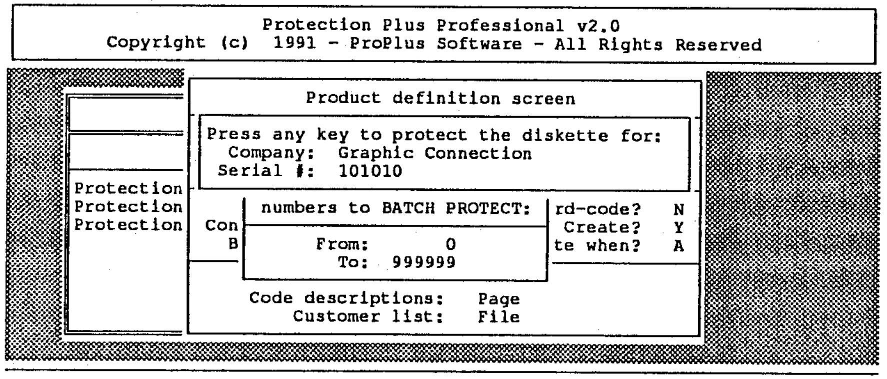


```
                 Protection Plus Professional v2.0
   Copyright (c)  1991 - ProPlus Software - All Rights Reserved

                 Product definition screen
     Product Name:   Protection Plus/Clipper
         Version:    1.3a
     Nex
EZM - Easy   Enter the range of serial
PLD - Proj   numbers to BATCH PROTECT:   rd-code?   N
Protection   Con                          Create?   Y
          B      From:        0           te when?  A
                   To:  999999
         Code descriptions:   Page
             Customer list:   File
```

Serial number range

You are permitted to enter a range of serial numbers to be updated during this procedure. Enter the lower limit into the "From" field and the upper limit in the "To" field. Hit ESC or choose Cancel to abort this process.

When you accept a range to protect, a screen showing a company name and serial number similar to the following will appear. For each company that falls in the serial number range you specified, you are prompted to enter a diskette to be protected. This makes a batch update simple.



```
                 Protection Plus Professional v2.0
   Copyright (c)  1991 - ProPlus Software - All Rights Reserved

                 Product definition screen
          Press any key to protect the diskette for:
            Company:  Graphic Connection
           Serial #:  101010
Protection
Protection      numbers to BATCH PROTECT:  rd-code?   N
Protection Con                              Create?   Y
         B      From:        0             te when?   A
                  To:  999999
         Code descriptions:   Page
             Customer list:   File
```

3.4 Distributors

If you are preparing to protect a bulk number of copies of your package to send to a distributor, Protection Plus has a feature built-in to automate the process.

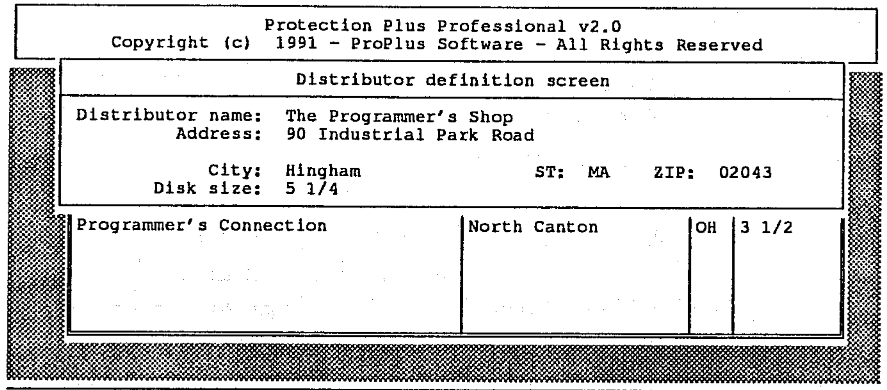
A distributor database contains a list of all valid distributors, their addresses, and disk size information. The master settings has a toggle flag called "Distributor." When this is set to Yes, during the add of a client, the distributor field will be enabled. You will be given the opportunity to choose from or type in a distributor name. The information from the distributor screen will automatically be copied into the current client record. You will easily be able to identify all serial numbers that have been sent to a distributor in the browse display of clients. The company name field will be changed to "Distributor." Once an end-user purchases the product from the distributor and either calls or sends in a registration card, the record can immediately be updated with the correct information. There will always be a history of which distributor sold the product to which clients.

Usually this option is most often used in the following way:

- Add all of the possible distributors to the distributor database.
- Turn the master settings distributor flag on.
- Add and protect the bulk client records.
- Turn the master settings distributor flag off.

During insert or if any changes are made during edit, you will be presented with a prompt that will ask to Accept, Retry, or Cancel. If all changes are correct, press [A]. If you wish to edit a field, press [R]. Should you decide that you do not want the changes to be saved, press [C].

The next figure shows an example of the distributor screen.



```
                  Protection Plus Professional v2.0
       Copyright (c)  1991 - ProPlus Software - All Rights Reserved

                     Distributor definition screen

   Distributor name:  The Programmer's Shop
            Address:  90 Industrial Park Road

               City:  Hingham               ST:  MA    ZIP:  02043
          Disk size:  5 1/4

  Programmer's Connection            North Canton        OH   3 1/2
```

Distributor name

This is the company name of the distributor. Every distributor record in the database must have a unique name.

Distributor information

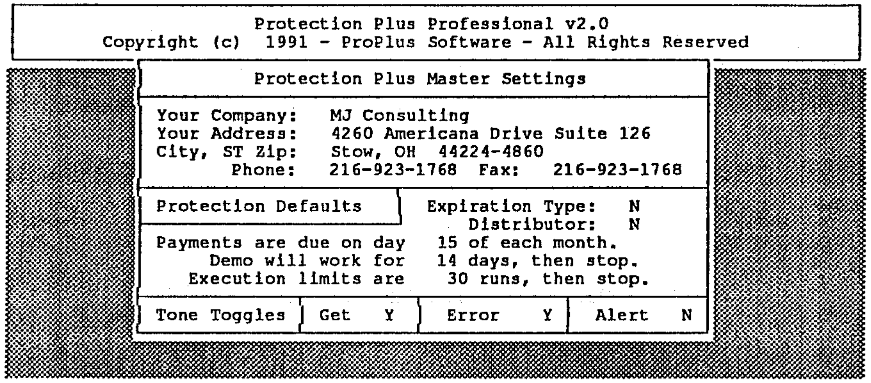
The following information is kept for every distributor:

- two lines for address
- city
- state
- ZIP
- disk size

Enter the appropriate information for this distributor in the spaces provided. This information will be copied into a client screen when a new client is added and the distributor flag in the master settings is enabled.

3.5 Master Settings

This menu option gives you access to the system parameters file. This file keeps track of your address, default protection parameters, as well as the various tone toggles. The next figure shows an example of the master settings screen.



```
                    Protection Plus Professional v2.0
       Copyright (c)  1991 - ProPlus Software - All Rights Reserved

                    Protection Plus Master Settings

        Your Company:   MJ Consulting
        Your Address:   4260 Americana Drive Suite 126
        City, ST Zip:   Stow, OH  44224-4860
               Phone:   216-923-1768  Fax:   216-923-1768

        Protection Defaults       Expiration Type:   N
                                      Distributor:   N
        Payments are due on day   15 of each month.
             Demo will work for   14 days, then stop.
           Execution limits are    30 runs, then stop.

        Tone Toggles | Get    Y | Error    Y | Alert    N
```

Company name

Enter your company name here. The data in this field will be placed in the control file during "Protect Diskette." It will then be available for display in your target application.

Address and phone information

The following information is kept for your company:

- one line for the street/PO box address
- second line for city, state, ZIP
- phone number
- FAX telephone number

This information is stored in the control file during "Protect Diskette." It is also available to be displayed in your target application.

Default expiration type

Enter the default expiration type. The expiration type chosen here will be the one that is selected when you add a new client. This field makes adding a bulk set of clients with the same expiration type easier since the expiration type field will be able to be skipped.

The following types of expiration are allowed:

Exe Count	Will cause the program to stop after "X" number of program executions. The "X" is set in the execution counter limit field. Note that this type of expiration is not limited to program executions.
Payments	Allows you to have the product stop working on a particular day of the month. If the expiration day lies on a weekend, the next business day will be used instead. That way, the user can use the product over the weekend and get re-unlocked during a business day.
Demo	You set the date of expiration. Unlike payments, the date to expire is absolute-independent of the day of the week.
Shareware	Used if you would like to distinguish between a demo and shareware. The mechanics of protection are the exact same as demo.
None	Will add no expiration to the program.

<u>Distributor flag on?</u>

If you are preparing to protect a bulk number of copies of your package to send to a distributor, set this option to "Yes." Otherwise, set it to "No." The distributor flag is used when adding new clients to a particular product. If the flag is enabled, during the add of a client, a popup list will appear of valid distributors. You may choose one and press enter. All of the information from that distributor will automatically be copied into the current client record.

You will easily be able to identify all serial numbers that have been sent to a distributor in the browse display of clients. The company name field will be changed to "Distributor." Once an end-user purchases the product from the distributor and either calls or sends in a registration card, the record can immediately be updated with the correct information. There will always be a history of which distributor sold the product to which clients.

Usually this option is most often used in the following way:

- Add all of the possible distributors to the distributor database.
- Turn this flag on.
- Add and protect the bulk client records.
- Turn this flag off.

<u>Payment day</u>

This number is used to help calculate the expiration date for a product being protected when you set the protection type to "Payments." This is best illustrated with an example:

Set this number to 20 and assume it is now July 10th. If you set the expiration type of a particular client to "P" for payments, the expiration date will be changed to August 20th - or the next weekday after August 20th. The date is calculated with the pp_npdate() function. You still have the ability to change the date once it is placed in the field.

<u>Demo days</u>

This number is used to help calculate the expiration date for a product being protected when you set the protection type to "Demo." This is best illustrated with an example:

Set this number to 20 and assume it is now July 10th. If you set the expiration type of a particular client to "D" for demo, the expiration date will be changed to July 30th. Basically, this number is added to the current date and that date is placed in the expiration date field. You still have the ability to change the date once it is placed in the field.

If you enter a zero for this number, the expiration date will be left blank so that in your source code, you could test for a Demo type and test for a blank date (first run of the program) and set the expiration date to X number of days from the first date it is executed. This eliminates the shelf life of a demo.

<u>Expiration count</u>

This number is used to help calculate the execution count limit for a product being protected when you set the protection type to "Exe Count." This is best illustrated with an example:

Set this number to 30. If you choose the expiration type to be "Exe Count" then this number is placed in the counter limit field automatically. You still have the ability to change the number once it is placed in the field.

Get beep

If this field is set to TRUE, a tone will sound whenever you fill a field.

Error beep

If this field is set to TRUE, a tone will sound whenever you make a data entry error.

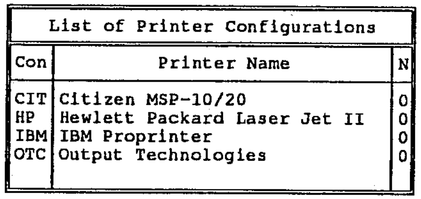
Alert beep

If this field is set to TRUE, a tone will sound at certain points to make you aware of various conditions. These are conditions that are not necessarily wrong, but you should still be woken up!

3.6 Reports Menu

Selecting this option from the main menu provides a sub-menu. One of the sub-menu functions allows for the editing of printer definition strings while the other allows one to build custom reports given database and field information.

