SB FW Memory Details (build 198)

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

The following color coding is used:

Black text: 1802 code

Green text: unused / spare memory

Blue text: data areas

Red text: info about routines that are removed

In the first chapter ‘BASIC ROM’ and 3rd chapter ‘EXPANSION ROM E800 – EFFF’ only locations are listed that are different from the original COMX BASIC ROM and EXPANSION BOX ROM.

For chapter ‘SB FW ROM E000 – E7FF’ and ‘SB FW ROM F800 – FFFF’ ALL routines are listed.

For the remaining chapters most routines are listed but in some cases only USB command entry locations are mentioned.

BASIC ROM

All changed locations compared to the original COMX ROM are listed.

0018 - 004A	- Removed OUT 4, 3C41 (start of logo tune) - Code from 0021 – 0053 moved back to 0018 – 004A. - Changed stack pointer to LDL 2,BDFF instead of 4FFF, to make sure 4FFF RAM area is not destroyed at restart.
004B - 0051	Code from 007B – 007F moved back to 004B – 004F
0052 - 0056	spare
007B - 0087	Code moved back from 0080 – 008C
0088 - 008A	LBR 0DA7, logo tune start-up
008B - 008C	spare
00D9 - 00F5	Start-up text: ‘COPYRIGHT © 1983 BY. C O M X’ changed to: ‘SCREEN EDITOR’ and ‘© 1983 C O M X’ rest of the new copyright text is on FFA5 – FFE6
0144 - 0146	LBR E2E3, Check on cold/warm restart and display boot message if needed. Text stored on FF96 – FFA4. After that display new © text (FFA5 - FFE6).
014B	Changed 7C to 7F to put boot text on new screen location. Text taken from 00D9 – 00E5, ‘SCREEN EDITOR’
015E	Changed AB to BB to put boot text on new screen location. Text taken from 00E6 – 00F5, ‘© 1983 C O M X’
0187 - 019B	Check on logo tune on/off, if on branch E181
019C - 01A3	Check on 80 column auto boot, if on branch to E3B2 and search for 80 col card. Then (if found) skip 40 column boot screen.
01A2 - 01A7	spare

01A8	-	01C4	Moved back boot screen routine from 01AD – 01B6, including branches on 01BD, 01C1 and 01C3 as well as XRI on 01B7 (which still checks on space bar). Added check on ‘C’ key for ‘Copyright message’ display on boot screen. If pressed branch to E438.
01CA			Changed AE to A9 so ‘moved’ boot screen routine is called which is now on 01A9.
01E5			Changed 0A to 24, to show new copyright screen longer
01F6			Changed 20 to 00, to change clear screen to 0 on startup. This as the screen editor uses 0 as separators instead of spaces.
0206			LBR E728, get SCREEN/COLOR/CTONE info from NVRAM and store it on 41C0.
021A	-	0222	spare
02E4	-	02E6	LBR FC1A, new scroll routine for proper clock clearing
046C	-	046F	LBR E751; 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.
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
0C62	-	0C63	Part of COLOR routine, changed sub routine call from 0C80 to FEA9, FEA9 will call to 0C80 and after that set the last color shape mask back to NVRAM (b7/b6 on F3EA).
0D1A	-	01DC	Part of TONE, MUSIC and NOISE routine, change to call 0DE9 which will skip sound setting if VOLUME is set 0.
0D5A	-	0D5C	Part of VOLUME routine, call routine on FD10 which will store VOLUME value in NVRAM.
0DA7	-	0DF4	Removed POUT, TOUT routine and replaced with routines below.
0DA7	-	00CF	Read logo tune on/off: OUT 4,3C41: on, OUT 4,3C40: off. Loop 0E88 to zero if logo tune on; go back to 008D.
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	-	0DF2	Part of TONE, MUSIC and NOISE routine, which will skip sound setting if VOLUME is set 0.
0DF3	-	0DF4	spare
0F2B	-	0F2D	Call to FF28 as part of new CPOS routine to check if DWIDTH is active
1003	-	1005	Call to 1330 to set OUT 1 to 0 to handle start-up of SB
100D	-	100E	Call to FD19 to: - Call to 1AD5 (original code) - 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.
1010	-	1011	Call to FD80, call 1A6C (original code), set 80 column if applicable and shape line 10 to character number for all characters.
1013	-	1015	Call to E248 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
1016	-	1018	Call to E743 which will call DDFA in bank 0. DDFA will request initiation NVRAM if not done and then initiate NVRAM. Initiate NEW unless WARM start was selected. NEW jump to 102D otherwise 103E
1019	-	1026	Check if USB GRAPH is active (FF09) then check if 80 column is active E685) Clear line after READY unless 80 column is active
1027	-	1029	Call E3E7 to print the 'return' after 'READY' and reset b8 of F3F7 (CLOCK) as no program is running.
102A	-	102B	Branch back to 1052
102C			spare
104F	-	1051	Call to 1019, to clear line after READY, print the return after ready (original code) and reset CLOCK b8 of F3F7.
1083	-	1084	Call to FCD2 to perform line feed including printer checks.
108A	-	108C	Call to E10B to print error code in text. OUT 1 with 1 removed.
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 and replace with routines below.
1330	-	1339	Set OUT 1 to 0 to handle start-up of SB.
133A	-	1342	spare
13C6	-	13DB	F&M Screen editor adaptation
1400	-	1402	
1407			
140A			
1414	-	142A	
1506	-	1507	Changed start address 'RUN' to E3D5 to handle CLOCK off during run.
1530	-	1531	Changed start address 'PLOAD' to FCFD to do an INP 1 which will re-activate EF handling for tape.
1542	-	1543	Changed start address 'DLOAD' to FD03 to do an INP 1 which will re-activate EF handling for tape.
1550	-	1551	Changed start address 'CALL' to E3DB to handle CLOCK off during CALL.
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 E6F6 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 E3E1 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 E442 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 FB8C to handle different EDIT behavior in screen editor.
1B9E	-	1BA0	Part of LET routine, calling E383 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.

