COMX-35

Superboard V1.2

(FW Build 200 – PC SW build 91)

Manual

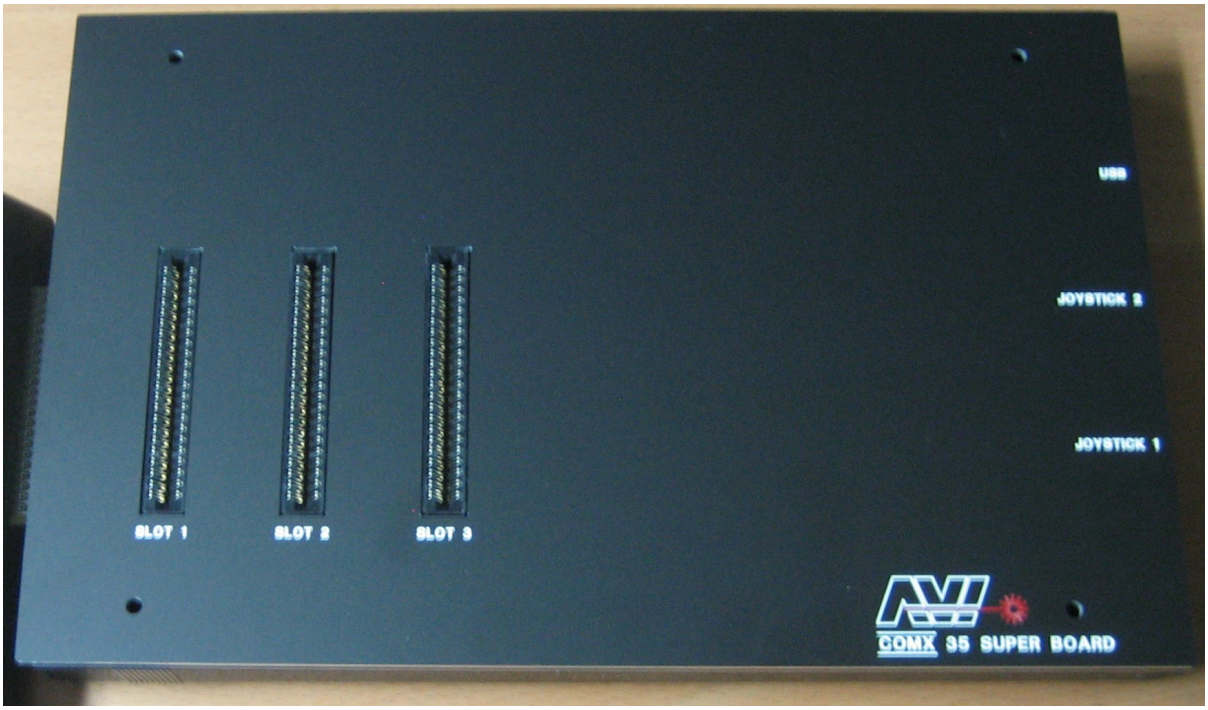

Table of contents

COMX USB Interface – PC Software

Installation and set-up

Before installation of the USB SW first connect the SB to the COMX and the SB via a USB cable to the PC. This should install the device driver software. If windows cannot find the drivers automatically they can be downloaded from: http://www.ftdichip.com/FTDrivers.htm (download VCP Drivers?).

After installation of the USB SW start it via the start menu (note that the next time your PC is started the USB SW will start automatically), this will add an icon in the taskbar that looks like:

This means the USB SW is running but no connection is found to the COMX as yet.

To get the connection working first identify the serial port used by the driver SW. To do this: start the Control Panel, select Hardware and Sound and then the Device Manager. Search for the entry Ports (COM & LPT) and take note of the port number indicated as 'USB Serial Port', this value is needed to be filled in the settings dialog.

To open the settings window, go to the icon in the taskbar and select 'Setting…' with the right mouse button. Now fill in the port number as 'COM Port' and make sure the 'Speed' is set to the same value as is selected by the jumper on the SB board (default 57600).

Press 'Save'

The taskbar icon should now change to:

Note that the icon will change back to the yellow warning sign as soon as the COMX is switched off. When the COMX is switched back on it should switch back to the green icon within a few seconds.

If in some cases the green icon does not appear, the following could help: open the Device Manager via the control panel and select 'Scan for hardware changes'.

About

To open the about window: go to the icon in the taskbar and select 'About…' with the right mouse button. This will show the COMX USB Interface version and build number as well as the FW build number (if available), copyright message, link to COMX SB web site, special thanks and license information.

USB Settings

To open the settings window: go to the icon in the taskbar and select 'USB settings…' with the right mouse button. Note that when settings are saved by pressing the 'Save' button the USB interface is restarted and the current directory on the COMX will also be reset to the 'Root' directory.

Other settings:

Root – Either use the ‘Root’ button or change the path manually to change the root directory seen from the COMX. The selected root is the highest directory level that can be accessed by the COMX. This is either shown as directory \ or \top-directory on the COMX (depending on ‘Show to directory in USB CD’ setting)

COM Port – Specify the COM Port as used by the ‘USB Serial Port’ found in the Device Manager.

Speed – 57600 or 115200 baud. When changed make sure the jumper on the SB board is changed accordingly

Show top directory in USB CD – Select Yes / No, ‘No’ will show the root directory as \, ‘Yes’ will show the root directory as \top-directory (i.e. \Comx when the default root folder is used).

Case conversion – These settings are only used if USB CHAR(0), normal COMX character set or USB CHAR(1), lower case ‘default’ standard COMX characters on ‘SHIFT’ are selected. USB CHAR(0) is set as default. For these character sets the USB SW will convert any filename or directory name as specified: in lower case, upper case or capitalized. Default all filenames will be in lower case and directory names capitalized. When USB CHAR(2) or (3) are used the capitalization is as typed on the COMX.

Back-up – Select Yes / No, ‘Yes’ will result in creation of a back-up file for the specified commands with the format ‘nameNumber.type.bak’. So when saving a file ‘happiehap.comx’ the first back-up file created will be named happiehap1.comx.bak, second happiehap2.comx.bak etc. ‘No’ will result in a question on the COMX to overwrite the file Y/N when the file already exist on the PC.

Save – Save current settings

Online Settings

To open the online settings window: go to the icon in the taskbar and select ‘Online settings…’ with the right mouse button.

Define the Player name and location as used in the USB HSWEB command. Note that the maximum length of both player name and location is 25 characters and the following characters are not allowed and will be ignored: # ~ ; :.

To be able to get your email address specified as sender with USB EMAIL make sure to fill in a correct and valid email in the ‘Email address’ field

Home page and bookmark fields are used by the USB BROWSER command and can be shown on the COMX by pressing the ‘H’ or 0 to 9 keys during a browser session. Max length of these fields is 126 characters. To store a bookmark press the ‘Store’ button.

Email address and aliases are used by the USB EMAIL command. Any defined alias can be edited when selected by the Aliases choice button. A new alias can be stored when ‘alias / email address’ is shown. To store the alias press the ‘Store’ button.

Delete back-up

To open the delete back-up window: go to the icon in the taskbar and select ‘Delete back-up…’ with the right mouse button. This will open the Delete back-up window, showing all back-up (.bak) files in the specified root directory and sub directories. Pressing ‘Yes’ will delete all these files ‘No’ will just close the window.

PC Disk Access

[STR=] USB PLOAD,"filename" [,start] [,S] [,R] [,start,B] [,Bx]

Load any program .comx file into COMX RAM memory. When optional parameter 'R' is used the loaded program will automatically be started with a RUN or CALL command.

Default BASIC .comx file content will be loaded starting at address @4400. If however a 'start' value is used the content will be loaded starting at the specified address (rounded to lower 256 bytes). So 'start' value @4560 will result in a load to @4500. The 'start' value can be any 16 bit hexadecimal or decimal value as well as a variable expression. Note however that a variable expression starting with 'R', 'S' and 'B' is not allowed.

Machine code .comx files (saved with 'start' and 'end' address) will always be loaded at the same address used during the save command.

To load a binary (.bin) file specify optional parameter B and the start address.

Optional parameter 'Bx' will execute a BLOAD command loading the specified .ram file to bank x or all banks (BA) after the .comx or .bin file is loaded. So using parameter 'Bx' will result in loading of two files first the .comx or .bin file and then the .ram file, both with the same filename. If the first file is not found the loading will be stopped and a 'file not found' error will be shown. If the second file is not found the first file will already have been loaded and a 'RAM bank file not found' will be shown.

Optional parameter S will show:

- The start, end and exec addresses if a pure assembler (ASM) file is loaded
- The start and end addresses if a binary (BIN) file is loaded
- The start address if an assembler & BASIC (A&B) file is loaded
- 'Basic only' if a BASIC (BAS) file is loaded

USB PLOAD can also be used to load directly to the SB RAM banks or a COMX 32K RAM card. To do so select the desired RAM bank (CARD Bx) and the use a USB PLOAD command. The program to be loaded needs to be able to RUN in @C000-@DFFF range. Also the program should have been saved from @C000 or loaded with a start address of @C000.

NOTE: USB PLOAD with option R will not work when loading directly to a RAM card as the SB FW will be switched in before execution of the RUN or CALL command.

If optional parameter STR is specified the return string will contain the possible error code: 'File not found in directory' or 'RAM bank file not found' if the specified file or RAM bank file (option Bx) is not found. In this case any running program will not 'break' and the user program can handle the error code. If the file is found an empty string will be returned.

USB PSAVE,"filename" [,start] [,start,end [,B] [,D]] [start,end,exec] [,Bx]

This will save the current program to a .comx file. Any BASIC program will be saved as well as any assembler code before DEFUS address. When F&M BASIC 2.0 is loaded this command will only save BASIC (from address @6700).

When only optional parameter 'start' is specified the current program in memory is only saved from specified address. This can be used for saving only the BASIC part of a program by using the DEFUS value as start address.

When optional parameters 'start' and 'end' are used the specified part of COMX memory between 'start' and 'end' address will be saved to a .comx file. Optionally 'exec' can be specified as well for the execution address. The execution address will be used when a USB PLOAD with parameter 'R' is given to start the machine code program at the correct address.

Parameter ‘B’ can be used to save the specified part of COMX memory from ‘start’ to ‘end’ address to a .bin file.

Optional parameter ‘Bx’ will execute a BSAVE command saving content of bank x or all banks (BA) to the specified .ram file after the .comx or .bin file is saved. So using parameter ‘Bx’ will result in saving of two files first a .comx or .bin file then the .ram file, both with the same filename.

Parameter ‘D’ can be used to save (or dump) the specified part of COMX memory from ‘start’ to ‘end’ address to a .txt file.

Start, end and exec values can be any 16 bit hexadecimal or decimal value as well as a variable expression. Note however that a variable expression starting with ‘B’ and ‘D’ is not allowed.

USB PSAVE can also be used to save directly from any selected expansion card. To do so select the desired cards (CARD Sx) and the use a USB PSAVE command with a range of @C000-@DFFF.

[STR=] USB DLOAD,”filename”

Load data area for string and arrays from a .comx file.

If optional parameter STR is specified the return string will contain the possible error code: ‘File not found in directory’ if the specified file name is not found. In this case any running program will not ‘break’ and the user program can handle the error code. If the file is found an empty string will be returned.

USB DSAVE,”filename”

Save data area for string and arrays to a .comx file.

[STR=] USB BLOAD,”filename” [,Bx]

Load data from indicated .ram file directly to (RAM) bank x, where x can be 8 to 15. In this case max 8KB is loaded from the .ram file.

If x=A is used (or Bx is omitted) all RAM banks will be loaded (max 8*8K = 64K).

If optional parameter STR is specified the return string will contain the possible error code: ‘RAM bank file not found’ if the specified file is not found. In this case any running program will not ‘break’ and the user program can handle the error code. If the file is found an empty string will be returned.

USB BSAVE,”filename” [,Bx]

Save data to indicated .ram file directly from (RAM) bank x, where x can be 8 to 15. In this case 8KB is saved to the .ram file.

If x=A is used (or Bx is omitted) all RAM banks will be saved (resulting in a 8*8K = 64K file).

[STR=] USB DEL,"filename" [,A] [,B] [,I] [,R] [,T] [,X]

Delete 'filename'.

If no additional parameter is given the file with filename.comx will be deleted. If other file types need to be deleted the following parameters can be used:

A. All 5 defined file types: Emma 02 (.comx), binary (.bin), image (.img), RAM bank (.ram) and text (.txt) files
B. Binary files (.bin)
I. Image files (.img)
R. RAM bank files (.ram)
T. Text files (.txt)
X. Delete all back-up files with extension .comx or with extension as indicated with A, B, I, R or T parameters.

If optional parameter STR is specified the return string will contain possible error code: 'Could not delete file' if the specified file name is not found or an error caused the file not to be deleted. In this case any running program will not 'break' and the user program can handle the error code. If the file is found an empty string will be returned.

INT = USB COMP("filename1","filename2" [,B] [,I] [,R] [,T])

Compare filename1 with filename 2, if the files are identical 0 is returned if not the location of the first difference is returned.

If no additional parameter is given files with filename1.comx and filename2.comx will be compared. If other file types need to be compared one of the following parameters can be used:
B: Binary files (.bin)
I: Image files (.img)
R: RAM bank files (.ram)
T: Text files (.txt)

[STR=] USB CD [,"dirname"]

Change to sub directory 'dirname', also return the path from the PC (if the path is >127 characters, just the last 127 are printed). If no directory name is specified the PC path is returned without any change. Directory name ".." will go up one directory level. Note that only one directory name can be given per USB CD command.

Examples:
A$=USB CD
-> Assign current path to A$

USB CD,"Games"
-> Change directory to directory 'Games' if it is available

PR USB CD,"Games"
-> Change directory to directory 'Games' if it is available and print full path

USB MKDIR,"dirname"

Create sub directory 'dirname'

USB RMDIR,"dirname"

Remove sub directory 'dirname'

USB CAT [/W]

Show current directory content (or catalog to use COMX terminology). USB CAT will use the COMX DOS format. USB CAT/W will show a condensed format with 2 filenames per line and all directories marked with '*'

RAM Bank Access

USB RLOAD (bank)

Load program stored in (RAM) bank 8 to 14 to main RAM. Program to be loaded will be stored in main RAM location 4400h to 63F8h.

USB RLOAD/RSAVE/OLD can be used to design a program using > 32K as follows:

1. Main RAM area 4400h - 63FFh (8K) to be used as 'swap' area.
2. Main RAM area 6400h – end main RAM (~B6xx) to be used for the main program. This is about 20K.
3. Bank 8 to 14 to be used to swap to 4400h – 63FFh area, this is 7*8K which is 56K. In theory allowing a 56K + 20K = 76K program.

Example:

Write the following program in main RAM:

```
DEFUS @4400
10 PRINT "BANK 8"
20 USB OLD,@6400
30 GOTO 40: REM Continue main program on line 40
USB RSAVE(8)

DEFUS @4400
10 PRINT "BANK 9"
20 USB OLD,@6400
30 GOTO 10: REM Continue main program on line 10
USB RSAVE(9)

DEFUS @6400
10 PRINT "MAIN"
20 USB RLOAD(8)
30 GOTO 10: REM Start program swapped in from bank 8 on line 10
40 PRINT "MAIN"
50 USB RLOAD(9)
60 GOTO 10: REM Start program swapped in from bank 9 on line 10
```

This stores 2 small programs in bank 8 and 9. The 3rd and main program starting at 6400h will load the program from bank 8 and start it at line 10, then a jump will be made back to the main program at line 40 which will load the program from bank 9 etc.

Note 1: after an USB RLOAD the program loaded at 4400h area HAS TO be continued, the main program can only continue after a USB OLD,@6400.

Note 2: Before any string or array variable is set the main program needs to have been started. Any sting or array variable that is assigned in the program in the swap area BEFORE the main program has executed will be erased when the main program starts execution. This will NOT be the case after first execution as long as the program is not 'ended' or stopped with 'ESC'.

USB RSAVE (bank)

Save current program in main RAM to RAM bank 8 to 14. Program to be saved can be max 8K large and should be stored in main RAM location 4400h to 63F8h.

See USB RLOAD for an example of how USB RLOAD/RSAVE can be used to design a program using more than 32K.

USB MOVE (start, end, destination [,force [destination slot, destination bank]])

Move / copy memory part from start to end to the requested destination. Source and destination can overlap.

If optional force parameter is omitted or 0 a warning will be given on any move command with a destination overlap to ROM or unsafe RAM areas (4000-43FFh, BF00-BFFFh and stack area). With force parameter <> 0 the move command will be executed without any warning.

Source location (if between C000h-DFFFh) is always from the current selected slot and bank (as selected via CARD Bx), destination slot and bank can optionally be specified in the command. Note a copy to C000h-DFFFh will always give a warning unless force = 1 is specified.

Clock

USB CLOCK (x)

x=0: Remove and stop the clock at top right of screen
x=1: Show clock at top right of screen, when program is running (after a RUN, CALL, USR or USB PLOAD,"filename",R, DOS RUN) the clock will not be shown.
x=2: Always show clock at top right of screen even when a program is running (some SW might turn the clock of if the SW has their own interrupt routine).
x=3: Same as 0

USB RTCFT (x)

Set RTC FT to x (0 or 1), for details see M48T58 data sheet.

Any other uneven value of x will have the same result as 1, even same as 0.

USB RTCPC

Fetch PC date and time and set RTC accordingly. This command will also fetch the date and time (24H or 12H) format from the PC and store it into NVRAM for use with USB DATE and USB TIME commands.

STR = USB DATE

Read the current date and return it to specified string variable or print it. Resulting string is formatted according to the PC format as received by USB RTCPC.

USB DATE,"xx/yy/zz"

Set date according to specified string value, string value should be formatted according to the PC date format as received in USB RTCPC. Note that the separator is also fetched from the PC and could for example be a '/' or '-'. A PR USB DATE will show the current separator format.

Note that an error code will be given if an incorrect date is specified. The command will check on format, invalid month (>12) and invalid day (depending on month and/or leap year).

STR = USB TIME

Read the current time and return it to specified string variable or print it. Resulting string is formatted according to the PC format as received by USB RTCPC, i.e. if the PC is using a 24 hour format the TIME command will do the same, if the PC uses a 12 hour format the TIME command with do the same and add AM or PM to the end of the string.

USB TIME,"hh:mm:ss"

USB TIMEAM,"hh:mm:ss"

USB TIMEPM,"hh:mm:ss"

Set time according to specified string value. Use TIMEAM to set the time in AM format, TIMEPM for PM format. All 3 commands will work independently of set clock type.

COMX Configuration

USB LINE

Deactivate F&M Screen Editor and activate COMX Line Editor

USB SCREEN

Activate F&M Screen Editor and clear screen

USB SCREEN (x)

Set COMX BASIC current and startup background screen color, where x is defined as:

1: Black
2: Green
3: Blue
4: Cyan
5: Red
6: Yellow
7: Magenta
8: White

Original COMX BASIC command SCREEN(x) still changes the current background color as before. However the value is not saved in NVRAM, so after a restart or power off/on the background color as defined with USB SCREEN (x) will be used as start-up color. Default startup color is black.

USB COLOR (x)

Set COMX BASIC current and startup character color, where x is defined as:

	Normal text / <comxcharlow>	**URL links / <comxcharhigh>**
1	Black	Green
2	Red	Yellow
3	Blue	Cyan
4	Magenta	White
5	Black	Blue
6	Red	Magenta
7	Green	Cyan
8	Yellow	White
9	Black	Red
10	Blue	Magenta
11	Green	Yellow
12	Cyan	White

Original COMX BASIC command COLOR(x) still changes the current character color as before. However the value is not saved in NVRAM, so after a restart or power off/on the character color as defined with USB COLOR (x) will be used as start-up color. Default startup color is 12, cyan/white.

