DiskWorks V2.0

Contents

This manual is translated from German with the free version of DeepL.com and ChatGPT. Thanks a lot!

Introduction

This is a user guide for DiskWorks (hereinafter referred to as DW). DiskWorks is a user interface for the Sharp PC-1600.
With this program you can forget about the tiresome and relatively uncomfortable floppy commands (Kill the wildcards). Even the knowledge of entire command sequences can - once it has been defined - be consigned to the waste paper basket.

DiskWorks makes it easy to copy, delete, rename, protect and view files. DW formats and copies discs in their entirety.

As a special bonus, DiskWorks can be used to start all kinds of programs (Basic and machine language) in different ways. DW works with the CE-1600P floppy drive and all compatible RAM discs without any problems (tested by me with 32, 64 and 128, by Gernot with 256KB). The copy functions even support the serial interface (can also be configured from DiskWorks).

DW also offers an info function that provides information about the selected drive (free space, number of files), the status of the batteries and the date and time.

If you own a ‘big’ PC (Windows, Linux, Mac), you can also use a program for backing up data and programs to the PC (via the serial interface). There are several programs available on the market - even free of charge. However, you will have to solder the cables yourself or buy them. A lot is possible here, even with Arduino or ESP32 units.

DiskWorks is now available in version 2.0. Compared to versions 1.0 and 1.1, some major and minor changes have been made.

The input routine for entering file names etc. has been revised and now works correctly. A cursor now flashes when asking for yes/no or when asked to press any key. This makes the whole thing a little easier to understand. You now know when the PC1600 expects something from the user.

The macro language (see DW.CFG) has been extended by two important features.
And last but not least, a few tweaks have been made to the copy routine. I can now also change the name of the file to be copied when copying. This has sometimes proved useful for me.

DW 2.0 starts with the address &DOOO and occupies approx. 6000 bytes on a DW RAM disc, plus a DW.CFG file that you have to create yourself or adapt to your needs. Short basic programs are therefore not affected even if no additional RAM cards are used.

DW.CFG is stored by DW from &CCOO.

19 bytes are required for each file on the source drive (after DW). This means that if MAXFILES=1 is set, there will be approx. 130 files on the medium. More information on this in a later section. At this point it should only be said that the file DW.CFG should be present on the drive S2: in order to operate DW. Otherwise DW will issue error message 152 (file not found) at startup and macros cannot be used.

But first a word about the minimum requirements for DW. Since it is intended as a user interface to start programs, copy them, give them different names, delete them and contains many other useful functions that simplify daily work with the PC-1600, it must run on every system. In other words, even on the basic system without any other programs.

It does – with a few exceptions. Almost all functions can be used if there is no memory module in the PC-1600. The function for copying floppy disks is the exception here. Since the copying process takes place in five steps, the 64KByte of the floppy disk must also find space somewhere in the memory (divided by 5 in each case).

DW then looks for the necessary memory, namely on the memory module S1: and there in the memory area from &80C5 to &BFFF of bank 0. If the module is not there, this will of course go wrong. If there is important data there, it is wiped away after copying the diskette.

I have made another compromise that I don't want to conceal. It's about copying files from drive X: to drive Y:. Unfortunately, this is no faster than with the original operating system version of the copy command. You become a disk jockey. But for this function to work without memory modules, there is no other way (maybe a little, maybe it will come again).

You can also start DW from cassette, but it always tries to search for drive S2: first and read in existing files.

Overview

But back to the operation. First of all, a description of the appearance of DiskWorks after starting and after the input message.

```
-> FILE1 .     4711
   FILE2 .     4712
   FILE3 .BAS! 4712
INF CPY KIL MSK S2: X:
```

This is what it can look like when DW is started. Provided, of course drive S2: is present and contains the displayed files. If the PC is not docked to the CE1600P (with floppy), then COM: is entered as the target drive instead of X:. The file displayed at the top is always the 'current' file, also indicated by the arrow (->). I can mark it for editing, start it (if program) or view it.

File3.BAS in the example is terminated by an exclamation mark “!”. This means that this file has a protection flag and cannot be deleted accidentally.

The number after the file name indicates the size of the respective file in bytes. If more than three files are stored on the drive, you can use the cursor keys (up and down) to jump file by file. Cursor down in the situation shown above would make FILE2 the current file. FILE1 disappears and FILE3.BAS moves up

This works in the same way with the cursor up key when you have reached the respective edges (first or last file), pressing the cursor keys has no effect. It will not beep (DW is a user interface and not a music program!).

