



QUICK REFERENCE HANDBOOK

































































































































VERSION 0.3b

JEFFREY DANIELS

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	
7680	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	00	01	22
7702	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	44
7724	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	66
7746	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	88
7768	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	110
7790	90	91	92	93	94	95	96	97	98	99	00	01	02	03	04	05	06	07	08	09	10	11	132
7812	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	154
7834	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	176
7856	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	198
7878	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	220
7900	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	242
7922	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	264
7944	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	286
7966	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	308
7988	88	89	90	91	92	93	94	95	96	97	98	99	00	01	02	03	04	05	06	07	08	09	330
8010	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	352
8032	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	374
8054	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	396
8076	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	418
8098	98	99	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	440
8120	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	462
8142	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	484
8164	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	506

+30720 color

04 CHARTS	40 GRAPHICS	105 INPUT
12 BASIC	48 MEMORY	110 EXPANSIONS
18 VARIABLES	76 LABELS	114 STORAGE
22 CHARACTERS	80 ASSEMBLY	125 SOUND
30 SCREEN	100 KEYBOARD	126 ERRORS
36 VIDEO	102 HARDWARE	128 INDEX

BLK	WHT	RED	CYN	PUR	GRN	BLU	YEL	
								BLK
8	9	10	11	12	13	14	15	
								WHT
24	25	26	27	28	29	30	31	
								RED
40	41	42	43	44	45	46	47	
								CYN
56	57	58	59	60	61	62	63	
								PUR
72	73	74	75	76	77	78	79	
								GRN
88	89	90	91	92	93	94	95	
								BLU
104	105	106	107	108	109	110	111	
								YEL
120	121	122	123	124	125	126	127	
								ORG
136	137	138	139	140	141	142	143	
								LT ORG
152	153	154	155	156	157	158	159	
								LT PNK
168	169	170	171	172	173	174	175	
								LT CYN
184	185	186	187	188	189	190	191	
								LT PUR
200	201	202	203	204	205	206	207	
								LT GRN
216	217	218	219	220	221	222	223	
								LT BLU
232	233	234	235	236	237	238	239	
								LT YEL
248	249	250	251	252	253	254	255	

VIC 20

QUICK REFERENCE HANDBOOK

VERSION 0.3B

EDITED BY JEFFREY DANIELS

This compendium presumes basic understanding of VIC 20 programming and is designed around my personal preferences as a user. It contains tables and maps in many configurations informed by the foundational works of the following:

Jim Butterfield	Dale Gilbert	Michael Krause
Andy Finkel	Brian Grainger	Andreas Dripke
Neil Harris	Karl Hildon	Sven Petersen
Paul Higginbottom	Raeto Colin West	David Barron
Michael Tomczyk	Jim Wilcox	and others
Russ Davies	Ward F. Shrake	
Bob Yannes	Paul A. Le Brasse	

including ongoing research by members of the Denial community where more detailed information and explications can be found.

Special thanks to Denial contributors: chysn, wimoos, DarwinNE, Schlowski, srowe, Vic-20-Ian, Noizer, joshuadenmark, Kweepa, orion70, AndyH, Mike, necronom, vicist, mathom, thegg, doug_in_nc, pixel, and others for offering suggestions, proofreading, and contributions.

This book is a work in progress; future revisions will respond to user needs. It is currently available in digital and print forms: Version 0.3 was made in September 2022. Spiral bound (lay-flat) editions are available by special request directly from the author.

QUICK START	
LOAD FROM TAPE:	LOAD then press play on tape
LOAD FROM DISK:	LOAD"* [or <filename>]",8,1
LOAD 2 PART IMAGE:	LOAD"<A000 file>",8,1 NEW LOAD"<2000 or 6000 file>",8,1 SYS64802
START BASIC PROGRAM:	RUN
WARM RESET (START ROM):	SYS64802
SPECIFIC SOFTWARE SYS:	32592 Scott Adams 64096 Vicmon 28681 Programmers Aid 24576 Vicmon

Special thanks for the generous support of the following:

Adrian Fox	Francios Leveille	Leif Bloomquist	Renato Bugge
Alessandro Ubiali	Fredric Blåholtz	Leo LaFlamme	Robert Harbron
Anders Kator	Gabriel Angelos	Louis Mazzei	Roberto Bernardo
Andrew Karlsson	Geoff Nelson	Marco Bergomi	Robin Hurst
Andrew Layden	Ghislain de Blois	mark gladson	P1XL Games
Bindi Hawkey	Glen Richards	Massimiliano Certelli	Ryan Liston
Björg Stojalowski	Heather Kent	Mat Allen	Sampo Rintanen
Bo Goeran Kvamme	Ian Buxton	Matt Dawson	Scott Walters
Brad Bell	James Happel	Matthew Walworth	SignalDEV
Brent Santin	Jan-Erik Sundh	Michael Ingleby	Simon Henstock
Bret Kellihan	Jani Iltanen	Michael Kircher	Stefan Termén
Brian Lyons	jeff smaldon	mega-cart.com	Steve McCrea
Brian Thomson	Jess Ragan	Olli Savia	Tobias Andersson
Bryan Henry	Jonathan Brawn	Paul Lambert	VGA Foundation
Bryce Wilson	Joonas Merienn	Paul Quirk	Wayne Tomlinson
Christopher Prest	K Bergman	Pedro Bermejo	William M
Clockwork Logic	Kevin Potts	Pedro Lambrini	and others
Colton T Stephens	Larry Cameron	Peter Hurst	
David T	Laurent DIZY	Peter Jensen	



CC BY-NC-SA 2022 Attribution — NonCommercial — ShareAlike

This work is licensed under a Creative Commons Attribution-NonCommercial License (US/v3.0). The presentation of the information may be duplicated with attribution. Should this publication contain copyrighted materials, all rights remain with the respective copyright holders. Permissions beyond the scope of this license are administered by Jeffrey Daniels. Information on how to request permission may be found at the Denial forum website: sleepingelephant.com/denial

