USB CTONE (x)

Set COMX BASIC current and startup character color mode, where x is defined as:

0: Normal color mode
1: Black and white mode

Original COMX BASIC command CTONE(x) still changes the current character color mode as before. However the value is not saved in NVRAM, so after a restart or power off/on the character color mode as defined with USB CTONE (x) will be used as start-up color mode. Default startup color mode is 0.

USB CHAR [(x)]

x = 0 to 3: save 'x' in NVRAM
x = 4 to 7: do NOT save 'x' in NVRAM

Shape character set:
x = 0 or 4: Standard COMX characters (upper case 'default')
x = 1 or 5: Lower case 'default' standard COMX characters on 'SHIFT'
x = 2 or 6: Upper case 'default' and lower case characters on 'SHIFT'
x = 3 or 7: Lower case 'default' and upper case characters on 'SHIFT'

For value 0 to 3 the selected character set will be re-shaped at the next COMX boot if NVRAM is available.

When no (x) is specified USB CHAR will re-shape the last selected character set.

Every USB CHAR will also re-shape ALL other default COMX characters.

Note that after a COLOR(x) only part of the COMX character set is re-colored (character numbers 33 until 96), this means that characters 'on shift' (i.e. lower case in USB CHAR(2)) will not be re-colored. SB FW is not adapted to change the COLOR command as this would cause some issues with existing software. To be able to get all characters in the right 'COLOR' use a USB CHAR command after the COLOR command. USB CHAR will re-shape all characters and will re-color character numbers 33 until 127 as defined in the latest given COLOR command.

USB FLASH (x)

The COMX BASIC switches the screen output off during every SHAPE, COLOR and USB CHAR command. Since FW build 196 this has been removed. However as it changes COMX behavior (look and feel of some SW) slightly it can be switched back on by this command. Latest state is saved in NVRAM.

Switch screen flash on (x=1) or off (x=0).

Any other uneven value of x will have the same result as 1, even same as 0.

USB LOGOTUNE (x)

Switch COMX boot tune on (x=1) or off (x=0)

Any other uneven value of x will have the same result as 1, even same as 0.

USB BOOTMSG (x)

Switch warm start message on boot screen on (x=1) or off (x=0)

Any other uneven value of x will have the same result as 1, even same as 0.

USB GO40

Switch off 80 column card mode and switch back to normal COMX 40 column video.

If a colon (:) is used after USB GO40 any BASIC commands after the colon will be ignored.

USB GO80 [(x)]

Without option parameter: switch on 80 column card mode.

With (x) specified: switch 80 column card auto start on (x=1) or off (x=0). No start-up screen will be shown on the 80 column output. On a reset a question for warm (W) or cold (any key except W) start-up will be shown.

Any other uneven value of x will have the same result as 1, even same as 0.

VOLUME (x)

Set volume of BASIC sound functions (NOISE, TONE and MUSIC) with a range of 0 to 4. Where 0 is no sound and 4 is the loudest. This value is saved in NVRAM and restored at every power-up.

Note that not all SW makes use of the VOLUME setting and as such this command will not have effect for all SW. For example SW that uses 1802 out instructions to make sound (USB HIT). Also SW that on start-up use the VOLUME command will off course change this setting.

USB SYSDISP

Show current settings for NVRAM system area.

Note: the printer settings will only be shown if the related card and printer type are connected.

USB SYSDEF

Initiate NVRAM system area to default values:
Boot message: on
Logo tune: on
Screen editor: on
Character set: Standard characters
Clock display: off
Date format: dd/mm/yy
Time format: 24H
80 column auto boot: off
Colors: Screen 1, Color 12, Ctone 0
Volume: 4
Online: On
Flash: Off

Printer TV: off
Printer keyboard: on
Printer CNTL: normal characters
Serial printer:

- Baud rate 600
- 8 data bits
- No parity
- LF
- 2 stop bits
- CR pause 5

Note 1: The screen will be cleared if the screen editor is not active before USB SYSDEF is given.
Note 2: NVRAM Start address is NOT changed or reset, use USB NVSTART instead

NVRAM

USB NVLOAD

Restore NVRAM SW and user data part (F000h to F3E7h) from back-up file on the PC.

Location of the back-up file (nvram.bin) should be the 'root' directory as set in the COMX USB Interface SW on the PC.

USB NVSAVE

Save NVRAM SW and user data part (F000h to F3E7h) to back-up files on the PC.

Location of the back-up file (nvram.bin) will be the 'root' directory as set in the COMX USB Interface SW on the PC.

USB SYSLOAD

Restore NVRAM system area (F3E8h-F3F7h) from back-up file on the PC

Location of the back-up file (system.bin) should be the 'root' directory as set in the COMX USB Interface SW on the PC.

Note 1: The screen will be cleared if the screen editor is not active before USB SYSLOAD is given.

USB SYSSAVE

Save NVRAM system area (F3E8h-F3F7h) to back-up files on the PC.

Location of the back-up file (system.bin) will be the 'root' directory as set in the COMX USB Interface SW on the PC.

USB NVLIST

List content of NVRAM SW area including:

- Current defined user area (if any is defined)
- Used and free SW area
- Stored SW IDs, number of bytes used and SW name if available

Current reserved SW names:

ID	Name
1	Happiehap
2	Happiehap 2
3	Hit & run (USB HIT)
4	Space invaders (USB SPACE)
5	Eat a worm (USB WORM)
6	Get your gadget
7	GateMaster
8	COMX Dragon
9	Crossfire

10 Look before you
11 Tennismania (USB TENNIS)

NVRAM is divided in 4 areas:

F000 – Fxxx: User area, this area can be used by any user or user program. No defined format is available so any address location can be filled as desired.USB NVPOKE can be used to store data and COMX BASIC PEEK can be used to retrieve data. Fxxx is defined via USB NVSTART.

Fxxx – F3E7: SW area, this area can be used by any SW. The area is formatted so every SW program can find stored data back. USB NVPUT, NVGET, NVIGET, NVSGET, NVCLR and NVDEL can be used to store, retrieve or manipulate data to/from this area.

F3E8 – F3F7: System area, reserved for use by the SB FW.

F3F8 – F3FF: Reserved for the RTC.

USB NVPOKE (nvram address,value)

Store value in NVRAM address location, NVRAM address range should be between F000h and F3FFh. This command will first enable NVRAM write then store the value and last reactivate NVRAM write protection again.

[STR =] USB NVPUT (id, source, number of bytes)

Put data in NVRAM 'SW area' for the indicated SW id (1-255). Data that will be put into NVRAM will be from memory location 'source' with a length the of indicated number of bytes.

Returned string will be empty ("") if the command is successful. If NVRAM is full the returned sting will be "NVRAM full, data not stored".

The data stored can be retrieved via USB NVGET.

Possible return stings:
- NVRAM full, data not stored
- No rtc or nvram found, or nvram not inititiated
- Nvram write enable error

If no return string is specified, i.e. USB NVPUT (1, @4500, 4), any error will result in 'breaking' the program and printout of the error code.

Machine code: USB NVPUT can be called from any machine code program by calling subroutine on E019h. Before the call the following registers should be load:

R8.1: id
RC: source
R8.0: number of bytes

The COMX standard SEP 4 / SEP 5 should be used for the subroutine call. Content of registers: R7.0, RE and RF will be destroyed

Example:
```
LDI 86
PHI RC
LDI 00
PLO RC
LDI 0B
```

```
PLO R8
LDI 06
PHI R8
SEP 4   E019
SEP 5
```

This routine will save content of software id 6 from 8600h with a length of bh bytes.

[STR =] USB NVGET (id, destination)

Get data from NVRAM 'SW area' for the indicated SW id (1-255). Data that will be received from NVRAM will be stored starting at memory location 'destination' with a length of the indicated number of bytes.

Returned string will be empty ("") if the command is successful. If the requested SW id is not found the returned sting will be "Software ID not found".

Possible return stings:

- Software ID not found
- No rtc or nvram found, or nvram not inititiated

If no return string is specified, i.e. USB NVGET (1, @4500), any error will result in 'breaking' the program and printout of the error code.

Machine code: USB NVGET can be called from any machine code program by calling subroutine on E01Ch. Before the call the following registers should be load:

R8.1: id
RC: destination

The COMX standard SEP 4 / SEP 5 should be used for the subroutine call. Content of registers: R7.0, RE and RF will be destroyed

Example:

```
LDI 86
PHI RC
LDI 00
PLO RC
LDI 06
PHI R8
SEP 4   E01C
SEP 5
```

This routine will load content of software id 6 to 8600h.

[STR =] USB NVPUT (id, value <INT>, string <STR>)

Put BASIC variable data in NVRAM 'SW area' for the indicated SW id (1-255). The value if the INT variable and the string content will be stored in NVRAM.

Returned string will be empty ("") if the command is successful. If NVRAM is full the returned sting will be "NVRAM full, data not stored".

Example:

```
10 DEFINT A
20 A=12345
30 B$="STORED IN NVRAM"
```

```
40 R$=USB NVPUT(2,A,B$)
```

The above program will store value 12345 and string STORED IN NVRAM for SW id '2'.

The data stored can be retrieved via USB NVIGET (INT value) and USB NVSGET (STR content)

If no return string is specified, i.e. USB NVPUT (1, A, A$), any error will result in 'breaking' the program and printout of the error code.

STR = USB NVSGET (id)

Get string content from NVRAM 'SW area' for the indicated SW id (1-255).

Returned string will be empty ("") if:
- the requested SW id is not found
- no NVRAM is found or NVRAM is not initiated

INT = USB NVIGET (id)

Get INT value from NVRAM 'SW area' for the indicated SW id (1-255).

Returned value will be 0 if:
- the requested SW id is not found
- no NVRAM is found or NVRAM is not initiated

Example:

```
10 DEFINT A
20 A=USB NVIGET(2)
30 B$=USB NVSGET(2)
```

USB NVSTART (start)

Change the NVRAM 'SW area' start address to the specified address. This will mean that the command USB NVPUT, USB NVGET, USB NVSGET and USB NVIGET will use the NVRAM range of 'start' to @F3E7. Start address can range from @F000 (default) to @F3E4 and in steps of 4 bytes only.

Changing the start address will move all stored data to the new address so no data will be lost.

If the newly specified NVRAM size is smaller than the already occupied space the command will not be accepted (error code 'Requested NVRAM size to small to fit current data').

NVRAM space @F000 until start address is available for user data as required. It is recommended that USB NVSTART is never used in any BASIC program to avoid changing NVRAM size and setup for other users.

USB NVCLR

Clear the complete NVRAM 'SW area'.

USB NVDEL (id)

Delete the data for indicated SW id (1-255) from the NVRAM ‘SW area’.

I/O

INT = USB JOY (1) / INT = USB JOY (2)

Read value for joystick 1 or 2 and return it to specified variable or print it.

Any other uneven value of x will have the same result as 1, even same as 0.

INT = USB INP (port)

Read value for specified input port and return it to specified variable or print it. Port value can be 1 to 7.

USB OUT (port, value)

Send value to specified output port. Port value can be 1 to 7, value can be a 8 or 16 bit depending on OUT port specified (port 1, 2 and 3 are 8 bit, 4, 5, 6 and 7 are 16 bit).

USB Q (value)

Set Q=value

INT = USB Q

Read Q value and return it to specified variable or print it.

INT = USB EF (x)

Read specified EF flag value and return it to specified variable or print it. The x value can be 1 to 4.

Internet Access

USB ONLINE (x)

Switch online features on or off, on (x=1) or off (x=0).

Any other uneven value of x will have the same result as 1, even same as 0.

After USB ONLINE (0) the command USB HSWEB, USB URL, URLGET, USB URLDISP and USB BROWSER will not attempt to access any online / internet features, USB ONLINE (1) enables online features again.

Note: USB ONLINE (0) can also be used to disable games to sync to the high score server. Also when using the COMX / SB without connection to a PC, USB ONLINE (0) will eliminate any additional delays when running the build in games.

[STR=] USB URL,"url"

[STR=] USB URL (link number)

Set url string to be used in USB URLGET (screen). When the url command is used with a string the indicated string is used to set the URL address. If a link number is given the link number will used to set the URL address according to the link specified on the current screen. So USB URL (3) will set the URL address to the 3rd link currently displayed on the screen.

If the command is successful the optional return string will contain the requested URL for USB URL,"url" and will be empty ("") for USB URL (link number).

Possible error return strings:
- USB transfer timeoutb
- USB transfer error
- Online features disabled

[STR=] USB URLGET (screen)

Fetch screen number x from the PC based on the previously set url. If there is no data in the requested screen the error code 'No more data found'' will be returned.

Optional return string will be empty ("") if the command is successful. Possible error return strings:
- USB transfer timeout
- USB transfer error
- Online features disabled
- Network error
- No more data found
- URL not found

USB URLDISP

Display latest fetched URL screen on COMX.

During execution of this command USB CHAR(7) will be executed to set lower case 'default' and upper case characters on 'SHIFT'.

Screen color and character color will follow information as specified in the fetched URL screen.

Warning: if no valid data is loaded via USB URLGET a USB URLDISP command will most likely crash the COMX due to an invalid OUT 3 command based on invalid data.

USB BROWSER [,"url"]

Open specified url string in a text browser, if no url is specified the default 'home page' is opened.

Key commands - valid with magenta (or no) cursor:

- Movement:
  - Joystick down: Move cursor to next topic / link
  - Joystick up: Move cursor to previous topic / link
  - Joystick right or CR: Follow selected link
  - Joystick left: Return to previous topic
- Scrolling:
  - + or Space: Scroll down to next page
  - -: Scroll up to previous page
- Other:
  - CNTL B: Return back to BASIC
  - G: Go to specific URL. Latest selected URL will be specified as default.
  - H: Go to home page
  - SHIFT H: Set current page as home page
  - 0-9: Go to bookmark 0 to 9
  - SHIFT 0-9: Set current page as bookmark 0 to 9
  - ?: Show help page

Key commands - valid with yellow or diamant cursor (edit mode):

- Movement:
  - Joystick down: Move cursor to next topic / link or line
  - Joystick up: Move cursor to previous topic / link or line
- Other:
  - CNTL B: Return back to BASIC
  - CR or CNTL T: Store form text or URL

Key commands - valid with yellow cursor (edit mode):

- Movement:
  - Joystick right : Cursor to right
  - Joystick left: Cursor to left
- Other:
  - CNTL Q: Quit 'go to' URL feature and return to text browser

Default home page and bookmarks:

Homepage: Emma 02
Bookmark 0: Super Board - High Scores
Bookmark 1: AVI - COMX 35
Bookmark 2: AVI - Super Board
Bookmark 3: Emma 02 - COMX 35

Bookmark 4: Emma 02 - COMX BASIC
Bookmark 5: Elf/1802 Emulation
Bookmark 6: COSMAC ELF
Bookmark 7: Spare Time Gizmos
Bookmark 8: AVI - COSMAC VIP
Bookmark 9:

HTML ASCII characters:

HTML characters 161 to 255 will be shown as long as not more than 31 of these characters need to be shown on one COMX screen at the same time. If more than 31 characters are needed an X will possibly be shown for the 32nd and onwards characters. This is depending on how many 'regular' characters are shown on the same screen.

If any input checkboxes or radios are used 4 characters from the available 30 characters per COMX screen will be reserved.

Reserved shapes:

0: Empty space in screen editor functions
1 – 30: used for <comxshape> and &shapeX; tags. For more info see COMX HTML tag section.
31: X used for special characters that can't be displayed
127: Cursor

COMX HTML tags:

<comx shapeX="112233445566778899">

Shape character with identified number X (1-126), where 11 is the hexadecimal shape code for line 1, 22 for line 2 etc.

Characters 1 to 30 will work as any HTML ASCII character and can be shown by using &shapeX; where X is 1 to 30.

Characters 32 to 126 are the normal ASCII characters and a re-shape via <comx shapeX> will result in changing the shape for that character until another <comx shapeX> for the same character is done.

Note that the USB BROWSER command uses a USB CHAR (7) so lower and capital case characters are reversed!

Example:
<comx shape65="FFFFFFFFFFFFFFFF">a
Will show a cyan colored block instead of all characters a

Example:
<comx shape1="FFFFFFFFFFFFFFFF">&shape1;
Will show a cyan colored block for every &shape1; that is specified in the HTML file.

&shapeX;

Used to show shape characters 1 to 30.

<comx char="format">

When format is:

High or white: change character mode to use characters 128-255. For all regular characters this means: show in color white which is the same as is used for URL links.

Low or cyan: Change character mode to use characters 0-128. For all regular characters this means: show in color cyan which is the same as is used for normal text.

<comx screen="color">

Change COMX background color, where color can be either a number or the color written in text:

1: Black
2: Green
3: Blue
4: Cyan
5: Red
6: Yellow
7: Magenta
8: White

Note: when using the comxscreen tag for a page that page will also be shown with default comxcolor (12) and comxctone (0) settings. That is unless also comxcolor and comxctone tags are specified.

<comx color="colormix">

Change COMX character color, where colormix can be a number from 1 to 12:

	Normal text / <comxcharlow>	**URL links / <comxcharhigh>**
1	Black	Green
2	Red	Yellow
3	Blue	Cyan
4	Magenta	White
5	Black	Blue
6	Red	Magenta
7	Green	Cyan
8	Yellow	White
9	Black	Red
10	Blue	Magenta
11	Green	Yellow
12	Cyan	White

These color changes will be effective for all standard ASCII characters including characters 161-255. However when a character is changed by <comxshape> that character will NOT follow the coloring above but as defined in the shape commando following the definition below:

	Red	**Blue**	**Green**
1-4	CCB0	CCB1	PCB
5-8	CCB0	PCB	CCB1
9-12	PCB	CCB0	CCB1

CCB0: Bit 6 as defined in character shape
CCB1: Bit 7 as defined in character shape
PCB: bit 7 of character number (see comxcharlow/comxcharhigh tags)

Note: when using the comxcolor tag for a page that page will also be shown with default comxscreen (1) and comxctone (0) settings. That is unless also comxscreen and comxctone tags are specified.

<comx ctone="format">

Change COMX color format, where format can be: 0 / color or 1 / gray. Either a number or text is allowed.

Note: when using the comxctone tag for a page that page will also be shown with default comxscreen (1) and comxcolor (12) settings. That is unless also comxscreen and comxcolor tags are specified.

USB EMAIL

Start the COMX Email editor.

Key commands - valid with magenta (or no) cursor:

- Movement:
  - Joystick down: Move cursor to next topic / link
  - Joystick up: Move cursor to previous topic / link
  - Joystick right or CR: Follow selected link
- Scrolling:
  - + or Space: Scroll down to next page
  - -: Scroll up to previous page
- Other:
  - CNTL B: Return back to BASIC
  - ?: Show help page

Key commands - valid with yellow cursor (edit mode):

- Movement:
  - Joystick down: Move cursor to next topic or line
  - Joystick up: Move cursor to previous topic or line
  - Joystick right : Cursor to right
  - Joystick left: Cursor to left
- Other:
  - CNTL B: Return back to BASIC
  - CR or CNTL T: Store form text
  - CR in message field: Go to next line

Aliases can be used in the 'To' field if they are defined in the PC SW. See also chapter 'Online settings'.

[STR=] USB HSWEB

Synchronize latest high scores with the Super Board high scores server (http://www.comx35.com/comx35sbhs/hs2.php). Scores for the software IDs as listed under USB NVLIST will be synchronized but only if a score is already stored in NVRAM. If no score is stored no synchronization for that game will be done; to store a score just play the game once (make sure to play a game that is adapted to include NVRAM high score code).

Player name and location will be taken from the USB PC SW. Except for Happiehap, Happiehap 2 and Get your gadget where the player name will be used from the game itself.

Optional return string will be empty ("") if the command is successful. Possible error return strings:
- USB transfer timeout
- USB transfer error

- Network error
- Server error
- Online features disabled

If no return string is specified, i.e. USB HSWEB, any error will result in ‘breaking’ the program and printout of the error code.

Machine code: USB HSWEB can be called from any machine code program by calling subroutine on E013h.