If you want to go quickly to the beginning or end of the file list, use the left and right cursor keys (on the block of ten). Right jumps to the end, left to the beginning.

To be able to edit files (rename, copy, delete), they must be selected. This is done file by file with the SPACE key. It is used to invert the marking in each case. This means that it is both switched on and off.

The RCL key can be used to invert the marking of all files that are currently read in and match the file mask. The double arrow key deselects all files.

At this point I must briefly refer to the menu functions. The KBII key is used to quickly swap the displayed source and destination drives. More on the subject of drives later.

The current file (first in the display) can be started, whether BASIC or ML program, by pressing the ENTER key. DW recognizes the file type automatically and independently of the name.

The computer can be switched off at any point in DW with OFF. MODE is used at the top menu level (files are displayed, a menu line as the bottom line in the display) to exit DiskWorks. Otherwise, MODE can be used to cancel any operation before final execution (within security queries etc.).

Function menus

And now to the function key menus. The lowest menu line (INF, CPY, KIL, MSK, S2:, X:) has a counterpart on the so-called second menu level (NAM, PRO, HEX, COM, INI, DCP). DEF switches back and forth between these two lines.

The displayed functions are activated by pressing the function key below (!, ”, #, S, %, &)

But now to the meaning of the abbreviations in the menu lines.

First Menu Level:

INF

This menu item provides information about the currently selected drive, the available files, the selected file mask, the date and the battery status (internal and, if connected, also external).

The top line shows the free space on the RAM disk S2: or the source drive. The file mask is the display template for files to be displayed. The '*.*' is converted into the example mask shown above. The third line provides information about the number of files that match the file mask and how many of them are currently selected (0). The bottom line shows the date, time, internal battery voltage (I=) in percent and the external battery voltage (CE-1600P, E=) if the device is connected.

If the power supply unit is connected, it may happen that a voltage above 100% is displayed. Naturally, the conversion of the analog voltage on the PC-1600 is quite inaccurate. In order to measure the voltage as accurately as possible, several measurements are taken and an average value is then calculated. Nevertheless, small jumps and inaccuracies can still occur. So if a measurement result seems 'strange', simply call up the INF function again...

CPY

This is used to copy files (CoPY). To do this, they must be marked. The files are then transferred from the source drive to the target drive. The date and time of the files are retained.

For copying, DW asks whether the target drive is ready. Pressing 'Y' starts the copying process. It is not started with “N” MODE aborts.

The CoPY command can of course also be used to copy via the serial interface (COM1:). A PC (Linux, Windows, Mac) can and should be used to make data backups on the PC. No backup, no mercy!

To copy to COM1:, all files to be copied are marked and then the CoPY function is started. DW asks before each file whether the interface is ready to accept the data. MODE or 1~ interrupts the copying process here. 'J' starts the transfer.

The return path from the PC is somewhat more laborious. As DW cannot know the name of the file to be transferred, it asks for a file name before each transfer to the PC-1600. The procedure is then the same as for the transfer to the PC.

Now let's talk about backing up larger RAM disks.

If you own a 128 or even a 256 KByte RAM disk, you will certainly want to make a backup from time to time. Of course, this is only possible on several disks. With DW, however, copying is very easy. Simply mark all files on the RAM disk (RCL after start) and start the copy function. When the disk is full, an error message appears and DW has transferred all the files that have already been successfully copied and removed the marker.

Now simply insert the next diskette for the backup and start the copy process again. Do this until DW has copied all files.

(!) DiskWorks always overwrites files with the same name on the target diskette. Caution is absolutely necessary!

The copy progress is indicated in the bottom line of the display by “.” (read operation) and “o” (write operation).

KIL

KIL can be used to delete (KIL1). If files are selected, they are deleted. DiskWorks also asks the user whether the deletion of each file should be confirmed individually. 'Y' ensures that a confirmation prompt is displayed for each marked file. 'N' deletes immediately. MODE aborts.

If no file is selected, wildcards can also be used for deletion. For example, '*.*' deletes all files on the source drive, regardless of the file mask set!

The same applies here: be careful! Deleted files are irretrievably gone!

MSK

Defines the file mask (MaSKe). To display all Basic files, for example, '*.BAS' can be entered after pressing 'S' or F4. Only files with the ending *.BAS are then shown in the display. To delete an existing mask, simply press the ENTER key. MODE aborts and does not change anything.

S2

S2 shows the currently selected source drive. The drive can be changed by pressing '%' or F5. The selection is again made using function keys F2 to F6. F2=COM:, F3=X:, F4=Y:, F5=S1:, F6=S2:. MODE aborts and does not change anything. COM refers to the serial port of the PC-1600 selected by SETDEV. The transmission parameters set in BASIC are also adopted.

