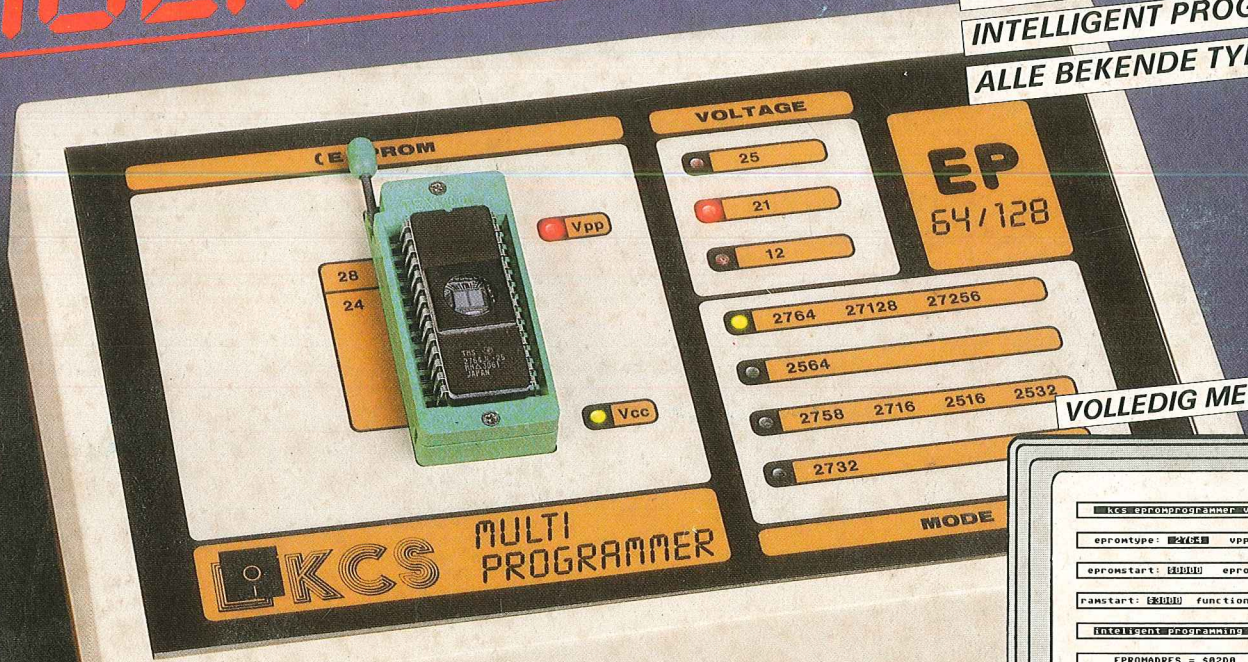
KCS

MULTI-PROGRAMMER

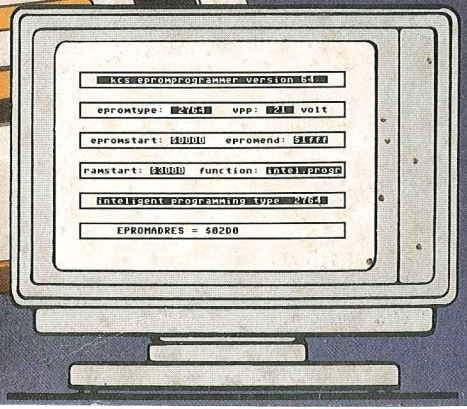
INKLUSIEF SOFTWAREPAKKET

INTELLIGENT PROGRAMMEREN

ALLE BEKENDE TYPEN (E)PROMS



VOLLEDIG MENU GESTUURD!



VOOR UW COMMODORE 64/128

FILE_ID.DIZ - Notepad

File Edit Format View Help

```
Kolff Computer Supplies (KCS) EP64/128 MULTI-PROGRAMMER
Commodore 64/128

Richard Lagendijk (2013)  :: Manual Scan
N.A. Hoijtink     (2013)  :: KCS-PROMMER64.PRG
Ruud Baltissen    (2013)  :: KCS-PROMMER64.2.PRG
```

KCS-EP64.128.d64

```
0 "                    " 2A
0 "                    " DEL
0 "       KCS          " DEL
0 "     MULTI          " DEL
0 "  PROGRAMMER        " DEL
0 "    EP64/128        " DEL
0 "                    " DEL
57 " KCS-PROMMER64     " PRG
0 "                    " DEL
57 " KCS-PROMMER64.2   " PRG
0 "                    " DEL
0 "  BROUGHT TO YOU    " DEL
0 "       BY           " DEL
0 "  N.A. HOIJTINK     " DEL
0 "     IN 2013        " DEL
0 "                    " DEL
550 BLOCKS FREE.
```

Contents

What is an EPROM?