Commodore VIC 20	66 keys QWERTY full stroke
VIC-1001 Japan 1980	cartridge port: 44 pin
VIC-20 USA May 1981	1 game port: 9 pin
VC-20 Germany 1981	user port: RS-232 24 pin
Discontinued: January 1985	serial port: IEEE-488 6 pin
CBM Basic V2 (8 KB)	audio/Video: 5 pin
KERNAL (8 KB)	sound: VIC-1 6561
Charset (4 KB)	mono sound: 3 square, 1 noise
CPU: 6502A	Power supply: 9 V
1.0227 MHz NTSC	25 W consumption
1.108 MHz PAL	23 rows x 22 columns
Coprocessor: VIC-I (6560)	184 x 176 3-bpp (184 x 88 mc)
20 KB ROM 5 KB RAM	8 screen colors
3583 bytes free	16 background/aux colors
Computer dimensions (common):	
403 x 204 x 72 mm, 1800 g (15.9 x 8 x 2.9 in., 4 lbs.)	
Cartridge dimensions (common):	
139.7 mm x 85.725 x 15.9 mm (5.5 x 3.375 x 0.625 in.)	
Total disk storage capacity: 174,848 bytes per disk on 35 tracks	
Sequential files: 168,656 bytes per disk	
Relative files: 167,132 bytes per disk, 65,535 records per file	
Entries of a directory (track 18): 144 per disk	
Sectors per track: 17 (inner) to 21 (outer) depending on zone	
Blocks: 683 (664 blocks free)	

DEC	HEX	MSB	SCRN	BASIC	6502	BIN	PETSCII
0	00	0	@	EoL	BRK	□□□□□□□□	dot mode
1	01	256	A		ORA(I,X)	□□□□□□■□	double width
2	02	512	B		JAM	□□□□□□■□	
3	03	768	C		SLO(I,X)	□□□□□□■□	
4	04	1024	D		NOP Z	□□□□□■□□	
5	05	1280	E	WHT	ORA Z	□□□□□■□■	WHT
6	06	1536	F		ASL Z	□□□□□■□□	
7	07	1792	G		SLO Z	□□□□□■□■	
8	08	2048	H		PHP	□□□□■□□□	enable case
9	09	2304	I		ORA #	□□□□■□□■	disable case
10	0A	2560	J		ASL A	□□□□■□□□	line feed
11	0B	2816	K		ANC #	□□□□■□□■	
12	0C	3072	L		NOP	□□□□■□□□	
13	0D	3328	M		ORA	□□□□■□□■	car return
14	0E	3584	N		ASL	□□□□■□□□	lower case
15	0F	3840	O		SLO	□□□□■□□■	end double
16	10	4096	P		BPL	□□■□□□□□	tab
17	11	4352	Q	down	ORA(I),Y	□□■□□□□■	down
18	12	4608	R	rvs on	JAM	□□■□□□□□	rvs on
19	13	4864	S	home	SLO(I),Y	□□■□□□□■	home
20	14	5120	T	insert	NOP Z,X	□□■□□□□□	insert
21	15	5376	U		ORA Z,X	□□■□□□□■	
22	16	5632	V		ASL Z,X	□□■□□□□□	
23	17	5888	W		SLO Z,X	□□■□□□□■	
24	18	6144	X		CLC	□□■□□□□□	
25	19	6400	Y		ORA Y	□□■□□□□■	
26	1A	6656	Z		NOP	□□■□□□□□	repeat
27	1B	6912	[SLO, Y	□□■□□□□■	dot address
28	1C	7168	£	RED	NOP X	□□■□□□□□	RED
29	1D	7424]	right	ORA X	□□■□□□□■	right
30	1E	7680	↑	GRN	ASL X	□□■□□□□□	GRN
31	1F	7936	←	BLU	SLO X	□□■□□□□■	BLU

DEC	HEX	MSB	SCRN	BASIC	6502	BIN	PETSCII
32	20	8192	space	space	JSR	□□■□□□□□	space
33	21	8448	!	!	AND(I,X)	□□■□□□□■	!
34	22	8704	"	"	JAM	□□■□□□□□	"
35	23	8960	#	#	RLA(I,X)	□□■□□□□■	#
36	24	9216	\$	\$	BIT Z	□□■□□■□□	\$
37	25	9472	%	%	AND Z	□□■□□■□■	%
38	26	9728	&	&	ROL Z	□□■□□■□□	&
39	27	9984	/	/	RLA Z	□□■□□■□■	/
40	28	10240	((PLP	□□■□■□□□	(
41	29	10496))	AND IMM	□□■□■□□■)
42	2A	10752	*	*	ROL A	□□■□■□□□	*
43	2B	11008	+	+	ANC #	□□■□■□□■	+
44	2C	11264	,	,	BIT	□□■□■□□□	,
45	2D	11520	-	-	AND	□□■□■□□■	-
46	2E	11776	.	.	ROL	□□■□■□□□	.
47	2F	12032	/	/	RLA	□□■□■□□■	/
48	30	12288	0	0	BMI	□□■□■□□□	0
49	31	12544	1	1	AND(I),Y	□□■□■□□■	1
50	32	12800	2	2	JAM	□□■□■□□□	2
51	33	13056	3	3	RLA(I),Y	□□■□■□□■	3
52	34	13312	4	4	NOP Z,X	□□■□■□□□	4
53	35	13568	5	5	AND Z,X	□□■□■□□■	5
54	36	13824	6	6	ROL Z,X	□□■□■□□□	6
55	37	14080	7	7	RLA Z,X	□□■□■□□■	7
56	38	14336	8	8	SEC	□□■□■□□□	8
57	39	14592	9	9	AND Y	□□■□■□□■	9
58	3A	14848	:	:	NOP	□□■□■□□□	:
59	3B	15104	;	;	RLA Y	□□■□■□□■	;
60	3C	15360	<	<	NOP	□□■□■□□□	<
61	3D	15616	=	=	AND X	□□■□■□□■	=
62	3E	15872	>	>	ROL X	□□■□■□□□	>
63	3F	16128	?	?	RLA X	□□■□■□□■	?