COM

Displays the currently selected target drive. It can be changed using '&' or F6. See S2: or source drive.

Second Menu Level:

NAM

NAM is used to rename files. Either all selected files are displayed individually and asked for the new name or (if no file is selected) a reNAMe with wildcards is requested. NAME “*.BAK” AS “*.OLD” renames all files with the extension 'BAK' to files with the extension 'OLD'. MODE terminates at every level of the command. .

PRO

Protecting files. If a file is write-protected (indicated by a '!' after the file name in DW), it cannot be deleted or overwritten (by the operating system or DW functions).

PRO can now be used to switch the protection of the current file in the display (top line) on or off.

HEX

Viewing files in hex mode (incl. ASCII). This can be a very useful function You can 'take a quick look' at any files. They are displayed in hex mode. The corresponding ASCII character appears behind the hex digits in the display.

When the display is full, DW stops. Press any key to display the next line (all others are scrolled by one line).

MODE is also used here to end the file display.

COM

Sets the communication parameters for the serial interface. The default setting after starting DiskWorks corresponds to the BASIC commands:

```
        SETDEV "COM1:" and

        SETCOM"COM1:",9600,8,N,1,X,N
```

iin succession.

The Setcom setting can now be changed as required using the function key menus in this function.

INI

This function is used to format a floppy disk in drive X: (INIt command from BASIC). DW first asks whether a floppy disk has been inserted in X. If the answer is 'Y', the diskette in X: is formatted (initialized) without mercy. So be careful, all data is deleted from the floppy disk.

DCP

Copies entire disks from drive X: to drive X: (DiskCoPy). Please note the remarks made at the beginning about the required hardware configurations.
After starting the function, DW asks for the source disk (source disk), reads part of the disk into memory and then asks for the target disk (target disk). This is repeated until the entire diskette has been copied (5 times).

Configuration File (DW.CFG, DWMac)

Overview

As mentioned at the beginning, a file called DW.CFG is required to operate DW. This file is created by any text editor (e.g. BASIC with LOAD*, SAVE$ or Text+). It contains commands or command sequences (macros, hence the name DWMac) to be configured by the user, which can then be called by DW.

The file must contain a hat (” ^ ” = Shift + SPACE) as the first and last character. Otherwise DW may crash.

And in the version 2.0 of DiskWorks available with this manual, there is another special feature with these macros. First of all, some bad news: DW is no longer automatically exited when a macro is executed. And now a lot of good news: you can now use macros in the usual way (with a small change) as well as within DiskWorks. E.g. marking all *.BAK and automatic deletion. There are many such applications. But there is no limit to the user's imagination.

„Macro-Language“ DWMac

But now for more information on the 'DWMac' language extension.

A little exaggeration is always nice. So don't take the macro language so seriously. It is not an excellent programming language, but rather the writing down of key sequences which DW can then play back.

This is called 'starting a macro'. Macros can be executed both inside and outside DW. Actually 'only' the KEYBUFF$ of the operating system is used.

It is only important that you do not inadvertently execute a macro that was intended for 'outside' within DiskWorks. This will certainly cause the computer to crash.

Structure of the DW.CFG file

In principle, a macro has a very simple structure. It starts with the key which, when pressed, starts the macro. This is followed by a space and then the sequence of commands to be executed begins.

So:

<key><SPACE><command sequence>

The <key> must be a key that can be reached via the keyboard, i.e. the set [A..Z,a..z,0..9,!,“,#,$,%,&,?,:,',{,},[,],(,),|,_,@,;”,]. Everyone should experiment with this. But that should be about all the characters.

<SPACE> does not need to be explained any further, but is necessary in any case.

And now for the <command sequence> Any character that can be reached via the keyboard can be entered. For example, the sequence <LOAD “S2:DW”>. Well, and then a small problem is revealed (you think), because how do I make it clear to the computer that I also want to press ENTER after the Load...? No problem. There are actually two solutions to this problem. The first is the character “~” (the tilde). Instead of the tilde, the macro interpreter (sounds good, doesn't it?) inserts an ENTER when executing the macro.

Special Features

However, it is said to have happened that you not only want an ENTER, but also, for example, the code of the OFF key in a macro. Or even the code of the CTRL key.

Or the SML key…

Well, there's a solution for that too. Key codes of any kind can be written in DW.CFG as the following character string:

\<ASCII-CODE\

So the code is enclosed in two backslashes, e.g. \15\ This means the OFF key, as the ASCII code of the OFF key is 15.