The COMX standard SEP 4 / SEP 5 should be used for the subroutine call. Content of registers: R7.0, R8, RC, RE and RF will be destroyed

Turbo Tape

USB TLOAD[+]

USB TSAVE[+]

USB TDLOAD[+]

USB TDSAVE[+]

F&M Turbo tape routines work exactly like BASIC save/load commands except much faster. The '+' versions are improved versions resulting in different wav formats. The wav files created with the regular versions can be loaded with the regular and '+' versions. Wav files created with the '+' versions are not loadable with the regular versions.

Printer

USB PON

Turn the printer on with CNTL characters suppressed.

This command will first search for a standard printer card, if found it will be selected and the printer output will be activated. If no standard printer card is found a search will be done for the thermal printer card.

The printer output can be switched off via USB PROFF. Also note that most USB and DOS commands will switch off the printer output except for:

- USB CAT
- USB VER
- USB CD
- USB HEX
- USB BIN
- USB NVLIST
- USB HELP
- USB SYSDISP
- USB HISTORY

USB POFF

Turn printer output off

USB PTV (x)

Select printer setting to enable or disable TV output during 'printer on'. x=0 (default) no output will be done to the TV screen after USB PON, x=1 output will be done to both TV and printer after USB PON.

If output to TV is selected USB PON will also temporarily switch the clock off until the printer is switched off again.

USB PCNTL (x)

Switch printer CNTL character suppression on (x=1, suppressed; normal characters) or off (x=0, hex values). Set value will be stored in NVRAM and used for the USB PON, USB PTVON, USB PTEST, USB PLIST and USB PPRINT commands.

Any other uneven value of x will have the same result as 1, even same as 0.

USB PKB (x)

Select keyboard output setting to the printer. x=1 (default) keyboard output will be send to the printer, x=0 no keyboard output will be send to the printer

USB PMEMDUMP (start, end)

Start memory dump routine and send result to printer.

If memory location C000h to DFFFh is included the content of the current selected slot and bank will be printed. To select a different slot use the CARD Sx command, to select a different bank use the CARD Bx command.

USB PTEST

Start printer test routine.

USB PSET

Set serial printer configuration. The set configuration will be stored in NVRAM and used for USB PON, USB PMEMDUMP, USB PLIST, USB PPR and USB PPRINT commands.

USB PLIST

Same as LIST command but output is send to the printer.

USB PPR / USB PPRINT

Same as PRINT command but output is send to the printer.

F&M Monitor

USB MON

Start F&M Monitor!

Note that the ‘Program start’ value is save in regular RAM on location 4403h/4404h so that the value is save in the current loaded program and saved within the USB PSAVE command. This also means the ‘Program start’ value should never be defined as a value below 4405h as that will crash the monitor when using the CNTL I/D features.

Machine code: The register read function can be called from any machine code program by calling subroutine on E016h. The COMX standard SEP 4 / SEP 5 should be used for the subroutine call. Content of registers RF will be destroyed.

USB DMON

Start F&M Disk Monitor

Note F&M Disk Monitor uses some BASIC RAM, USB DMON will save these RAM areas to (RAM) bank 15 and restore to original values at exit to BASIC (key 3).

Games

USB TENNIS

Start Tennismania.

Note Tennismania uses some BASIC RAM, USB TENNIS will save these RAM areas to (RAM) bank 15 and restore to original values at exit to BASIC (key CNTL B).

High scores are saved in NVRAM with SW ID=11.

USB SPACE

Start Space invaders.

Additional key functions compared to original game:
CNTL B: Return to BASIC
S: Toggle sound on / off

High score and sound setting are saved in NVRAM with SW ID=4.

Note Space invaders uses some BASIC RAM, USB SPACE will save these RAM areas to (RAM) bank 15 and restore to original values at exit to BASIC (key B).

USB WORM

Start Eat a worm.

Additional key functions compared to original game:
CNTL B: Return to BASIC
S: Toggle sound on / off

This version of the game does not support the ‘bonus’ and ‘level’ features as in the original. However it does support two new features instead:

1. WALL counter, which counts the number of worms that need to be eaten until a wall forms
2. Speed increase, the longer you play the faster your snake will run

High score and sound setting are saved in NVRAM with SW ID=5.

Note Eat a worm uses some BASIC RAM, USB WORM will save these RAM areas to (RAM) bank 15 and restore to original values at exit to BASIC (key B).

COMX DOS to PC

USB IMGTEST,"filename"

Check format of image file on PC and show if it needs a floppy disk formatted to single/double sided and single/double density.

USB IMGTODISK,"filename" [/DR]

Copy a disk image file from the PC to a floppy disk. PC filename should have extension .img, however no extension should be given in the USB command. During the image copy the disk side and track currently being written is shown on the screen.

Example:
USB IMGTODISK,"GAMES"

will put the games.img image on the current floppy disk.

Note that the destination floppy disk HAS TO be formatted with DOS INIT with the same parameters as the disk image. See USB IMGTEST to find out what format is needed.

ESC key (long press) will stop the command but note that destination disk in this case will not be usable.

USB DISKTOIMG,"filename" [/DR]

Copy a floppy disk to a disk image file on the PC. PC filename will get extension .img, however no extension should be given in the USB command. During the image copy the disk side and track currently being written is shown on the screen.

Example:
USB DISKTOIMG,"GAMES"

will put the current floppy disk content to an image called games.img.

ESC key (long press) will stop the command but note that destination image in this case will not be usable.

Graphics

Most commands described in this chapter were added in FW build 195, released July 2016. With the exception of commands USB TV, USB DHEIGHT and USB DWIDTH.

USB GRAPH

Change back to normal text screen, this will include a clear screen and re-shape of current selected character set. Note USB GRAPH (except for the clear screen) is executed automatically when a program ends, this to avoid a 'hanging' COMX as no command entry is possible in the 'graphic' modes.

USB GRAPH (graphic screen)

Graphic screen values range from 0 to 7 where:
Bit 0 = 0: Characters size 6*9 or 6*8 pixels
Bit 0 = 1: Characters size 6*16 pixels

Bit 1 = 0: Normal pixel size
Bit 1 = 1: Double width / double height pixels

Bit 2 = 0: PAL resolution (6*9 character size, 108 pixel screen height),
Bit 2 = 1: NTSC resolution (6*8 character size, 96 pixel screen height)

If bit 0=1 then setting bit 2 has no effect and the character size will be 6*16 with a screen height of 108 pixels. Note that these cases (graphic screen 5 and 7) are used for extended / different versions of graphic screen 1 and 3.

For all graphic screen variants (except 7) the complete defined area will NOT be usable. This is because not all pixels can be addressed individually (max 127 blocks of 6*16 pixels is possible) due to the fact that USB GRAPH uses 127 characters to simulate the graphic screen. The 127 characters can't fill any of the graphic screen resolution. When no more characters are available USB PLOT, LINE, CIRCLE and ELLIPSE will just stop working; i.e. nothing is plotted.

Graphic screen 0, 2, 4 and 6 allow use of PRINT and CPOS features but PRINTed text will be re-shaped if too many USB PLOT, LINE, CIRCLE or ELLIPSE commands are executed.

Graphic screen 0:
(normal text screen, including limited plot feature like PECOM 64:

X: 0-227, Y: 0-107
*Character size: 6*9 pixels*
Pixel size: normal

Note: First screen line and first 2 character columns are not used, so USB PLOT (0, 0, 8) will plot in the bottom left hand corner of the character in CPOS (12,2), i.e. line 13 and character 3.

Graphic screen 1:

X: 0-239, Y: 0-215
*Character size: 6*16 pixels*
Pixel size: normal

Note: The last line on the screen will use character size 6*8.

Graphic screen 2:

X: 0-119, Y: 0-107
*Character size: 6*9 pixels*
Pixel size: Double width / double height

Graphic screen 3:

X: 0-119, Y: 0-107
*Character size: 6*16 pixels*
Pixel size: Double width / double height

Graphic screen 4:

X: 0-227, Y: 0-95
*Character size: 6*8 pixels*
Pixel size: normal

Note: This is an NTSC version of graphic screen 0. When using this screen on a PAL COMX the first 3 character lines (16 pixel lines) will be repeated on the bottom of the screen.

Graphic screen 5:

X: 0-239, Y: 0-207
Character size: 6*16 pixels
Pixel size: normal

Note: This is more or less the same as graphic screen 1 except the last line on the screen is never used. This makes USB PLOT (0, 0, 8) start 8 pixels higher compared to graph 1. As the last line on graphic screen 1is only 8 pixels high graphic screen 5 in a way allows more pixels to be drawn but on a smaller area.

Graphic screen 6:

X: 0-119, Y: 0-95
*Character size: 6*8 pixels*
Pixel size: Double width / double height

Note: This is an NTSC version of graphic screen 2. When using this screen on a PAL COMX the first 12 pixel lines will be repeated on the bottom of the screen.

Graphic screen 7:

X: 0-107, Y: 0-107
Character size: 6*16 pixels
Pixel size: Double width / double height

Note: This is more or less the same as graphic screen 3 except the first and last 'column' of the screen are not used. This makes USB PLOT (0, 0, 8) start 6 pixels to the right compared to graphic screen 3. Due to this fact all (defined) positions of the screen can be filled by using 126 characters, meaning USB PLOT will never stop working as the characters never 'run out'.

USB PLOT (x, y, color)

Plot one pixel on position X, Y. Note that 0, 0 is the bottom left hand corner of the screen (so NOT like CPOS).

Color identifies the color of the pixel (1=black to 8=white), following the COMX SCREEN color numbering. Note that the last color ‘plotted' can affect color of pixels already on the screen. This because only one pixel color is possible for every 6 pixels in an horizontal ‘row' (and of course the screen background color), also on every character (6*8, 6*9 or 6*16) only color 1 to 4 or 5-8 can be displayed simultaneously. The last use USB PLOT color will change colors on the same line of 6 pixels and on the same character if needed.

USB PLOT (x, y, color, character)

Plot character on position X, Y. Note 0, 0 is the bottom left hand corner of the screen.

Color identifies the color of the pixel (1=black to 8=white). Colors have the same limitation as for USB PLOT.

Character number is drawn as defined in the standard COMX ROM character set, where ‘A' is 65. However lower case characters are printed if character number is between 97-122 as in a standard ASCII definition.

Character is drawn ‘down' from x, y; so x, y is the top left corner of the character. DOS INIT

USB LINE (x, y, color, x2, y2)

Plot a line from position X, Y to X2, Y2. Note 0, 0 is the bottom left hand corner of the screen.

Color identifies the color of the pixel (1=black to 8=white). Colors have the same limitation as for USB PLOT.

USB CIRCLE (x, y, color, radius)

Plot a circle with the centre at position X, Y and radius. Note 0, 0 is the bottom left hand corner of the screen.

Color identifies the color of the pixel (1=black to 8=white). Colors have the same limitation as for USB PLOT.

USB ELLIPSE (x, y, color, a, b)

Plot an ellipse with the centre at position X, Y, horizontal radios a and vertical radius b. Note 0, 0 is the bottom left hand corner of the screen.

Color identifies the color of the pixel (1=black to 8=white). Colors have the same limitation as for USB PLOT.

USB DCHAR (state)

Double character lines:

Bit 0 = 0: default 9 (or 8 on NTSC) character lines
Bit 0 = 1: 16 character lines

Bit 1 = 0: do not clear shape lines 9 (or 8 on NTSC) to 16
Bit 1 = 1: clear shape lines 9 (or 8 on NTSC) to 16

USB DCHAR (1) or USB DHAR (3) will almost double the graphic capacity from 128 characters with 9 (or 8 for PAL) lines to 128 characters with 16 lines. This mode is giving similar resolution as when using USB GRAPH with graphic screen bit 0 set to 1.

In double character line mode the lines 10 to 16 will show random shapes until changed via the USB SHAPE command u3nless bit 1 is set to 1 with USB DCHAR (3). Clearing shape lines 9 to 16 will also disable the screen editor so after any USB DCHAR (3) on return to BASIC prompt a USB GRAPH will be executed to reset screen and shapes to normal like is done after USB GRAPH (x).

Note that the COMX and the Screen editor will also not be able to handle doubling of the character lines very well when using USB DCHAR (1). Half of the lines will not be shown anymore. To return back to normal give the USB DCHAR command with state=0.

USB SHAPE (character, shape string)

Extended SHAPE command where the shape string can consist of a string with 16 bytes to allow ‘shaping’ of character line 10 (or 9 on PAL) up to 16. This command is added so it can be used to ‘shape’ line 10 to 16 when using USB DCHAR (1).

Note that using USB SHAPE in combination with the screen editor can cause some input to be disabled.

For example after:
USB SHAPE (65,”00112233445566778899AABBCCDDEEFF”)

Input of the character A will become impossible. Meaning no SHAPE command or any command using character ‘A’ will be accepted by the screen editor. Only a reset or execution of the USB GRAPH command will resolve this issue. Because of this it is probably good practice to start any program using USB SHAPE with a USB LINE command so the screen editor is disabled. At the end of program execution a USB GRAPH and USB SCREEN will reset the screen editor to default settings. Reason for this issue is because the screen editor uses character line 10 to identify the character number. So character 65 has a line 10 shaped as 64 or 41 hexadecimal. This is used to read what character is located where because the page memory is not readable but the character memory is on a COMX.

USB TV (state)

Turn TV screen on (1) or off (0).

This command can be used to get a picture or text shown on screen very fast by first turning the TV off with USB TV(0), then printing the screen and at the end turning the TV back on with USB TV(1).

Note USB TV(0) will also disable the keyboard!

USB PPOKE (page RAM address ,value)

Store value in CDP1870 page RAM address location, page RAM address range is between 0 and 3FFh. Specified address will always be converted to a value within this range. So a USB PPOKE on address 400h will result in a USB PPOKE on address 0.

Page RAM location 0 is not necessarily the top left hand corner of the screen as this is depending on scrolling done since restart. To find the current top of screen location use value stored on address 4195h and 4196h.

Example:
10 P=PEEK(@4195)*256+PEEK(@4196)
20 USB PPOKE (P, 65)

Will store character ‘A’ somewhere in the page RAM which on a COMX screen will show an ‘A’ on the top left corner of the screen.

The main advantage of using USB PPOKE instead of a PR CHR$() is that with USB PPOKE it is possible to put any character on the screen including characters 10 (Ah) and 13 (Dh).

USB CPOKE (address, value) / USB CPOKE (character, line, value)

Store value in CDP1870 character RAM on specified address, character RAM address range is between 0 and 7FFh. Specified address will always be converted to a value within this range. So a USB CPOKE on address 800h will result in a USB CPOKE on address 0.

CDP1870 character RAM is build up of 16 bytes per character starting with character 0 and using 1 byte per character line.

One USB CPOKE command will change the shape of 1 line of a character.

Example:
USB CPOKE (1040,#FF)

This will shape the top line of character ‘A’ to a line.

1040 is character 65 (‘A’) multiplied by 16 (16 lines per character)

The following formula can be used to calculate the character RAM address:
character_number*16+line_number

For convenience a second USB CPOKE command is available where character number and line can be specified. So USB CPOKE (65, 0, #FF) is the same as USB CPOKE (1040,#FF).

INT = USB CPEEK (address) / INT = USB CPEEK (character, line)

Similar to USB CPOKE however will read a value from the CDP1870 character RAM on specified address, character RAM address range is between 0 and 7FFh. Specified address will always be converted to a value within this range.

For convenience a second USB CPEEK command is available where character number and line can be specified. So A = USB CPEEK (65, 0) is the same as A = USB CPEEK (1040).

USB DWIDTH (state)

Set pixel width to double size. State 1 = double and 0 = normal.

Note that the COMX and the screen editor will not be able to handle double pixel width very well. Half of the lines will not be shown anymore. To return back to normal pixel width and height press CNTL S (or give the applicable command with state=0).

Note that on FW version 1.91 and before: When using CPOS or USB POS when double width is active the first line on the screen will be line 0 and column 0 to 19, the second line will also still be line 0 but column 20 to 39. From FW version 1.92 this is corrected so CPOS(1,0) is the first location of the second line.

USB DHEIGHT (state)

Set pixel height to double size. State 1 = double and 0 = normal.

Note that the COMX and the screen editor will not be able to handle double pixel width very well. Half of the lines will not be shown anymore. To return back to normal pixel width and height press CNTL S (or give the applicable command with state=0).

REM adaptation

Commands described in this chapter were added in FW build 195, released July 2016.

USB LIGHTRM line

Change color of text in REM command on indicated line from output color (default cyan) to input color (default white). Colors cyan and white are depending on which COLOR combination is used.

To return to normal output color: execute the USB LIGHTRM command a second time for the same line number.

USB HIDERM line

Hide line number of indicated REM line. The text following the REM command will overwrite the line number and REM command text.

To return to normal output: execute the USB HIDERM command a second time for the same line number.

Examples:
10 REMABCDEFG
20 REM---REM Text
30 REMXX

After USB HIDERM will show:
ABCDEFG
---REM Text
XX REM

Obviously when using the Screen editor the hidden REM lines cannot be edited when ‘listed’’ on the COMX screen. Also the EDIT command when the LINE editor is active will not work correctly. To edit these lines return to normal output with a second USB HIDERM.

RENUMBER extension

Commands described in this chapter were added in FW build 196, released July 2016.

RENUMBER start, end [,line] / RENUMBER [line]

With the new extended RENUMBER command it is possible to renumber only part of the BASIC code by specifying the start and end address of the part to be renumbered. Note that the address specified as 'end' will NOT change number to make it easier to renumber subroutines.

Examples:
1000 REM Subroutine 1
1005 REM some code
2000 REM Subroutine 2

RENUMBER 1000,2000
This will result in:
1000 REM Subroutine 1
1010 REM some code
2000 REM Subroutine 2

Optional 3rd parameter 'line' specifies the line increment which if omitted will default to 10.

If during the RENUMBER the COMX detects that the new renumbering doesn't fit between specified 'start' and 'end' it will automatically decrease the increment with 1.

The extended RENUMBER command will also handle BASIC line numbers >65280 which were not handled in the original command. Note however that COMX BASIC does have some other limitations with line numbers >65280 so it is recommended not to use any line number between 65280 and 65535.

Slot And Card Access

CARD B (x) or CARD Bx

Select ROM or RAM bank in slot 4, where x=0 to 14.

Bank 0-7 are used for USB FW and bank 8 to 14 for 7 8K RAM banks.

Note that selecting bank 15 will not be accepted in this command. This is because RAM bank 15 is reserved for use by the SB FW and when overwritten will cause the COMX to behave incorrectly.

() is optional for hard coded values, however if variables or calculations are used as 'x' () are mandatory. E.g. B=7: CARD B(B) and CARD B7 are correct but B=7:CARD BB is invalid.

[INT =] CARD B

Return the current selected bank, possible return values are bank 0 to 15.
If CARD B is used without preceding variable or PRINT the bank value will be printer to the screen.

CARD S (x) or CARD Sx

Select slot x, where x=1 to 4.

() is optional for hard coded values, however if variables or calculations are used as 'x' () are mandatory. E.g. S=2: CARD S(S) and CARD S2 are correct but S=2:CARD SS is invalid.

CARDS0 and CARDS4 also have the same behavior as the USB command.

INT = CARD S

Return the current selected slot, possible return values are slot 1 to 4.

[INT =] CARD F (x) or [INT =] CARD Fx

Find and select card type x. Optional return value is the selected slot with card type x. If the card type is not found 0 is returned.

Known types:
1: Standard printer card
2: Thermal printer card
3: Floppy disk card
4: 80 Column card
5: EPROM programmer (I have never seen this one!)
6: Network card
33: USB FW, always slot 4

() is optional for hard coded values, however if variables or calculations are used as 'x' () are mandatory. E.g. F=1: CARD F(F) and CARD F1 are correct but F=1:CARD FF is invalid.

USB

Select the SB card (slot 4 and bank 0).
Commands CARDS0 and CARDS4 also have the same behavior as USB.