DEC	HEX	MSB	SCRN	BASIC	6502	BIN	PETSCII
64	40	16384	▬	@	RTI	□■□□□□□□	@
65	41	16640	♣	À	EOR(I,X)	□■□□□□□■	À
66	42	16896		B	JAM	□■□□□□□□	B
67	43	17152	▬	C	SRE(I,X)	□■□□□□□■	C
68	44	17408	▬	D	NOP Z	□■□□□□□□	D
69	45	17664	▬	E	EOR Z	□■□□□□□■	E
70	46	17920	▬	F	LSR Z	□■□□□□□□	F
71	47	18176		G	SRE Z	□■□□□□□■	G
72	48	18432		H	PHA	□■□□■□□□	H
73	49	18688	↖	I	EOR #	□■□□■□□■	I
74	4A	18944	↙	J	LSR A	□■□□■□□□	J
75	4B	19200	↘	K	ALR #	□■□□■□□■	K
76	4C	19456	▬	L	JMP	□■□□■□□□	L
77	4D	19712	↘	M	EOR	□■□□■□□■	M
78	4E	19968	↗	N	LSR	□■□□■□□□	N
79	4F	20224	▬	O	SRE	□■□□■□□■	O
80	50	20480	▬	P	BVC	□■□■□□□□	P
81	51	20736	●	Q	EOR(I),Y	□■□■□□□■	Q
82	52	20992	▬	R	JAM	□■□■□□□□	R
83	53	21248	♥	S	SRE(I),Y	□■□■□□□■	S
84	54	21504		T	NOP	□■□■□□□□	T
85	55	21760	↗	U	EOR Z,X	□■□■□□□■	U
86	56	22016	✕	V	LSR Z,X	□■□■□□□□	V
87	57	22272	□	W	SRE Z,X	□■□■□□□■	W
88	58	22528	♣	X	CLI	□■□■□□□□	X
89	59	22784		Y	EOR Y	□■□■□□□■	Y
90	5A	23040	♣	Z	NOP	□■□■□□□□	Z
91	5B	23296	+	[SRE Y	□■□■□□□■	[Å Æ
92	5C	23552	⊗	£	NOP X	□■□■□□□□	£ Ö Ä Ø
93	5D	23808]	EOR X	□■□■□□□■] Å Ü
94	5E	24064	⊗	↑	LSR X	□■□■□□□□	↑
95	5F	24320	↙	←	SRE X	□■□■□□□■	←

DEC	HEX	MSB	SCRN	BASIC	6502	BIN	PETSCII
96	60	24576	space		RTS	□■■■■■■□□	
97	61	24832			ADC(I,X)	□■■■■■■□■	チ
98	62	25088			JAM	□■■■■■■□□	ツ
99	63	25344			RRA(I,X)	□■■■■■■■■	テ
100	64	25600			NOP Z	□■■■■■■□□	ト
101	65	25856			ADC Z	□■■■■■■□■	ナ
102	66	26112			ROR Z	□■■■■■■□□	ニ
103	67	26368			RRA Z	□■■■■■■■■	ヌ
104	68	26624			PLA	□■■■■■■□□	ネ
105	69	26880			ADC #	□■■■■■■□■	ノ
106	6A	27136			ROR A	□■■■■■■□□	ハ
107	6B	27392			ARR #	□■■■■■■■■	ヒ
108	6C	27648			JMP(I)	□■■■■■■□□	フ
109	6D	27904			ADC	□■■■■■■□■	ヘ
110	6E	28160			ROR	□■■■■■■□□	ホ
111	6F	28416			RRA	□■■■■■■■■	マ
112	70	28672			BVS	□■■■■■■□□	ミ
113	71	28928			ADC(I,Y)	□■■■■■■□■	ム
114	72	29184			JAM	□■■■■■■□□	メ
115	73	29440			RRA(I,Y)	□■■■■■■■■	モ
116	74	29696			NOP Z,X	□■■■■■■□□	ヤ
117	75	29952			ADC Z,X	□■■■■■■□■	ユ
118	76	30208			ROR Z,X	□■■■■■■□□	ヨ
119	77	30464			RRA Z,X	□■■■■■■■■	ラ
120	78	30720			SEI	□■■■■■■□□	リ
121	79	30976			ADC Y	□■■■■■■□■	ル
122	7A	31232			NOP	□■■■■■■□□	レ
123	7B	31488			RRA Y	□■■■■■■■■	リ
124	7C	31744			NOP X	□■■■■■■□□	ワ
125	7D	32000			ADC X	□■■■■■■□■	リ
126	7E	32256			ROR X	□■■■■■■□□	リ
127	7F	32512			RRA X	□■■■■■■■■	ヲ

DEC	HEX	MSB	SCRN	BASIC	6502	BIN	PETSCII
128	80	32768		END	NOP #	■□□□□□□□	
129	81	33024		FOR	STA(I,X)	■□□□□□■□	end double
130	82	33280		NEXT	NOP #	■□□□□□■□	
131	83	33536		DATA	SAX(I,X)	■□□□□□■□	
132	84	33792		INPUT#	STY Z	■□□□□■□□	
133	85	34048		INPUT	STA Z	■□□□□■□■	F1
134	86	34304		DIM	STX Z	■□□□□■□□	F3
135	87	34560		READ	SAX Z	■□□□□■□■	F5
136	88	34816		LET	DEY	■□□□■□□□	F7
137	89	35072		GOTO	NOP #	■□□□■□□■	F2
138	8A	35328		RUN	TXA	■□□□■□□□	F4
139	8B	35584		IF	ANE #	■□□□■□□■	F6
140	8C	35840		RESTORE	STY	■□□□■□□□	F8
141	8D	36096		GOSUB	STA	■□□□■□□■	shift return
142	8E	36352		RETURN	STX	■□□□■□□□	upper case
143	8F	36608		REM	SAX	■□□□■□□■	
144	90	36864		STOP	BCC	■□□■□□□□	BLK
145	91	37120		ON	STA(I),Y	■□□■□□□■	up
146	92	37376		WAIT	JAM	■□□■□□□□	rvs off
147	93	37632		LOAD	SHA(I),Y	■□□■□□□■	clear
148	94	37888		SAVE	STY Z,X	■□□■□■□□	insert
149	95	38144		VERIFY	STA Z,X	■□□■□■□■	
150	96	38400		DEF	STX Z,Y	■□□■□■□□	
151	97	38656		POKE	SAX Z, Y	■□□■□■□■	
152	98	38912		PRINT#	TYA	■□□■□■□□	
153	99	39168		PRINT	STA Y	■□□■□■□■	
154	9A	39424		CONT	TXS	■□□■□■□□	
155	9B	39680		LIST	TAS Y	■□□■□■□■	
156	9C	39936		CLR	SHY	■□□■□■□□	PUR
157	9D	40192		CMD	STA X	■□□■□■□■	left
158	9E	40448		SYS	SHX Y	■□□■□■□□	YEL
159	9F	40704		OPEN	SHA Y	■□□■□■□■	CYN