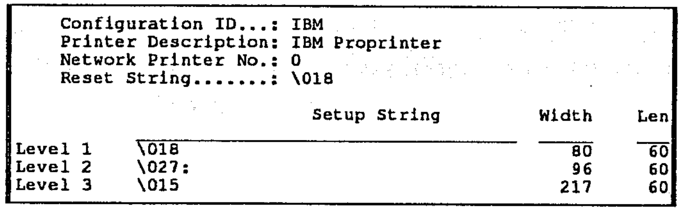
3.6.1 Printer setup



```
 List of Printer Configurations
Con           Printer Name                N
CIT Citizen MSP-10/20                     0
HP  Hewlett Packard Laser Jet II          0
IBM IBM Proprinter                        0
OTC Output Technologies                   0
```

The printer setup utility will allow you to define your own printer configurations in addition to the four (4) standard configurations included with the program.

1. To view/edit a configuration, highlight it with the cursor keys, and press [Return]. A window will appear displaying the information for that printer configuration.



```
     Configuration ID...: IBM
     Printer Description: IBM Proprinter
     Network Printer No.: 0
     Reset String.......: \018

                       Setup String            Width     Len

Level 1    \018                                   80      60
Level 2    \027:                                  96      60
Level 3    \015                                  217      60
```

<u>Configuration ID</u>

The configuration ID is a unique identifier to identify the printer configuration.

<u>Printer description</u>

This field is a brief description of the printer. It should contain information relevant as to what features the configuration has.

Network printer number

The network printer number is the printer that all reports should be routed to. If the system is running a network, the number should correspond to the network assigned printer number, where the first network printer is labeled one (1). If the program is running on a stand alone system, the number should correspond to the first parallel printer port on the PC. The number should then be set to zero (0).

Reset string

The reset string is the escape sequence that will be sent to the printer on completion of a report. This should also be set as to reset the printer to its default power-up condition.

Setup strings, width, and length

Next are three sets of set-up strings, widths, and lengths. The report generator uses these to determine how to lay out the lines of text. The report generator automatically selects the one best suited for the report. If one line of the report is too wide for Width 1, but is less than Width 2, the second set-up string will be used. If it is wider than Width 2, then Width 3 will be used. If the report is still too wide, the report generator will utilize more than one line on the page per record printed, printing only as many fields on each line as will fit without breaking and will wrap to the next line. If a line has to wrap, the report generator will automatically adjust the number of lines per page since fewer records will fit on a page. Refer to your owner's manual for the codes relevant to your printer.

To edit any of the fields, move the arrow and press [return].

To leave, press the [ESC] key.

If any changes were made, you will be presented with a prompt that will ask to Accept, Retry, or Cancel. If all changes are correct, press [A]. If you wish to edit a field, press [R]. Should you decide that you do not want the changes to be saved, press [C].

3.6.2 Printing reports

This menu options lets you define your own reports. Each report is based on a primary database you select from a window. You select the fields you want in the report, specify upper and lower limits of field values to select for the report, and you select a printer configuration. If the configuration you want does not exist, you may add a new one.

You can define as many reports as you want for each file. All reports print in a columnar format with headings. Based on the printer configuration you select, an effort will be made to print one record per line by switching between print sizes and types. If a complete record does not fit on one line, it will wrap to the next line between two fields.

This option allows you to generate reports by file name. Each file name includes one or more field titles. Each field title has different data that may be chosen for a report.

1. Field titles may be given a range by assigning an upper, and lower limit. The user may also request that a title total be reported by typing [Y] when prompted by; "Would you like a total figure on this title (Y/N)?" This is only in effect for numeric fields, obviously.

2. Use the cursor keys to highlight the file name that you want to generate a report on.

3. Press [Return]. The report title window will then appear. Names of previously created reports will be listed. If no reports have been created, the window will be empty, and you will go to the insert mode automatically.

4. To add a report, press the [INS] key. A window will appear. Enter the name of the report that you wish to create, and press [Return].

5. At this time, you will be asked for a printer configuration file to be used for the report. By pressing return, a window will appear with your current printer configurations in it. Use the cursor keys to highlight the configuration file that you wish to use, and press [Return].

6. You will then be returned to the window with your report name in it. Highlight the report name using the cursor keys, then press [Return].

7. Press the [INS] key to bring a window up that will allow you to select the available fields for your report. Highlight your choice, then press [Return]. If you are defining a new report and no fields have been added yet, the insert mode will automatically be entered.

8. You will then be presented with a window that will allow you to choose the fields that you want contained in the report. By highlighting the field that you want then pressing [Return], that field will be selected.

9. Additional fields may be selected by repeating steps 7 & 8.

10. An upper and lower limit may be placed on the fields in your report. Entering a range will limit the listing to data belonging to or falling within the upper and lower limits of the range specified.

11. To enter a range for a field, highlight the field, and press [Return]. A window will appear requesting a upper limit, lower limit, and if you wish to have the field total at the conclusion of the report.

12. To enter additional ranges, repeat step 11.

13. Once you have finished adding titles, and ranges to your report the [ESC] key should be pressed. A window will appear presenting you with all the information that you have selected for your report.

14. A prompt will appear. Press [Y] to save & print the report. Press [N] to save the report to disk.

15. If you press [Y] then you will be presented with a prompt asking for an optional heading for the report. Enter up to two lines of text followed by a [Return].

16. After this point you will be asked to select what sort option you wish to use for your report if the database selected has indexes. Highlight the option that you wish to use, then press [Return].

17. Your report will then begin to print. To abort the printout at any time, just press a key. You will be returned to the reports menu.

3.7 Utilities Menu

3.7.1 Re-index database files

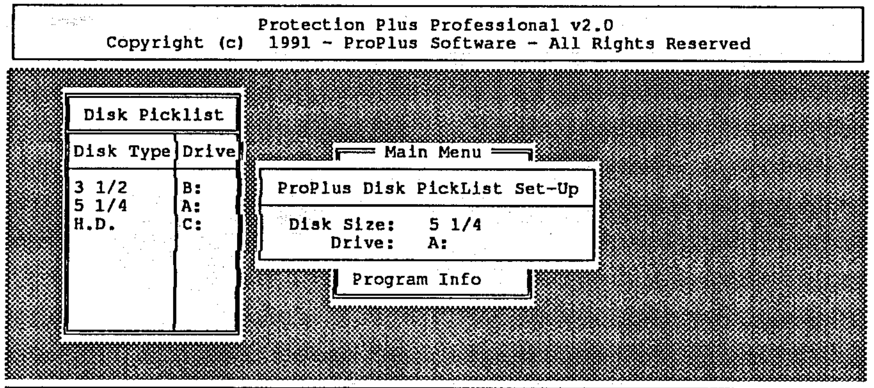
This menu option packs and re-indexes all files in the system.

3.7.2 Adjust ProPlus colors

If you are using a color or composite monitor, this utility will let you change the color settings for the screens. You can also specify whether you want windows to explode and implode.

3.8 Disk Picklist

The disk picklist is used to tell Protection Plus which drives and disk sizes are available on the current computer system. It is important that this information is entered before any other menu option is chosen. The next figure shows an example of the disk picklist screen.



```
                    Protection Plus Professional v2.0
      Copyright (c)  1991 - ProPlus Software - All Rights Reserved

 Disk Picklist
 Disk Type  Drive          Main Menu
 3 1/2      B:        ProPlus Disk PickList Set-Up
 5 1/4      A:          Disk Size:   5 1/4
 H.D.       C:              Drive:   A:
                           Program Info
```

Disk type

Enter the diskette size or type for the current drive to be added to the disk picklist.

Drive

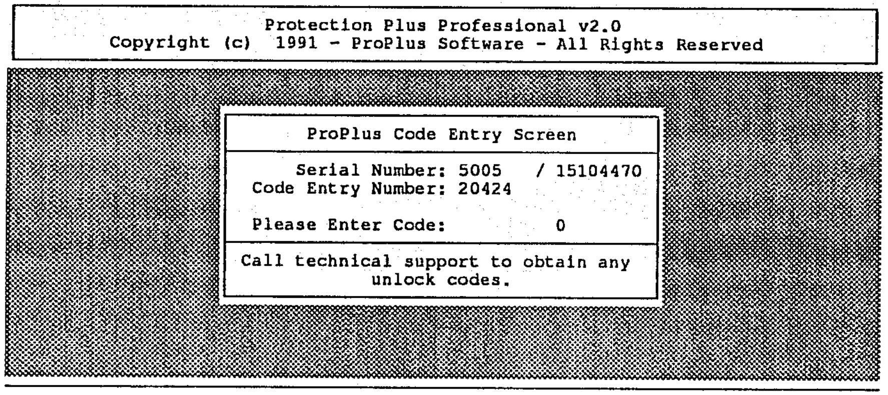
Enter the logical DOS device drive for the current drive to be added to the disk picklist.

3.9 Program Info

By selecting the Program Info option from the main menu, a display of release date, author, and phone number information is available.

3.10 Enter Code - F5 Popup

A sample code entry screen can be seen by hitting F5 from the main menu in PROPLUS.EXE. This code entry screen is the actual one used to unlock people who purchase Protection Plus. A sample of the screen can be seen in the following figure.



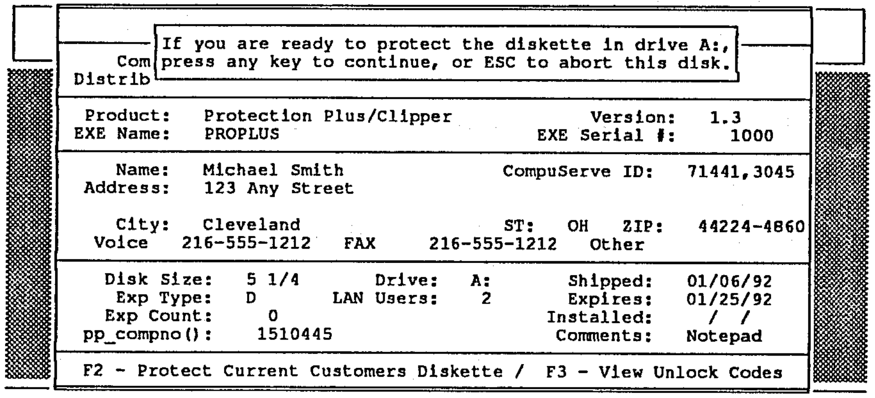
```
                    Protection Plus Professional v2.0
        Copyright (c)  1991 - ProPlus Software - All Rights Reserved

                    ProPlus Code Entry Screen

                     Serial Number: 5005   / 15104470
                 Code Entry Number: 20424

                 Please Enter Code:            0

                 Call technical support to obtain any
                            unlock codes.
```

3.11 F2 - Protect Diskette

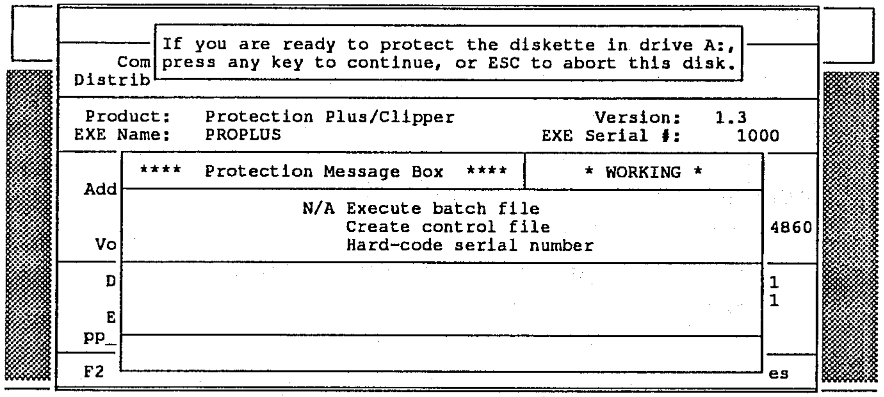
When it comes time to distribute copies of your software, there are certain circumstances upon which you need to protect the diskettes before distribution. You actually need to protect **every** diskette you send to clients. The only exception to this rule is when you are sending an update of the software. Be sure to read the "Commonly Asked Questions" where one of the questions addresses this topic.