221C - 22D8	Old RENUMBER location, RENUMBER moved to SB bank 7.
221C	USB CPEEK
221C - 22D8	spare
241A - 241D	LBR E36D, NOP. String assignment handling, allowing A$=USB commands.
2600 - 2602	LBR E3A2, part of IF handling to allow USB commands to be used in IF statements.
280D - 280F	LBR E392, part of PRINT routines, to allow USB commands to print return values.
2A05 - 2A07	LBR FDD8, string assignment handling, allowing A$=USB commands.
2AEE	Changed 3A to 30 (BNZ to a BR)
2D7B	Changed 3A to 30 (BNZ to a BR)
3D0B - 3D0C	LBR FAC7. To handle bug in READ feature which crashed if no DATA statements are available.

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 E70E)
E016 - E018	USB MON ‘register check' call (call to E6E5)
E019 - E01B	NVPUT: R8.1 SW ID, source = RC, R8.0 = number of bytes (call to E6D1)
E01C - E01E	NVGET: R8.1 SW ID, dest = RC, R7.0 = 2 (call to E&DB)
E01F - E10A	Updated interrupt routine including printing of clock if needed and JOY handling.
E10B - E147	Print error in TEXT via C10B routine in bank 1
E148 - E157	Return to slot stored on BFFE
E158 - E16E	Write NVRAM location F300 + M[R6]
E16F - E180	Read NVRAM location F300 + M[R6]
E181 - E19E	Logo tune routine, branch back to 0196.
E19F - E1C3	NOT USED 80 column select routine
E1C4 - E235	- Switch in bank 15 and reset CNTL X/V and R buffers. - Get screen/line editor info from NVRAM and store on 43F8. - 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
E236 - E247	Auto boot 80 column CARD
E248 - E2E2	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 FD39 for a DOS NEW
E2E3 - E34D	Check on cold/warm restart and display boot message if needed. Text stored on FF96 – FFA4. After that display new © text (FFA5 - FFE6). Branch back to start-up routing 0147.

E34E - E36C	Check if NVRAM is working correctly
	Return:
	D=0, OK
	D=FF, error

E36D - E382	String assignment handling, allowing A$=USB commands
E383 - E391	Part of LET routine, added check is on both CARD and USB statements to allow A=USB commands.
E392 - E3A1	Part of PRINT routines, to allow USB commands to print return values.