COMX DOS

DOS BOOT

Force re-boot of COMX DOS 1.4 from ROM.

DOS INIT

Formats a disk.

DOS STARTUP

Formats a disk like the INIT program however with an F&M startup boot sector which will attempt to load and run a program called 'STARTUP' when DOS is booted from this disk after a power up (like with DOS NEW).

Note: After a DOS STARTUP command the next DOS command will fail with an error message ‘File not found in directory’. This is caused by the fact that DOS is booted after every DOS STARTUP and DOS INIT, in this case the DOS boot cannot find the ‘STARTUP’ file on the disk and will report that in an error message. It is recommended to always give a DOS NEW after the DOS STARTUP is finished to ‘flush’ out the first boot including the file not found error.

Also remember that DOS is only booted on a DOS command from disk after a power-up and or after DOS INIT / STARTUP. DOS BOOT boots from ROM and as such will not use the STARTUP function.

Example:
DOS STARTUP

COMX DOS V1.4
COPYRIGHT BY COMX – 1984
INSERT FORMAT DISK THEN PRESS ANY KEY

ARE YOU SURE ? (Y/N) Y
DRIVE NO. (1 OR 2) ? 1
DOUBLE TRACK DENSITY ? (Y/N) N
SINGE(1) OR DOUBLE(2) SIDE ? 2
ARE YOU SURE ? (Y/N) Y
DISK NAME (MAX.8 CHARS) ? TEST
DATE (DD-MM-YY) ? 05-07-13

READY
DOS NEW

F&M STARTUP © 1986

File not found in directory
READY
10 PRINT “STARTUP PROGRAM”
DOS SAVE,”STARTUP”

READY

DOS LABEL,"diskname" [/DR]

Change the disk name to specified string. Maximum 8 characters can be specified for the diskname.

DOS DATE,"date" [/DR]

Change the disk date to specified string. Maximum 8 characters can be specified for the disk date. Any date format will be allowed.

DOS LOCK [,"filename"] [,D] [,W] [/DR]

Lock specified software on the disk. Option D is used for the delete protection and option W for write protection. /DR is the drive number (1 or 2). When no filename is specified all software on the disk is locked according to remaining options. When no D or W is specified both protections will be set, when no /DR is specified drive 1 will be used.

DOS UNLOCK [,"filename"] [,D] [,W] [/DR]

Unlock specified software on the disk. Option D is used for the delete protection and option W for write protection. /DR is the drive number (1 or 2). When no filename is specified all software on the disk is unlocked according to remaining options. When no D or W is specified both protections will be removed, when no /DR is specified drive 1 will be used.

DOS DISKCOPY

Make an exact copy of a disk including a format/initialization of the destination disk before copy begins.

Note this command will destroy content of (RAM) bank 13 and 14. This is done to save BASIC RAM content from address @5000 to @9FFF which is used by the DISKCOPY command. BASIC RAM is only restored if the command is successfully executed.

DOS FILECOPY

Make a copy of a specific file.

Note this command will destroy content of (RAM) bank 13 and 14. This is done to save BASIC RAM content from address @5000 to @9FFF and @AC00 to @AFFF which is used by the FILECOPY command. BASIC RAM is only restored if the command is successfully executed.

Other Commands

USB VER

Show version, copyright and date information.

USB OLD [,defus address]

Restore ‘old’ BASIC program. When the optional defus address is specified the program at the specified address will be recalled. If no address is specified the program at default defus address of 4400h will be restored.

If a recall is done for defus address 4400h the restore will only be successful if the original program started with 10 REM.

USB HELP [,“command”]

Show command format for specified command or any command that contains the specified string. For example USB HELP “USB” will show all USB commands. If more than 24 lines of command help is shown the command will pause until a key is pressed, if Q is pressed help will quit.

If the command string is equal to a “*” all supported BASIC commands will be shown.

If the command string is omitted a short explanation and format for help will be shown.

USB HISTORY

Show last 8 commands (including USB HIST) which can be ‘recalled’ by using CNTL R, see also Screen Editor help.

STR = USB HEX (value)

Return the hexadecimal value of the variable or number specified. Max value is 8388607 or hexadecimal 7FFFFF.

STR = USB BIN (value)

Return the binary value of the variable or number specified. Max value is 8388607, binary values are returned in either 8 bit, 16 or 24.

INT = USB POS (row, column)

Return the ASCII value of the character on screen position row / column. Row = 0 and column = 0 represent the top left corner of the screen.

USB MEMDUMP (start, end)

Start memory dump routine and send result to screen.

If memory location C000h to DFFFh is included the content of the current selected slot and bank will be printed. To select a different slot use the CARD Sx command, to select a different bank use the CARD Bx command.

Screen Editor

Joystick: move cursor
CNTL D: Delete character at cursor and shift rest of line left
CNTL S: Clear the screen and switch character size back to normal
CNTL I: Insert space at cursor and shift rest of line right
CNTL R: Repeat last line, a total of 8 lines are kept in a buffer if a RAM chip is installed in the SB
CNTL X: Copy line of current cursor position to 'copy' buffer, the buffer is 96 characters long. If the cursor is on a line with more than 96 the cursor will be moved to the position after the characters that are copied to the buffer. If no RAM chip is installed in the SB the CNTL R buffer will be used.
CNTL V: Paste the content of the 'copy' buffer to the current cursor position
CNTL E: Move cursor to end current line
CNTL W: Wipe current line from cursor position
SPACEBAR during LISTING: Pause LIST function until next SPACEBAR press

Note 1: Some software, like the original version of the F&M Monitor+, might not be fully compatible with the screen editor and when giving control back to BASIC, after the program is ended, the screen looks empty but at every return error: 'Line buffer overflow' is given. To solve this just press CTRL S to really clear the screen.

Note 2: Some software, like Kleiduiven Schieten, Torens van Hanoi, Ijskikker and F&M Schaak might not be fully compatible with the screen editor because they use the character 0 for a different shape. In this case the screen will be filled with funny shaped characters instead of spaces. To avoid this problem you can switch off the screen editor by using USB LINE.

Note 3: Some software, like Killerwatt might not be fully compatible with the screen editor and will cause the game to hang.

Other features

- Saved in NVRAM and initiated at startup:
  - Clock display on / off
  - Boot message (W = WARM START) on / off
  - Selected character set (USB CHAR)
  - 80 Column auto start-up (USB EIGHTYCOL)
  - Logo tune (USB LOGOTUNE)
  - Line editor / screen editor selection
  - Time and date format
  - Volume
  - Printer settings
  - Online status
  - SHAPE, COLOR and USB CHAR screen flash

Bug fixes

- Red dot in shape of exclamation mark
- Temporary stack 4FFFh changed to BDFFh to avoid change of RAM memory locations 4FF7h-4FFFh during restart
- Typing '10 TOUT' on clean startup hanging fixed, TOUT is replaced by USB anyway but still '10 USB' will not cause any hanging either.
- Crash on typing READ x when no DATA statement is available in the loaded BASIC program.
- Bug fixed in F&M expansion ROM where CARDS caused a crash
- Bug fixed when 65535 line number is typed
- Fixed COMX BASIC bug that crashed EDIT x where x was not an integer value like EDIT (10
- Corrected error code when typing SHAPE without parenthesis

Extra Features

- When 80 column card is plugged in it is possible to start directly in 80 column mode
- Press 'W' on start-up to boot with 'warm' start, i.e. BASIC will not be reset. Plus added Warm start text on logo screen
- Press 'C' on start-up to show copyright screen
- Extended text on 4th boot screen including longer delay
- Extended BASIC start-up message
- All (x) can be replaced by basic expressions like (1+4) or (A) etc
- All string values can be replaced by string expressions like A$, A$(1), A$+B$, MID$(A$,1,4) etc
- Changed the expansion ROM error code 'ERROR' or 'SYNTAX ERROR' to 'Syntax error in CARD statement'. Also using DOS will give error 'NO DISK CARD' if no disk card is found. This was removed in the F&M expansion version to save space.
- All ERR CODE numbers replaced by text including BASIC, DOS, F&M BASIC and USB error codes
- All ERR CODE text and HELP text will be capitalized correctly independent of used USB CHAR
- Added line of spaces after 'READY' prompt
- Changed EDIT x to show line number only in screen editor mode to simplify editing as well as to avoid issues that caused EDIT not to work correctly in screen edit mode
- Moved shape '&' 1 line up
- Moved top line for shape '#' 1 line lower
- Added a call to 'DOS NEW' on startup (only if DOS was booted before reset), so no DOS NEW needs to be given manually anymore
- Keyboard is cleared after every 'ESC' press so no unwanted characters show up at the READY prompt
- Command entry in 'SHIFT' characters is allowed in screen editor with CHAR (2) or (3) active. All commands are stored as 'normal' characters. i.e. in capitals for CHAR (2) and lower case for CHAR (3).
- Map joystick 1 values to the COMX keyboard. This will allow BASIC SW and in some cases machine code SW (only if BASIC key read routines are used) to be used with joystick 1. Fire key is mapped to space bar and up/down/left/right to the build in joystick. This function only works when slot 4 is selected and active.

APPENDIX A – MEMORY MAP

COMX & Super Board Memory MAP

Address	Original COMX 35	COMX + Super Board
0000 - 3FFF	COMX BASIC ROM	SB BASIC ROM
4000 - 43FF	BASIC System Parameters	BASIC System Parameters
4400 - BDFF	USER RAM	USER RAM
BE00 - BFFF	Reserved for use by expansion cards	Reserved for use by expansion cards and used by SB
C000 - DFFF	Reserved for 4 expansion cards Selection via OUT 1: b1: Expansion slot 1 (#02) b2: Expansion slot 2 (#04) b3: Expansion slot 3 (#08) b4: Expansion slot 4 (#10)	Slot 1, 2 and 3 reserved for 3 expansion cards Selection via OUT 1: b1, b2, b3 and b4 as for the original COMX b0: 0 select ROM banks, 1 select RAM banks b5-b7: select bank number
E000 - E7FF	Reserved for expansion ROM (replacing part of COMX BASIC ROM)	SB FW
E800 - EFFF	Reserved for expansion ROM	Expansion box ROM
F000 - F3FF	Not used	NVRAM & RTC
F400 - F7FF	Character Memory	Character Memory
F800 - FFFF	Page Memory	Page Memory (write) & SB FW (read)

Super Board EPROM Content

	SYSTEM ROM
0000-3FFF	SB BASIC ROM
4000-47FF	SB FW E000-E7FF
4800-67FF	Not used
6800-6FFF	Expansion box ROM
7000-77FF	Not used
7800-7FFF	SB FW F800-FFFF

	EPROM 1 (SB Banks 0-3)	EPROM 2 (SB Banks 4-7)
0000-1FFF	SB Main command handling	MON & SYSDISP
2000-3FFF	Error handling, NVLIST & VER	TENNIS
4000-5FFF	Help, printer commans & MEMDUMP	DMON, WORM, JOY, Q, MOVE, TV, PCNTL, PTV, HEX, BIN, GO40 GO80, BOOTMSG, LOGOTUNE, DWIDTH, DHEIGHT, POS, PKB, EF, LINE, SCREEN, JOYLOAD, NVPOKE, CLOCK, OLD, RTCFT, CARDB, CARDF
6000-7FFF	DOS and Tape commands	SPACE & Graphic commands

APPENDIX B – DETAILED MEMORY MAP

This document lists detailed memory locations for most SB FW routines as well as data locations.

The following color coding is used:
Black text: SB 1802 code
Green text: unused / spare memory
Blue text: data areas
Red text: info about code that is removed compared to the original COMX ROM
Orange text: Main routine description
Unchanged: info about original COMX ROM code

Note that not all code / locations are described!

BASIC ROM (0000 - 3FFF)

0000	-	000C	RB = 0013, SEP B
000D	-	000F	Branch to 037D
0010	-	0012	Branch to 0258
0013	-	001E	Disable interrupts and set OUT 3 to F0 (normal pixel width, COLB12 = 11, Display off, No CFC, black BKG).
0018	-	0246	Start-up routine which is optimized and rewritten
0018	-	001E	Set R2 to BDFF (instead of 4FFF which cause 4FFF RAM area to be destroyed at restarts) and change stack pointer to R2
001F	-	0032	Copy initial values from 0DA7-0DBE to 41C0-41D7
0033	-	004B	- Initialize R7 and RF.0 to 0 - Check on PAL/NTSC (EF2) and correct 41C3/41CA and set R7 to 8h and RF.0 to 80h for NTSC version. - R7 is used for CDP1870 OUT 5 value and RF for the character of the COMX bars on the start-up screen. Values will be different for PAL/NTSC
004C	-	005A	Set RD=FFFF for pointer to PAGE RAM and clear PAGE RAM
005B	-	005D	Give Q pulse to start DMA RAM Refresh
005E	-	0071	Fetch LOGOTUNE bit from NVRAM (bit 7 F3F0h), put it in bit 0 of RC and fill rest of RC with 3C40 which will start tune if bit 0 = 1 with the OUT 4 on 0070h.
0072	-	0080	Delay if LOGOTUNE is on
0081	-	0088	OUT 7 with F800 to define top of screen
0089	-	00A2	SHAPE COMX characters as defined from 0605-0A84. Note this routine uses R4=0C80 and R5=00AA followed by a SEP 4 to call the SHAPE routine at 0C80 and return with a SEP 5 to 00AD.
00A3	-	00A9	Entry: 00AA, print D on screen pointed in RD, INC D and return by branching to 00A3 which executes a SEP5.
00AA	-	00DE	PRINT COMX logo and bars on the screen
00DF	-	00F9	Check location 4281/4282 (DEFUS value), if between 4400h and C400h and the low byte is 0Ch continue to add the 'W = WARM START' message on the screen, if not skip to 0108.
00FA	-	0104	Check F0F3 bit 5 if 'W = WARM START' (boot message) should be printed
0105			Print boot message followed by remaining text as below.
0108	-	0117	Print remaining start-up text, loading R6.0 with 63 (R6.1 is already #FF)
0118	-	0120	OUT 3 with 40h(Double Pixel width, COLB10=01, Display on, No CFC, black BKG) OUT 5 with R7 (Double pixel height and 8/9 lines depending on PAL/NTSC mode)
011E	-	012C	Fetch LOGOTUNE bit from NVRAM (bit 7 F3F0h) and branch over logotone routine if switched off

012D	-	014D	Logotone routine
014E	-	0152	Fetch GO80 bit from NVRAM (bit 6 F3F0) and branch over 80 column selection if switched off
0153	-	0175	GO80 selection routine - Search for 80 column card by switching slots from 1 to 3. - If not found continue regular boot on 019B - If found skip over logo screen cycle routine by using a SEP 4 to 01C2
0176	-	017E	Force copyright screen during boot screen (C pressed) and continue on 01AC by using a SEP 5. This routine is called when 'C' is pressed via check on 0189
017F	-	019A	Start routine in 0180. Wait routine including key press checks 'C' and 'space'. Key C will branch to 0176 to force copyright screen, other keys (not 'space or 'C') will exit routine to 01C2. Wait loop length is set to byte stored after 'SEP 4' call to 0180.
019B	-	01A1	Set X to R5; R4=0180
01A2	-	01C1	Logo screen cycle routine - SEP 4, X calls the wait routine with a delay of 'X' - After wait routine screen is changed by an OUT 3 to change COLB10 (color) or pixel width/height by an OUT 5. The OUT 5 used R7.0 to select NTSC or PAL mode. - Delay on copyright screen is increased compared to original ROM
01C2	-	01D0	Clear screen (with '00' instead of '20' as in original ROM
01D1	-	01EB	Get COLOR, CTONE and SCREEN info from NVRAM (F3E9, bit 0-3 and 5-6), use value with bit 7=1 (normal pixel width) with an OUT 3 to use colors in start-up.
01EC	-	01F1	OUT 5 with R7 (normal pixel height and 8/9 lines depending on PAL/NTSC mode)
01F2	-	0203	Clear 4100-41AE
0204	-	0209	Set R1=0496 (start interrupt routine)
020A	-	021D	Set 4193 to 419B to default values, 419C to 41AE default set to 0 above
021E	-	0220	Set OUT 1 to 0 to handle start-up of SB.
0221	-	0226	R3=0229
0227	-	0228	Enable interrupt, X=R2 and P=R3
0229	-	0246	Initialize register, R4=2E14, R5=31EB, R9=4370, RD=3302
			Continue on 0FFF
0244	-	0257	Spare
02E4	-	02E6	LBR FBF8, new scroll routine for proper clock clearing
046C	-	046F	LBR E6C7; NOP, call to updated print routine
054E	-	0561	Updated interrupt branching to E025/E01F. Including a LDA 0 on 054E to handle a COMX without DMA.
056E	-	0570	LBR 0DE0, part of updated interrupt routine clearing key press if ESC was pressed.
0605	-	0A84	COMX Standard character SHAPE definition
0733			Changed 08 to 00, to remove 'red dot' in shape for '!'
0741	-	0742	Changed FE D4 to D4 FE to make shape '#' more correct
075B	-	0662	Moved 1 address up, to move shape '&' on line up
0A85	-	0AF7	Optimized SHAPE command handling to allow routine to be used by USB SHAPE
0BFC	-	0C7F	COLOR routine
0C5F			Changed LDI 63, PLO 9 to LDI 3F, PLO9. This value is used for the number of characters to be shaped in the 0C80 routine. The old routine shaped a max of 64 (40h) characters so even if 63h was loaded it would only change 40. The new optimize SHAPE routine shapes a max of 128 characters so using 63h causes too many characters to change color/shape. I changed the number to 3Fh as that excludes the 'green' 0.
0C62	-	0C63	Part of COLOR routine, changed sub routine call from 0C80 to FE5E, FE5E will call to 0C80 and after that set the last color shape mask back to NVRAM (b7/b6 on F3EA).
0C80	-	0D0A	Optimized 'SHAPE' routine to include a check on USB FLASH setting (bit 7 NVRAM

			F3EB)
0D0B	-	0D34	Part of TONE, MUSIC and NOISE routine, changed to skip sound setting if VOLUME is set 0.
0D35	-	0D61	Optimized VOLUME routine to include storing of VOLUME value in NVRAM (including parts on 22B2-)
0D62			Spare
0D63	-	0D96	CTONE Routine (unchanged)
0D97	-	0DA1	POP values from stack and restore RD, RC.0, RA.0 and R8.0. Used from CTONE and other routines
0DA7	-	0DF4	Removed POUT, TOUT routine and replaced with routines below.
0DA7	-	0DBE	Initial values 41C0 to 41D7
0DBF	-	0DCF	Spare
0DD0	-	0DDF	NOT used as it will be overwritten by the FDC if connected
0DE0	-	0DE8	Part of updated interrupt routine, including a call to E01F and clearing of key press if ESC was pressed
0DE9	-	0DF4	Spare
0E00			PLOAD
0E03			DLOAD
0E05	-	0E75	PLOAD / DLOAD
0F2B	-	0F2D	Call to FEDD as part of new CPOS routine to check if DWIDTH is active
0FFF	-	1027	Continuation of re-written start-up routine
0FFF	-	1005	Set 42BB = R2.1, store high byte stack pointer on 42BB Set 42B7 = 0
1006	-	1007	Call to 18D0, to set initial values for some system locations (key and character values)
1008	-	100B	Call to E18A to: - Call to 1AB5 (original code) to initialize default I/O routines on 428A-4291 - Get USB VOLUME setting from NVRM - Reset CNTL V/R and X buffers - Set LINE/SCREEN editor - Set PRINTER slot - Set USB COLOR - Set USB CHAR - Auto boot 80 column CARD.
100C	-	100E	Call to FD39, call 1A6C (original code), set 80 column if applicable and shape line 10 to character number for all characters.
100F	-	1011	Call to E215 to give 80 col WARM BOOT message if applicable. Give BASIC start-up message for 80/40 column modes; If needed do a call to FD39 for a DOS NEW
1012	-	1016	Call to E6B9 which will call DDFA in bank 0. DDFA will request initiation NVRAM if not done and then initiate NVRAM. Routine will return 0 if 'WARM' was selected and then skip the NEW routine below.
1017	-	1027	Initialize 4281/4282 to 440C (default BASIC DEFUS address), 4400 = 0 and call 1A47 (NEW)
1028	-	106D	Main BASIC interpreter loop first clearing flags and printing READY prompt
1028	-	1030	Call 1451 to clear bit 0 to 6 in R8.1 and initialize 429E-42A6 Check flags bit 0 and 1 on 429D if either = 1 skip 'READY' prompt and continue on 104B.
1031	-	1039	Print READY prompt on screen
103A	-	1047	Check if USB GRAPH is active (FEBE) then check if 80 column is active (E616) On return clear line after READY unless 80 column is active (if D != 0)
1048	-	104A	Call E37F to print the 'return' after 'READY' and reset b8 of F3F7 (CLOCK) as no program is running.
104B	-	104D	Call to 18FC, command line input routine