If the client is not already in the client file of the appropriate product, add it. Select the client for editing by hitting [return] on their record in the client database. Press the F2 key. The following figure will appear.

If you named an .EXE file in the product database and set the hard code serial number flag on, make sure that the file is on the diskette so that its serial number can be recorded correctly.


```
       If you are ready to protect the diskette in drive A:,
   Com press any key to continue, or ESC to abort this disk.
 Distrib

  Product:   Protection Plus/Clipper          Version:   1.3
 EXE Name:   PROPLUS                       EXE Serial #:    1000

     Name:   Michael Smith               CompuServe ID:   71441,3045
  Address:   123 Any Street

     City:   Cleveland                      ST:   OH   ZIP:   44224-4860
   Voice   216-555-1212   FAX     216-555-1212   Other

   Disk Size:   5 1/4      Drive:   A:       Shipped:   01/06/92
    Exp Type:   D      LAN Users:   2        Expires:   01/25/92
   Exp Count:     0                        Installed:    /  /
 pp_compno():    1510445                    Comments:   Notepad

  F2 - Protect Current Customers Diskette /  F3 - View Unlock Codes
```

The drive letter in the above screen will change according to your disk picklist database and which size diskette this client requires. When you insert a disk into the correct drive and hit any key, the following screen is displayed.



```
        If you are ready to protect the diskette in drive A:,
    Com press any key to continue, or ESC to abort this disk.
Distrib

 Product:   Protection Plus/Clipper           Version:   1.3
EXE Name:   PROPLUS                       EXE Serial #:    1000

       ****  Protection Message Box  ****       * WORKING *
 Add
                         N/A Execute batch file
                             Create control file              4860
  Vo                         Hard-code serial number

   D                                                          1
                                                              1
   E
 PP_

 F2                                                           es
```

The protection box appears with several messages and symbols. In the top of the box, there are three options (run batch file, hard code serial number, and create control file). If any of these options are disabled, an N/A appears to the left. The current procedure has an arrow pointing to it. If an error occurred, the arrow points to what procedure was being executed. As the procedures are completed, a checkmark will appear to the left. Any messages will appear in the box below the three procedures. The lowest box informs you when to press any key to continue.

You can have errors occur from the following:

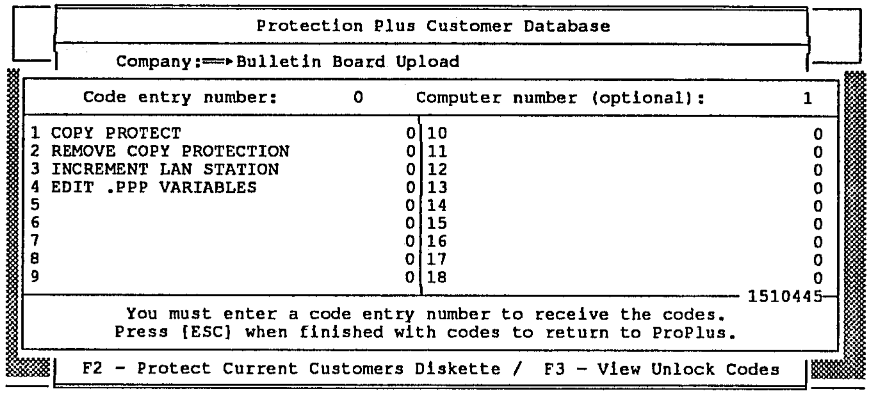
GENERAL
- -Drive not ready, open, disk upside down, write protected, etc.
- -Warning - Control file already exists on the destination disk.

HARD CODING
- -EXE file not found.
- -PP_SERIAL() function not found or EXE file already serialized.

BATCH FILE EXECUTION
- -Procedure in batch file returned a non-zero DOS errorlevel.

3.12 Unlock Codes

The unlock codes provide access to routines that you hide in your applications. Be sure to read about the unlock codes in the "Definitions" section on page 33. The unlock codes are accessed in PROPLUS.EXE via the client file. You must select a client for editing and press the F3 key. A screen similar to the following will appear. You must enter a code entry number and have the descriptions filled in order to receive a code for that number. If you are using maximum security, you must also fill in the computer number field. If you are not using maximum security, leave the default of 1 in the computer number field. Note that the value entered for pp_compno() on the client record will appear just below the 18th unlock code. This is for verification purposes.


```
                 Protection Plus Customer Database
      Company:══►Bulletin Board Upload

   Code entry number:        0     Computer number (optional):        1

 1 COPY PROTECT                   0|10                                  0
 2 REMOVE COPY PROTECTION         0|11                                  0
 3 INCREMENT LAN STATION          0|12                                  0
 4 EDIT .PPP VARIABLES            0|13                                  0
 5                                0|14                                  0
 6                                0|15                                  0
 7                                0|16                                  0
 8                                0|17                                  0
 9                                0|18                                  0
                                   ────────────────────────────── 1510445
          You must enter a code entry number to receive the codes.
         Press [ESC] when finished with codes to return to ProPlus.

   F2 - Protect Current Customers Diskette /  F3 - View Unlock Codes
```

4. LIBRARY REFERENCE

4.1 Compiling and Linking

There are no special instructions for compiling your program when using functions from the Protection Plus library. However, there is one immediate change to your application source code. The public variable PP_CFILE must be defined in your application containing the control filename. You may add any extension to this filename with the exception of ".DRV". See the "Function Reference" section below for a coding example of this. Review the section "Reserved Words" on page 138 to make sure that variable names in your program do not conflict with the public variables used in the library functions.

There are two changes to your link file. Add the appropriate library to your link library list and the appropriate .OBJ file to the ROOT of the program. That is, if you are using Clipper® Summer '87, use PPLUSS87.LIB and PP_S87.OBJ; if you compile with Clipper® 5.x, use PPLUS50.LIB and PP_50.OBJ. **The location of the Protection Plus library on your library link list must fall between CLIPPER.LIB and EXTEND.LIB.** The extend library must be included. The Protection Plus library may be allocated to an overlay; do **not**, however, overlay the .OBJ file.

All function names used in the Protection Plus system are preceded with a "pp_" prefix. This is to easily (1) distinguish a name so that it is associated with Protection Plus, and (2) keep your function names from possibly conflicting.

NOTE: If your product initially goes out as an executable limit demo and gets unlocked, you might want to destroy (overwrite, not just erase) the control file from the floppy upon initialization. This is so that the smart user can't copy it over and have more time with the demo than desired.

NOTE: One thing to keep in mind is that this library was set up so that you have **total control** of what happens when. The suggestions made in the sample source code are only recommendations. You may set your system up however you choose. You can be pretty creative at how the security is structured!

PP_ADVECT()

Increment the execution-limit counter

Syntax

```
PP_ADVECT()  →  lStatus
```

Returns

.T. if successful, else .F.

Description

Increments the execution counter by one. The execution counter is stored in the control file. The execution counter should be incremented once just before the main menu of your application is executed for the first time. That way, if password protection is used and the user types in the wrong password and the program exits, he won't miss out on an execution.

Example

```
The first thing to do is right before the main menu is accessed for
the first time, the execution counter should be incremented if the
expire type is set to execution limit.  EXE_LIMIT is a logical
defined in the beginning of the program that is .T. of the
expiration type is "E" for EXE limit.  We only need to advance the
counter if the EXE counter limiting is used

if exe_limit
    pp_advect()
endif
```

See Also PP_EXECT(), PP_EXELM(), PP_NEXECT(), PP_NEXELM(), PP_NEXPTYPE()

PP_CENUM()

Return the current code entry number

Syntax

```
PP_CENUM()  →  nNumber
```

Returns

The random number.

Description

Generates a random number for the user code entry routine to use later. This is used when either copy protection is used or other hidden routines are needed. On the power-up screen, this number is displayed for the user to see. When special access is needed to a routine, such as pp_copywrite(), the user tells you this number over the phone, so you can find the correct code using either PROPLUS.EXE or UNLOCK.EXE.

The number is random, and leads to the ability to secure hidden procedures in your source code and only allow a user access when they are online with you. Protection Plus gives you eighteen hidden procedures. For example, say you have a password function in your application and your user forgets his password. He can call you, tell you the code entry number on the screen, and you can generate an unlock code from PROPLUS.EXE or UNLOCK.EXE with his code entry number as a parameter and instantly tell him a number to type in response to a special prompt, and the password test is overridden. He can then run the program, go to the password change function, and create a new password for future use. Keep in mind that the code entry number is random, but the actual code to enter at the special prompt changes EVERY DAY according to that number. It would be nearly impossible to break the code, eliminating the possibility of one user calling another user and using unauthorized codes.

There are so many possibilities available for this and associated functions. Use your imagination.

Note that this function does NOT create a public variable as it in previous versions of Protection Plus. It is up to you to make a variable, set it equal to pp_cenum(), display it on the screen and pass it as the second parameter to pp_ucode().

Example

```
PUBLIC code
code = pp_cenum()
@ 5,2 say "Code entry: " + ltrim( str( code ) )
```

See Also PP_UCODE()

PP_CHKPP()

Verify the integrity of the control file

Syntax

```
PP_CHKPP()  →  lStatus
```

Returns

.T. if successful, else .F.

Description

Checks the validity of the control file. It is possible, however, for this function to return a .T. for "file okay" when the data is garbaged. This function call should be placed at the beginning of your application, if even desired at all. It will also test to make sure that the control file is for the current version of Protection Plus Professional.

NOTE: All functions in the Protection Plus library that depend on the control file call pp_chkpp() to verify that control file is sound. If this function returns a false, the function called will return either the "worst case" value or an error value. For example, if pp_expired() is executed and pp_chkpp() failed during the execution, pp_expired() will return .T., to keep the program from running.

Example

```
if !pp_chkpp()
   ?? "Error with application data file"
   ?
   cancel
endif
```

See Also PP_CRPPP(), PP_SAVEPP()

PP_COMPNO()

Return the computer number

Syntax

```
PP_COMPNO()  →  nNumber
```

Returns

A number representing a unique number of the computer.

Description

The pp_compno() function is used optionally to give you a computer "number." This number is actually the checksum of the ROM BIOS. When the user first calls you to be unlocked (if copy protection is used), you enter the number that appears on his screen into the customer record of PROPLUS.EXE. If the user calls and says that his computer hard drive crashed, and you need to copy protect him again, when he calls back to unlock the "new hard drive" and this number is DIFFERENT, he is trying to fool you and install your software on another computer. This adds another level of security when the user "acts" stupid!

Refer to the "Definitions" section on page 33 for a more detailed description of this computer number.

Example

```
In this example, the computer number was appended on to the end of
the serial number.  This is to make the number a little more
indistinguishable, since it appears to be part of the serial
number.

@ 4,2 say "    Serial: "+pp_serial()+"/"+ltrim(str(pp_compno()))
```

See Also PP_SERIAL()

PP_COPYCHK()

Verify the copy protection

Syntax

```
PP_COPYCHK( <nType> )  →  lResult
```

Arguments

<nType> is the copy protection type:

1 = external file, single user (default)
2 = internal data, single user
3 = external file, network/multiple user

Returns

.T. if successful, else .F.