E3A2	-	E3B1	Part of IF handling to allow USB commands to be used in IF statements.
E3B2	-	E3D4	Part of boot routine: search for 80 column card if found skip 40 column boot screens
E3D5	-	E3DA	Part of RUN code; call E409 to handle CLOCK off during RUN and CALL original RUN code at 1F76
E3DB	-	E3E0	Part of CALL code; call E409 to handle CLOCK off during RUN and CALL original CALL code at 2C00
E3E1	-	E3E6	Part of USR code; call E409 to handle CLOCK off during RUN and CALL original USR code at 2C03
E3E7	-	E437	CLOCK off during RUN/CALL handling. If F3F7 contains invalid value do nothing. If b8=1 reset it to 0.
E409	-	E437	Switch clock off during RUN / CALL
E438	-	E441	Force copyright screen during boot screen (C pressed) and branch back to 01A8
E442	-	E4E3	Part of LIST code to properly format USB commands
E4E4	-	E50F	Part of error text routine to handle slot handling if printer or 80 column is active
E510	-	E574	Store current line in CNTL R buffer (triggered on ‘CR’)
E575	-	E5BC	CNTL R routine (called when CNTL R is pressed)
E5BD	-	E5CC	Get current CNTL R buffer (from bank 15, DCFE) and return location in RE.
E5CD	-	E5D8	Select slot as stored on BF42
E5D9	-	E5F2	Store new CNTL buffer location (RE) to bank 15, DCFE. If no RAM found RE=4000
E5F3	-	E5FF	Call routine on 42A3 with slot as in D; always return to slot 10.
E600	-	E643	Call disk routine Input: address: M[R6]+M[R6+1] return slot m[R6+2]
E644	-	E659	Select disk card
E65A	-	E66D	Error code routine which will return as from a normal subroutine call Input: M[R6]= error code number
E66E	-	E684	Check on WARM boot possibility Return: 0=WARM NOT 0=COLD
E685	-	E68F	Check if 80 column card is active Return: NOT 0=80 column active 0=80 column NOT active
E690	-	E6A4	Reset F3ED bit 5, printer off on error code
E6A5	-	E6B3	Set slot back to BFFE and call 2e25
E6B4	-	E6C3	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
E6C4	-	E6DB	USB CARD F search for slot routine
E6DC	-	E6E5	NVPUT: R8.1 SW ID, source = RC, R8.0 = number of bytes (call to bank 0, slot 10, D3DE)
E6E6	-	E6EF	NVGET: R8.1 SW ID, dest = RC, R7.0 = 2 (call to bank 0, slot 10, D4F9)
E6F0	-	E6F5	USB MON ‘register check' call (call to bank 4, slot 90, D50E)

E6F6 - E70C	Part of EDIT command, check command line to make sure a valid argument is specified like for example ().
E70B - E718	HSWEB for use from USB commands (call to bank 0, slot 10, DD0D)
E719 - E727	HSWEB for use from external SW
E728 - E742	Check if NVRAM is initiated, if not store E0 on 41C0 If initiated get SCREEN/COLOR/CTONE info from NVRAM and store it on 41C0.
E743 - E750	Call DDF6 in bank 0.
E751 - E75D	Updated print routine, on B1 / EF1 store character on screen directly, if not store it in print buffer for printing via interrupt routine.
E75E - E76D	Called on line buffer overflow, check is done to see if USB BROWSER is active (43F8 = 2) if so continue otherwise give error code 27
E76E - E77C	Called on CNTL S press, if USB BROWSER is active ignore otherwise call CNTL S routine to clear screen
E77D - E79D	Called on cursor down, if USB BROWSER is active ignore otherwise call scroll routine
E79E - E7FF	62 (98 decimal) bytes 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 (43F8 = 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 FD93 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 4SB 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 E6B9 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: E76E 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 FD63 to enable TV out (if it was

	disabled) and reset DHEIGHT and WIDTH settings to normal.
EE8E - EE90	Part of cursor shape routine, calling FD4A to select line 8/9 for cursor depending on NTSC/PAL machine
EEC4 - EEC9	Check for CNTL keys; continue on F9C7 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 E575
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 E510 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