There is another small but nonetheless subtle special feature. With the character string \@, the file name of the top display line can be inserted at the position in the macro. This is practically like a variable that has a different value depending on the situation.

So that you don't have to find out all the codes first, there is a table with the special codes. You can also map the normal ASCII characters in the macros in this way. But after all, it's easier to write “LOAD” than \76\\79\\65\\68\

Key	**Code**
F1	17
F2	18
F3	19
F4	20
F5	21
F6	22
DEF	27
KBII	4
CTRL	3
SHIFT	1
SML	2
Double Arrow	9
RCL	25
DOWN	10
UP	11
BS	5
OFF	15
CL	24
MODE	31
Left	8
Right	12
\	92 or \\
@	64
~	126

And now once again to the 'inside' and 'outside'. In version 1.2 of DW, DW was automatically exited for a macro and thus automatically 'released' to the Basic interpreter.

This no longer happens automatically, but you simply have to create the exit from DW with the MODE key. But how to generate the MODE key? Sure, just learned: simply \31\ as the first character of the command sequence and off you go in the direction of the basic interpreter.

Attention!

This also makes it clear why you have to be careful not to play an 'outside' macro 'inside'. The following is assumed: Loading the Text+ program from DW using a macro. Text+ is on S2: and is called TXT.

The command sequence in the Basic interpreter would therefore be 'BLOAD "S2:TXT<ENTER>'. However, the following should be noted in DW.CFG (we assume that we want to assign the "T" key):

```
T \31\BLOAD"S2:TXT~
```

This is preceded by the Gur code for the Mode button, then the <SPACE> and finally the code for the ENTER button.

If the \31\ is forgotten, DW executes the macro 'Inside' and nothing bad happens until the first "T" of TXT.

But at the T, DW 'thinks' it should execute the macro of the "T" key... This leads to the macro being executed again and again. It may then no longer be possible to cancel it. The only thing that helps is the reset switch. If the error does occur, try the BRK key first. Perhaps the endless macro can be interrupted.

The respective command (or command sequence) is executed immediately after pressing the defined key. There are no security prompts, as is the case when starting programs via Enter.

However, the DW.CFG file must never be longer than 1024 bytes! Otherwise DW will crash at startup! So watch out. DW does not do this at this point.

My Example

The following is a short description of my current DW.CFG file. First of all: the first and last character must be a hat ^! ! The first hat must be in the first line, the last hat must be in a separate line!

```
^C \31\BLOAD"S2:TXT~LS2:DW.CFG~E
T \31\BLOAD"S2:TXT~
X \31\BLOAD"S2:DEB~
F \31\LOAD"S2:FINANZ",R~
D \31\BLOADnS2:DB~
! \31\BLOAD"S2:FKEY,#0,&C008~BLOAD"S2:DW~
U \31\BLOAD"S2:CLOCK~BLOAD"S2:DW~
K \31\BLOAD"S2:TK~
```

```
O \15\
M \9\\25\
B M\18\
A \31\LOAD"S2:DWASM~
E \31\BLOAD"S2:TXT~L\@~E
Q \31\LOAD S2: ST. BAS",R~
^
```

First Line

The first line is quite complicated. The first character is the little hat, so it has nothing to do with the actual macro. Then comes the 'C'. The macro is therefore assigned to the C key. The following space character must separate the key definition from the actual macro. Now the macro begins. \31\ is the key code for the MODE key, so it causes the macro interpreter DW to exit (you can also say: the mode key is pressed for the user). BLOAD "S2:TXT~ loads the program Text+ (is called TXT in my case and is located on S2:). The tilde ~ replaces the Enter key. But this is not the end of the program. Now press the L key for the user and enter the file name S2:DW.CFG with Enter ~. Now that the file has been loaded, the macro does the last thing. It starts the edit mode of Text+ with E. To summarize, the macro loads the text editor and the file DW.CFG for editing. And it does all this from DW with a single keystroke. So the first example already shows a lot of the "macro language" DWMac.

Exclamation Mark!

Now let's look at the macro that starts with the exclamation mark ! (assigned to the ! key). This is also a nice little thing, which the attentive reader has probably already noticed. First the macro loads a function key definition (BLOAD "S2:FKEY) to the position #0,&C008 and then starts the DW program again, i.e. DiskWorks.

U as in detoUr

The same principle is used in the U macro. Only the background clock is loaded there (can be activated with the Shift keys, then remains in the background and waits for activation). DiskWorks is then also started again. So back to the "universal user interface".