Description

Tests the value of the copy protection stored on disk and compares it to the current computer. The user will not be able to decode this encryption. The external file, single user (default) is exactly like the old-style copy protection of previous versions of Protection Plus. This type is used on non-network environments. The internal file, single user keeps the copy protection inside the .EXE file itself. This can be handy, but it requires an unlock code for every updated .EXE file that you send your users. If you give frequent updates, you should use type 1 so that any future updated .EXE files will not have to be unlocked.

The external file, network/multiple user is used for either network environments or where the entire directory will be shared (copied) between two computers.

For example, if a client frequently takes data from work to home and copies the entire directory, simply sell that client a license for two LAN workstations. When they become unlocked on the first computer, run the program, copy the entire directory to the second computer, run the program, and copy the entire directory back to the first computer, the application is completely limited to ONLY working on those two computers. If this sequence of events is not followed exactly, extra copies of the application may become unlocked. The other approach would be to unlock a copy at home and unlock a copy at work and then just copy the important data files without disturbing the copy protection file. (maybe make the copy protection file hidden or system so that a DOS file copy will not copy the copy protection file).

When a network environment is active, only the FIRST licensed number of workstations will be allowed to use the software. When another non-licensed workstation attempts to execute the application, you can default the program to work in demo mode or not work at all. Refer to the "Program Definitions" in the .NG database or the section called "Definitions" in the user's manual for a detailed description of network layout.

Example

```
If you would like to revert this program to a demo version in the
event that copy protection is invalid or not found, simply create a
public or static variable and code the following.

demo_ver = !pp_copychk( 1 )
```

If you want the program not to work, code the following. Keep in mind that when the user first installs the software, there will not be any copy protection on his hard drive. He will need to call you first to create the appropriate file. So, you will never be able to just "terminate" an application if copy protection is not found. Because of this, refer to the function pp_ucode() on how to integrate this.

```
clear
get_code=0
If !pp_copychk()
    @ 10,10 say "Enter code: " get get_code
    read
    if pp_ucode(get_code)=1
        pp_copywrite( 1 )
    else
        clear
        cancel
    endif
endif
```

See Also PP_COPYWRITE(), PP_KILLPROT(), PP_UCODE()

PP_COPYWRITE()

Create the copy protection file

Syntax

```
PP_COPYWRITE( [<nType>], [<nAttrib>] )
                         → lResult
```

Arguments

<nType> is the copy protection type:

1 = external file, single user (default)
2 = internal data, single user
3 = external file, network/multiple user

<nAttrib> is the attributes of the external file:

0 = normal attributes (default)
1 = read only attribute (read warning below)
2 = hidden attribute
4 = system attribute

Returns

.T. if successful, else .F.

Description

Writes the copy protection file to be used by the pp_copychk() function. There are three types of copy protection available. The user will not be able to decode the encryption used. When you use the external data file (types 1 and 3), you can choose which attributes are set on the created file. DO NOT USE THE READ-ONLY ATTRIBUTE UNLESS YOU HAVE A SPECIAL REASON TO DO SO. This is because pp_killprot() will not be able to erase the file and pp_copywrite() will not be able to overwrite the existing file.

The external file, single user (default) is exactly like the old-style copy protection of previous versions of Protection Plus. This type is used on non-network environments. The internal file, single user keeps the copy protection inside the .EXE file itself. This can be handy, but it requires an unlock code for every updated .EXE file that you send your users. If you give frequent updates, you should use type 1 so that any future updated .EXE files will not have to be unlocked.

The external file, network/multiple user is used for either network environments or where the entire directory will be shared (copied) between two computers. For example, if a client frequently takes data from work to home and copies the entire directory, simply sell that client a license for two LAN workstations. When they become unlocked on the first computer, run the program, copy the entire directory to the second computer, run the program, and copy the entire directory back to the first computer, the application is completely limited to ONLY working on those two computers. If this sequence of events is not followed exactly, extra copies of the application may become unlocked. The other approach would be to unlock a copy at home and unlock a copy at work and then just copy the important data files without disturbing the copy protection files. (maybe make the copy protection file hidden or system so that a DOS file copy will not copy the copy protection file).

When a network environment is active, only the FIRST licensed number of workstations will be allowed to use the software. When another non-licensed workstation attempts to execute the application, you can default the program to work in demo mode or not work at all. Refer to the "Program Definitions" in the .NG database or the section called "Definitions" in the user's manual for a detailed description of network layout.

This function is usually part of a code entry function list. Your application can be distributed as a demo. If the user wishes to give you a credit card number or mail you a check, you can immediately unlock their software by running this function.

IMPORTANT: If you will be using the single internal form of copy protection, you MUST name the control file the same as the .EXE filename. The extension of the control filename is not important, just so the first 8 characters of the control filename match the running .EXE filename! ALSO, if you force a type #3 (multiple external) copy protection, pp_copywrite() will create an empty protection file as should even if the control file has a LAN users of 0. That is, you should do a test using pp_lanactive() before calling pp_copywrite(). The sample program LAN.PRG and SAMPLES.TXT illustrate the proper use.

Example

```
See the example in pp_copychk() and the sample source code.
```

See Also PP_UCODE(), PP_COPYCHK(), PP_KILLPROT()

PP_CRPPP()

Create a new control file with blank data

Syntax

```
PP_CRPPP()  →  lResult
```

Returns

.T. always

Description

If the control file gets erased and the master diskettes were lost, this function can be used to create a dummy control file with blank string data, serial number 1, demo expire type, and date()+15 day expire date. This could be used temporarily until a new control file is received. If your product initially goes out as an execution limit demo and gets unlocked, you might want to destroy (overwrite, not just erase) the control file from the floppy upon initialization. This is so that the smart user can't copy it over and have more time with the demo than desired.

Example

```
This is most often implemented in one of the hidden user-code
positions.  Refer to pp_ucode() for more information on
integrating.

get_code=0
@ 20,10 say "Enter code: " get get_code
read

&& define 5 as the "create blank ppp file" code
if pp_ucode(get_code)=5
    pp_crppp()
endif
```

See Also PP_CHKPP(), PP_SAVEPP(), PP_UCODE()

PP_CTCODES()

Find out the unlock code numbers for display

Syntax

```
PP_CTCODES(<nNumber>, <nCode>, [<nCompno>] )
                       → nResult
```

Arguments

<nNumber> is the code entry number (1 to 18).
<nCode> is the code entry number given by pp_cenum().
[<nCompno>] (optional) is the computer number or fixed number.

Returns

Number corresponding to the unlock code, zero if error.

Description

Returns the corresponding unlock code number based upon the number passed. The number passed must be between 1 and 18. If it is not in this range, a zero is returned. This function is used in a client tracking type application where a list of unlock codes is needed based upon the code entry number on the computer of the client to be unlocked.

NOTE: Unlock codes are based on the system date. When this function is used, both the unlocking computer and the recipient computer must have the same system date.

THIRD PARAMETER: This parameter is used if you would like to have the computer number as part of the unlocking sequence. If you wish to have maximum security, using this parameter would prohibit a user from telling you his hard drive crashed, when he only wanted to unlock your program on another computer.

That is, he wants to unlock your program on another computer, so he calls you up while he is sitting at his second computer and tells you he needs a new unlock code for his computer (you think it is his first one). If you asked him what the pp_compno() was on his screen, and he lied and told you the number that was on his FIRST machine (which, according to your records is a legal copy), the unlocking sequence would not work on his second machine. This parameter is optional and defaults to 1. This paragraph mentioned the unlocking sequence. However, you can use the computer number as a basis for ANY of the 18 hidden user codes.

Example

```
*This example does not show maximum security as defined in the
*"Definitions" section.

PRIVATE code_no

@ 10,10 say "Code entry number from client's computer: " get
code_no
read
@ 12,10 say "Code 1: " + str( pp_ctcodes( 1, code_no )
@ 13,10 say "Code 2: " + str( pp_ctcodes( 2, code_no )
```

See Also PP_CENUM(), PP_COMPNO()

PP_CTCONFILE()

Creates a control file based on given array

Syntax

```
PP_CTCONFILE( <cFilename>, <aArray> )
                    → lResult
```

Arguments

<cFilename> is the complete path and filename of the control file. This assumes that the file does not exist or should be overwritten.

<aArray> contains an array with the data to be placed in the control file. NOTE THAT the array must contain the correct type data and must have 20 elements in it.

Returns

.T. if successful, else .F.

Description

This function creates the file <cFilename> and places the control file data from <aArray> into it. The following table lists the elements the correct type of data. Refer to the "Library CT Funcs" in the .NG database or the section called "Client tracking functions" in the user's manual for a detailed description of the client tracking functions.

ELEMENT	TYPE	DATA
1	C	company name
2	C	name
3	C	address 1
4	C	address 2
5	C	city
6	C	state
7	C	zip

8	C	phone
9	C	product name long form
10	N	product serial number
11	C	distributer name
12	C	pp company
13	C	pp address line 1
14	C	pp address line 2
15	C	pp phone
16	C	pp fax number
17	D	demo expiration date
18	C	expire type "N", "E", "S", "P", or "D"
19	N	exe run-count limit
20	N	LAN workstation limit

Example

```
Refer to the "Library CT Funcs" in the .NG database or the section
called "Client tracking functions" in the user's manual for a
detailed description and example of pp_ctconfile().  The file
SAMPLES.TXT describes an example source file of how to use the
client tracking functions.
```

See Also PP_CTSERIAL(), PP_UCODE(), PP_SERIAL()

PP_CTSERIAL()

Hard-codes the serial number into a .EXE file

Syntax

PP_CTSERIAL(<cFilename>, <cString>) → lResult

Arguments

<cFilename> is the complete path and filename of the .EXE file to put the serial number into. This assumes that the file exists, and that the function PP_50.OBJ was linked into the ROOT of the program.

<cString> is the string serial number that is placed into the .EXE file. Note that the length of this string MUST BE 6 CHARACTERS!

Returns

.T. if successful, else .F.

Description

This function opens the <cFilename> and places the <cString> data in the .EXE file. Any call to pp_serial() will return the contents of <cString>. Also, be sure to place the serial number in the control file using pp_ctconfile() if PROPLUS.EXE is not being used. Refer to the "Library CT Funcs" in the .NG database or the section called "Client tracking functions" in the user's manual for a detailed description of the client tracking functions.

Example

```
if pp_ctserial( "A:\PROPLUS.EXE", "005000" )
    ? "Serial number written okay."
else
    ? "Error writing serial number to file."
endif
```

See Also PP_CTCONFILE(), PP_UCODE(), PP_SERIAL()

PP_EXECT()

Return the current execution counter value

Syntax

```
PP_EXECT()  →  nNumber
```

Returns

Counter <nNumber> if successful, 0 upon error

Description

Retrieves the value of the EXE execution counter stored in the control file.

Example

```
@ 10,20 say "Number of executions: " + ltrim( str( pp_exect() ) )
```

See Also PP_NEXECT(), PP_EXPIRED(), PP_ADVECT(), PP_EXELM(), PP_NEXELM()

PP_EXELM()

Return the current execution counter limit value

Syntax

PP_EXELM() → nNumber

Returns

Counter limit <nNumber> if successful, 0 upon error

Description

Retrieves the value of the EXE execution counter limit stored in the control file.

Example

```
@ 10,20 say "Execution limit: " + ltrim( str( pp_exelm() ) )
```

See Also PP_NEXELM(), PP_ADVECT(), PP_EXECT(), PP_NEXECT(), PP_NEXPTYPE()

PP_EXPDATE()

Return the expiration date

Syntax

```
PP_EXPDATE()  →  dExpireDate
```

Returns

<dExpireDate> is the expiration date of the product or date()-1 if the control file is corrupted.