SB FW ROM F800 – FFFF

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 FC8B)
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 C88A.
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 E3D5.
F9A8	-	F9C6	NOT USED 'MORE' key routine
F9C7	-	F9D8	CNTL check routine, if no CNTL key pressed continue at EECA: CNTL E -> FBB2 CNTL W -> FBD0 CNTL X -> F9D9 CNTL V -> FA33
F9D9	-	FA32	CNTL X routine
FA33	-	FA72	CNTL V routine
FA73	-	FAAC	Print char (D) on screen and scroll if needed, used by CNTL V, R routines
FAAD	-	FAC6	Step current cursor position and check for end of line/screen, used by CNTL V, R, X, E routines
FAC7	-	FACE	Handle bug in READ feature which crashed if no DATA statements are available.
FACF	-	FAF1	Clear line from current position
FAF2	-	FB06	Search for first character in current input line (i.e. search for '0')
FB07	-	FB1D	Step current position one position back
FB1E	-	FB65	Print current CNTL R buffer to screen
FB66	-	FB8B	Check on invalid line numbers, i.e >= FFFF. Introduced to fix bug in original COMX BASIC which crashes on line number 65535.
FB8C	-	FBB1	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.
FBB2	-	FBC4	CNTL E routine
FBC5	-	FBCF	Print character on cursor position back on screen (i.e. remove cursor)
FBD0	-	FBF2	CNTL W routine
FBF3	-	FBF8	Error code routine Input: M[R6] = error code
FBF9	-	FBFA	Disk routine: SEP RE / SEP R5

FBFB	-	FC14	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

FBFD	-	FC14	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

FC15 - FC19	Call DOS routine in RAM (B700)
FC1A - FC56	New scroll routine to handle clearing of clock
FC57 - FC8A	Call subroutine in other bank Input: M[P+1]=slot M[P+2/3]=address
FC72 - FC8A	Call subroutine in other bank Input: M[R2]=slot M[R2+1]=RF.1 M[R2+2]= return slot
FC8B - FC90	EDIT function in screen editor
FC91 - FCD1	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].
FCD2 - FCE3	Call to E685, to check printer off on error code bit, if set reset to 0. If needed switch off printer. Perform line feed including printer checks.

FCE4 - FCFC	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

FCE6 - FCFC	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

FCFD - FD02	Part of PLOAD routine, call FD0A to do an INP 1 which will re-activate EF handling for tape.
FD03 - FD08	Part of DLOAD routine, call FD0A to do an INP 1 which will re-activate EF handling for tape.
FD09 - FD0F	INP 1 which will re-activate EF handling for tape.
FD10 - FD18	Part of VOLUME routine, call routine in bank 6 which will store VOLUME value in NVRAM.
FD19 - FD28	Store VOLUME value from NVRAM to 41C9, then CALL E1C4 to reset CNTL X/V and R buffers; set LINE/SCREEN editor, set PRINTER slot, set USB COLOR, USB CHAR and auto boot 80 column CARD.
FD29 - FD37	Find card Input: Card id stored on RC = BF41 return to slot M[R6]

FD38	-	FD44	Find FDC and call DOS NEW
FD45	-	FD49	Find FDC, return D = 0 if not found
FD4A	-	FD5B	Select line 8/9 for cursor depending on NTSC/PAL machine
FD5C	-	FD62	NOT USED, set slot to 10 and branch to 2E25
FD63	-	FD7F	Part of CNTL S routine to enable TV out (if it was disabled) and reset DHEIGHT and WIDTH settings to normal.
FD80	-	FD92	If 80 column is booted set 42AD = 0D and call ECBB, if not call ECB8
FD93	-	FD9C	End CARD Q routine, added setting of 42AD = 85, call 1A6C and branch to E800 (original part which is end of CARD routine)
FD9D	-	FDAD	Cursor COL40 switch
FDAE	-	FDCA	Cursor COL80 switch
FDCB	-	FDD7	Start is on FDCC, print command on screen as stored on memory pointed by RC. Used by LIST routine.
FDD8	-	FDE2	String assignment handling, allowing A$=USB commands. If command input is USB branch to 2A0E otherwise 2A08.
FDE3	-	FE45	Part of line input routine converting lower case characters to capitals in commands.
FE46	-	FE92	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