104E	-	1050	Call to 1E31, interpret command line
1051	-	1056	Check bit 0 of R8.1 if 1 (RUN mode) 'return' with a SEP 5 otherwise continue on 1057
1057	-	105C	Set RB = 42D0, which will store the interpreted BASIC command line
105D	-	1061	Check bit 2 of R8.1 if 1 execute 1062 if 0 execute 106A
1062	-	1069	Call 1C00 Call 2E42 (LF + CR) Loop back to 1028 / start of main loop
106A	-	106E	Call 1CFA Loop back to 1028 / start of main loop
106F	-	10B5	Error code handling
107B	-	107D	Call to FCAB to perform line feed including printer checks.
1085	-	1087	Call to E115 to print error code in text.
10AE	-	10B5	Set stack pointer back to bottom (top) and return to main loop
1227	-	122A	Replace POUT with CARD as in original expansion box
124E	-	132F	Command table shifted to replace TOUT with USB
1330	-	1342	Removed part of unused branching table
1330	-	1342	spare
13C6	-	13DB	F&M Screen editor adaptation
1406	-	142C	Default I/O Routine, output 1406, input 1409
1451	-	1474	Clear bit 0 to 6 in R8.1 Copy content 1475-147D to 429E-42A6
1475	-	147D	Default values for address 429E-42A6 (44h, start user RAM and 'empty call/return for BASIC functions
1400	-	1402	Changed start address 'RUN' to E36D to handle CLOCK off during run.
1407			
140A			
1414	-	142A	
1506	-	1507	
14A0	-	14A9	DEL
14A0	-	14A4	Print space (42AC) value on current cursor position
14A5	-	14A9	Print DEL (4280) value on current cursor position
1530	-	1531	Changed start address 'PLOAD' to FCD6 to do an INP 1 which will re-activate EF handling for tape.
1542	-	1543	Changed start address 'DLOAD' to FCDC to do an INP 1 which will re-activate EF handling for tape.
1550	-	1551	Changed start address 'CALL' to E373 to handle CLOCK off during CALL.
1552	-	1553	Changed start address VOLUME to 0D36 to allow a little more space to include storing of value in NVRAM
1558	-	1559	Changed start address 'DOS' to E010 which will call USB command routines which will check if a new SB DOS command is give. If not normal DOS routines will be called.
155A	-	155B	Changed start address 'CARD' to E817 as in original expansion ROM.
1564	-	1565	Changed start address 'EDIT' to E687 to handle incorrect arguments like ().
1568	-	1569	Changed start address 'USB' to F816 for command handling
156A	-	156B	Changed address 0DA7 to 0DA1, this is part of the TOUT and/or POUT routines and shouldn't really be used anymore. To be safe I changed it to an address with SEP 5.
15C0	-	15C1	Changed start address 'USR' to E379 to handle CLOCK off during USR.
17C9	-	17CC	Part of LIST code, intercepting if a USB command is detected. If so a call is made to E3CF to handle proper formatting of USB commands.
17F2	-	17F4	F&M Screen editor adaptation (part of EDIT code)

1840	-	1844	Part of EDIT code, calling FB6C to handle different EDIT behavior in screen editor.
18D0	-	18F8	Initialize: 4280 = 86, DEL key value 42AA = 8C, CNTL R key value 42AB = 8D, CNTL C key value 42AC = 20, spacebar key value 42AD = 85, clear to end of screen output value 42AE = 87, CNTL S key value 42B5 = 44 = ‘,’ 42B6 = 49 = ‘1’ 42CE = 43 = ‘+’ 42CF = 8D, CNTL C key value (used in EDIT, and 19AEh)
18F9	-	18FB	Call 2E42 to give a LF+CR before continuing command line input routine on 18FC
18FC	-	1988	Command line input routine R8.1 bit 0 1: program is running 0: command line mode
18FC	-	190A	RC.0 = R8.1 bit 0 (1 = RUN MODE, 0 = Command line) RC.1 = bit 1 and 0 from 429D If 429D bit 1 or 0 are 1 skip ‘:’ or ‘?’ If R8.1 = 0 branch to E10B to check status USB SCREEN and print ‘:’ if not active and return to 1910. If USB SCREEN is active return to 1913
190B	-	190D	If R8.1 = 1 branch to E180 to print ‘?’ followed by a ‘0’ character, return to 1910
1910	-	1912	Call 2CD3 to print character in D to current screen position (: or ‘0’)
1913	-	191C	Set R7 to command line buffer 4200, call ‘In hook’ on 428A and store resulting key input to stack (M(R2)).
191D	-	1922	If RUN mode (RC.0 != 0) or if command line buffer is not empty (R7.0 != 0) branch to 192D. For USB SCREEN mode R7.0 will always be != 0 as it is pointing to the 42D0 interpreted line buffer, possibly not fully optimized
1923	-	1928	Check if input is ‘CR’ (0Dh) if so loop back to 18F9 for a LF+CR and new input
1929	-	192C	Check if input is ‘space’ if so loop back to 1913 for reset to start of command line buffer and new input
192D	-	1932	Check if input is ‘CNTL C’ (42DA) if so loop back to 18F9 for a LF+CR and new input
1933	-	1937	Check if input is ‘CNTL R’ (42AA) if not continue on 1952
1938	-	1951	CNTL R (USB LINE) Copy 4000-405F (CNTL R buffer) to 4200-405F (command line buffer) and screen via the out-hook (428E). The buffer is copied until 0dh is found or 405F is reached.
1952	-	1956	Check if input is ‘DEL’ (4280) if not continue on 1961
1957	-	1960	DEL pressed If R7.0 = 0 (empty line buffer) do nothing and return to 18FC for ne input. Otherwise call 14A0 to handle ‘DEL’ and then wait for new input by branching to 1919.
1961	-	196D	Store input (stack value) on command line buffer M(R7) and CNTR buffer and increct command line buffer pointer with 1 (R7+1)
196C	-	1971	If CR was not pressed continue on 197C
1972	-	197B	CR pressed Set TAB value to 0 (429B), end routine on 1988 if RC.1 != 0 (429D bit 0/1). Otherwise print a LF (0Ah).
197C	-	1987	Check for command line buffer full (R7.0 = 60) if not continue waiting for next input at 1919 otherwise print ‘DEL’ character on screen via outhook 428E and continue on 195B to handle ‘DEL’
1988			SEP 5 to end command line input routine
1A47	-	1A7F	NEW
1A6C	-	1A7F	CLS, initialize 429D (flags), 429C, 42B3 to 0, 42BC to 41h, 42BA to 7Ah and 42B9 to B1.
1AB5	-	1ACE	Initialize 428A – 4291, copied from 1ACF-1AD6 (call to input/output routines)

1ACF	-	1AD6	Call to default input (1409) and output (1406) routines
1B9E	-	1BA0	Part of LET routine, calling E33E where a check is done on both CARD and USB statements to allow A=USB commands
1D01	-	1D02	Changed to FB66, to call subroutine to check if a line number >= FFFF has been given. Bug fix of original COMX BASIC.
1DFA	-	1E30	Spare
1E31	-	1EDC	Interpret command line
221C	-	22D8	Old RENUMBER location, RENUMBER moved to SB bank 7.
221C	-	22B1	USB CPEEK
22B2	-	22C8	Store VOLUME value on RA (41C9) and NVRAM
22C9	-	22D8	Spare
241A	-	241D	LBR E328, NOP. String assignment handling, allowing A$=USB commands.
2600	-	2602	LBR E35D, part of IF handling to allow USB commands to be used in IF statements.
26C8	-	26F8	INPUT
26C8	-	26D0	Check is BASIC is in RUN mode (R8.1 bit 0 = 1) if not give ERROR 11h
26D1	-	26D8	If an open quote (CFh) is detected on BASIC line (M(RB)) call 2E35 to print all text specified on line until a close quote is detected (CEh)
26D9	-	26DD	Fetch low byte of current interpreted command line buffer from 4287 and high byte from Rf.1 store as one pointer in RE (42D0 range)
26DE	-	26E6	Check if INPUT is requesting input to a string variable (M(RB) = D7h) if so branch to input string routine on 2E49.
26E7	-	26EA	Call to E2B0 to handle input of the integers (104B) and handle empty lines as well as error message for string. On return the correct low byte of the interpreted command line is in D and stored in RE.0.
26EB	-	26ED	Store the integer on the interpreted command line in the correct variable (call to 3000)
26EE	-	26F1	Check if more variables are specified on BASIC line (comma / C2h found), if so loop back to 26E7
26F4	-	26F8	Store current low byte of interpreted command line (42D0) on 4287, command end.
26F9	-	26FF	Spare
280D	-	280F	LBR E34D, part of PRINT routines, to allow USB commands to print return values.
2A05	-	2A07	LBR FD91, string assignment handling, allowing A$=USB commands.
2AEE			Changed 3A to 30 (BNZ to a BR)
2CD3	-	2CFF	Print D on current screen position
2D7B			Changed 3A to 30 (BNZ to a BR)
2E35	-	2E41	Print data starting on M(RB) on current screen position until a close quote (CEh) is detected
2E42	-	2E48	Print LF + CR
2E49			INPUT string
3000			Store the integer on the interpreted command line in the correct variable
321E	-	3225	CLS, one NOP removed on 3225
3226	-	322A	Spare
322B	-	3222	Print characters after SEP 4 call to current screen position until ‘0’
3D0B	-	3D0C	LBR FAA9. To handle bug in READ feature which crashed if no DATA statements are available.

RAM (4000 - BFFF)

4000 - 405F	CTRL R Buffer
4100 - 41xx	Scratch area?
4193 - 4194	Cursor position within @F800-@FFFF location, default F800h
4195 - 4196	Top left screen position within @F800-@FFFF location, changes after scrolling, default F800h
4197 - 4198	Cursor position, assuming top left corner = 0, default 0
4199	? default FFh
419A	Column counter, default 0
419B	Cursor on/off (off = 0), default FFh
419D	Timer on/off (off = 0)
419E - 41A0	24 bits timer value, count back to 0 if @419D != 0
41A1 - 41A2	Start address timer, a jump is needed to @054E at the end of timer routine. Usable registers: B, A and E, register 1 is program counter.
41A3	Raw keyboard value, changed after changing value to 0
41A4	Direct keyboard value
41A5	Counter for key repeat
41A6 - 41A7	Start address ESC routine
41A8	If ESC is pressed equals to #FF
41AE - 41B3	Printer Settings and info
41B4 - 41BF	Spare?
41C0	Last value OUT 3
41C1	Low byte of last OUT 4
41C2	High byte of last OUT 4
41C3	Low byte of last OUT 5
41C4	High byte of last OUT 5
41CC - 41CF	Call to MUSIC routine
41D0 - 41D3	Call to TONE routine
41D4 - 41D7	Call to NOISE routine
41D8 - 41E0	Spare
4280	DEL key value, 86h
4281 - 4282	DEFUS value
4283 - 4284	EOP (End Of Program) value
4287	Low byte of current position on 42D0 interpreted command line
428A - 428D	In hook, call to input routine
428E - 4291	Out hook, call to output routine
4292 - 4293	End array/start string value
4294 - 4295	Start array value
4296 - 4297	Line number TIMOUT
4299 - 429A	EOD (End Of Data) and end string value
429B	X value TAB
429C	? default 0
429D	Flags, default 0 Bit 0: if 1 no READY prompt should be given Bit 1: if 1 no LF should be done
429E	Start address user RAM, default 44h
429F - 42A2	Call to BASIC functions
42A0	Used by COMX BASIC for start address of latest used command. FW uses this in

	certain cases to detect what type of command is being used. For example to differentiate between PRINT USB Q and A=USB Q
42A1	Used during command handling to 'save' R8.1.
42A3 - 42A6	Subroutine call to the requested USB or DOS command. The same locations are also used by COMX BASIC for some command handling subroutine calls
42A8	Used by the F&M Screen Editor, 0 line editor, 1 screen editor, 2 USB BROSWER editor
42A9	RE.1 at call of 1AB5 routine (initialize normal I/O calls), used in CALL/USR to restore to RE.1 before calling user routine. Don't understand the purpose.
42AA	CNTR R key value, 8Ch
42AB	CNTR C key value, 8Dh
42AC	Spacebar key value, 20h
42AD	Clear to end of screen output value, 85h for 40 column and 0Dh for 80 column mode
42AE	CNTL S key value, 87h
42B1 - 42B2	DATA pointer
42B3	?, default 0
42B5	',', 44h
42B6	'1', 49h
42B7	?, default set to 0
42B9	?, default B1h
42BA	?, default 7Ah
42BB	High byte value of R2 / Stack pointer, normal start value BDh
42BC	?, default 41h
42CE	'+', 43h
42CF	CNTL C key value (used in EDIT and 19AEh), 8Dh
42D0 - ?	Interpreted BASIC command line
4300 - 4368	Storage A, B, C.... Z, 4 bytes per variable
4370 - ?	Command parameter buffer
43F9	Used by the F&M Screen Editor for the last character on the cursor position
43F9 - 43FF	USB TDSAVE / TDSAVE+ buffer
43FD - 43FF	USB DATE buffer
4400 - Stack	User RAM
BDFF	'Start' of stack
BE80 - BEFF	DOS buffer but also used by USB commands for the USB PC <-> COMX buffer. Also used by some USB commands (WORM, TENNIS, SPACE, MON, DMON) for storing temporary data.
BF42	Current OUT 1 value (selected slot/bank)
BFFB	USB PLOT Character number pointer (to DF80-DFFF)
BFFC	Graphic Screen + 1 (0 = off)
BFFD	Printer slot if connected
BFFE	OUT 1 value (selected band/slot) at USB/DOS command entry
BFFF	Last specified USB CHAR

SB ROM Bank 0 (C000 - DFFF)

C000	0, indicating ROM
C001	21, indicating SB FW ROM bank 0
C002	10, indicating bank 0 slot code 10
C003 - C00F	Identification text: SBV1.2 - MAIN
C010 - C012	LBR C915, USB command handling routine called is ROM is switched in via a CARD command
C013 - C3F5	USB command table: - Command text, end is indicated by a code with b6 and b7 = 0 Instead of text also a command code could be used (identified by >80) i.e. SCREEN = A4. - Type code: b0 = 0 command always returns a value (STR/INT) b1 = 1, command can return STR / INT b2 = 1, command will exit 'USB command' routine and branch back to BASIC b3 = DOS command b4 = 1, command will switch printer off b5 = not used b6/b7 = 0 (used as end code for command text) - Start address (2 bytes) - Slot code (if != 0) - Second type (if != 0, used when different types are needed for one command. Example: USB Q and INT = USB Q. - Second start address (2 bytes, if second type was detected) - Second slot code (if != 0) - End command code = 0 Note for commands including sub text of other commands the shortest command should come latest in the table. i.e. TIME should come after TIMEAM.
C417 - …	USB PSAVE
C53A - …	USB PLOAD
C8C1 - …	USB DSAVE
C9E3 - …	USB CD
CA1B - …	USB MKDIR
CA34 - …	USB CAT
CC45 - …	USB RMDIR
CC77 - …	USB DEL
CD77 - …	USB RTCPC
CDBA - …	USB SYSDEF
CE12 - …	USB DATE
CFFD - …	USB TIME
D0B8 - …	USB TIMEPM
D0D0 - …	USB TIMEAM
D163 - ...	USB NVSAVE
D17E - …	USB SYSSAVE
D199 - …	USB NVLOAD
D201 - …	USB SYSLOAD
D26E - ...	USB INP
D2CF - …	USB OUT
D35F - ...	USB HISTORY

D3CB - ...	USB NVPUT
D533 - ...	USB NVGET
D595 - ...	USB NVIGET
D62F - ...	USB NVSTART
D5F4 - ...	USB NVSGET
D730 - ...	USB NVCLR
D766 - ...	USB NVDEL
D78B - ...	USB NVLIST: startup, continue in bank 1 on DAED
D7D6 - ...	USB RSAVE
D818 - …	USB RLOAD
D888 - ...	USB MOVE
D8F0 - ...	USB IMGTODISK
D98F - ...	USB IMGTODISK
D9F9 - ...	USB IMGTEST
DA13 - …	USB JOY
DA1E - …	INT = USB Q
DA2D - …	USB EF
DA38 - ...	USB POS
DA43 - …	USB DLOAD
DAEF - ...	USB BSAVE
DB7E - ...	USB BLOAD
DC57 - ...	USB COMP
DD51 - DD57	Check online mode if off return 0
DD58 - ...	USB HSWEB
DD84 - ...	USB URL
DDA6 - ...	USB URLGET
DE25 - DE47	USB BROWSER
DE48 - DEC6	If NVRAM is not initiated, request initiation and then initiate NVRAM. Initiate NEW unless WARM start was selected. NEW jump to 102D otherwise 103E
DEC7 - ...	URL DISP
DE9D -	USB BROWSER: URL ‘input’ save routine
DEEF - …	USB EMAIL
DF09 - DFB7	Some text
DF81 - DFDF	Spare
DFE0 - DFE3	Text: ‘ AM’
DFE4 - DFE8	Date/time mapping table
DFE9 - DFEC	Text: ‘ PM’
DFED - DFEF	RTC mapping
DFF0 - DFF3	Date/time mapping table
DFF4 - DFFF	Number of days per month (DFF4 = Jan, DFFF = Dec)

SB ROM Bank 1 (C000 - DFFF)

C000			0, indicating ROM
C001			22, indicating SB FW ROM
C002			30, indicating bank 1 slot code 30
C003	-	C00F	Identification text: SBV1.2 - ERR
C010	-	C012	LBR F800, USB command handling routine called is ROM is switched in via a CARD command
C013	-	C13B	Text error routines
C10B	-	C125	Find and print error text Input: RA.0 = error code
C126	-	C13B	Find error text and store in BE80 buffer Input: RA.0 = error code
C13C	-	C15C	Call subroutine in other bank Input: M[P+1]=slot M[P+2/3]=address
C15D	-	D792	Error text
D818	-	D8BF	Spare
D8C0	-	D95B	Game/program names, used in USB NVLIST
D95C	-	D969	Store new CHAR value in NVRAM Input: D=new CHAR
D96A	-	D98C	Fetch command arguments
D98D	-	DA7F	USB CHAR
DADB	-	DCC4	Continue USB NVLIST (main code)
DD08	-	DD7A	USB VER
DDC5	-	DE05	- Get USB COLOR mask and set as last COLOR mask in NVRAM. - Get USB CHAR from NVRAM and call CHAR. (DDEF) - Get 80 column auto boot from NVRAM is set call E203
DE14	-	DE3D	USB BROWSER re-shape routine
DE3E	-	DE6C	Switch off USB GRAPH (called if at ‘READY’ prompt) no USB GRAPH (0) was done.
DE6D	-	DF12	spare
DF13	-	DFFC	Lower case shapes
DFFD	-	DFFF	FW Version date: DD - MM - YY

SB ROM Bank 2 (C000 - DFFF)