DEC	HEX	MSB	SCRN	BASIC	6502	BIN	PETSCII
160	A0	40960	█	CLOSE	LDY #	█□□□□□□□	space
161	A1	41216	█!	GET	LDA(I,X)	█□□□□□□█	ア
162	A2	41472	█	NEW	LDX #	█□□□□□□□	イ
163	A3	41728	█#	TAB(LAX(I,X)	█□□□□□█	ウ
164	A4	41984	█\$	TO	LDY Z	█□□□□█□□	エ
165	A5	42240	█%	FN	LDA Z	█□□□□█□█	オ
166	A6	42496	█&	SPC(LDX Z	█□□□□█□□	カ
167	A7	42752	█'	THEN	LAX Z	█□□□□█	キ
168	A8	43008	█(NOT	TAY	█□□□□□□□	ク
169	A9	43264	█)	STEP	LDA #	█□□□□□□█	ケ
170	AA	43520	█*	+	TAX	█□□□□□□□	・
171	AB	43776	█+	-	LXA #	█□□□□□█	ト
172	AC	44032	█,	*	LDY	█□□□□█□□	ス
173	AD	44288	█-	/	LDA	█□□□□□□█	ル
174	AE	44544	█.	↑	LDX	█□□□□█□□	レ
175	AF	44800	█/	AND	LAX	█□□□□█	°
176	B0	45056	█0	OR	BCS	█□□□□□□□	ル
177	B1	45312	█1	>	LDA(I),Y	█□□□□□□█	上
178	B2	45568	█2	=	JAM	█□□□□□□□	下
179	B3	45824	█3	<	LAX(I),Y	█□□□□□█	下
180	B4	46080	█4	SGN	LDY Z,X	█□□□□█□□	年
181	B5	46336	█5	INT	LDA Z,X	█□□□□□□█	月
182	B6	46592	█6	ABS	LDX Z,Y	█□□□□█□□	日
183	B7	46848	█7	USR	LAX Z,Y	█□□□□█	タ
184	B8	47104	█8	FRE	CLV	█□□□□□□□	ロ
185	B9	47360	█9	POS	LDA Y	█□□□□□□█	ン
186	BA	47616	█:	SQR	TSX	█□□□□□□□	コ
187	BB	47872	█;	RND	LAS Y	█□□□□□█	サ
188	BC	48128	█<	LOG	LDY X	█□□□□□□□	シ
189	BD	48384	█=	EXP	LDA X	█□□□□□□█	ル
190	BE	48640	█>	COS	LDX Y	█□□□□█□□	セ
191	BF	48896	█?	SIN	LAX Y	█□□□□█	ソ

DEC	HEX	MSB	SCRN	BASIC	6502	BIN	PETSCII
192	C0	49152		TAN	CPY #	■■■■■■■■	
193	C1	49408		ATN	CMP(I),X	■■■■■■■■■	
194	C2	49664		PEEK	NOP #	■■■■■■■■	
195	C3	49920		LEN	DCP(I,X)	■■■■■■■■■	
196	C4	50176		STR\$	CPY Z	■■■■■■■■	
197	C5	50432		VAL	CMP Z	■■■■■■■■■	
198	C6	50688		ASC	DEC Z	■■■■■■■■	
199	C7	50944		CHR\$	DCP Z	■■■■■■■■■	
200	C8	51200		LEFT\$	INY	■■■■■□□□	
201	C9	51456		RIGHT\$	CMP #	■■■■■□□■	
202	CA	51712		MID\$	DEX	■■■■■□□□	
203	CB	51968		GO	SBX #	■■■■■□□■	
204	CC	52224		?SYNTAX	CPY	■■■■■□□□	
205	CD	52480		FOR	CMP	■■■■■□□■	
206	CE	52736		NEXT	DEC	■■■■■□□□	
207	CF	52992		DATA	DCP	■■■■■□□■	
208	D0	53248		INPUT#	BNE	■■■■■□□□	
209	D1	53504		INPUT	CMP(I),Y	■■■■■□□■	
210	D2	53760		DIM	JAM	■■■■■□□□	
211	D3	54016		READ	DCP(I),Y	■■■■■□□■	
212	D4	54272		LET	NOP Z,X	■■■■■□□□	
213	D5	54528		GOTO	CMP Z,X	■■■■■□□■	
214	D6	54784		RUN	DEC Z,X	■■■■■□□■	
215	D7	55040		IF	DCP Z,X	■■■■■□□■	
216	D8	55296		RESTORE	CLD	■■■■■□□□	
217	D9	55552		GOSUB	CMP Y	■■■■■□□■	
218	DA	55808		RETURN	NOP	■■■■■□□□	
219	DB	56064		REM	DCP Y	■■■■■□□■	
220	DC	56320		STOP	NOP X	■■■■■□□□	
221	DD	56576		ON	CMP X	■■■■■□□■	
222	DE	56832		WAIT	DEC X	■■■■■□□□	
223	DF	57088		LOAD	DCP X	■■■■■□□■	

DEC	HEX	MSB	SCRN	BASIC	6502	BIN	PETSCII
224	E0	57344	■	SAVE	CPX #	■■■■□□□□	space
225	E1	57600	■	VERIFY	SBC(I),X	■■■■□□□■	ア
226	E2	57856	■	DEF	NOP #	■■■■□□□■	イ
227	E3	58112	■	POKE	ISC(Z,X)	■■■■□□■	ウ
228	E4	58368	■	PRINT#	CPX Z	■■■■□■□□	エ
229	E5	58624	■	PRINT	SBC Z	■■■■□□■	オ
230	E6	58880	■	CONT	INC Z	■■■■□□■	カ
231	E7	59136	■	LIST	ISC Z	■■■■□□■	キ
232	E8	59392	■	CLR	INX	■■■■□□□□	ク
233	E9	59648	■	CMD	SBC #	■■■■□□□■	ケ
234	EA	59904	■	SYS	NOP	■■■■□□□■	・
235	EB	60160	■	OPEN	USBC #	■■■■□□■	ト
236	EC	60416	■	CLOSE	CPX	■■■■□■□□	ス
237	ED	60672	■	GET	SBC	■■■■□□□■	ル
238	EE	60928	■	NEW	INC	■■■■□□□□	リ
239	EF	61184	■	TAB(ISC	■■■■□□■	°
240	F0	61440	■	TO	BEQ	■■■■□□□□	レ
241	F1	61696	■	FN	SBC(I),Y	■■■■□□□■	上
242	F2	61952	■	SPC(JAM	■■■■□□□■	下
243	F3	62208	■	THEN	ISC(I),Y	■■■■□□■	リ
244	F4	62464	■	NOT	NOP Z,X	■■■■□■□□	年
245	F5	62720	■	STEP	SBC Z,X	■■■■□□■	月
246	F6	62976	■	+	INC Z,X	■■■■□□■	日
247	F7	63232	■	-	ISC Z,X	■■■■□□■	タ
248	F8	63488	■	*	SED	■■■■□□□□	ロ
249	F9	63744	■	/	SBC Y	■■■■□□■	ン
250	FA	64000	■	↑	NOP	■■■■□□■	コ
251	FB	64256	■	AND	ISC Y	■■■■□□■	サ
252	FC	64512	■	OR	NOP X	■■■■□□□□	シ
253	FD	64768	■	>	SBC X	■■■■□□□■	ル
254	FE	65024	■	=	INC X	■■■■□□□□	セ
255	FF	65536	■	TT	ISC X	■■■■□□■	ソ