Description

Retrieves the date of expiration stored in the control file. If pp_getvar() is used, propl19 contains the same information - use it instead.

Example

```
Note that in the following example, propl19 and pp_expdate() are
totally interchangeable.  It is quicker, however to use propl19
since the disk access will be limited to once and not twice.  If
the expire date is empty, we do not print anything on the screen.

@ 7,61 say if( !empty( propl19 ),"Expires: "+ dtoc( propl19 ), "" )
```

See Also

PP_NEXPTYPE(), PP_EXPIRED(), PP_UNLOCK(), PP_EXPTYPE(), PP_GETVAR()

PP_EXPIRED()

Is this application expired (by execution limit or date)?

Syntax

```
PP_EXPIRED() → lResult
```

Returns

.T. if the application is expired or the control file is corrupted, .F. otherwise.

Description

Checks to see if first the current date is greater than the expire date. If the expire type is set to "E", it checks to make sure that the execution count is greater than the execution count limit. If either of these conditions is true, the function returns .T..

Example

```
NOTE that if it has expired (by date) it is important to perform a
pp_upddate() (described later) so that the combination of the date
checking and the expiration date checking prohibit unauthorized use
of your application.

if pp_expired()
    ? "Sorry, but the demo for this product has expired... Please"
    ? "call and order from ProPlus Software at 216-923-1768"
    ?
    pp_upddate()
    cancel
endif
```

See Also PP_EXPDATE(), PP_NEXPTYPE(), PP_UNLOCK(), PP_EXPTYPE(), PP_UPDDATE()

PP_EXPTYPE()

Return the expiration type

Syntax

```
PP_EXPTYPE()  →  cString
```

Returns

" " for error, or one of the following <N>one, <E>xe count, <D>emo, <P>ayments, <S>hareware.

Description

Retrieves the expiration type from the control file. If the control file or the data in it is corrupted, a " " is returned.

Example

```
Set up some global flags
Is this an execution limit version?  If it is, set the flag
EXE_LIMIT to .T. for use later in the program.

exe_limit=( pp_exptype() = "E" )
```

See Also PP_NEXPTYPE() PP_EXPDATE() PP_EXPIRED() PP_UNLOCK()

PP_GETVAR()

Create PUBLIC variables from control file

Syntax

```
PP_GETVAR()  →  lResult
```

Returns

.T. if data successfully retrieved, .F. if control file is corrupted.

Description

This function creates the following public variables and fills them with the data contained in the control file.

PROPL1	company name
PROPL2	name
PROPL3	address 1
PROPL4	address 2
PROPL5	city
PROPL6	state
PROPL7	zip
PROPL8	phone
PROPL9	product name long form
PROPL10	product serial number (character data type)
PROPL11	distributer name
PROPL12	pp company
PROPL13	pp address line 1
PROPL14	pp address line 2
PROPL15	pp phone
PROPL16	pp fax number
PROPL17	last date system used (date data type)
PROPL18	last time system used
PROPL19	expiration date

Example

```
if !pp_getvar()
    ? "Error reading control file... Call technical support!"
    ?
    cancel
endif
```

See Also PP_SAVEPP(), PP_CHKPP(), PP_CRPPP()

PP_ISDRIVE()

Determine the presence and status of floppy

Syntax

```
PP_ISDRIVE( <nDrive> ) → nStatus
```

Arguments

<nDrive> is the diskette drive number, 0 = A:, 1 = B:

Returns

-1 - wrong parameters
0 - drive loaded and ready to read or write
1 - drive door open or diskette inserted upside down
2 - diskette is unformatted
3 - write protected
4 - undetermined

Description

This function is designed as a full replacement for isdrive(). Where isdrive() returns just .T. or .F. depending if the diskette drive is ready or not, pp_isdrive() returns a numeric code designating the diskette drive's status.

Example

```
iStatus := PP_ISDRIVE( 0 )

DO CASE
  CASE iStatus == 1
     Qout( "The door to drive A is open." )
  CASE iStatus == 2
     Qout( "The diskette in drive A is not formatted." )
  CASE iStatus == 3
     Qout( "The diskette in drive A is write-protected." )
  CASE iStatus == 4
     Qout( "Something unknown is wrong with drive A." )
ENDCASE
```

PP_KILLPROT()

Remove the copy protection

Syntax

```
PP_KILLPROT( <nType> )  →  lResult
```

Arguments

<nType> is the copy protection type:

1	= external file, single user (default)
2	= internal data, single user
3	= external file, network/multiple user

Returns

.T. if successful, else .F.

Description

Will remove all copy protection from the current application. This function, like most others, relies on the public pp_cfile variable to be defined as mentioned in the beginning notes. This function is most commonly found as one of the hidden user codes. Refer to pp_ucode() to see how to integrate this. Note that this file removes the copy protection file from the user's hard drive. It will do so for all DOS attributes except for read-only. If the user is smart enough to back up his application directory before he calls you, he could restore his backup files and continue to use your application. There is really no way to stop the use of the EXE file on a particular computer where it was once unlocked.

NOTE: Depending on which type of copy protection you are using in the application, you must use the corresponding number in this function. If the correct number is not used, the copy protection might not be disabled.

Example

```
if pp_killprot( 1 )        && kill copy protection on hard drive
    @ 2,0 say "Protection erased!"
endif
```

See Also PP_COPYWRITE(), PP_COPYCHK(), PP_UCODE()

PP_LANACTIVE()

Determine if application was network protected

Syntax

```
PP_LANACTIVE()  →  lResult
```

Returns

.T. if application was network protected (LAN limit > 0), else .F.

Description

When a LAN limit is entered into the control file by either PROPLUS.EXE, the client tracking functions, or library functions during run-time, the application becomes "LAN active." This function is used to determine which parameters to pass to pp_copywrite() and pp_copychk(). Refer to the "Program Definitions" in the .NG database or the section called "Definitions" in the user's manual.

Example

```
Refer to SAMPLES.TXT for more examples of the network functions.

if pp_lanactive()
   pp_copychk( 3 )
else
   pp_copychk( 1 )
endif
```

See Also PP_COPYCHK(), PP_COPYWRITE(), PP_UCODE(), PP_LANLIM(), PP_LANUSERS()

PP_LANCHECK()

Verify that workstation count < workstation limit

Syntax

```
PP_LANCHECK()  →  lResult
```

Returns

.T. if the workstation limit has not been reached, .F. otherwise.

Description

If the current workstation count is equal to or greater than the workstation limit, this function returns .F.. This function is used in pp_copychk() so that when a workstation logs on, if it is not currently licensed and the count is less than the limit, it will automatically be granted a license. Refer to the "Program Definitions" in the .NG database or the section called "Definitions" in the user's manual.

Example

```
Refer to SAMPLES.TXT for a description of source examples that are
available.

if pp_lancheck()
    pp_lanplus()
else
    ? "Workstation limit exceeded!"
    cancel
endif
```

See Also PP_LANLIM(), PP_LANPLUS(), PP_LANMINUS(), PP_NLANLIM(), PP_UCODE()

PP_LANDECR()

Decrements the current number of workstations by 1

Syntax

```
PP_LANDECR()  →  lResult
```

Returns

.T. if successful, else .F.

Description

Decrements the current number of workstations by 1. This is used for limiting the number of workstations that use the application AT ONE TIME. Note that copy protection is not an issue here. The function pp_lanincr() is used upon startup after testing pp_lancheck() and then this function is used both in exit AND in errorsys. Refer to the "Program Definitions" in the .NG database or the section called "Definitions" in the user's manual for a detailed description of network layout.

Example

```
&& clean up and exit application
&& delete temporary files
pp_landecr()                    && log this workstation out
```

See Also PP_LANMINUS(), PP_LANACTIVE(), PP_LANCHECK(), PP_LANLIM(), PP_UCODE()

PP_LANINCR()

Increments the current number of workstations by 1

Syntax

```
PP_LANINCR()  →  lResult
```

Returns

.T. if successful, else .F.

Description

Increments the current number of workstations by 1. This is used for limiting the number of workstations that use the application AT ONE TIME. Note that copy protection is not an issue here. This function is used upon startup after testing pp_lancheck() and then pp_landecr() is used both in exit AND in errorsys. Refer to the "Program Definitions" in the .NG database or the section called "Definitions" in the user's manual for a detailed description of network layout.

Example

```
if pp_lancheck()                && am I allowed to add another user?
    pp_lanincr()                && add another user
else
    ? "User limit exceeded."
    cancel
endif
```

See Also PP_LANPLUS(), PP_LANACTIVE(), PP_LANCHECK(), PP_LANLIM(), PP_UCODE()

PP_LANLIM()

Returns the current workstation limit

Syntax

```
PP_LANLIM()  →  nNumber
```

Returns

Counter limit <nNumber> if successful, 0 upon error

Description

Retrieves the value of the LAN workstation counter limit stored in the control file. This function returns the limit of the number of workstations that was originally set by either PROPLUS.EXE or by pp_ctconfile() and resides in the control file.

Example

```
@ 10,20 say "Workstation limit: "+ltrim( str( pp_lanlim() ) )
```

See Also PP_LANUSERS(), PP_NLANLIM(), PP_LANCHECK(), PP_LANACTIVE()

PP_LANMINUS()

Decrements the allowed number of workstations by 1

Syntax

```
PP_LANMINUS()  →  lResult
```

Returns

.T. if successful, else .F.

Description

Decrements the allowed number of workstations by 1. This is used for copy protecting or limiting the number of workstations using the application. If three workstations are currently licensed and they have run the application, a fourth workstation will be denied access. If the company calls you up and requests to purchase another license, you simply need to provide a pp_ucode() number to increment the limit by 1. If you would like to change the actual limit back down by one, you would need to provide a pp_ucode() to decrement the allowed number. ***Note that if you increase the number by 1 an a new workstation logs on, decreasing the limit will not prevent that workstation from continued use.*** You need to decrement the limit counter AND ALSO execute pp_copywrite() to overwrite the current copy protection file. Refer to the "Program Definitions" in the .NG database or the section called "Definitions" in the user's manual for a detailed description of network layout.

Example

```
pp_lanminus()    && decreases the workstation counter limit by 1.
```

See Also PP_LANDECR(), PP_UCODE(), PP_LANUSERS(), PP_NLANLIM(), PP_LANCHECK()

PP_LANPLUS()

Increments the allowed number of workstations by 1

Syntax

```
  PP_LANPLUS()  →  lResult
```

Returns

.T. if successful, else .F.

Description

Increments the allowed number of workstations by 1. This is used for copy protecting or limiting the number of workstations using the application. If three workstations are currently licensed and they have run the application, a fourth workstation will be denied access. If the company calls you up and requests to purchase another license, you simply need to provide a pp_ucode() number to increment the limit by 1. Refer to the "Program Definitions" in the .NG database or the section called "Definitions" in the user's manual for a detailed description of network layout.

Example

```
pp_lanplus()      && increases the workstation counter limit by 1.
```

See Also PP_LANINCR(), PP_UCODE(), PP_LANUSERS(), PP_NLANLIM(), PP_LANCHECK()

PP_LANUSERS()

Returns the current workstation user-count

Syntax

```
PP_LANUSERS()  →  nNumber
```

Returns

Counter <nNumber> if successful, 0 upon error

Description

Retrieves the value of the LAN workstation counter stored in the control file. This number starts at zero, and if networking is active, it is incremented every time a NEW workstation executes pp_copychk(). This number is compared to pp_lanlimit() to see if another workstation is licensed.