Off

A particularly short macro is assigned to the O button. The only action is to press the key with the key code 15, which is nothing other than the OFF key. So O switches the computer off (Off).

„Passing Parameters"

After the particularly short macro O, here's a particularly versatile macro that doesn't look very special at first glance. It starts with E, then exits with \31\ DW, starts S2:TXT~ and activates the load function (L) and? Yes, what is that? There is the parenthesized monkey @ after the backslash \! And what's that for? Quite simply E stands for Edit (of course, the editor Text+ is loaded) and is supposed to edit a file. But which file? Also very simple. The current file, i.e. the file at the top of the display. That's great, isn't it? A universal macro for editing any file. One for all! However, the user now has a little responsibility here, because it may not make much sense to use a debugger with the word processor. DW's macro interpreter does not check anything at this point! How should it?

Final Remarks on DWMac

In conclusion, it is crucial to understand the limitations and requirements for using macros within DiskWorks (DW). Here are the key points to keep in mind:

1. **Activation Requirement**: Macros can only be executed when DiskWorks is active. If you exit DW (except through the macro command \31), macros will no longer run. This is important to prevent unwanted macro executions that could interfere with the operation of the PC1600.
2. **Macro Execution**: The procedure after the macro command \31\ is straightforward. Once the macro that started with \31\ is completed, it will end. If the final macro command restarts DW and you want to run another macro, it is possible to link and nest macros. However, this should be done with caution to avoid complex and potentially problematic configurations.
3. **Avoid Infinite Loops**: Be careful not to create infinite loops within your macros. Such loops can lead to situations where only a reset can stop the loop. For example, the following infinite loop:

   ```
   Q \31\BLOAD"S2:DW~Q
   ```

4. **Duplicate Macro Keys**: Assigning a macro key multiple times (double or triple assignment) does not produce negative side effects, except that the subsequent definitions will be ignored. Once the macro interpreter finds the first macro assigned to a key, it will execute it, disregarding any additional macros assigned to the same key.
5. **Error Messages**: The error messages in DWMac correspond to those of the Basic interpreter, particularly for file functions. The error numbers are consistent, allowing you to refer to the original computer manual for explanations. The only exceptions are errors 98 and 99, 'Destination = Source' and 'other error!'. The latter indicates an error that cannot be precisely analyzed, requiring the use of logical reasoning to identify the cause.

By keeping these points in mind, you can effectively use macros within DiskWorks, ensuring smooth and efficient operation without unintended interruptions or errors.

```
98 Ziel = Quelle!          Source = Destination
99 sonstiger Fehler!"      Other Error
```

151	Datei gibt's schon	File already exists
152	Datei nicht gefunden!	File not found
154	Datei bereits offen!	File already open
155	Gerät nicht vorhanden	Device not present
157	Falscher Dateiname!	Wrong file name
159	Disk schreibgeschützt!	Disk is write protected
160	Keine Diskette!	No disk
161	Disk nicht formatiert!	Disk not formatted
162	Schreib- Lesefehler!	Read/Write error
163	Falsche Disk (Wechsel)!	Wrong disk (during change)
164	Diskette voll!	Disk full
167	Fataler Fehler!	Fatal error
168	Akkus leer!	Batterie (almost) empty

Final Remarks on DW and the Universe

If you notice any errors, please report them to me. Nobody is perfect (I'm Nobody). You can reach me at crido@gmx.de, among other contact points which are not relevant here ;-)

Background Information: DiskWorks originated from an idea I had in the spring of 1992. In early summer, after a Basic pilot, I began its development. At the 1992 PC-1600 Club Meeting in July, a 'dummy' version was presented, roughly indicating the user interface. At that time, Gernot and Harald were very enthusiastic, which motivated me to continue developing the program.

Since late summer 1992, DW has been in a state where I constantly use it (some others had the "pleasure" of working with error-laden pre-release versions as well). However, the development of version 1.0 was only completed in January 1993. In April, version 1.1 was released, and from late May to early June, version 2.0 was launched. The problem is that I already have a few more ideas on what could be added to DiskWorks.

Overall, the program currently reflects over a year of development time (with many thinking breaks and suggestions from Gernot and Harald – thank you for that).

Thank you for your attention, and have fun with DiskWorks 2.0!

XianBild
July 18, 1993 (originally; today is July 18, 2024, and I have reactivated the PC-1600 and revised this manual). As the software was never considered to be internationally used (1982 was different to today), you will have to live with some German text in the user interface. But it should be easy to translate or understand intuitively.