ABS	A shift B	182	returns absolute value of X ABS(<expression>)	DC58	59408
AND	A shift N	175	AND logical operand <expression>AND<expression>	CFE9	53225
ASC	A shift S	198	returns ASCII code ASC(<string>)	D78B	55179
ATN	A shift T	193	returns arctangent ATN(<number>)	E308	58123
CHR\$	C shift H	199	returns ASCII string CHR\$(<number 0-255>)	D6E6	55020
CLOSE	CL shift O	160	closes file engaged by OPEN CLOSE<file number>	FFC3	65475
CLR	C shift L	156	clears all defined variables CLR	C65E	50782
CMD	C shift M	157	changes output to device CMD<file number>[, <string>]	CA86	51846
CONT	C shift O	154	resume from STOP (direct mode) CONT	C857	51287
COS	None	190	returns cosine in radians COS(<number>)	E261	57953
DATA	D shift A	131	hold list of values for READ DATA<value>[<,value...>]	C858	51448
DEF	D shift E	150	defines a function FN DEF FN<name>(<variable>)=<expression>	D3B3	54195
DIM	D shift I	134	defines an array DIM<variable>(<subscripts>)[, <var>(<sub>...)]	D081	53377
END	E shift N	128	exits program END	C831	51249
EXP	E shift X	189	r: exponential (X>0 and X<88.03) EXP(<number>)	DFED	57325
FN	none	165	function defined by DEF DEF FN<name>(<variable>)=<expression>	D3F4	54260
FOR	F shift O	129	repeats section before NEXT FOR<variable>=<start>TO<limit>[STEP<increment>]	C742	51010
FRE	F shift R	184	returns free memory in bytes FRE(<dummy number>)	D37D	54141

GET	G shift E	161	defines var from key pressed GET<variable>	FFE4	65508
GET#	none	none	returns variable from device GET#<file number>,<variable>		
GO	none	203	GO TO variation of GOTO GO TO<line number>	C8A0	51360
GOSUB	GO shift S	141	goes to subroutine GOSUB<line number>	C883	51331
GOTO	G shift O	137	goes to line number GOTO<line number>	C8A0	51360
IF	none	139	evaluates condition IF<expression>THEN<line number/statement>	C928	51496
INPUT	none	133	variable from user input INPUT["<prompt>"];<variable>[, variable...>	FFCF	65487
INPUT#	I shift N	132	defines value from device INPUT#<file number>,< variable [,variable ...]>	CBA5	52133
INT	none	181	returns integer INT(<number>)	DCCC	56524
LEFT\$	LE shift F	200	returns leftmost chars of string LEFT\$(<string>,<length>)	D700	55040
LEN	none	195	number of chars in string LEN(<string>)	D77C	55164
LET	L shift E	136	assigns value to variable [LET]<variable>=<value>	C9A5	51621
LIST	L shift I	155	lists the program in memory LIST[line number][-[line number]]	C69C	50844
LOAD	L shift O	147	transfers program from device LOAD["<file>"][,<device>][,<command>]	FFD5	65493
LOG	none	188	returns natural logarithm LOG(<number>)	D9EA	55786
MID\$	M shift I	202	returns string from middle MID\$(<string>,<start>,<length>)	D737	55095
NEW	none	162	erases current program NEW	C642	50754
NEXT	N shift E	130	loops back to original FOR NEXT<counter>][,<counter>][,<counter>]	CD1E	52510

NOT	N shift O	168	NOT logical operand NOT<expression>	CE4D	52948
ON	none	145	sets branch point on condition ON<variable>GOTO/GOSUB<line>[,<line>]>	C94B	51531
OPEN	O shift P	159	prepares device for access OPEN<file number>[,<device>][,<command>]...	FFC0	65472
OR	none	176	OR logical operand <expression>OR<expression>	CFE6	53222
PEEK	P shift E	194	byte from memory location PEEK(<memory location>)	D80D	55309
POKE	P shift O	151	places value in RAM location POKE<memory location>,<value>	D824	55332
POS	none	185	returns current cursor position POS(<dummy>)	D39E	54174
PRINT	?	153	displays on screen PRINT[<variable> or <string>...]	FFD2	65490
PRINT#	P shift R	152	sends data to device PRINT#<file num>,<variable>[,<variable>...]	CA80	51840
READ	R shift E	135	retrieve next DATA READ<variable>[,<var>,<var>...]	CC06	52230
REM	none	143	holds programmer's remark REM<any text>	C93B	51515
RESTORE	RE shift S	140	resets DATA pointer RESTORE	C81D	51229
RETURN	RE shift T	142	completes GOSUB subroutine RETURN	C8D2	51410
RIGHT\$	R shift I	201	rightmost chars of string RIGHT\$(<string>,<length>)	D72C	55084
RND	R shift N	187	returns random number RND(<number>)	E094	57492
RUN	R shift U	138	executes current program RUN[<line number>]	C871	51313
SAVE	S shift A	148	saves program to device SAVE["<file>"][,<device>][,<command>]	FFD8	65496
SGN	S shift G	180	returns sign SGN(<number>)	DC39	56377