Example

```
@ 10,20 say "Workstation count: "+ltrim( str( pp_lanusers() ) )
```

See Also PP_LANUSERS(), PP_NLANLIM(), PP_LANCHECK(), PP_LANACTIVE()

PP_LASTDAY()

Return the last date of the month

Syntax

```
PP_LASTDAY( <dDate> )  →  dDate
```

Arguments

<dDate> is a date in the month of which you would like to know the last day.

Returns

The last date of the month of the date you passed.

Description

This function is used to determine the last date of a particular month. It is used by other Protection Plus library functions such as pp_npdate() to help determine the next payment date. This function is now documented for your use also, if desired.

Example

```
@ 4,2 say "Last day of this month: " +dtoc( pp_lastday( date() ) )
```

See Also PP_NPDATE()

PP_NEXECT()

Reset the execution counter value

Syntax

```
PP_NEXECT( <nNewCount> )  →  lResult
```

Arguments

<nNewCount> is the number to which to set the execution counter.

Returns

.T. if successful, else .F.

Description

Sets the execution counter, kept in the control file, to the number specified. This function allows the execution counter to be changed from within the application itself. This could be one of the hidden user codes.

Example

```
pp_nexect(0)     && sets execution counter to zero
```

See Also

PP_EXELM(), PP_UCODE(), PP_EXECT(), PP_ADVECT(), PP_NEXELM()

PP_NEXELM()

Reset a new execution limit value

Syntax

```
PP_NEXELM( <nNewLimit> )  →  lResult
```

Arguments

<nNewLimit> is the number to which to set the execution counter limit.

Returns

.T. if successful, else .F.

Description

Sets the execution counter, kept in the control file, to the number specified. This function allows the execution counter limit to be changed from within the application itself. This could be one of the hidden user codes.

Example

```
pp_nexelm(50)    && extends the execution counter limit to 50.
```

See Also PP_EXECT(), PP_EXELM(), PP_ADVECT(), PP_NEXECT(), PP_UCODE()

PP_NEXPTYPE()

Reset the expiration type and date or execution limit

Syntax

```
PP_NEXPTYPE( <sType>, <dData> OR <nData> )
                       →  lResult
```

Arguments

<sType> is the new expiration type.
<dData> OR <nData> is the data associated with the expiration type.

Returns

.T. if successful, else .F.

Description

Resets the expire type to the <sType> given. Depending on what the value of <sType> is, sets other applicable data. <sType> must be upper-case!

TYPE	DATA
<D>emo	DATE of expiration
<E>xe count	NUMBER of executions allowed
<P>ayments	DATE of next payment (expiration)
<S>hareware	DATE of expiration

Note that <N> is not a choice. To set expiration type to "N", use the pp_unlock() function. The function pp_nexptype() is also used to reset the expiration date. Also note that when the type is changed to <E>, the expiration date field is blanked and the execution counter is zeroed. When the type is changed to <P>, <D>, or <S>, the execution limit and counter is zeroed. This could be one of the hidden user codes. Refer to pp_ucode() for information on how to integrate this.

Example

```
*Set next expiration type to <P>ayments and make it expire on
*the next payment date around the 20th of next month

if pp_nexptype("P",pp_npdate(date(),20))
    @ 2,0 say "Application payment expire date extended to"+;
        dtoc(pp_npdate(date(),20))
endif

-OR-

*Set the next expiration type to <D>emo and make it expire on
*12/31/92

if pp_nexptype("D",ctod("12/31/92"))
    @ 2,0 say "Application demo expire date set to 12/31/92"
endif
```

See Also PP_EXPIRED(), PP_UNLOCK(), PP_NPDATE(), PP_EXPDATE(), PP_UCODE()

PP_NLANCT()

Reset a new workstation count value

Syntax

```
PP_NLANCT( <nNewLimit> )  →  lResult
```

Arguments

<nNewLimit> is the number to which to set the workstation counter.

Returns

.T. if successful, else .F.

Description

Sets the workstation counter, kept in the control file, to the number specified. This function allows the workstation counter to be changed from within the application itself. This could be one of the hidden user codes and is typically used to zero the counter.

Example

```
pp_nlanct( 0 )    && resets the workstation limit counter to zero.
```

See Also PP_UCODE(), PP_LANLIM(), PP_LANPLUS(), PP_LANMINUS(), PP_NLANCT()

PP_NLANLIM()

Reset a new workstation limit value

Syntax

```
PP_NLANLIM( <nNewLimit> )  →  lResult
```

Arguments

<nNewLimit> is the number to which to set the workstation counter limit.

Returns

.T. if successful, else .F.

Description

Sets the workstation limit counter, kept in the control file, to the number specified. This function allows the workstation limit counter limit to be changed from within the application itself. This could be one of the hidden user codes.

Example

```
pp_nlanlim(50)    && extends the workstation counter limit to 50.
```

See Also PP_UCODE(), PP_LANLIM(), PP_LANPLUS(), PP_LANMINUS(), PP_NLANCT()

PP_NPDATE()

Return next payment date

Syntax

```
PP_NPDATE( <dDate>, <nDay> )  →  dDate
```

Arguments

<dDate> is a date in the current month.
<nDay> is the day of next month that payment is due.

Returns

Date of next payment due.

Description

The <nDay>th day in the month after the month of <dDate> is returned. If this day happens to be on a weekend, the next Monday date is returned. This function is used in conjunction with pp_nexptype(). Together with the user-code, your client can call you up after you receive his monthly check, type in the appropriate user-code and have his product work until next month.

Example

```
*Set next expiration type to <P>ayments and make it expire on
*the next payment date around the 20th of next month

if pp_nexptype("P",pp_npdate(date(),20))
    @ 2,0 say "Application payment expire date extended to"+;
        dtoc(pp_npdate(date(),20))
endif
```

See Also PP_NEXPTYPE(), PP_UCODE()

PP_OLDCOPY()

Verify the old-style copy protection

Syntax

```
PP_COPYCHK()  →  lResult
```

Returns

.T. if successful, else .F.

Description

All security has been modified from all versions previous to the Professional Edition. This function is used to upgrade an existing client to a new version of your application without needing to unlock them. That is, when you start linking in the Professional Edition libraries AND if you were and still are using the copy protection functions, you should use this function to upgrade their copy protection file to the new version. Do this only if you do not want to force your clients to call to become unlocked.

Example

```
If you would like to automatically upgrade the copy protection on
the computer to the latest and revert to a demo mode if no
protection exists, follow the sample code:

PUBLIC demo_mode

if pp_oldcopy()
   pp_copywrite( 1 )
endif
demo_mode = !pp_copychk( 1 )
```

See Also PP_COPYCHK(), PP_COPYWRITE(), PP_KILLPROT()

PP_PATH()

Determine the path where EXE running resides

Syntax

```
PP_PATH()  →  cString
```

Returns

cString - path to the executable file being executed

Description

This function returns the DOS path (including drive) location where the EXE file being run resides. This is useful for determining where any .DBF or other data files also exist. It is mandatory to use this function as part of the PP_CFILE when a control file is used and is located with the .EXE, and the .EXE is being run from another directory.

That is, the Protection Plus functions cannot "find" the control file or the copy protection file automatically. If you allow your users to place your .EXE file in the DOS PATH statement, this function will allow you to add the path to the control file (and copy protection file) by adding the string returned as part of the PP_CFILE.

NOTE: This function ends the string in a "\" backslash character automatically. **Requires DOS v3.xx and above.**

Example

```
PUBLIC pp_cfile, path           && make variables

path = pp_path()                 && fill path variable with path
pp_cfile = path+"proplus.ppp"   && make the PP_CFILE have path also

set default to (path)            && so Clipper can find its .DBFs
```

PP_SAVEPP()
Save changed control file data

Syntax
```
PP_SAVEPP()  →  lResult
```

Returns
.T. always

Description
This function is used if you would like to give the user the ability to change the string data stored in the configuration file upon request. It is necessary to first call the pp_getvar() function to create the public variables. The client-specific variables can then be changed. Call this function after the variables are changed to store them into the control file. Note the importance of the data type. This function is most common-ly found as one of the hidden user codes. Refer to pp_ucode() to see how to integrate this.

Example

```
*Assuming that pp_getvar() has already been run:

@ 10,10 say "Enter new company: " get propl1
@ 11,10 say "   Enter new name: " get propl2
@ 12,10 say "Enter new address: " get propl3
@ 12,10 say "Enter new address: " get propl4
@ 13,10 say "   Enter new city: " get propl5
@ 14,10 say "  Enter new state: " get propl6
@ 15,10 say "    Enter new ZIP: " get propl7
@ 16,10 say "  Enter new phone: " get propl8
read
pp_savepp()
```

See Also PP_CRPPP(), PP_CHKPP()

PP_SERIAL()

Return the product serial number

Syntax

```
PP_SERIAL()  →  cString
```

Returns

Serial number as a string, or "ERROR!".

Description

Returns the serial number of the product. If hard-coded serial numbers are used, the hard-coded number is returned; otherwise, the serial number in the control file is returned. If hard-coded serial numbers are not being used and the control file is corrupted, "ERROR!" will be returned. If either method is used, but the product has not been stamped with a serial number by PROPLUS.EXE or the client tracking functions, "ERROR!" will be returned.

NOTE: This function MUST be included in your source code if you intend on using copy protection. Whether or not you display the actual number, you must include it so that the application is stamped as unique. If you do not use copy protection, however, it is not required.

IMPORTANT: A separate .OBJ file must be linked into the ROOT of your application whether or not you are using overlays. The two files PP_50.OBJ and PP_S87.OBJ are distributed with the application.

Example

```
@ 4,2 say "Serial: " + pp_serial()
```

See Also PP_COMPNO()

PP_UCODE()

Return user (hidden) code

Syntax

```
PP_UCODE( <nNumber>, <nCode>, [<nCompno>] )
                         →  nResult
```

Arguments

<nNumber> is the number input from the user.
<nCode> is the code entry number given by pp_cenum().
[<nCompno>] (optional) is the computer number or fixed number.

Returns

Number between -1 and 18.

Description

Compares the code entry value passed to today's code entry numbers for the given code entry random number generated previously with pp_cenum(). If the value is correct, it returns a number from 1-18 to represent the "hidden" function to be run. Otherwise, a zero is returned indicating an invalid code and a -1 is returned if a bad parameter was sent. This is where the great flexibility lies in the Protection Plus library. You have the complete control over what you want to do, and what you would like to hide from your users, while still keeping the capability of your procedures available at your request.

NOTE: Unlock codes are based on the system date. When this function is used, both the unlocking computer and the recipient computer must have the same system date.

THIRD PARAMETER: This parameter is used if you would like to have the computer number as part of the unlocking sequence. If you wish to have maximum security, using this parameter would prohibit a user from telling you his hard drive crashed, when he only wanted to unlock your program on another computer. That is, he wants to unlock your program on another computer, so he calls you up while he is sitting at his second computer and tells you he needs a new unlock code for his computer (you think it is his first one). If you asked him what the pp_compno() was on his screen, and he lied and told you the number that was on his FIRST machine (which, according to your records is a legal copy), the unlocking sequence would not work on his second machine. This parameter is optional and defaults to 1. This paragraph mentioned the unlocking sequence. However, you can use the computer number as a basis for ANY of the 18 hidden user codes.