An EPROM is a memory unit (chip) that can retain a program without being powered. Other memory IC ‘s (Integrated Circuits) like RAMs will lose their data as soon as the power is switched off. At the time of this writing the following kinds of memory chips exist:

RAM RANDOM ACCESS MEMORY
This type of memory can be written to and read from. The BASIC or machine code programs that are loaded from tape or disk are stored in this type of memory.

ROM READ ONLY MEMORY
This kind of memory is fixed forever, and can only be read from. Creating and filling ROMs is done with specialized machines and processes by the manufacturer.
It is used to store system software for example, so it is available when we switch the computer on. The Kernal and BASIC interpreter of Commodore computers are stored in this kind of memory unit.

PROM PROGRAMMABLE READ ONLY MEMORY
This kind of memory can be written to ONCE with special hardware, including the EP64/128. Once a PROM is ‘burned’ the contents can’t be erased anymore.

EPROM ERASABLE PROGRAMMABLE READ ONLY MEMORY
This kind of memory can, like a PROM, be written with special programming hardware, but has as an advantage that it can be erased and rewritten.

Erasing is done with UV (Ultra Violet) light. Special EPROM erasers are available for this purpose, but a solarium may also be used. Once an EPROM is erased it can be filled with data (software) again. This has great advantages for the user.

An EPROM can be recognized by a small transparent window on the top. This window is usually taped over after the EPROM has been programmed to keep out light and prevent unwanted loss of data.

The EPROM’s number indicates the memory capacity of the chip:

For example: 2716 = 2K memory.
2732 = 4K memory.
2764 = 8K memory.
27128 = 16K memory.
27256 = 32K memory.

1K is 1024 bytes of memory, or 1024 addresses where data can be stored.

How does it work.

The connection pins on an EPROM can be divided in 4 categories:

ADDRESS LINES	depending on its capacity an EPROM has 10, 11, 12, 13 or 14 address lines so it can address 1K, 2K, 4K, 8K or 16K memory locations.
DATA LINES	on these lines the contents of the memory locations is available.
POWER LINES	two: one for 5V and one for GND.
CONTROL LINES	a programming voltage line (Vpp) and two select lines, CE for chip enable and OE for output enable.

When a specific address in the chip must be read, the required address is put on the address lines and both CE and OE are pulled low. The data bits can then be read from the data lines (D0-D7).

When a specific address in the chip must be written to, the destination address is put on the address lines, the data to be written is put on the data lines. Depending on the type of EPROM used a programming voltage of 12, 21 or 25V is applied to the Vpp pin and with a programming pulse of 50 milliseconds on CE (or PGM) the data is then 'burned' in the EPROM. This process is repeated for each address that needs to be programmed.

The 50ms duration of the programming pulse is advised by the EPROM -manufacturers, but in practice a few ms suffice for programming. Instead of one long 50ms pulse a few short ones can be applied. When the data is not written correctly a new pulse can be given. This process repeats until the data is written or the total pulse duration of 50ms is reached.

This approach is known as intelligent programming and yields huge time savings compared to traditional EPROM -programming. Not all EPROMs can be programmed this way!

The EP64/128 can be used both as a 'traditional' and as an 'intelligent' programmer.

In an empty EPROM all data bits contain a '1' (high): all addresses contain $FF. During programming the '1' bits are converted to '0' bits where necessary.

Installing the EP64/128

Turn off the computer and insert the connector, with the KCS logo facing UP, into the USER port (on the back-left side of the computer). Now turn the computer back on.

Several LED's on the panel will be switched on. Before the device can be used the operating software needs to be loaded from tape. Both sides of the tape contain a copy of the software. Type in this command to load the software:

```
LOAD "EPROMSOFT"
```

When it has finished loading, the program can be started by issuing the RUN command.

The program consists of 2 parts:

A- Menu-driven main program
B- Command driven Monitor program

The software starts in the main program and the following menu appears:

The main program is menu-driven, all changes in the menu are applied to the programmer. Both the screen and the programmer will show what type of EPROM is selected. As long as no command for programming, verifying, reading or empty testing is given the Textool-Socket is not powered. This can be verified with the Vpp and Vcc LEDs. Only when BOTH are OFF an EPROM may be inserted or removed from the socket.

When inserting an EPROM the socket's lever must be in the upright position. The EPROM may then be placed in the socket with the notch facing upward (away) and the lever can be pushed down.