SIN	S shift I	191	returns sine in radians SIN(<number>)	E268	57960
SPC(S shift P	166	prints blank spaces SPC(<number 0-255>)	CAF8	51960
SQR	S shift Q	186	returns square root SQR(<number>)	DF71	57201
STATUS	ST	none	returns status of I/O operation STATUS	C795	51093
STEP	ST shift E	169	sets increment of FOR loop FOR<variable>=<start>TO<limit>[STEP<increment>]	C82F	51247
STOP	S shift T	144	breaks program STOP	D465	54373
STR\$	ST shift R	196	returns string value of variable STR\$(<number>)	E127	57639
SYS	S shift Y	158	begins machine code at location SYS<memory location>	E127	57639
TAB(T shift A	163	positions print starting next line TAB(<number 0-255>)	CAFB	51960
TAN	none	192	returns tangent in radians TAN(<number>)	E2B1	58033
THEN	T shift H	167	decides outcome based on IF IF<expression>THEN<line number/statement>		
TIME	TI	none	returns timer as variable TI		
TIME\$	TI\$	none	timer string HH MM SS TI\$="000000"		
TO	none	164	sets end point of FOR loop FOR<var>=<start>TO<limit>[STEP<increment>]		
USR	U shift S	183	starts ML code with variable USR(<number>)		
VAL	V shift A	197	returns numeric value VAL(<string>)	D7AD	55213
VERIFY	V shift E	149	compares program to storage VERIFY["<file>"][,<device>][,<command>]	FFDB	65499
WAIT	W shift A	146	halts prog until mem condition WAIT<memory location>,<AND-mask>[,<XOR-mask>]	D82D	55341

BASIC PROGRAM TOKENS							
0	end of line	67	C	135	READ	172	*
1-31	unused	68	D	136	LET	173	/
32	{space}	69	E	137	GOTO	174	↑
33	!	70	F	138	RUN	175	AND
34	"	71	G	139	IF	176	OR
35	#	72	H	140	RESTORE	177	>
36	\$	73	I	141	GOSUB	178	=
37	%	74	J	142	RETURN	179	<
38	&	75	K	143	REM	180	SGN
39	'	76	L	144	STOP	181	INT
40	(77	M	145	ON	182	ABS
41)	78	N	146	WAIT	183	USR
42	*	79	O	147	LOAD	184	FRE
43	+	80	P	148	SAVE	185	POS
44	,	81	Q	149	VERIFY	186	SQR
45	-	82	R	150	DEF	187	RND
46	.	83	S	151	POKE	188	LOG
47	/	84	T	152	PRINT#	189	EXP
48	0	85	U	153	PRINT	190	COS
49	1	86	V	154	CONT	191	SIN
50	2	87	W	155	LIST	192	TAN
51	3	88	X	156	CLR	193	ATN
52	4	89	Y	157	CMD	194	PEEK
53	5	90	Z	158	SYS	195	LEN
54	6	91	[159	OPEN	196	STR\$
55	7	92	£	160	CLOSE	197	VAL
56	8	93]	161	GET	198	ASC
57	9	94	↑	162	NEW	199	CHR\$
58	:	95	←	163	TAB(200	LEFT\$
59	;	96-127	unused	164	TO	201	RIGHT\$
60	<	128	END	165	FN	202	MID\$
61	=	129	FOR	166	SPC(203	GO
62	>	130	NEXT	167	THEN	204-254	unused
63	?	131	DATA	168	NOT	255	π
64	A	132	INPUT#	169	STEP		
65	B	133	INPUT	170	+		
66	C	134	DIM	171	-		

EXAMPLE OF BASIC PROGRAM LINE STRUCTURE

```

1 A=2:PRINTA
2 GOTO1

```

memory	BASIC	value	Program Explication
4096		0	Program start ↓ pointer 43-44 (2B-2C)
4097		12	Next line (low byte): 12
4098		16	Next line (high byte, *256 = 4096) = 4108
4099	1	1	Line Number (low byte)
4100		0	Line Number (high byte *256)
4101	A	65	Variable A (PETSCII A)
4102	=	178	Relational operator token
4103	2	50	Value of variable (PETSCII 2)
4104	:	58	Colon
4105	PRINT	153	BASIC keyword token
4106	A	65	Variable A (PETSCII A)
4107		0	End of line marker (EoL)
4108		19	Next line (low byte): 19
4109		16	Next line (high byte *256 = 4096) = 4115
4110	2	2	Line Number (low byte)
4111		0	Line Number (high byte *256)
4112	GOTO	137	BASIC keyword token
4113	1	49	Destination line number for GOTO
4114		0	End of line marker (EoL)
4115		0	0 (end of program indicator) 0
4116		0	0 (end of program indicator) 0
4117-	Variables start		pointer 45-46 (2D-2E)
	Arrays start (variables end)		pointer 47-48 (2F-30)
	Array end and start of empty space		pointer 49-50 (2B-2C)
	String start (direction ↑)		pointer 51-52 (33-34)
	Active string start		pointer 53-54 (35-36)
-7679	Program memory end		pointer 55-56 (37-38)

VARIABLES

Only the first two characters, plus the identifier symbol, are significant in identifying a variable.

REAL NUMERIC (FLOATING POINT)		\$2D \$2E
RANGE	any decimal ranged $\pm 2.93 \times 10^{-38}$ to $\pm 1.70 \times 10^{38}$	
EXAMPLES	A AB A1	reserved: ST TI
MEMORY	7 bytes: 1 byte for exponent, 4 bytes for mantissa	
ARRAY MEMORY	5 bytes per element	
POINTERS	45-46 (\$2D-\$2E)	

NUMERIC FUNCTIONS

ABS(X)	returns absolute value of X
ATN(X)	returns arctangent of X
COS(X)	returns cosine of X in radians
DEF FN	defines and returns a function FN X
EXP(X)	returns exponential of X ($X > 0$ and $X < 88.03$)
FRE(X)	returns free memory in bytes (X may be any value)
INT(X)	returns integer value of X
LOG(X)	returns natural logarithm of X
PEEK(X)	returns byte value from memory location X
RND(X)	returns random number based on X as seed
SGN(X)	returns sign of X as value (-1,0,1)
SIN(X)	returns sine of X in radians
SQR(X)	returns square root of X
ST	returns system status (last I/O operation) as variable
TAN(X)	returns tangent of X in radians
TI	returns timer as variable
USR(X)	starts ML code with variable X
VAL(A\$)	converts and returns numeric value of A\$

INTEGER		\$2D \$2E
RANGE	any whole number ranged -32768 to 32767	
EXAMPLES	A% AB% A1%	
MEMORY	7 bytes: 2 bytes for a 16-bit value, rest is 0	
ARRAY MEMORY	2 bytes per element	
POINTERS	45-46 (\$2D-\$2E)	

STRING		\$33 \$34
RANGE	0 to 255 PETSCII characters	
EXAMPLES	A\$ AB\$ A1\$	reserved: TI\$
MEMORY	7 bytes: 1 byte for length, 2 bytes pointer, rest is 0	
ARRAY MEMORY	3 bytes plus length of string per element	
POINTERS	51-52 (\$33-\$34)	