Example

The following example is from an application's power-up screen. The first thing here is the entering of the password. Notice how a set key statement is used for the user-code function.

```
set key -4 to ucode           && F5 for hidden routines
password=space(8)
do while empty(password) .AND. !lastkey()=27
    @ 20,28 clear to 20,60
    @ 20,30 say "Password: " get password picture "@!"
    read
enddo
set key -4 to

* Write your own code here to test password and continue program
* if ok.
if password#"PROPLUS "
    pp_upddate()
    set color to
    clear
    @ 2,0 say "Incorrect password entered, program aborted!"
    @ 4,0 say ""
    cancel
endif
* At this point, the program is allowed to continue.  This is where
* you would stick the first "DO" command to execute your main
* menu or whatever you do next.
```

```
do main_menu
```

This function UCODE() you create to get the secret number after the user signals the code entry key sequence, whatever you define. For the purposes of this demo, only the result routine number is printed on the screen. In your application, you could set up a set of case statements for each value (1-18) to carry out your routines. A zero is returned if an invalid value was entered. The program can execute a function and continue with the program or execute a function and cancel to DOS. You have complete control! All of these functions drop to DOS after executing.

```
FUNCTION ucode
    PRIVATE _routine, code_no
    clear gets
    code_no = pp_cenum()
    @ 10,10 say "Code entry: " + ltrim( str( code_no ) )
    get_code=0
    @ 20,28 say "Enter code: " get get_code picture "@Z"
    read
    if lastkey()=27
        retu .T.
    endif
    _routine = pp_ucode( get_code, code_no, pp_compno() )
    set color to
    clear
    do case
        case _routine=1              && unlock application
            if pp_unlock()
                @ 2,0 say "Application unlocked..."
            endif
        case _routine=2              && advance payment date
            if pp_nexptype("P",pp_npdate(date(),20))
                @ 2,0 say "Application payment expire date "+;
                    extended to "+dtoc(pp_npdate(date(),20))
            endif
        case _routine=3              && kill protection from disk
            if pp_killprot()
                @ 2,0 say "Protection erased!"
            endif
        case _routine=4
            if pp_nexptype("D",ctod("4/10/92"))
                @ 2,0 say "Application demo expire date set "+;
                  "to 4/1/92"
            endif
        case _routine=10              && write copy protection file
            pp_copywrite()
        case _routine=0               && error
            retu .F.
    endcase
    @ 4,0 say ""
    exit_program()
return .T.
```

See Also PP_CENUM(), PP_COMPNO(), PP_NEXPTYPE(), PP_NPDATE(), PP_UNLOCK()

PP_UNLOCK()

Unlock the program (from date or execution count expiration)

Syntax

```
PP_UNLOCK()  →  lResult
```

Returns

.T. if successful, else .F.

Description

Updates the control file to make the expiration type "N" for none and empties the expiration date field. This is most commonly found as one of the hidden user codes. Upon the completion of payment from your client, you may unlock him from any type of expiration (either date or exe counts). This leaves all copy protection intact. Refer to "Commonly Asked Questions" for more information about application limiting and unlocking versus copy protection. Refer to pp_ucode() to see how to integrate this function as a user-code.

Example

```
if pp_unlock()
    @ 2,0 say "Application unlocked..."
endif
```

See Also PP_UCODE(), PP_NEXPTYPE()

PP_UPDDATE()

Updates the last date and time used fields

Syntax

```
PP_UPDDATE()  →  lResult
```

Returns

.T. if successful, else .F.

Description

Stores the current value of date() and time() in the control file. This should be run upon exiting the program. This is because if the user keeps setting back the clock, it would be impossible to avoid the narrowing window. If he uses the program for an hour every day, eventually, no matter how much he changes the date and time, will not be able to access the program. This occurs provided that you run the check date function pp_valdate() upon entry.

Example

```
A function to clean up screen and update the last used date and
time is recommended.  The last used date and time should be updated
upon exiting because if the user keeps setting back the clock, it
would be impossible to avoid the narrowing window. If he uses the
program for an hour every day, eventually, no matter how much he
changes the date and time, will not be able to access the program.
This occurs provided that you run the check date function
pp_valdate upon entry.

FUNCTION exit_program
    pp_upddate()
    set color to
    clear
    cancel
return .T.
```

(continued on next page)

```
NOTE that if it has expired (by date) it is important to perform a
pp_upddate() (described later) so that the combination of the date
checking and the expiration date checking prohibit unauthorized use
of your application.

if pp_expired()
    ? "Sorry, but the demo for this product has expired... "
    ? "Please call and order from ProPlus Software at
           216-923-1768."
    ?
    pp_upddate()
    cancel
endif
```

See Also PP_UCODE(), PP_VALDATE()

PP_VALDATE()

Validates last date/time used

Syntax

```
PP_VALDATE()  →  lResult
```

Returns

.T. if the current date and time > last date and time used, .F. otherwise or on error.

Description

Makes sure that the last date and time used, stored in the control file, is less than the current date and time.

Example

```
The following checks the date and the time to make sure that the
last date and time used is less than the current date and time.  If
not, the user is trying to "back up" the computer clock in order to
gain more time from your running demo or payment package.  This
check is only valid IF the current expire type is NOT "None" (which
means unlocked).  The test is run once.  If it fails, the user can
enter the new date and time.  If it still fails then he is given an
option to enter a code number.  This is just in case the user
accidentally forwards his DOS date to an advanced value and runs
the program and then sets it back correctly.  If you do not follow
this two step check as shown below, the user will NEVER be able to
set his clock back to the right date and run your application.
Upon entering the correct code number, the current DOS date and
time is saved into the last used date and time field in the control
file.

if !pp_valdate() .AND. pp_exptype()#"N"
    clear
    curr_date=date()
    curr_time=time()
    @ 2,0 say "Please set the date and time correctly:"
    ?
    ?
    run date
    @ 7,0 say ""
    run time
endif

if !pp_valdate() .AND. pp_exptype()#"N"
    clear
    @ 2,0 say "The date and time is still incorrect.  Please
              call" + "technical support"
    @ 3,0 say "for assistance.              Code entry #:"+;
        ltrim(str(pp_cenum()))+"  Date: "+dtoc(date())
    get_code=0
    @ 5,0 say "Enter code: " get get_code picture "@Z"
    read
    if pp_ucode(get_code)=5   && arbitrary number chosen by you
        pp_upddate()
        pp_getvar()     && get new values since they just changed
    else
        clear
        cancel
    endif
endif
```

See Also PP_UPDDATE(), PP_UCODE()

4.2 Function Quick-Reference List

pp_advect()	Increment the execution-limit counter
pp_cenum()	Return the current code entry number
pp_chkpp()	Verify the integrity of the control file
pp_compno()	Return the computer number
pp_copychk()	Verify the copy protection
pp_copywrite()	Create the copy protection file
pp_crppp()	Create a new control file with blank data
pp_ctcodes()	Find out the unlock code numbers for display
pp_ctconfile()	Creates a control file based on given array
pp_ctserial()	Hard-codes the serial number into a .EXE file
pp_exect()	Return the current execution counter value
pp_exelm()	Return the current execution counter limit value
pp_expdate()	Return the expiration date
pp_expired()	Is this application expired by date or exec. count?
pp_exptype()	Return the expiration type
pp_getvar()	Create public variables from control file
pp_isdrive()	Determine the presence and status of floppy
pp_killprot()	Remove the copy protection
pp_lanactive()	Determine if application was network protected
pp_lancheck()	Verify that workstation count < workstation limit
pp_landecr()	Decrements the current number of workstations
pp_lanincr()	Increments the current number of workstations
pp_lanlim()	Returns the current workstation limit
pp_lanminus()	Decrements the allowed number of workstations
pp_lanplus()	Increments the allowed number of workstations
pp_lanusers()	Returns the current workstation user count
pp_lastday()	Return the last day of the month
pp_nexect()	Reset the execution counter value
pp_nexelm()	Reset a new execution limit value
pp_nexptype()	Reset the expiration type and date or exec. limit
pp_nlanct()	Reset a new workstation count value
pp_nlanlim()	Reset a new workstation limit value
pp_npdate()	Return next payment date

pp_oldcopy()	Verify the old-style copy protection
pp_path()	Determine the path where .EXE running resides
pp_savepp()	Save changed control file data
pp_serial()	Return the product serial number
pp_ucode()	Return user (hidden) code
pp_unlock()	Unlock the program from date or exec. expiration
pp_upddate()	Updates the last date and time used fields
pp_valdate()	Validates last date/time used

4.3 Client Tracking Functions

Some people have existing custom software designed to track their clients. If you fall into this category, you should know that the functionality of PROPLUS.EXE can be brought over to your custom software. That is, there are functions in the Protection Plus library to hard code a serial number into an .EXE files, create a control file, and generate a table of unlock codes.

These functions give you a tremendous ability to adapt to special protection requirements by allowing you to write a custom Clipper® application to do things like track your clients or simply protect a diskette. Refer to the library reference section for the syntax of the client tracking functions.

The client tracking functions are:

pp_ctcodes()
pp_ctconfile()
pp_ctserial()

Note that these same functions are used in the PROPLUS.EXE program!

4.4 Reserved Words

The following variable names are reserved. They cannot be used as variable names elsewhere in your application. These variables are only reserved if you call pp_getvar(). The purpose of these variables is to allow you to display licensing information stored in the control file. If you intend to use pp_getvar(), do not use these variable names elsewhere in your program.

PROPL1
PROPL2
PROPL3
PROPL4
PROPL5
PROPL6
PROPL7
PROPL8
PROPL9
PROPL10
PROPL11
PROPL12
PROPL13
PROPL14
PROPL15
PROPL16
PROPL17
PROPL18
PROPL19

5. COMMONLY ASKED QUESTIONS

These questions were asked through the Nanforum on CompuServe. Thank you all for asking questions, for it helps us explain how the product works.

"Once the program is unlocked, what's to prevent someone from copying the unlocked files to another computer? Obviously you must have some way of uniquely identifying the computer or hard drive or both, so what about backups and restores? What happens if they run a disk defragger that moves the files around? How about if they replace the hard drive? What if they have to reformat and/or repartition the drive because of a drive failure or they want different partition sizes then they had before or they upgrade to a different version of DOS? Will any of these things require a new installation?"

When it comes to the term "unlock," there is a distinction that you need to know. That is, the difference between **copy protecting** and **execution limiting.** If you would like to "copy protect" your application, once the application is received by the client he calls you, follows a 10 second sequence of events, and the product is protected. This protection CAN be backed up, restored, unfragmented (moved), copied from one hard drive to a new one (on the same computer), and transferred to a new version of DOS (also on the same computer). The protection is stored in a file with the DOS file attributes you specify. Any utility that can copy this file can manipulate the protection at will. The application will only work, however, on the computer that the protection was installed on.

Copy protection is separate from application limiting. That is, the "unlocked" that we talk about means that the application is free to run without expiring on a certain date or after a certain number of executions.

If you choose to copy protect the application AND limit it to a demo version, you can unlock the demo version but still keep the copy protection. There is **NO WAY** for you to remotely "eliminate" copy protection from your application without changing the source. You can, however, remotely disable the protection from one computer and then remotely enable it on another by a simple telephone call from your user. This is great when he or she calls you up and asks to move the application to another computer. You do not need to send them new disks. If they copy the files to another computer, the copy protection routines will tell you that this is not a valid computer. You could, if you choose to, add a copy protection file to any computer using your software anywhere remotely over the phone. You can also run any other of your special "hidden" routines that you do not want your user to perform unaided, remotely!

"What is to keep a person from using a disk copy program, or disk copy card from copying the distribution disks and then using the same code to install it on several systems? I have found that using a "hardware key" is the best copy protection around. They add about $40.00 to the cost of the software, allow customers to make backups of the distribution disks and keeps the software from being stolen. The customer can restore the code to a drive without having to bother me. (I tend to be out of the office a lot and like to keep the Cell. Phone bill down). Can your product cause the code to revert to a "DEMO" mode if the correct key is not entered? I let all my customers share their disks with others. If the hardware key is missing they get a demo version. (To encourage the sharing I pay them a finders fee for every copy sold off of a shared version.) I have seen a similar system used where the customer had to call in every year or so, dependant upon the number of times the program was started, for access codes."