C000			0, indicating ROM
C001			22, indicating SB FW ROM
C002			50, indicating bank 2 slot code 50
C003	-	C00F	Identification text: SBV1.2 - HELP
C010	-	C012	LBR F800, USB command handling routine called is ROM is switched in via a CARD command
C013	-	..	HELP print routines
C216	-	D95D	HELP text per command
D95E	-	DB40	spare
DB41	-	DCA2	HELP text for HELP command
DCA3	-	DCCB	Shape cursor to original
DD4B	-	..	USB PLIST
DD7D	-	..	USB PPR / USB PPRINT
DD97	-	..	USB PTEST
DDD3	-	..	USB PMEMDUMP
DDEE	-	..	USB PON
DE26	-	..	USB POFF
DE57	-	..	USB PSET
DF7C	-	..	USB HELP
DfC1	-	DFDD	Shape cursor to original and save R8, RC and RE on stack
DFDD	-	DFF5	spare
DFF6	-	DFFF	Original cursor shape

SB ROM Bank 3 (C000 - DFFF)

C000			0, indicating ROM
C001			22, indicating SB FW ROM
C002			70, indicating bank 3 slot code 70
C003	-	C00F	Identification text: SBV1.2 - DISK
C010	-	C012	LBR F800, USB command handling routine called is ROM is switched in via a CARD command
C013	-	..	DOS BOOT
C0E0	-	..	DOS LOCK
C260	-	..	DOS UNLOCK
CEA1	-	..	DOS INIT
CEF5	-	..	DOS STARTUP
CF2B	-	..	DOS LABEL
D031	-	..	DOS DISKCOPY
D0B7	-	..	DOS FILECOPY
D100	-	D129	Copy USB BROWSER text (DE32-DE67) to RAM for use in browser
D12A	-	D19A	Checkbox and radio routine (USB BROWSER)
D19B	-	D219	7F (127 decimal) bytes spare
D21A	-	D6F6	Part of COPY code, will be copied to AF00 on DOS DISKCOPY
D6F7	-	D743	Part of COPY code, will be copied to B400 on DOS DISKCOPY
D744	-	D834	Part of INIT code, will be copied to B400 on DOS INIT and DOS STARTUP
D834	-	D8A8	Part of INIT code, will be copied to B300 on DOS INIT and DOS STARTUP
D8A9	-	DAFF	Part of INIT code, will be copied to B000 on DOS INIT and DOS STARTUP
DB00	-	DEFF	BOOT code will be copied to B700 on DOS BOOT
DE32	-	DE7F	Text: ‘gO TO url:’, ‘lOADING PAGE…’ and ‘sENDING EMAIL’ including spacing and reversed to allow for USB CHAR(3) use in USB BROWSER
DE80	-	DE9F	20 (32 decimal) bytes spare
DEA0	-	..	USB TSAVE and TSAVE+
DEA3	-	..	USB TDSAVE and TDSAVE+
DEA6	-	..	USB TLOAD and TLOAD+
DEA9	-	..	USB TDLOAD and TDLOAD+
DF00	-	DF80	Part of INIT code, will be copied to B4F0 on DOS INIT
DF81	-	DFFF	Part of STARTUP code, will be copied to B4F0 on DOS STARTUP

SB ROM Bank 4 (C000 - DFFF)

C000	0, indicating ROM
C001	22, indicating SB FW ROM
C002	90, indicating bank 4 slot code 90
C003 - C00F	Identification text: SBV1.2 - MON
C010 - C012	LBR F800, USB command handling routine called is ROM is switched in via a CARD command
C013 - C028	MON: text 'PROTECTED and 'UNPROTECTED'
C029 - C034	MON: text 'HEX.' and 'TEXT'
C035 - C057	Find printer CARD, if found store slot code on BFFD
C058 - C095	Print number routine 1
C05D - C095	Print number routine 2
C096 - C245	MON: screen 'READ REGISTERS'
C246 - C300	MON: screen 'AUTO STORE'
C300 - C44A	MON: screen 'CHANGE' 2nd screen
C44A - C4E2	MON: screen 'DUMP'
C4E3 - C4EB	MON: text '↑DELETE'
C4EC - C51D	MON: table for address positions 'CHANGE'
C51E - C526	MON: text '↑INSERT'
C527 - C531	MON: text 'WHITE' and 'BLUE'
C532 - C612	MON: screen 'MOVE'
C613 - C6E7	MON: screen 'CHANGE' 1st screen
C6E8 - D80B	Continue USB MON (main code)
D80C - D823	Part of SYSDISP printing off or on
D824 - DC31	USB SYSDISP
DC44 - DC4C	Print to screen routine via DC3B Print all characters M[R6] and increase R6 until '0' is found
DC4D - DC9D	Print routine checking USB CHAR setting, so depending on setting lower case will or will not be used.
DC9F - DCBD	Part of USB SYSDISP, routine will switch to printer card and check if a serial or parallel card is connected.
DCBE - DCE2	Part of USB SYSDISP
DCE3 - DD05	Check if NVRAM is writable / available, if not give error message 5B or 61
DD06 - DD3A DD54 - DD60	Check if printer is active, if active select printer, branch to M[R6+1/R6+2] and switch back to slot on M[R6]. If no printer active just branch to M[R6+1/R6+2].
DD3B - DD53	Error code routine
DD61 - DD6D	Part of USB SYSDISP, printing 'Printer'
DD6E - DE13	USB BROWSER passwd input routine
DE13 - DE57	spare
DE58 - DEEF	USBBROWSER passwd input return jump table
DE60 - DFFF	MON: startup screen

SB ROM Bank 5 (C000 - DFFF)

C000	0, indicating ROM
C001	22, indicating SB FW ROM
C002	B0, indicating bank 5 slot code B0
C003 - C00F	Identification text: SBV1.2 - TENN
C010 - C012	LBR F800, USB command handling routine called is ROM is switched in via a CARD command
C013 - C20E	TENNIS: Start-up code
C20F - C6DA	BROWSER: Main code
C6DB - C670C	32 (50 decimal) bytes spare
C70D - C7BB	TENNIS: Screen A / B
C7BC - C7CB	TENNIS: GAME OVER text
C7CC - C8FE	TENNIS: Screen C / D
C8FF - CA1D	TENNIS: Screen E / F
CA1E - CAB4	TENNIS: Shapes
CAB5 - CADD	TENNIS: Screen G / H
CADE - CD81	TENNIS: Game tables (byte 1: ?, 2: game number, 3: game letter, 4+5: high score location, 6+7: screen map location)
CD82 - CE9F	TENNIS: Screen I / J
CEA0 - CFFF	TENNIS: Screen K / L
D000 - DE46	TENNIS: Code main part
DE47 - DE48	2 bytes spare
DE49 - DFAC	TENNIS: HELP text
DFB8 - DFD7	TENNIS: Some table
DFD8 - DFFF	TENNIS: row 24 text

SB ROM Bank 6 (C000 - DFFF)

C000			0, indicating ROM
C001			22, indicating SB FW ROM
C002			D0, indicating bank 6 slot code D0
C003	-	C00F	Identification text: SBV1.2 - DMON
C010	-	C012	LBR F800, USB command handling routine called is ROM is switched in via a CARD command
C013	-	C015	WORM: Interrupt routine
C016	-	C023	DMON: WHITE / BLUE text
C024	-	C04B	DMON: Shapes
C04C	-	C069	WORM: Shapes
C06A	-	C099	DMON: Screen locations 'change'
C09A	-	C179	DMON: Start screen
C17A	-	C188	DMON: Error text
C188	-	C18C	DMON: 15 spaces
C18C	-	C26E	DMON: Change screen
C26E	-	C2DE	DMON: Dump screen
C2Df	-	C309	DMON: shape routine
C30A	-	C324	Print routine, print value on m[R8] and increase R8. If R8 = 0, take next byte as character and following as the number and repeat that character 'number' times. End by 2 zeros.
C544	-	..	DMON: main code
C9AE	-	CA1B	DMON: startup code, initiate memory etc.
CA40	-	..	USB CTONE
CA48	-	..	USB COLOR
CA6B	-	CA8D	Spare
CABB	-	..	USB CLOCK
CB03	-	..	WORM: main code
D1B9	-	D1D9	Call subroutine in other bank Input: M[P+1]=slot M[P+2/3]=address
D1DA	-	D1F1	WORM: switch interrupt off, store high score etc
D1F2	-	D1FE	WORM: sound routine
D1FF	-	D227	Sound off routine
D228	-	..	USB RTCFT
D253	-	..	USB OLD
D2E0	-	..	USB LINE
D312	-	..	USB SCREEN
D34F	-	..	USB NVPOKE
D529	-	D539	Get (x) from command line and return x in D
D53A	-	..	USB TV
D55D	-	..	USB Q
D567	-	..	USB DMON: start code like CHAR and CLS routines
D6F2	-	..	USB WORM: start code like CHAR and CLS routines
D75A	-	D786	USB SPACE: start code like CHAR and CLS routines
D79C	-	D7B9	USB GRAPH
D7BA	-	D7BE	spare

D87B - ..	USB PCNTL
D8DB - ..	USB DWIDTH
D8F7 - ..	USB DHEIGHT
D917 - ..	USB TENNIS: start code like CHAR and CLS routines
D939	USB POS
D9D1 - ..	USB MON: start code like CHAR and CLS routines
D9F0 - ..	USB LOGOTUNE
DA0A - ..	USB GO (40, 80 and 80(x))
DA60 - ..	USB BOOTMSG
DA97 - ..	USB PTV
DAC6 - ..	USB PKB
DAE3 - ..	USB HEX
DB38 - ..	USB BIN
DB7B - ..	Continue LET CARD command check
DBB3 - ..	LET CARD F routine
DBF7 - ..	USB ONLINE
DC83 - ..	USB CLS
DCDF - DDB1	USB MEMDUMP
DDB2 - DDE7	Shape routine for USB BROWSER
DDE8 - DDF7	USB BROWSER: CLS + Shape (was C650)
DDF8 - DE16	USB BROWSER: Shape standard char set (was C510)
DE17 - DE56	USB BROWSER: input CR routine
DE57 - DE6C	Check on end of command
DE6D - DE85	spare
DE86 - DF09	USB BROWSER: Shapes
DF0A - DF17	WORM: text ‘C R A S H !!’
DF18 - DF3B	WORM: text first line
DF40 - DFFF	WORM: table

SB ROM Bank 7 (C000 - DFFF)

C000			0, indicating ROM
C001			22, indicating SB FW ROM
C002			F0, indicating bank 7 slot code F0
C003	-	C00F	Identification text: SBV1.2 – SPAC
C010	-	C012	LBR F800, USB command handling routine called is ROM is switched in via a CARD command
C013	-	C01A	SPACE: Jump table
C01B	-	C022	SPACE: Data address table
C023	-	C037	SPACE: table
C038	-	C03F	SPACE: data
C040	-	C0A6	SPACE: data
C0A7	-	C0C0	Copy data to RAM bank routine
C0C1	-	C0E2	SPACE: data
C0E3	-	C104	SPACE: data
C105	-	C133	SPACE: data
C134	-	C157	SPACE: data
C158	-	C17A	SPACE: data
C17B	-	C1A7	SPACE: data
C1A8	-	C1CD	SPACE: first screen row
C1CE	-	C277	SPACE: shapes
C278	-	D28F	SPACE: code
D290	-	D316	SPACE: code, initiation and high score routines
D317	-	D337	Call subroutine in other bank (start-up; calling common routine at FC51)
D338	-	D355	SPACE: high score handling and call to HSWEB
D356	-	DE34	SPACE: return to BASIC (B key)
D388	-	D3A0	SPACE: all sound off
D3A1	-	D3B9	SPACE: Sound on/off (S key)
D3BA	-	D3C4	SPACE: sound
D3C5	-	D4E6	USB BROWSER input routine
D52B	-	D5EF	USB GRAPH (continued, jump in from bank 6 D7B7
D5D2	-	D649	USB PLOT (X, Y, color, char) USB PLOT (X, Y, color) Command handling resulting in: R8.0 = X RA.0 = Y R7.1 = Shape color bits (7/6), character color bit (0) R8.1 = character number
D64A	-	D737	PLOT (X, Y, color) On input: R8.0 = X RA.0 = Y R7.1 = Shape color bits (7/6), character color bit (0) RB = DF4x screen info pointer Destroys content of R9, RC, RF Code: - Set RC = Cxxxx (screen location in bank 7 based on X, Y and RB)

	- Check if X / Y are in screen range if not exit - RD, R8.0, RA.0 SAVED on Stack - RD = F808 / F80F to store and read from RAM bank 7 - Calculate RC to specify character covering X, Y-> rest values stored in R8.0/R8.1 - If char height != 10 read current character from char memory (D4 EF55 uses R9 as screen position by adding 3800 to RC) otherwise read from RAM bank 7 with SEP D. - If current char = 0 get next char from R9 (BFFB) pointer to DF80 area - Re-shape character to space (only for height != 10) using R9 and RF - Store character on screen (height !=10) or in RAM bank 7 with SEP D - Add 3800 to RC so it can be used as screen pointer - Store character and color bit on RC and R9 (i.e. print char on screen and on shape pointer) - Shape correct line via rest values of X/Y R8.0/R8.1 using RD and RF and pixel color from R7.1
D738 - D7BE	PLOT (X, Y, color, char)
D7BF - D820	LINE OCTANT 1
D821 - D881	LINE OCTANT 0
D882 - D8B3	LINE (X, Y, color, X2, Y2)
D8B4 - D8CD	USB LINE(X, Y, color, X2, Y2) Command handling R8.0 = X RA.0 = Y R7.1 = Shape color bits (7/6), character color bit (0) R8.1 = X2 RA.1 = Y2
D8CE - D8DD	USB CIRCLE(X, Y, color, radius) Command handling
D8DE - D99A	CIRCLE(X, Y, color, radius)
D99B - D9AA	USB ELLIPSE (X, Y, color, A, B)
D9AB - DA87	ELLIPSE(X,Y, color, A, B)
DA88 - DA8F	Error codes USB LIGHTRM / HIDERM
DA90	USB LIGHTRM entry
DA93	USB HIDERM entry
DA90 - DAD6	USB LIGHTRM / USB HIDERM handling
DAD7 - DB0B	LIGHTRM
DB0C - DBBA	HIDERM
DBBB DD5C	RENUMBER
DD5D - DD88	Clear all character shapes from line 9
DD67 - DD88	Clear all character shapes
DD89 - DDBE	USB DCHAR
DDBF - DDEE	USB CPOKE
DDEF - DDFF	spare
DE00	Return from calculation routine
DE01 - DE02	Select calculation routine RD = DB01 RF is 16 bit input for multiply R9, B and C used in calculation routines as pointers to variables
DE03 - DE59	Multiply, call via SEP D 03 var1 Result RF = var2 Result (RC)= var1(R9)*var2

			Variables: 87 (BE84-BE87), 8b, 8f, 93 etc.) 94-97 USED as addition factor in multiply Routine does not work with negative numbers
DE5A	-	DE72	Add, call via SEP D 5A var1 var2 Result Result(RC)=var1(R9)+var2(RC)
DE73	-	DE88	Sub, call via SEP D 73 var1 var2 Result Result(RC)=var1(R9)-var2(RC)
DE89	-	DE9D	Multiply * 2, call via SEP D 89 var1 Result Result(RC)=var1*2 Works for negative numbers in var1.
DE9E	-	DEA5	USB SHAPE
DEA6	-	DEB5	USB FLASH
DEB6	-	DEE1	USB PPOKE
DF0D	-	DF35	spare
DF36	-	DF3F	Line numbers 10, 100, 1000, 10000, FFFF for USBHIDE RM space calculations
DF40	-	DF80	USB GRAPH screen details Graph 0 (DF40) START LOC: C1E2/F9E2, X: 38*6=228 (#E4), Y: 12*9=108 (#6C), Y offset = 0, char height=9, number of char per line 40 / #28 Graph 1 (DF48) START LOC: C208/FA08, X: 40*6=240 (#F0), Y: 13.5*16=216 (#D8), Y offset = 8, char height =16, number of char per line 40 / #28 Graph 2 (DF50) START LOC: C0DC/F8DC, X: 20*6=120 (#78), Y: 12*9=108 (#6C), Y offset = 0, char height =9, number of char per line 20 / #14 Graph 3 (DF58) START LOC: C078/F878, X: 20*6=120 (#78), Y: 6.75*16=108 (#6C), Y offset = 4, char height =16, number of char per line 20 / #14 Graph 4 (DF60) START LOC: C1E2/F9E2, X: 38*6=228 (#E4), Y: 12*8=96 (#60), Y offset = 0, char height =8, number of char per line 40 / #28 Graph 5 (DF68) START LOC: C1E0/F9E0, X: 40*6=240 (#F0), Y: 13*16=208 (#D0), Y offset = 0, char height =16, number of char per line 40 / #28 Graph 6 (DF70) START LOC: C0DC/F8DC, X: 20*6=120 (#78), Y: 12*8=96 (#60), Y offset = 0, char height =8, number of char per line 40 / #14 Graph 7 (DF78) START LOC: C079/F879, X: 18*6=108 (#6C), Y: 6.75*16=108 (#6C), Y offset = 4, char height=16, number of char per line 40 / #14
DF81	-	DFFF	USB PLOT, LINE, CIRCLE character list, pointed by M(BFFB) + DF80

SB RAM Bank 8-12 (C000 - DFFF)

Not used by SB FW and as such available for user programs.