STRING AND PRINT FUNCTIONS

ASC(A\$)	returns ASCII code of A\$
CHR\$(X)	returns ASCII string of X
LEFT\$(A\$,X)	returns leftmost X characters of string A\$
LEN(A\$)	returns the number of characters in string A\$
MID\$(A\$,S,L)	returns L characters of string A\$ starting with Sth character
POS(X)	returns current cursor position
RIGHT\$(A\$,X)	returns rightmost X characters of string A\$
SPC(X)	move print position X spaces
STR\$(X)	returns string value of variable X
TAB(X)	move print position X spaces relative to left margin
TI\$	returns six character clock string (hours, minutes, seconds)
VAL(A\$)	converts and returns numeric value of A\$

ORDER OF OPERATIONS	1 st	()	6 th	< = >
	2 nd	↑	7 th	NOT
	3 rd	negative	8 th	AND
	4 th	* /	9 th	OR
	5 th	+ -		

TRUE = -1 (1=1)=-1
FALSE = 0 (1=2)=0

XOR	$X \text{ XOR } Y = (X \text{ AND NOT } Y) \text{ OR } (\text{NOT } X \text{ AND } Y)$
NOR	$A \text{ NOR } B = \text{NOT } (A \text{ OR } B)$
secant	$\text{SEC}(X) = 1/\text{COS}(X)$
cosecant	$\text{CSC}(X) = 1/\text{SIN}(X)$
cotangent	$\text{COT}(X) = 1/\text{TAN}(X)$
inverse sine	$\text{ARCSIN}(X) = \text{ATN}(X/\text{SQR}(-X*X+1))$
inverse cosine	$\text{ARCCOS}(X) = -\text{ATN}(X/\text{SQR}(-X*X+1))+\pi/2$
inverse secant	$\text{ARCSEC}(X) = \text{ATN}(X/\text{SQR}(X*X-1))$
inverse cosecant	$\text{ARCCSC}(X) = \text{ATN}(X/\text{SQR}(X*X-1))+(\text{SGN}(X)-1*\pi/2)$
inverse cotangent	$\text{ARCOT}(X) = \text{ATN}(X)+\pi/2$
hyperbolic sine	$\text{SINH}(X) = (\text{EXP}(X)-\text{EXP}(-X))/2$
hyperbolic cosine	$\text{COSH}(X) = (\text{EXP}(X)+\text{EXP}(-X))/2$
hyperbolic tangent	$\text{TANH}(X) = \text{EXP}(-X)/(\text{EXP}(X)+\text{EXP}(-X))*2+1$
hyperbolic secant	$\text{SECH}(X) = 2/(\text{EXP}(X)+\text{EXP}(-X))$
hyperbolic cosecant	$\text{CSCH}(X) = 2/(\text{EXP}(X)-\text{EXP}(-X))$
hyperbolic cotangent	$\text{COTH}(X) = \text{EXP}(-X)/(\text{EXP}(X)-\text{EXP}(-X))*2+1$
inverse hyperbolic sine	$\text{ARCSINH}(X) = \text{LOG}(X+\text{SQR}(X*X+1))$
inverse hyperbolic cosine	$\text{ARCCOSH}(X) = \text{LOG}(X+\text{SQR}(X*X-1))$
inverse hyperbolic tangent	$\text{ARCTANH}(X) = \text{LOG}((1+X)/(1-X))/2$
inverse hyperbolic secant	$\text{ARCSECH}(X) = \text{LOG}((\text{SQR}(-X*X+1)+1)/X)$
inverse hyperbolic cosecant	$\text{ARCCSCH}(X) = \text{LOG}((\text{SGN}(X)*\text{SQR}(X*X+1)/X))$
inverse hyperbolic cotangent	$\text{ARCCOTH}(X) = \text{LOG}((X+1)/(X-1))/2$

NOT	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
=	6	5	4	3	2	1	0	-1	-2	-3	-4	-5	-6	-7	-8
















AND	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
2	0	2	2	0	0	2	2	0	0	2	2	0	0	2	2
3	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
4	0	0	0	4	4	4	4	0	0	0	0	4	4	4	4
5	1	0	1	4	5	4	5	0	1	0	1	4	5	4	5
6	0	2	2	4	4	6	6	0	0	2	2	4	4	6	6
7	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
8	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8
9	1	0	1	0	1	0	1	8	9	8	9	8	9	8	9
10	0	2	2	0	0	2	2	8	8	10	10	8	8	10	10
11	1	2	3	0	1	2	3	8	9	10	11	8	9	10	11
12	0	0	0	4	4	4	4	8	8	8	8	12	12	12	12
13	1	0	1	4	5	4	5	8	9	8	9	12	13	12	13
14	0	2	2	4	4	6	6	8	8	10	10	12	12	14	14
15	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

OR	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	3	3	5	5	7	7	9	9	11	11	13	13	15	15
2	3	2	3	6	7	6	7	10	11	10	11	14	15	14	15
3	3	3	3	7	7	7	7	11	11	11	11	15	15	15	15
4	5	6	7	4	5	6	7	12	13	14	15	12	13	14	15
5	5	7	7	5	5	7	7	13	13	15	15	13	13	15	15
6	7	6	7	6	7	6	7	14	15	14	15	14	15	14	15
7	7	7	7	7	7	7	7	15	15	15	15	15	15	15	15
8	9	10	11	12	13	14	15	8	9	10	11	12	13	14	15
9	9	11	11	13	13	15	15	9	9	11	11	13	13	15	15
10	11	10	11	14	15	14	15	10	11	10	11	14	15	14	15
11	11	11	11	15	15	15	15	11	11	11	11	15	15	15	15
12	13	14	15	12	13	14	15	12	13	14	15	12	13	14	15
13	13	15	15	13	13	15	15	13	13	15	15	13	13	15	15
14	15	14	15	14	15	14	15	14	15	14	15	14	15	14	15
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15