MAIN PROGRAM OPERATION

In the EP64/128's control software the C64's function keys do the following:

F1 key: change EPROM type
F2 key: change programming voltage
F3 key: change EPROM start address
F4 key: change EPROM end address
F5 key: change RAM start address
F6 key: change programmer's function
F7 key: start chosen programmer's function
F8 key: invoke monitor program

Pressing the required F-key (no RETURN needed!) activates its function. The bottom 2 lines of the menu-screen are used to show error messages or instructions.

F1: EPROM type selection

After pressing the F1 key the CRSR left and right keys will allow you to choose between the following types of EPROMs:

TYPE	Vpp	Capacity	End Address	Normal	Intelligent
2758	25V	1K	$03FF	Yes	No*
2516	25V	2K	$07FF	Yes	No*
2716	25V	2K	$07FF	Yes	No*
2532	25V	4K	$0FFF	Yes	No*
2732	25V	4K	$0FFF	Yes	No*
2564	25V	8K	$1FFF	Yes	No*
2532a	21V	4K	$0FFF	Yes	No*
2732a	21V	4K	$0FFF	Yes	No*
2764	21V	8K	$1FFF	Yes	Yes
27128	21V	16K	$3FFF	Yes	Yes
2764a	12V	8K	$1FFF	No	Yes
27128a	12V	16K	$3FFF	No	Yes
27256	12V	32K	$7FFF	No	Yes

While selecting the EPROM type with the cursor keys the corresponding programming voltage and end addresses will be adjusted as well. This can be verified with the LED indicators on the programmer. By pressing RETURN the programmer will be configured for the chosen EPROM type.

*remark: Even though the manufacturer states that these EPROM can't be programmed intelligently, practice shows they actually can! The latest generation EPROMs showed good results. When in doubt pick 'Normal' programming. EPROM types 2764a, 27128a and 27256 can only be programmed intelligently.

F2: Set programming voltage

After pressing the F2 key, the cursor keys can be used to select the PROGRAMMING VOLTAGE. This may be necessary if the manufacturer recommends a different programming voltage. After selecting the voltage pressing RETURN will configure the programmer to the selected voltage.

F3: Set EPROM start address

After selecting F3, the keys 0-9 and A-F can be used to enter the 4 character hexadecimal EPROM START ADDRESS. The lowest address value is $0000, the highest address depends on the type of EPROM that was selected. When the end address is too high this will be reported in the bottom 2 lines of the screen. Corrections can be made with the DEL key. To confirm the entered address, press RETURN.
When the RETURN key is pressed immediately after F3, no changes will take place.

F4: Set EPROM end address

After selecting F4, the EPROM END ADDRESS can be changed. The highest valid end address depends on the type of EPROM that was selected. See F3 key for how to enter an address. Errors are shown in the bottom 2 lines of the screen.

F5: RAM START ADDRESS:

After pressing F5 the RAM START ADDRESS can be changed. The lowest possible address is $3000. The highest Ram address depends on the number of bytes the will be read or written. The highest RAM address can be calculated as follows:
RAM START(max) = $C000 – number of bytes to read/write.
Example:

Bytes to program: $ 4000 (for a xx128 EPROM)
RAM START(max) = $C000-$4000 = $8000

F6: EPROM access method

After pressing F6, the cursor keys offer a choice between the following functions:

READ	Reading data from the EPROM, starting at EPROM START and ending at EPROM END addresses. The values will be stored in the computer from the RAM START address.
EMPTY TEST	Performs a test to check that the EPROM is empty from the EPROM START to EPROM END addresses. If a byte is found that is not equal to $FF an error message is displayed.
COMPARE	Compares the contents of the EPROM from the EPROM START to EPROM END addresses with what is stored in the computer memory from the RAM START address. Differences will be displayed. After 20 mismatches the process will pause. By pressing SPACE the process continues, pressing the back arrow (upper left of the keyboard) will abort the process and return to the menu.
NORM.PROGR	The EPROM memory bytes from the EPROM START to EPROM END addresses will be programmed with the values from the computer memory starting at RAM START. Each byte will be written with a 50ms programming pulse. After programming an automatic COMPARE will be done, and any differences found will be printed on screen.
INTEL.PROGR	The EPROM memory bytes from the EPROM START to EPROM END addresses will be programmed with the values from the computer memory starting at RAM START. Instead of 50ms pulses a much shorter pulse will be used. After each short pulse the programmed value will be checked. If the value doesn't match, another pulse is applied, until the value is correct or the maximum number of pulses is reached. The maximum number depends on the programming voltage. After all addresses are programmed a COMPARE will be done and any differences will be printed on screen.