SB RAM Bank 13 & 14 (C000 - DFFF)

C000 - DFFF	Used by commands DISKCOPY and FILECOPY

SB RAM Bank 15 (C000 - DFFF)

C000 - D5FF	RAM swap area, mainly used in DOS commands as well as USB GRAPH, PLOT, LINE & CIRCLE for storage of screen content
DB0A - DB0F	F&M Disk Monitor data storage
DB10 - DC3C	Spare
DC40 - DC7F	F&M Monitor data storage
DC80 - DCE0	CTNL X and V buffer
DCFE - DCFF	Pointer to current CNTL R buffer
DD00 - DFFF	CNTL R buffer

SB FW ROM (E000 - E7FF)

E000	-	E00F	Data table for joystick handling
E010	-	E012	DOS command entry which will call USB command entry at F816.
E013	-	E015	USB HSWEB for use from external SW (call to E6AA)
E016	-	E018	USB MON 'register check' call (call to E681)
E019	-	E01B	NVPUT: R8.1 SW ID, source = RC, R8.0 = number of bytes (call to E66D)
E01C	-	E01E	NVGET: R8.1 SW ID, dest = RC, R7.0 = 2 (call to E677)
E01F	-	E10A	Updated interrupt routine including printing of clock if needed and JOY handling.
E10B	-	E113	Part of command line input routine, checking if USB SCREEN (42A8 != 0) is active. If so skip printing of ':'
E115	-	E146	Print error in TEXT via C10B routine in bank 1
E147	-	E156	Return to slot stored on BFFE
E157	-	E16D	Write NVRAM location F300 + M[R6]
E16E	-	E17F	Read NVRAM location F300 + M[R6]
E180	-	E189	Part of line input routine printing a '?' followed by a byte 0 character.
E18A	-	E202	- Store VOLUME value from NVRAM to 41C9 - Switch in bank 15 and reset CNTL X/V and R buffers. - Get screen/line editor info from NVRAM and store on 42A8. - Find PRINTER (standard or thermal) CARD, store slot code on BEFD. - Select bank 1 and call DDD8, which sets USB COLOR, USB CHAR and auto boot 80 column
E203	-	E214	Auto boot 80 column CARD
E215	-	E2AF	If 80 column boot ongoing: give WARM BOOT message if applicable, give BASIC start-up message for 80/40 column modes. If needed do a call to FCF8 for a DOS NEW
E2B0	-	E308	Handling of empty line and string in INPUT integer command
E309	-	E327	Check if NVRAM is working correctly Return: D=0, OK D=FF, error
E328	-	E33D	String assignment handling, allowing A$=USB commands
E33E	-	E34C	Part of LET routine, added check is on both CARD and USB statements to allow A=USB commands.
E34D	-	E35C	Part of PRINT routines, to allow USB commands to print return values.
E35D	-	E36C	Part of IF handling to allow USB commands to be used in IF statements.
E36D	-	E372	Part of RUN code; call E3A0 to handle CLOCK off during RUN and CALL original RUN code at 1F76
E373	-	E378	Part of CALL code; call E3A0 to handle CLOCK off during RUN and CALL original CALL code at 2C00
E379	-	E37E	Part of USR code; call E3A0 to handle CLOCK off during RUN and CALL original USR code at 2C03
E37F	-	E3CE	CLOCK off during RUN/CALL handling. If F3F7 contains invalid value do nothing. If b8=1 reset it to 0.
E3A0	-	E3CE	Switch clock off during RUN / CALL
E3CF	-	E470	Part of LIST code to properly format USB commands
E471	-	E49A	Part of error text routine to handle slot handling if printer or 80 column is active
E49B	-	E505	Store current line in CNTL R buffer (triggered on 'CR') At end of routine if R7.0 = 0 set RF.1 to 8D to force an empty line in screen editor mode to come back on next line without error code
E506	-	E54D	CNTL R routine (called when CNTL R is pressed)

E54E - E55D	Get current CNTL R buffer (from bank 15, DCFE) and return location in RE.
E55E - E569	Select slot as stored on BF42
E56A - E583	Store new CNTL buffer location (RE) to bank 15, DCFE. If no RAM found RE=4000
E584 - E590	Call routine on 42A3 with slot as in D; always return to slot 10.
E591 - E5D4	Call disk routine Input: address: M[R6]+M[R6+1] return slot m[R6+2]
E5D5 - E5EA	Select disk card
E5EB - E5FE	Error code routine which will return as from a normal subroutine call Input: M[R6]= error code number
E5FF - E615	Check on WARM boot possibility Return: 0=WARM NOT 0=COLD
E616 - E620	Check if 80 column card is active Return: NOT 0=80 column active 0=80 column NOT active
E621 - E635	Reset F3ED bit 5, printer off on error code
E636 - E644	Set slot back to BFFE and call 2E25
E645 - E654	Set slot back to BFFE + Error code routine Input M[R6] = error code This routine DOES NOT set R8 back to entry value; i.e. only use from CARD routine
E655 - E66C	USB CARD F search for slot routine
E66D - E676	NVPUT: R8.1 SW ID, source = RC, R8.0 = number of bytes (call to bank 0, slot 10, D457)
E677 - E680	NVGET: R8.1 SW ID, dest = RC, R7.0 = 2 (call to bank 0, slot 10, D562)
E681 - E686	USB MON 'register check' call (call to bank 4, slot 90, D50E)
E687 - E69F	Part of EDIT command, check command line to make sure a valid argument is specified like for example ().
E6A0 - E6A9	HSWEB for use from USB commands (call to bank 0, slot 10, DD58)
E6AA - E6B8	HSWEB for use from external SW
E6B9 - E6C6	Call DE48 in bank 0.
E6C7 - E6D3	Updated print routine, on B1 / EF1 store character on screen directly, if not store it in print buffer for printing via interrupt routine.
E6D4 - E6E3	Called on line buffer overflow, check is done to see if USB BROWSER is active (42A8 = 2) if so continue otherwise give error code 27
E6E4 - E6F2	Called on CNTL S press, if USB BROWSER is active ignore otherwise call CNTL S routine to clear screen
E6F3 - E713	Called on cursor down, if USB BROWSER is active ignore otherwise call scroll routine
E714 - E71A	RENUMBER call to bank 7
E71B - E7FF	spare

EXPANSION ROM (E800 - EFFF)

All changed locations compared to the original EXPANSION ROM are listed. Note that the original EXPANSION ROM also had a feature which disabled COMX ROM location 1000-17FF and selected E000-E7FF instead. These locations are not listed but are listed as part of the COMX ROM chapter. E000-E7FF is used for SB FW instead.

E812 - E816	End part of CARD routine which will call routine on EBA2 which will check if a PR or LET statement was used for the CARD command. If so a call is made to 2E25 to return the value to BASIC.
E83E - E844	Start of CARD routine, added a check if PR or LET statements was used if so branch to EBB1 to handle those CARD statements.
E845 - E846	NOP; NOP
E847 - E867	Check CARD sub command: B -> EA69 (was EA6D in original ROM) F -> EBD5 (new) P -> E9AA (not changed) Q -> EB0A (not changed) S -> EB57 (not changed) T -> E8FB (not changed) Any other value branches to EC9D to give an error message (5A / decimal 90)
E868 - E86C	F&M Screen editor adaptation
E897 - E8AE	F&M Screen editor adaptation (branch to EC9D changed to ECA1)
E985 - E9A9	Removed error text message 'NO THERMAL PRINTER CARD' replaced by routines below.
E985 - E987	Call error code routine with error 5A / decimal 90 to replace text message.
E988 - E9A9	F&M Screen editor adaptation
EA50 - EA6C	Removed error text message 'NO PRINTER CARD' replace by routines below.
EA50 - EA52	Call error code routine with error 5A / decimal 90 to replace text message.
EA53 - EA6B	Check if screen editor is active (42A8 = 0). If not active execute line editor code at EA59-EA62. If active execute code at EA63-EA6B
EA6D - EB09	Removed (rewritten) CARD B code with below routines
EA69 - EA8C	New CARD Bx code
EA8D - EA97	Print FW version, EA91/EA92/EA93 contains the number
EA98 - EADA	Fetch 'x' value in CARD Fx. X can be 0 to FE
EA9B - EADA	Fetch 'x' value in CARD Bx. X can be 0 to E
EA9E - EADA	Fetch 'x' value in CARD Sx. X can be 0 to 4
EAED	1 byte spare
EAFB - EB09	New CARD B code
EB16 - EB18	End CARD Q routine: branch to FD4C to set 42AD = 85, call 1A6C and branch to E800 (original part which is end of CARD routine)
EB3B	Part of bank change routine changed E0 to E1 so all 4 SB bank bits are used
EB57 - EC32	Removed (rewritten) CARD S code with below routines
EB57 - EBA1	New CARD Sx code
EBA2 - EBB0	End part of CARD routine, to check if a PR or LET statement was used for the CARD command. If so a call is made to 2E25 to return the value to BASIC.
EBB1 - EBB5	Check LET CARD commands F -> EBC0 Other (S) -> EBB6
EBB6 - EBBF	Continue LET CARD command check in bank 6, slot D0 address DB74.
EBC0 - EBD0	LET CARD F code, search for indicated card via E655 routine, then call LET CARD

	F routine in bank 6, slot D0 address DBAC
EBD1 - EBD4	4 bytes spare
EBD5 - EBE0	New CARD Fx code
EBE1 - EC32	F&M Screen editor adaptation
EC9D - ECB7	Removed error text message 'SYNTAX ERROR' replaced by routines below.
EC9D - ECA4	Error code calls 5A and 5F
ECA5 - ECA7	3 bytes spare
ECA8 - ECB1	F&M Screen editor adaptation - Down
ECB2 - ECB7	6 bytes spare
ECB8 - EDA5	Removed CARD M and V code
ECB8 - EFFF	F&M Screen editor adaptation (some details / changes below)
ECBB - ECD3	Shape line 10 to character number for use screen editor
ED90 - EDA7	Jump table for keys: ED90 – ED91: 041C Up ED92 – ED93: EDEA right ED94 – ED95: EDA8 Down ED96 – ED97: 0357 Left ED98 – ED9E: 0322 CNTL C / CR ED9E – ED9F: E6E4 CNTL S
EDDD - EDDF	Call new scroll routine at FC1A
EDE0 - EDE6	F&M Screen editor adaptation, code moved from E861 to EDD0-EDE6
EDE7 - EDE9	Call to called on 'down' on last screen row to check is USB BROWSER is active, if not screen is scrolled
EE61 - EE62	Part of CNTL S routine which is change to call FD1C to enable TV out (if it was disabled) and reset DHEIGHT and WIDTH settings to normal.
EE8E - EE90	Part of cursor shape routine, calling FD0A to select line 8/9 for cursor depending on NTSC/PAL machine
EEC4 - EEC9	Check for CNTL keys; continue on F9A8 to check different key presses
EECA - EEDC	F&M Screen editor code moved from EEC4-EED6
EEDD - EEE2	Check for CNTL R, if pressed call CNTL R routine at E506
EEE3 - EEEF	F&M Screen editor code rewritten to fit in EEE1-EEEF
EF18	Changed F&M Screen editor branch to E3C8 to the actual address 13C8
EF25	Changed F&M Screen editor branch to E3D3 to the actual address 13D3
EF4A - EF4C	Call routine on E49B to store current line in CNTL R buffer (triggered on 'CR')
EF54	Corrected branch to EADB to fit changed code
EF5D - EF5E	Added RF.0 storage on stack
EF55 - EFA4	Read current screen location Return: Character in R8.0 and on 43F9
EFFE - EFFF	Call EDE0 instead of E861 as routine was moved

NVRAM & RTC (F000 - F3FF)

F000 - Fxxx	User area. This area can be used by any user or user program. No defined format is available so any address location can be filled as desired. USB NVPOKE can be used to store data and COMX BASIC PEEK can be used to retrieve data. Fxxx is defined via USB NVSTART.	
Fxxx - F3E7	Software area. This area can be used by any software. The area is formatted so every software program can find stored data back. USB NVPUT, NVGET, NVIGET, NVSGET, NVCLR and NVDEL can be used to store, retrieve or manipulate data to/from this area	
F3E8	NVRAM SW area start. Start = F000h + (value << 2)	0 = F000h
F3E9	SCREEN, COLOR & CTONE b7: not used b6/b5: colb1/colb0 b4: not used b3: ctone b2-b0: background color	E0h
F3EA	b7/b6: Last color shape mask, copied from 41CB during COLOR routine b5/b4: saved color shape mask (USB COLOR)	F0h
F3EB	Printer settings (also copied to 41B0h) b7: SHAPE flash 1 = on, 0 = off b6: Printer CNTL characters; 1 = normal, 0 = hex b0-b5: number of stop bits 0-63	42h = 2 stop bits, normal CNTL and no SHAPE flash
F3EC	NVRAM initialized (A5h = initialized)	
F3ED	b6: online mode; 1 = on, 0 = off b5: printer off on error code; 1 = yes, 0 = no b4: printer 'on' state before command; 1 = on, 0 =off b0-3: Volume	44h = online mode = on, Volume level 4
F3EE	Serial Printer settings (also copied to 41B1H/41B2H) b6: line feed suppression; 1 = yes, 0 = no b5: parity type; 1 = even, 0 = odd b4: parity; 1= yes, 0 = no b2-b3: number of data bits 00=5, 10=7 or 11=8 b0-b1: Baud rate 00=1200, 01=600, 10=300, 11=110	0Dh = Baud rate 600, 8 data bits, No parity and no LF suppression
F3EF	Serial Printer settings (also copied to 41B3h) CR pause 0-255	5 = CR pause 5
F3F0	b7: Logo tune; 1 = on, 0 =off b6: 80 column auto boot ; 1 = on, 0 =off b5: Boot message; 1 = on, 0 =off b4: Line editor; 0 = on, 1 =off (copied to 43F8H) b3: Printer TV; 1 = on, 0 =off (copied to b7 41b0h) b2: Printer keyboard output; 1 = on, 0 =off b1-b0: USB CHAR set	B4h = Logo tune: on 80 column auto boot: off Boot message: on Screen editor: on Printer TV: off Printer keyboard: on Character set: Standard COMX characters
F3F1	Date separator character (-, / etc)	2Fh = /
F3F2	Date format xx (xx/yy/zz)	FCh = day

	FCH = day, FDH = month, FEH = year	
F3F3	Date format yy (xx/yy/zz) FCH = day, FDH = month, FEH = year	FDh = month
F3F4	Date format zz (xx/yy/zz) FCH = day, FDH = month, FEH = year	FEh = year
F3F5	Time format; 12h (12 hour) or 24h (24 hour)	24h = 24 hour
F3F6	NVRAM check field, used by SB FW to check is NVRAM is available	
F3F7	Clock display; 1 = on when COMX is not 'running' any program 2 = always on 0 = off b8: 1 when RUN / CALL or USR is 'running'	0 = clock display off
F3F8 - F3FF	RTC	

Character Memory (F400 - F7FF)

CDP 1870 Character Memory

SB FW ROM / Page Memory (F800 - FFFF)

When writing data to F800 – FFFF area it will be used as CDP 1870 Screen page memory, when reading data below will be seen

F800	-	F801	BR16, USB Command entry; used from C010 in all banks
F802	-	F803	Set slot back to M[BFFE] and pull registers from stack (end USB command), if printer is active set slot to M[BFFD] (call F83A)
F804	-	F806	EDIT function in screen editor (call 1052)
F807	-	F80D	Read location from slot x. Always CALL this routine via SEP x to F808! Input: M[R2]=slot M[R2+1]=return slot Output: D= M[RC]
F80E	-	F815	Store to location M[RC] in slot x. Always CALL this routine via SEP x to F80F! Input: M[R2]=slot M[RC]=M[R2+1] M[R2+2]=return slot
F816	-	F839	USB Command entry which will switch in bank 0 and call command entry routines at C915.
F83A	-	F859	Set slot back to M[BFFE] and pull registers from stack (end USB command), if printer is active set slot to M[BFFD]
F83F	-	F859	Set slot back to M[BF42] and pull registers from stack
F85A	-	F8E3	USB PLOAD/PSAVE routines
F8E4	-	F93B	HEX / DEC Routine 1
F8E9	-	F93B	HEX / DEC Routine 1
F93C	-	F95B	Part of USB PLOAD,R command: set slot back to M[BF42] and pull registers from stack (as for end USB command). Then switch off clock if needed and execute 'CALL' by calling sub on 42A3.
F95C	-	F969	Force bank = 0 but leave slot as selected before
F96A	-	F98A	If printer is active set slot to M[BFFD] otherwise M[BFFE]
F975	-	F98A	Set slot back to code on BFFD (printer slot)
F978	-	F98A	Set slot back to code on BFFE (selected slot at USB/DOS command entry)
F97B	-	F98A	Set slot back to code on BF42 (current selected slot)
F98B	-	F9A7	Part of USB PLOAD,R command: set slot back to M[BF42] and pull registers from stack (as for end USB command). The execute 'RUN' by calling routine on E36D.
F9A8	-	F9B9	CNTL check routine, if no CNTL key pressed continue at EECA: CNTL E -> FB92 CNTL W -> FBB0 CNTL X -> F9BA CNTL V -> FA15
F9BA	-	FA14	CNTL X routine
FA15	-	FA54	CNTL V routine
FA55	-	FA8E	Print char (D) on screen and scroll if needed, used by CNTL V, R routines
FA8F	-	FAA8	Step current cursor position and check for end of line/screen, used by CNTL V, R, X,

	E routines
FAA9 - FAB0	Handle bug in READ feature which crashed if no DATA statements are available.
FAB1 - FAD3	Clear line from current position
FAD4 - FAE6	Search for first character in current input line (i.e. search for '0')
FAE7 - FAFD	Step current position one position back
FAFE - FB45	Print current CNTL R buffer to screen
FB46 - FB6B	Check on invalid line numbers, i.e >= FFFF. Introduced to fix bug in original COMX BASIC which crashes on line number 65535.
FB6C - FB91	Part of EDIT routine to handle different EDIT behavior if screen editor is active. If active the EDIT line number is just printed on screen, if not active normal original EDIT call is made.
FB92 - FBA4	CNTL E routine
FBA5 - FBAF	Print character on cursor position back on screen (i.e. remove cursor)
FBB0 - FBD2	CNTL W routine
FBD3 - FBD8	Error code routine Input: M[R6] = error code
FBD9 - FBDA	Disk routine: SEP RE / SEP R5
FBDB - FBF2	COPY TO/FROM bank x (1) Input: R7 = source start RF = length R8 = destination RE.0 = return slot RE.1 = destination slot RA.1=source slot
FBDD - FBF2	COPY TO/FROM bank x (1) R7 = source start RF = length R8 = destination RE.0 = return slot RA.1 = destination slot RE.1=source slot
FBF3 - FBF7	Call DOS routine in RAM (B700)
FBF8 - FC35	New scroll routine to handle clearing of clock
FC36 - FC69	Call subroutine in other bank Input: M[P+1]=slot M[P+2/3]=address
FC51 - FC69	Call subroutine in other bank Input: M[R2]=slot M[R2+1]=RF.1 M[R2+2]= return slot
FC6A - FCAA	Check if printer is active, if active select printer, branch to M[R6+1/R6+2] and switch back to slot on M[R6]. If no printer active just branch to M[R6+1/R6+2].
FCAB - FCBC	Call to E621, to check printer off on error code bit, if set reset to 0. If needed switch off printer. Perform line feed including printer checks.
FCBD - FCD5	COPY TO/FROM bank x (2) Input: R7 = source end RF = length R8 = destination

	RE.0 = return slot RE.1 = destination slot M[R2] = source bank x
FCBF - FCD5	COPY TO/FROM bank x (2) R7 = source end RF = length R8 = destination RE.0 = return slot RA.1 = destination slot M[R2] = source bank x
FCD6 - FCDB	Part of PLOAD routine, call FCE2 to do an INP 1 which will re-activate EF handling for tape.
FCDB - FCE1	Part of DLOAD routine, call FCE2 to do an INP 1 which will re-activate EF handling for tape.
FCE2 - FCE8	INP 1 which will re-activate EF handling for tape.
FCE9 - FCF7	Find card Input: Card id stored on RC = BF41 return to slot M[R6]
FCF8 - FD04	Find FDC and call DOS NEW
FD05 - FD09	Find FDC, return D = 0 if not found
FD0A - FD1B	Select line 8/9 for cursor depending on NTSC/PAL machine
FD1C - FD38	Part of CNTL S routine to enable TV out (if it was disabled) and reset DHEIGHT and WIDTH settings to normal.
FD39 - FD4B	If 80 column is booted set 42AD = 0D and call ECBB, if not call ECB8
FD4C - FD55	End CARD Q routine, added setting of 42AD = 85, call 1A6C and branch to E800 (original part which is end of CARD routine)
FD56 - FD66	Cursor COL40 switch
FD67 - FD83	Cursor COL80 switch
FD84 - FD90	Start is on FD.., print command on screen as stored on memory pointed by RC. Used by LIST routine.
FD91 - FD9B	String assignment handling, allowing A$=USB commands. If command input is USB branch to 2A0E otherwise 2A08.
FD9C - FDFA	Part of line input routine converting lower case characters to capitals in commands.
FDFB - FE47	Used by PR and IF handling to allow USB commands to be used: - Checks if first char is B (BIN or BLOAD), C (CD), D (DATE, DLOAD, DEL), H (HEX), U (URL, URLGET), N (NVGET, NVPUT, NVSGET). - If first char is a C also second char is checked on 'D' to make a difference between USB COMP which returns an INT and CD which returns a STR. - If first char is an N also 3rd char is checked on 'G', 'P' and 'S' to make sure NVIGET is not seen as returning a string - Last a check is performed on sub command code, if it is 98 (PLOAD), A1 (DLOAD) or AF (TIME) also returning a STR will be allowed
FE48 - FE5D	SWAP from bank x to RAM Input: R7 = source start RF = length R8 = destination RE.0 = return slot M[R2]= bank x
FE5E - FE81	Part of COLOR routine, call to 0C80 and after that set the last color shape mask back to NVRAM (b7/b6 on F3EA).
FE82 - FE98	Copy BE83 buffer to 4200 to return error code