FE93	-	FEA8	SWAP from bank x to RAM
			Input:
			R7 = source start
			RF = length
			R8 = destination
			RE.0 = return slot
			M[R2]= bank x

FEA9	-	FECC	Part of COLOR routine, call to 0C80 and after that set the last color shape mask back to NVRAM (b7/b6 on F3EA).
FECD	-	FEE3	Copy BE83 buffer to 4200 to return error code
FEE4	-	FEF2	Error code routine, if R7.0 = 2 call error code routine which ‘returns’ on E64F
FEF3	-	FF08	Check command buffer for end of command, return 0 it end of command
FF09	-	FF27	Check if USB GRAPH is active if so re-shape and set normal screen. Then jump to 80 column check.
FF28	-	FF3C	CPOS check if DWIDTH is active
FF3D	-	FF4C	
FF4D	-	FF53	RENUMBER call to bank 7
FF54	-	FF70	SHAPE flash if activated on bit 7 NVRAM F3EB
FF3D	-	FF93	.. (.. decimal) bytes spare
FF94	-	FF95	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.
FF96	-	FFA4	Text: ‘W = WARM START’

FFA5 - FFB0	Text: ‘SUPER BOARD’
FFB1 - FFC1	Text: ‘SUPER BOARD 2013’
FFC2 - FFCA	Text: ‘ED KEEFE’
FFCB - FFDD	Text: ‘MARCEL V. TONGEREN’
FFDE - FFE6	Text: ‘1985 F&M’
FFE7	Spare?
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

Bank 0 / 10

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 C8AA, USB command handling routine called is ROM is switched in via a CARD command
C013 - C37F	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.
C3C5 - ..	USB PSAVE
C4E9 - ..	USB PLOAD
C869 - ..	USB DSAVE

C98A - ..	USB CD
C9C2 - ..	USB MKDIR
C9DA - ..	USB CAT
CBF1 - ..	USB RMDIR
CC23 - ..	USB DEL
CD23 - ..	USB RTCPC
CD66 - ..	USB SYSDEF
CDBE - ..	USB DATE
CFB8 - ..	USB TIME
D073 - ..	USB TIMEPM
D08C - ..	USB TIMEAM
D122 - ..	USB NVSAVE
D13D - ..	USB SYSSAVE
D158 - ..	USB NVLOAD
D1C0 - ..	USB SYSLOAD
D22D - ..	USB INP
D28E - ..	USB OUT
D31A - ..	USB HISTORY
D386 - ..	USB NVPUT
D4EF - ..	USB NVGET
D552 - ..	USB NVIGET
D5EB - ..	USB NVSTART
D5B1 - ..	USB NVSGET
D6EB - ..	USB NVCLR
D721 - ..	USB NVDEL
D746 - ..	USB NVLIST: startup, continue in bank 1 on DAED
D791 - ..	USB RSAVE
D7D3 - ..	USB RLOAD
D844 - ..	USB MOVE
D8AC - ..	USB IMGTODISK
D952 - ..	USB IMGTODISK
D9BC - ..	USB IMGTEST
D9D5 - ..	USB JOY
D9E0 - ..	INT = USB Q
D9EF - ..	USB EF
D9FA - ..	USB POS
DA05 - ..	USB DLOAD
DAB1 - ..	USB BSAVE
DB40 - ..	USB BLOAD
DC1B - ..	USB COMP
DD14 - DD1A	Check online mode if off return 0
DD1B - ..	USB HSWEB
DD47 - ..	USB URL
DD68 - ..	USB URLGET
DDE7 - DE09	USB BROWSER
DE0A - DE8F	If NVRAM is not initiated, request initiation and then initiate NVRAM. Initiate NEW

			unless WARM start was selected. NEW jump to 102D otherwise 103E
DE90	-	..	URL DISP
DE9D	-		USB BROWSER: URL ‘input’ save routine
DEB8	-		USB EMAIL
DED2	-	DF80	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)

Bank 1 / 30

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 E236
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

Bank 2 / 50

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

Bank 3 / 70

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

Bank 4 / 90

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	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].
DD54 - DD60	
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

Bank 5 / B0

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

Bank 6 / D0

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
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

Bank 7 / F0

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 FC73)
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