F7: Execute EPROM command

This key may only be pressed when an EPROM has been properly inserted in the socket (facing the right way). Pressing the button will start the chosen programming process. Any error s will be displayed on the bottom 2 lines of the menu screen. Always check all entered values, both on screen and by checking the LEDs on the programmer. Wrong settings may damage the EPROM!

F8: THE MACHINE CODE MONITOR

By pressing F8 the monitor program is started. It resides in memory locations $C000-CFFF. The monitor command 'X' will quit the monitor and return to the menu.

	MONITOR COMMANDS	EXAMPLE
@	ASCII text assembler	@ 4000 'KCS
A	Assembler	A 2000 A9 12 LDA #$12
B	Break Set	B 1000 0075
C	Compare Memory	C 1000 2000 C000
D	Disassembler	D 2000
E	Examine Memory	E C000 D000 20 E3 FF
F	Fill Memory	F 1000 2000 FF
G	Go Run	G 1000
H	Hunt Memory	H C000 D000 20 E3 FF
I	Interpret Memory	I C000
J	Jump to subroutine	J 1000
L	Load from device	L or L "filename" 08 3000
M	Mask Set	M or M FF 00 F0
N	New Locate	N 7000 8000 2000 5000 5FFF,W
O	Output Enable	O
P	Printer Enable	P
Q	Quit Trace	Q or Q 1000
R	Register Display	R
S	Save to device	S "filename" 08 4000 5000
T	Transfer Memory	T 1000 2000 5000
V	Verify Memory	V or V "filename" 08 4000
W	Walk mode	W or W 1000
X	Return to main menu	X
#	Decimal to Hex conversion	# 32768
$	Hex to decimal conversion	$ C000

THE MONITOR

The monitor knows a large number of commands (see table on the previous page).

Besides those commands the monitor also supports the following built-in abbreviated dos commands:

Disk Commands	Explanation
	Disk Status
$	Directory
$0	Directory drive 0
$1:M	Directory drive 1, files starting with an M
V0	Validate disk in drive 0
R0:HELLO=HITHERE	Rename file on drive 0
S0:PR*	Scratch all files starting with PR*
N1:DISKNAME,ID	Format disk in drive 1, with name and ID
C1:*=:HELLO	Copy file from drive 0 to 1
D1=0	Backup drive 0 to drive 1
#:PROGRAMNAME	Show start and end address of program

MONITOR COMMAND DESCRIPTION

.@ ASCII TEXT ASSEMBLER

```
.@ 3000 4B 43 53 20 50 52 4F 2D 'KCS PRO-
.@ 3008 47 52 41 4D 4D 45 52 20 'GRAMMER
```

This command can be used to enter characters in memory without having to calculate their hex representation. The quote character indicates the start of the text and no more than 8 characters can be entered on one line. By pressing RETURN without typing text the monitor exits text assembly mode.

.A ASSEMBLER

```
.A 3000 LDA #$20
.A 3000 A9 20    LDA #$20
```

After entering the first line and pressing RETURN it will be overwritten and look like the 2nd line. The following line will have the correct next Hex address filled in.

.B BREAK SET

```
.B 2000 7F
```

This command will set a pseudo-breakpoint for the 127th time the instruction at memory location $2000. This function is only active during the Quick Trace command.

.C COMPARE MEMORY

```
.C 1000 2000 C000
```

Will compare memory locations $1000-$2000 with those starting at $C000. Memory locations with different values will be displayed.

.D DISASSEMBLER

```
.D 4000
., 4000 A9 20    LDA #$20
., 4002 9D 00 60 STA $6000
., 4005 AA       TAX
```

Machine language from memory locations $4000 up to $FFFF will be disassembled by this command. Pressing RUN/STOP will stop the disassembly process. Hitting SPACE will pause disassembly until SPACE is pressed again.

.E EXAMINE MEMORY

```
.E C000 D000 20 E3 FF
```

The monitor will read 3 bytes in the memory range C000-D000 and perform a logical and operation on those bytes with the first 3 bytes

[ed: text is missing here in the original, including the '.F' command, the F command explanation was added by the translator. For more, albeit sketchy, info about .E see the .M command]

.F FILL MEMORY

```
.F 1000 2000 FF
```

This command will fill memory locations $100-$2000 with the value $FF.

.G GO

```
.G
```

This command start execution of the program stored at the current program counter. Execution will terminate when a BRK instruction is encountered.

```
.G 1000
```

This command will start execution of the program from address $1000. Execution will terminate when a BRK instruction is encountered.