First, nothing is going to prevent a user from using diskcopy or the like. However, the application will only work on the computer that it was installed on. What happens is the application leaves your drive unprotected. The user calls you up on the phone when they receive the package. The tell you a number on their power-up screen. You type the number in the library unlock code utility and read them a number on your screen. Their application is now protected for that computer only - and they never have to call you again, even if they need to do backups and restores, unfragmenting, etc. This code that they read you changes randomly (every second). The codes on your computer change as a function of the random number, every day! Even if one user tells another user the unlock code, it will not work (unless by chance they received the same random number on the same date). The odds of this are minuscule. This protection is not as foolproof as the hardware key, but it is a lot cheaper (since you are not limited to the number of users per the one-time fee).

As far as the automatic demo mode goes, you have COMPLETE CONTROL over what happens when. The library was designed in such a way that if you use copy protection and the copy protection failed the test, you can make it a demo, abort, or whatever you want. You can make the user call every year, every day, every month (on the 20th), after 10 executions, or whenever! Upon demand, you can then remotely unlock it so that they never have to call you, but it is still copy protected. This would be great for people who pay you every month on lease, etc. **You have complete control!**

"How does the product protect against a legitimate user making a backup of the already installed product and restoring on another "identical" system (ie. same BIOS & hardware)?"

Unfortunately there is now way using either our method of copy protection **OR ANY** method, could anyone determine one computer from another (identical one) or stop unauthorized copies unless the hard disk itself is manipulated or a hardware key is used. This writing data to a sector on the hard drive technique (at least in our experience) requires protecting a floppy and mailing it to the user. Obviously this user may require special disk shipments in the event of hard disk failure and the like. We have eliminated that approach. Given two identical computers, the application might work. I say might because I haven't actually found two computers that are identical that give me the same copy protection password. Keep in mind that 99% of the unauthorized copying of applications occurs between two un-identical computers. For all practical purposes, the Protection Plus method is sufficient and definitely more convenient.

"How come when I am trying to use pp_ucode() with UNLOCK.EXE, the pp_ucode() function only returns a zero?"

Keep in mind that the unlock codes are based partially on the system date. When the pp_ucode() function is used, both the unlocking computer and the recipient computer **MUST** have the same system date. The codes generated are based upon 1) the current date, 2) the code entry number, and 3) the Protection Plus library serial number.

"I use the copy protection functions. When I ship out a updated version of my program, does the client need to be unlocked again?"

No! Well, it depends on what type of copy protection you are using. If you are using either single external (type #1) or multiple external (type #2, network), you do not need to unlock the client again. The image of the computer is stored in an external file. As long as that external file is valid, the program will pass the copy protection check. If you use single internal copy protection (type #2) the image of the computer is stored in the .EXE file itself. When you update the client with a new .EXE file, the image of the computer is gone! You should not use that type of copy protection when updates are given frequently.

When updating your programs to users, simply provide them with and updated .EXE file. Unless the information in the control file needs changed, you do not need to hard code the serial number or give them a new control file. If you do not use a control file and hard code the serial number in the application, you will need to hard-code the serial number in every updated application you provide your client. Keep in mind that the serial number is also included in the control file. If you update your product frequently, think about using the control file as your only source for the serial number.

Be aware that the copy protection scheme is based partially on the serial number of the end-product. If you unlocked a client and provide them with an updated .EXE file, the user will not need to be unlocked again. If their serial number changes for some reason, you will need to re-unlock them.

"Can I replace PROPLUS.EXE with our in-house client tracking program?"

Yes. There are client tracking functions in the Protection Plus library that give you access to the create control file, hard code serial number, and unlock codes technology. Refer to "Client Tracking Functions" for a better explanation.

6. DOS UTILITIES

6.1 UNLOCK.EXE

UNLOCK.EXE is used to obtain the unlock codes without having to load PROPLUS.EXE. This program is **NOT** to be distributed. The program has been changed since previous versions. It now allows you to enter an optional third parameter for the computer number. If you are going to use maximum security, you will want to add the third parameter to the pp_ucode() function. If you add the parameter to pp_ucode(), you must enter the third parameter in UNLOCK.EXE to get the correct unlock codes. Keep in mind that the third parameter is completely optional in both functions. Here is the way it works.

```
Usage: UNLOCK <code number> [<computer number>]
```

The code number is the code entry number that your customer will read to you. The second number is the computer number displayed on the user's screen if you choose to use "maximum security." If you are not using "maximum security" you may omit the second parameter. Here is a sample:

(found on next page)

```
C:\PROT>unlock 12345 998743

Protection Plus Professional v2.0
DOS unlocking facility              Serial number:  5005

Computer number:  998743,    Unlock codes for 12345:

Code 1:  5882277        Code 10:  8096115
Code 2:  6128259        Code 11:  8342097
Code 3:  6374241        Code 12:  8588079
Code 4:  6620223        Code 13:  8834061
Code 5:  6866205        Code 14:  9080043
Code 6:  7112187        Code 15:  9326025
Code 7:  7358169        Code 16:  9572007
Code 8:  7604151        Code 17:  9817989
Code 9:  7850133        Code 18:  10063971
```

6.2 HCSERIAL.EXE

HCSERIAL.EXE allows you to hard code a serial number into one of your applications from the DOS prompt. This does exactly the same thing that PROPLUS.EXE does during protect time if hard code serial number is enabled. This was written to be used in a protection batch file. Some people want to hard code the serial number and then archive the executable file. Using a protection batch file, that result could be achieved simply.

The parameters are:
   <file> - file name to hard code serial number
   <string> - serial number to be hard coded (6 char max)

If an error occurs, the DOS errorlevel will be set to one. This allows for perfect integration in a PROPLUS.EXE batch file during protection time. Here is a sample:

```
C:\PROT>hcserial \proplus\proplus.exe 193333

Protection Plus Professional v2.0
DOS hard code serialize utility

Working...
Serial number 193333 stored in file \proplus\proplus.exe.
```

7. SAMPLE SOURCE

Ok, do you like looking at source code? Good! This is the section of the manual that demonstrates various ways you can utilize the power and versatility of Protection Plus Professional. The file SAMPLES.TXT contains a list of files that are given to you to experiment with. Because these files are updated frequently, use SAMPLES.TXT as source for the list of sample files and their descriptions. The following sections give a brief explanation of the common uses of the Protection Plus library.

It is recommended that if you are experimenting with a sample program that uses pp_ucode(), print out the unlock codes for code entry #4500. These codes are valid only for the current date. You may print the unlock codes list using either PROPLUS.EXE or UNLOCK.EXE using the print screen or piping facility. Having these codes for the sample source is very handy!

7.1 To Copy Protect

There are five simple functions that you need to be familiar with in order to copy protect your application. First, pp_ucode() will be used to "hide" the functions pp_copywrite(), pp_cenum(), pp_copychk(), and pp_killprot(). Pp_ucode(), as we will now call the "user-code function," is added most commonly with the set key to statement. That way, when a user presses a function key (let's use F5), a pop up will then display a number and ask for a number. The number displayed is the code entry number. This number is explained under pp_cenum() in the function reference. The number the user has to type in is one of the unlock codes. The number he (the user) types in is passed to pp_ucode(). This function will then decode the number and return a number between zero and eighteen (0-18). This number corresponds to the user code to which you assign functions.

Usually a DO CASE structure is employed to determine which function to run. If the function pp_ucode() returns a zero (0), the code entered was invalid.

Once the structure is set up, you can place the other functions pp_copywrite() and optionally pp_killprot() in the DO CASE structure. These functions create the copy protection file and erase the protection file, respectively. Refer to the file SAMPLE.PRG function UCODE() for an example of this.

7.2 Monthly Unlock (payments)

Creating an application that must be unlocked every month on a certain day is simple. This would be used if, for example, your application is leased. The same basic method described in **7.1 To Copy Protect** dealing with pp_ucode() is used. If copy protection is used, the method described in that section will be implemented along with at least two other functions for the payments. Pp_npdate() will return the date for the given day of the month that you would like to expire. If that day is on a weekend, it will return the next Monday. Pp_nexptype() is used in conjunction with this function to advance the next expiration date. You would pass it a "P" for payments and pp_npdate() as shown in the file SAMPLE.PRG. Last, the application leaves your computer serialized and ready to expire on the day of next month that you setup. You have total control on how the system is configured!

7.3 Application with Modules

If your application is sold in modules, and the user can purchase only what he needs, you could actually ship him the entire package. How you would set it up, assuming that every module has a separate EXE file, is assign a different control file to each module. This would be done even if the program is not packaged as separate EXE files.

In PROPLUS.EXE, you would have a client record for every client and every module in that package. This might be a bit cumbersome at first, but look at the power you have. You could, at will, enable a module to work as a demo, remotely. This would be great if your user calls you up and says he is now interested in another one of your modules. You could, over the phone, enable the module he wants, and not have to worry about it stopping on a certain day or after so many executions.

This is how it is set up. First, the main EXE file (or main menu routine), will have its normal control file. When the user presses the menu function that calls a module, change the value of PP_CFILE to reflect the new module. You can then see if it is unlocked. If it is, run the program, else print a message or whatever else you want. BE SURE to change PP_CFILE back to its original value when complete with that module. When you change the status of a module for a client, use PROPLUS.EXE to keep track of when it was installed, when it will expire, etc. At the touch of a button, you will know which of your clients is using which modules.

The addition of protection or demo modes for the main program and its modules is all controlled through the pp_ucode() function. Refer to this function for coding examples. You could even "nest" the user code values to effectively achieve an unlimited number of hidden user codes.

8. PROPLUS PROGRAMMING SERVICE

We realize that adding this level of protection to your programs may not be easy, therefore we have developed a program to allow you to purchase Protection Plus and let us do the programming for you at a very nominal rate.

If you need help in installing Protection Plus functions into your program, send the top of your program (from line 1 till the first DO) on a 5 1/4 or 3 1/2 inch disk.

List the types of protection that you would like incorporated into your program. We will add the needed functions and send your fully tested source code back to you.

All you need to do is insert it back into the top of your program, recompile, link and it's ready to distribute.

<u>What do I do to get PPS working for me?</u>

List the protections that you would like to have added to your program here by entering the names in the action blanks and the meaning next to it. If you want a certain code to call your functions, enter your function_name() on the action blank. If you want a certain code to call a procedure, write DO your_proc on the action blank.

IE:

Code #	ACTION	MEANING
Code 1:	pp_copywrite()	Activate Protection File
Code 2:	DO zap	Empty users database

Enter your desired actions and meanings below.

Code #	ACTION	MEANING
Code 1:		
Code 2:		
Code 3:		
Code 4:		
Code 5:		
Code 6:		
Code 7:		
Code 8:		
Code 9:		
Code 10:		
Code 11:		
Code 12:		
Code 13:		
Code 14:		
Code 15:		
Code 16:		
Code 17:		
Code 18:		

The charge for this service is $75.00 per source file changed.

For us to return this via US Mail add $2.50 S&H. For us to return this via Federal Express Next Day add $18.00 S&H.

All checks and money orders must be made out in US Dollars.

Name ______
Company ______
Address ______
______
City ______ State ______ Zip ______
Phone (____)______-______ Country ______

PPS $ 75.00
Shipping $______
Total $______ (Please attach payment)

Comments/Suggestions:

______
______
______
______
______
______
______
______
______

9. INDEX