FE99 - FEA7	Error code routine, if R7.0 = 2 call error code routine which 'returns' on E64F
FEA8 - FEBD	Check command buffer for end of command, return 0 it end of command
FEBE - FEDC	Check if USB GRAPH is active if so re-shape and set normal screen. Then jump to 80 column check.
FEDD - FEF1	CPOS check if DWIDTH is active
FEF2 - FF01	? Adaptation to allow line numbers between @FF00-@FFFE in renumber
FF02 - FF49	spare
FF4A - FF4F	INPUT '0' data
FF50 - FF51	Data: 'SB' text used for check if data is valid to load with RLOAD. i.e. a compare is done if 'SB' is stored in the end of the bank.
FF52 - FF64	F8DF Text: 'W = WARM START'
FF63 - FF70	F890 Text: 'SUPER BOARD'
FF71 - FF83	F92E Text: 'SUPER BOARD 2014'
FF84 - FF8E	F946 Text: 'ED KEEFE'
FF8F - FFA3	F955 Text: 'MARCEL V. TONGEREN'
FFA4 - FFAE	F996 Text: '1985 F&M'
FFAF - FFBE	F97F Text: 'SCREEN EDITOR'
FFBF - FFCF	F9 BB Text: '© 1983 C O M X'
FFD0 - FFE6	F9CC Text: 'WORLD OPERATIONS LTD'
FFE7	End code Text #FF, possibly used for something else?
FFE8 - FFF7	NVRAM system area defaults
FFF8 - FFFF	Store to location M[RF] in slot x. Always CALL this routine via SEP x to FFF9! Input: M[R2]=slot M[RF]=M[R2+1] M[R2+2]=return slot

APPENDIX C – PC <-> COMX HEADERS

This chapter lists all headers used for USB communication between the PC SW and COMX

Standard Header (COMX <-> PC)

Address	Bytes	Address Description	Value	Value Description
0h	1	Command type	0	None
			1	Save
			2	Load
			3	Get first CAT content
			4	Get next CAT content
			5	CD - Change directory
			6	MKDIR
			7	RMDIR
			8	DEL
			9	Fetch time & date
			12h, 14h, 16h, 1Ah	Used in image header
			18h	COMP, Compare files
			19h	URL
			42h	DLOAD
			81h	Save and overwrite existing file
			82h	Load continue
			FFh	Cancel command
1h	18	Program or directory name		Name in ASCII characters
13h	1	Result	0	File found
			ADh, AEh, AFh	Used in image header
			F3h	URL not found
			F4h	No more data found
			F5h	Server error
			F6h	Network error
			F7h	File open
			F8h	File error
			F9h	File size > 8K
			FAh	File size > 64K
			FBh	File size < 8K
			FCh	File size < 64K
			FDh	File type error
			FEh	Address error
			FFh	File NOT found
14h	2	Actual file length (excl. header)		

16h	1	Program type	1	.comx machine code		
			2	.comx 'old' regular save/load (for backward compatibility reasons only)		
			3	.comx F&M basic save/load		
			4	.comx 'old' and incorrect regular data load (for backward compatibility reasons only)		
			5	.comx regular data save/load		
			6	.comx regular save/load		
			7h to 9h	reserved for .comx format		
			10h	.bin		
			11h	.txt (memory dump)		
			12h	.img		
			13h	.ram		
			14h	.bin, .txt, img, .ram, .comx		
			20h	.comx.bak		
			21h	NVSAVE / nvram.bin		
			22h	SYSSAVE / system.bin		
			23h	NVSYNC		
			24h	URL		
			25h	URL INPUT		
			30h	.bin.bak		
			31h	.txt.bak		
			32h	.img.bak		
			33h	.ram.bak		
			34h	.bin.bak, .txt.bak, img.bak, .ram.bak, .comx.bak		
			b7 = 1	BLOAD/BSAVE active		
				Basic	**Machine code**	**Data**
17h	2	Start address		DEFUS	Start	String
19h	2	End address		EOP	End-1	Array
1Bh	2	Exec address		String/Array/EOD	Exec	EOD
1Ch	18	Filename 2		Name in ASCII characters (used for USB COMP)		
2Fh	1	Category		Not used		
30h	1	Case type	1	Standard COMX characters (upper case 'default')		
			2	Lower case 'default' standard COMX characters on 'SHIFT'		
			3	Upper case 'default' and lower case characters on 'SHIFT'		
			4	Lower case 'default' and upper case characters on 'SHIFT'		
31h	1	High byte start load				
32h	1	BLOAD/BSAVE type	30h	One RAM Bank x		
			31h	RAM Banks 8 to 15		
			32h	RAM Banks 8 to 15 Continue		
33h	2	BLOAD/BSAVE length	2			
35h	3	FW build number		FW build (3 ascii values)		

CD Header (PC -> COMX)

Address	Bytes	Address Description	Value	Value Description
0h	1	Result	FFh	Directory not found
1h	126	Full path		Path name in ASCII characters
Last byte	1	End code	0	

URL,"url" Header (COMX -> PC)

Address	Bytes	Address Description	Value	Value Description
0h	1	Command type	19h	USB URL
1h	126	URL		URL name in ASCII characters
Last byte	1	End code	0	

URL (x) and URLLINK (x) Header (COMX -> PC)

Address	Bytes	Address Description	Value	Value Description
0h	1	Command type	19h	USB URL
1h	1	Command	< 32	0 = Normal USB URL / LINK 1 = PAGE BACK 2 = GO TO HOME 3 = SET HOME 4 = GO TO BOOKMARK 5 = SET BOOKMARK 6 = HELP
2	2	Screen number	0-65535	USB URL: Data screen number
2	1	Bookmark	0-9	Bookmark number
2	1	Link	0-255	USB URLLINK: New link number, link number 0 = default page

CAT Header (PC -> COMX)

Address	Bytes	Address Description	Value	Value Description		
0h-1Bh				Entry 1 + x Every header contains 4 directory entries and is send until no more entries are available (entry number = 0)		
0h	1	Entry number	0	No more entries		
			1 – FEh	Directory entry number of specified file		
			FFh	Directory name		
1h	18	Program or directory name		Name in ASCII characters		
13h	1	Result	0	File found		
			ADh	Format error SS / SD		
			AEh	Format error SS / DD		
			AFh	Format error DS / SD		
			F5h - FEh	Used in standard header		
			FFh	File NOT found		
14h	1	Size	0 - FFh	File size if > 255 KB; otherwise 0		
16h	1	Program type	1	.comx machine code		
			2	.comx 'old' regular save/load (for backward compatibility reasons only)		
			3	.comx F&M basic save/load		
			4	.comx 'old' and incorrect regular data load (for backward compatibility reasons only)		
			5	.comx regular data save/load		
			6	.comx regular save/load		
			7h to 9h	reserved for .comx format		
			10h	.bin		
			11h	.txt (memory dump)		
			12h	.img		
			13h	.ram		
			20h	.bak		
				Basic	**Machine code**	**Data**
17h	2	Start address		DEFUS	Start	String
19h	2	End address		EOP	End-1	Array
1Bh	2	Exec address		String/Array/EOD	Exec	EOD
20h-3Bh				Entry 2 + x		
40h-5Bh				Entry 3 + x		
60h-7Bh				Entry 4 + x		

Image file Header (COMX <-> PC)

Address	Bytes	Address Description	Value	Value Description
0h	1	Command type	0 – 9	Used in standard header
			12h	Get track image file
			14h	Save track image file
			16h	Test image file
			18h	Used in standard header
			1Ah	Get track image file continue
			42h	Used in standard header
			81h – 82h	Used in standard header
			94h	Save track image and overwrite existing file
			FFh	Cancel command
1h	18	Program or directory name		Name in ASCII characters
13h	1	Result	0	File found
			ADh	Format error SS / SD
			AEh	Format error SS / DD
			AFh	Format error DS / SD
			F5h - FEh	Used in standard header
			FFh	File NOT found
14h	1	Density	0	Single
			1	Double
15h	1	Sides	0	Single
			1	Double
16h	1	Track type	0	First track
			1	Second until last -1
			2	Last track
17h	1	Drive & side	14H	Drive 1 & side 1
			34H	Drive 1 & side 2
			18H	Drive 2 & side 1
			38H	Drive 2 & side 2
18h	1	Track number	0 to 34	
19h	1	Side	0 or 1	
30h	1	Case type	1	Standard COMX characters (upper case ‘default’)
			2	Lower case ‘default’ standard COMX characters on ‘SHIFT’
			3	Upper case ‘default’ and lower case characters on ‘SHIFT’
			4	Lower case ‘default’ and upper case characters on ‘SHIFT’

APPENDIX D – ERROR MESSAGES

This chapter lists all known COMX error messages, including the original COMX BASIC, PRINTER, DOS, F&M BASIC and SB error messages

COMX 35 BASIC Error Messages

DEC	HEX	ERROR CODE
0	0	Program halted by user
1	1	Syntax error in asc or len function
2	2	Array out of range or not dimensioned
3	3	Dimension error
4	4	Defint has illegal ending
5	5	Parentheses missing in argument
6	6	Argument out of range
7	7	Mixed mode calculation encountered
8	8	Divide by zero error, or log of negative number
9	9	Non-executable function encountered
10	A	Exit command must be used with for/next or gosub/return
11	B	For/next stack overflow, or for/next executed directly
12	C	Syntax error in for
13	D	Gosub stack overflow
14	E	Unacceptable character in hexadecimal number
15	F	Floating point number too large to be converted to integer or integer multiply overflow
16	10	Unacceptable operator in conditional statement
17	11	Input or fval cannot be directly executed
18	12	Must have variable or string name in read
19	13	Syntax error in read or input
20	14	Syntax error in len function
21	15	Syntax error in assignment statement
22	16	Missing quote
23	17	Syntax error in list
24	18	No such word found in library
25	19	Syntax error in mid$ function
26	1A	Unacceptable variable name found in next statement
27	1B	Either a number or a letter is expected
28	1C	Missing arithmetic parentheses
29	1D	Wrong number of arguments in poke statement
30	1E	Unacceptable last character in print statement
31	1F	Syntax error in data statement
32	20	No more data found
33	21	No such string found in input statement
34	22	Missing equal sign in assignment statement
35	23	Missing parentheses in string array
36	24	Too many arguments in usr or call
37	25	Syntax error in chr$ function
38	26	Unacceptable character in binary number

39	27	Line buffer overflow
40	28	File not opened for input
41	29	File not opened for output
42	2A	Unacceptable line end or non-executable statement
43	2B	Stack overflow
44	2C	Too many digits in number
45	2D	Unacceptable character in number fold
46	2E	No such line number found
47	2F	Unacceptable operation in if statement
48	30	Memory overflow
49	31	Wrong number of arguments in mod statement
50	32	Program too large for memory
51	33	Argument out of range
52	34	Wrong number of arguments
53	35	Wrong number of arguments
54	36	String variable not defined
55	37	Tape read error
56	38	Tape write error
57	39	File is not a basic program
58	3A	File is not basic data
59	3B	Reserved
60	3C	Reserved
61	3D	Reserved
62	3E	Rom or rom card not present
63	3F	Not enough memory for renumber to operate
64	40	Renumber located line number error
65	41	No such subscript variable defined
66	42	String is over 127 characters
67	43	Number of arguments in color must be 1
68	44	Number of arguments in screen must be 1
69	45	Number of arguments in ctone must be 1
70	46	Number of arguments in volume must be 1
71	47	Number of arguments in noise must be 2
72	48	Number of arguments in tone or music must be 3

F&M BASIC V2.00 Error Messages

DEC	HEX	ERROR CODE
73	49	Number of arguments in TEXT must be 1
74	4A	Number of arguments in POINT must be 2
75	4B	Number of arguments in DUMP must be 2
76	4C	Number of arguments in CHAR must be 1
77	4D	Incorrect use of ALPHA
78	4E	Line number missing
79	4F	TOS not found
81	51	Number of arguments in P-SCR must be 1
82	52	Not enough memory for P-SCR
83	53	Number of arguments in DEL must be 1 or 2
84	54	First argument in DEL must be lower than last
85	55	TOS command not recognized

PRINTER Error Messages

DEC	HEX	ERROR CODE
80	50	Printer error

COMX 35 DOS Error Messages

DEC	HEX	ERROR CODE
100	64	Text file record length is not fixed
101	65	Record number exceeds file size
102	66	Disk read error
103	67	Read or write pointer exceeds file size
104	68	Sector read or write error
105	69	Seek error
106	6A	Diskette write protected
107	6B	File locked
108	6C	Spare
109	6D	Spare
110	6E	Slash missing
111	6F	End of command found
112	70	Work id. Number error
113	71	Record length missing
114	72	More than 8 arrays already exist
115	73	String or array variable error
116	74	Not integer or too large integer exist
117	75	Not a fixed record length file type
118	76	String exceeds record length
119	77	Spare
120	78	Address syntax error
121	79	Drive number argument or syntax wrong
122	7A	Not enough disk memory
123	7B	Parenthesis missing
124	7C	Input file name error
125	7D	Comma missing
126	7E	Command does not exist
127	7F	Syntax error on ‘save’ command
128	80	Quotation mark missing
129	81	File name already opened or exists in directory
130	82	File not found in directory
131	83	No such kind of protection
132	84	More than 5 files already opened
133	85	File name not yet opened
134	86	More than 51 directory entries
135	87	File write protected
136	88	Disk or file non-copyable
137	89	No such handler command code

SB Error Messages

DEC	HEX	ERROR CODE
86	56	Only one parameter (B, I, R, T) allowed
87	57	Invalid parameter
88	58	Program will not fit into 8K RAM bank
89	59	Number of arguments in USB MOVE must be 3, 4 or 6
90	5A	Syntax error in CARD statement
91	5B	No rtc or nvram found, or nvram not initiated
92	5C	No valid data in RAM bank
93	5D	Could not delete file
94	5E	No data to save
95	5F	No disk card or disk not initiated
96	60	Too many parameters
97	61	Nvram write enable error
98	62	Specify date string
99	63	Incorrect date string format
138	8A	Incorrect I/O port specified
139	8B	Number of arguments in USB OUT must be 2
140	8C	Number of arguments in USB PPOKE or USB NVPOKE must be 2
141	8D	NVPOKE or NVSTART address out of nvram range
142	8E	USB command not recognized
143	8F	Help for command not found
144	90	Command ROM not detected
145	91	Unknown or incorrect file format
146	92	F&M BASIC not loaded
147	93	File name too long
148	94	No filename specified
149	95	Address range required
150	96	End address required
151	97	Invalid RAM bank
152	98	Invalid directory
153	99	Could not create directory
154	9A	Specify start < end address
155	9B	Could not delete directory
156	9C	Specify start address
157	9D	Incorrect line number
158	9E	Number of arguments in USB CPOS must be 2
159	9F	Specify start <= DEFUS address
160	A0	Program cannot be loaded in F&M BASIC
161	A1	Number of arguments in USB NVPUT must be 3
162	A2	NVRAM full, data not stored
163	A3	Software ID not found
164	A4	Number of arguments in USB NVGET must be 2
165	A5	Number of arguments in USB NVIGET , NVSGET, NVDEL and NVSTART must be 1
166	A6	Requested NVRAM size to small to fit current data
167	A7	Software ID should be between 1 and 255

168	A8	Label or date string too long
169	A9	No label or date string specified
170	AA	No or incorrect printer card found
171	AB	Printer not on
172	AC	Number of arguments in USB MEMDUMP and USB PMEMDUMP must be 3
173	AD	Wrong disk format, image is single sided/ single density
174	AE	Wrong disk format, image is single sided/double density
175	AF	Wrong disk format, image is double sided/single density
176	B0	Incorrect EF flag specified
177	B1	80 column card not found
178	B2	RAM bank file not found
179	B3	USB transfer timeout
180	B4	USB transfer error
181	B5	Two file names required
182	B6	Card type not found
183	B7	Network error
184	B8	Online features disabled
185	B9	Server error
186	BA	URL not found
187	BB	Number of arguments in USB PLOT must be 3 or 4
188	BC	Number of arguments in USB LINE or USB ELLIPSE must be 5
189	BD	Number of arguments in USB CIRCLE must be 4
190	BE	Number of arguments in USB CPEEK must be 1 or 2
191	BF	No REM statement found
192	C0	Number of arguments in USB CPOKE must be 2 or 3

APPENDIX E – BASIC COMMAND TABLE

This chapter lists all COMX commands and their command code, jump table entry and start address in ROM.

Command	Command code	Jump table entry	Start Address		
			Original COMX	Expansion / DOS	Super Board
REM	80	1500	31F9		
CLS	81	1502	321E		
NEW	82	1504	1A47		
RUN	83	1506	1F76		E36D
END	84	1508	10CC		
LET	85	150A	2500		
PRINT	86	150C	2800		
PR	86				
GOTO	87	150E	312A		
IF	88	1510	2600		
INPUT	89	1512	26C8		
LIST	8A	1514	1570		
GOSUB	8B	1516	2F3C		
RETURN	8C	1518	2F30		
WAIT	8D	151A	31D2		
DIM	8E	151C	2100		
FOR	8F	151E	2300		
NEXT	90	1520	2F00		
FIXED	91	1522	2EE/		
POKE	92	1524	2ABD		
DEG	93	1526	33FA		
RAD	94	1528	33FD		
MUSIC	95	152A	41CC		
DEFINT	96	152C	38E4		
PSAVE	97	152E	1672		
PLOAD	98	1530	0E00		FCD6
DEFUS	99	1532	3EFE		
EOP	9A	1534	25DA		
DATA	9B	1536	3234		
READ	9C	1538	3D00		
RESTORE	9D	153A	25F4		
EOD	9E	153C	25DD		
CLD	9F	153E	2BE9		
DSAVE	A0	1540	1675		
DLOAD	A1	1542	0E03		FCDC
TIMEOUT	A2	1544	0EE9		
NOISE	A3	1546	41D4		
SCREEN	A4	1548	0BC8		
COLOR	A5	154A	0BFC		
CTONE	A6	154C	0D63		
TRACE	A7	154E	2E00		

CALL	A8	1550	2C00		E373
VOLUME	A9	1552	0D37		0D36
TONE	AA	1554	41D0		
SHAPE	AB	1556	0A85		
DOS	AC	1558	0DD9	E817	E010
POUT	AD	155A	0DD4	Not available	Not available
CARD			Not available	E817	E817
EXIT	AE	155C	32D0		
TIME	AF	155E	1600		
CPOS	B0	1560	0F00		
RENUMBER	B1	1562	221C		E714
EDIT	B2	1564	1808		E678
FORMAT	B3		2D3F		
TOUT	B4	1568	14D4	14D4	Not available
USB			Not available	Not available	F816
AND	B5				
XOR	B6				
OR	B7				
CHR$	B9				
MID$	BA				
^	BB				
TAB	BC				
>=	BD				
<=	BE				
<>	BF				
STEP	C0				
TO	C1				
,	C2				
;	C3				
)	C4				
THEN	C5				
<	C6				
>	C7				
+	C8				
-	C9				
*	CA				
/	CB				
=	CC				
:	CD				
" (close)	CE				
" (open)	CF	159E	00FE		
CHAR	D1	15A2	2C6A		
INT	D2	15A4	2BB8		
SIN	D4	15A8	2999		
COS	D5	15AA	299C		
(	D6				
string var	D7	15AE	2400		
ATN	D8	15B0	299F		
EXP	D9	15B2	29A2		

LOG	DA	15B4	29A5		
SQR	DB	15B6	29A8		
INT	DC	15B8	1F00		
PEEK	DD	15BA	30E8		
ABS	DE	15BC	3FA5		
RND	DF	15BE	2900		
USR	E0	15C0	2C03		E379
INUM	E1	15C2	2B03		
#		15C4	2D00		
FNUM	E3	15C6	27C1		
ASC	E4	15C8	28B2		
LEN	E5	15CA	306D		
\	E6				
SGN	E7	15CE	2A74		
MOD	E8	15D0	1A80		
NOT	EA	15D4	19ED		
PI	EB	15D6	24E2		
FVAL	EC	15D8	28AF		
STR$	ED				
MEM	EE	15DC	3037		
KEY	F0	15E0	163F		