.H HUNT MEMORY (search)

```
.H C000 D000 'KCS
```

The monitor will search memory from $C000 to $D000 for the ASCII string 'KCS' and will print the address at which it was found. 0Max 20 characters are allowed as search string.

```
.H C000 D000
```

The monitor will search memory from $C000 to $D000 for the byte values specified, in this case 20 E3 FF, and will print the addresses at which it was found.

RUN/STOP will break the Hunt command.

.I INTERPRET MEMORY

```
.I A0C0
```

This example shows the Hex representation and corresponding ASCII text from memory location $A0C0 up to the end of memory ($FFFF). Pressing RUN/STOP will terminate the process. Hitting SPACE will pause the output until SPACE is pressed again.

.J JUMP TO SUBROUTINE

```
.J
```

This command will cause execution of the subroutine stored at the current program counter. Execution will terminate when a RTS instruction is encountered.

```
.J 3000
```

This command will cause execution of the subroutine at address $3000. Execution will terminate when a RTS instruction is encountered.

.L LOAD

```
.L “TEST” 01 or .L “TEST” 08
```

Loads the program with name “TEST” from cassette (01) or disk (08).

```
.L “TEST” 08 4000
```

Loads the program with name “TEST” from disk at address $4000.

.M MASK SET

```
.M
```

This command will show the bytes that are currently stored in the mask buffer. The mask buffer is used by the .E command.

```
.M FF 00 FF
```

This command stores the bytes specified after the .M command in the mask buffer. Afterwards, when the memory is examined with this command:

```
 .E 8000 9000 20 00 E0
```

All addresses in the memory range $8000-$9000 that contain a JSR to the area $E000-$FFFF will be displayed.

.N NEW LOCATE

```
.N 7000 7FFF 2000 5000 5FFF
```

This command will modify all 3 byte instructions stored in the memory area $7000-$7FFF that reference memory area $5000-$5FFF by adding $2000 to the target addresses.

.O OUTPUT ENABLE

```
.O
```

This command empties the printerbuffer by sending a carriage return, then closes the printfile and returns output to the screen only.

.P PRINTER ENABLE

```
.P
```

This command opens a file to device 4, so all output is sent to the printer, as well as to the screen.

.Q QUICK TRACE

```
.Q
```

This command will start the execution of the program stored at the current program counter. Execution is checked by the monitor and will terminate on an error or when a BRK instruction is encountered.

```
.Q 1000
```

This command will start execution of the program stored at address $1000. Execution is checked by the monitor and will terminate when an error is found or a BRK instruction is encountered.

Output of the Q command can be stopped by pressing RUN/STOP, which will switch the monitor into WALK MODE.

.R REGISTER DISPLAY

```
.R
```

Displays the contents of the internal CPU registers:

```
   PC   SR   AC   XR   YR   SP
.;8BCE 30 00 04 07 FF
```

Where:

PC = PROGRAM COUNTER	XR = X-REGISTER
SR = STATUS REGISTER	YR = Y-REGISTER
AC = ACCUMULATOR	SP = STACK POINTER

.S SAVE

```
.S “TEST” 01 4000 5000
```

Will write the contents of memory locations $4000-$5000 to cassette under the filename “TEST”. Replacing 01 with 08 will save the file to disk.

.T TRANSFER MEMORY

```
.T 1000 2000 8000
```

Copies the contents of memory area $1000-$2000 to address $8000.

.V VERIFY

```
.V “TEST” 01 or .V “TEST” 08
```

Compares the program named “TEST” from cassette (01) or disk (08) with the program in memory.

```
.V “TEST” 08 4000
```

Compares the program with name “TEST” from disk with the program in memory, starting at address $4000.

.W WALK MODE

```
.W 1000
```

Starts and walks through the machine language program from address $1000. When an illegal opcode or a BRK instruction is encountered execution will stop and the .R (show registers) command is executed. The walking speed is controlled with the SPACE bar. RUN/STOP will terminate WALK MODE.

.X EXIT MONITOR

```
.X
```

Will exit the monitor and return to the main menu of the program.

.# DECIMAL TO HEX CONVERSION

```
.#49152 will return $C000
```

.$ HEX TO DECIMAL CONVERSION

```
.$C000 will return 49152
```

GUARANTEE

Legal mumbo jumbo.

FINAL REMARKS

Programming EPROMs is done at your own risk. We advise you to check with your retailer for correct programming voltages of your EPROMs.

EPROMs and Circuit boards.

Without PCBs or EPROMs an EPROM programmer is of little use. L&S Electronics in Delft can inform you about this.

More information:

Phone: 015-618139 or 01738-9749 (both after 18:00)

Address:

L&S Electronics
Bosboom-Toussaintplein 65
2624 DG DELFT.