PETSII		useful printer codes					
0	dot mode	32		64	@ @	96	— —
1	double width	33	! !	65	À à	97	▲ A
2		34	" "	66	B b	98	▮ B
3		35	# #	67	C c	99	— C
4		36	\$ \$	68	D d	100	— D
5	WHT	37	% %	69	E e	101	— E
6		38	& &	70	F f	102	— F
7		39	/ /	71	G g	103	▮ G
8	enable case	40	((72	H h	104	▮ H
9	disable case	41))	73	I i	105	↙ I
10	line feed	42	* *	74	J j	106	↘ J
11		43	+ +	75	K k	107	↙ K
12		44	, ,	76	L l	108	▮ L
13	car return	45	- -	77	M m	109	↘ M
14	lower case	46	. .	78	N n	110	↘ N
15	end double	47	/ /	79	O o	111	▮ O
16	tab	48	Ø Ø	80	P p	112	▮ P
17	down (low)	49	1 1	81	Q q	113	● Q
18	rvs on	50	2 2	82	R r	114	— R
19	home	51	3 3	83	S s	115	♥ S
20	insert	52	4 4	84	T t	116	▮ T
21		53	5 5	85	U u	117	↙ U
22		54	6 6	86	V v	118	× V
23		55	7 7	87	W w	119	◻ W
24		56	8 8	88	X x	120	✻ X
25		57	9 9	89	Y y	121	▮ Y
26	repeat	58	: :	90	Z z	122	◆ Z
27	dot address	59	; ;	91	[Ä Æ	123	+ +
28	RED	60	< <	92	£ ö Ä ø	124	▮ ▮
29	right	61	= =	93] Å Ü	125	▮
30	GRN	62	> >	94	↑ ↑	126	π ✻
31	BLU	63	? ?	95	← ←	127	▮ ▮

PETSCII							
128		160		192	— —	224	
129	end double	161		193	A	225	
130		162		194	B	226	
131		163		195	— C	227	
132		164		196	— D	228	
133	F1	165		197	— E	229	
134	F3	166		198	— F	230	
135	F5	167		199	G	231	
136	F7	168		200	H	232	
137	F2	169		201		233	
138	F4	170		202		234	
139	F6	171		203		235	
140	F8	172		204		236	
141	shift return	173		205		237	
142	upper case	174		206		238	
143		175		207		239	
144	BLK	176		208		240	
145	up (up)	177		209		241	
146	rvs off	178		210	— R	242	
147	clear	179		211		243	
148	insert	180		212	T	244	
149		181		213		245	
150		182		214		246	
151		183		215		247	
152		184		216		248	
153		185		217	Y	249	
154		186		218		250	
155		187		219	+ +	251	
156	PUR	188		220		252	
157	left	189		221		253	
158	YEL	190		222		254	
159	CYN	191		223		255	

QUOTATION MODE

			
BLACK	WHITE	RED	CYAN
CTRL 1	CTRL 2	CTRL 3	CTRL 4
			
PURPLE	GREEN	BLUE	YELLOW
CTRL 5	CTRL 6	CTRL 7	CTRL 8
			
REVERSE ON	REVERSE OFF	HOME	CLEAR
CTRL 9	CTRL 0	HOME	SHIFT HOME
			
UP	DOWN	LEFT	RIGHT
SHIFT CURS DOWN	CURSOR DOWN	SHIFT CURS RIGHT	CURSOR RIGHT
			
F1	F2	F3	F4
F1	SHIFT F1	F3	SHIFT F3
			
F5	F6	F7	F8
F5	SHIFT F5	F7	SHIFT F7
			
INSERT	DELETE	LOWER CASE	?SYNTAX ERROR
SHIFT DEL	** DEL	** RVS N	** RVS SHIFT L
			
DISABLE CASE	ENABLE CASE	SHIFT RETURN	UPPER CASE
** RVS H	** RVS I	** RVS SHIFT M	** RVS SHIFT N

** must be entered manually into the line

SCREEN CHARACTER CODES

← 31	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	0 48	+ 43	- 45	£ 28
---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------

Q 17	W 23	E 5	R 18	T 20	Y 25	U 21	I 9	O 15	P 16	@ 0	* 42	↑ 30
---------	---------	--------	---------	---------	---------	---------	--------	---------	---------	--------	---------	---------

RUN STOP	A 1	S 19	D 4	F 6	G 7	H 8	J 10	K 11	L 12	: 58	; 59	= 61
----------	--------	---------	--------	--------	--------	--------	---------	---------	---------	---------	---------	---------

Z 26	X 24	C 3	V 22	B 2	N 14	M 13	, 44	. 46	/ 47	SPC 32
---------	---------	--------	---------	--------	---------	---------	---------	---------	---------	-----------

←	! 33	" 34	# 35	\$ 36	% 37	& 38	' 39	(40) 41	0 91	+ 93	▀ 105
---	---------	---------	---------	----------	---------	---------	---------	---------	---------	---------	---------	----------

■ 81	□ 87	▬ 69	▬ 82	▬ 84	▬ 89	▬ 85	▬ 73	▬ 79	▬ 80	▬ 122	▬ 64	π 94
---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	----------	---------	---------

RUN STOP	▲ 65	♥ 83	▬ 68	▬ 70	▬ 71	▬ 72	▬ 74	▬ 75	▬ 76	[27] 29	=
----------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	---

◆ 90	♣ 88	▬ 67	✕ 86	▬ 66	▬ 78	▬ 77	◀ 60	▶ 62	? 63	SHIFT
---------	---------	---------	---------	---------	---------	---------	---------	---------	---------	-------

←	1 BLK	2 WHT	3 RED	4 CYN	5 PUR	6 GRN	7 BLU	8 YEL	9 RVS	0 OFF	102	92	104
---	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	-----	----	-----

CTRL	▬ 107	▬ 115	▬ 113	▬ 114	▬ 99	▬ 119	▬ 120	▬ 98	▬ 121	▬ 111	▬ 100	▬ 95	↑
------	----------	----------	----------	----------	---------	----------	----------	---------	----------	----------	----------	---------	---

RUN STOP	▬ 112	▬ 110	▬ 108	▬ 123	▬ 101	▬ 116	▬ 117	▬ 97	▬ 118	:	;	=
----------	----------	----------	----------	----------	----------	----------	----------	---------	----------	---	---	---

▬ 109	▬ 125	▬ 124	▬ 126	▬ 127	▬ 106	▬ 103	,	.	/	↺
----------	----------	----------	----------	----------	----------	----------	---	---	---	---

	103	cmd M	0		32	SPC	0		160	RVS SPC
	89	shift Y	1		103	cmd M	1		231	RVS cmd M
	72	shift H	2		106	cmd N	2		234	RVS cmd N
	93	shift -	3		118	cmd L	3		246	RVS cmd L
	66	shift B	4		225	RVS cmd K	4		97	cmd K
	71	shift G	5		245	RVS cmd J	5		117	cmd J
	84	shift T	6		244	RVS cmd H	6		116	c cmd H
	101	cmd G	7		229	RVS cmd G	7		101	cmd G

	99	cmd T	0		69	shift E	1		68	shift D	2		67	shift C	3		64	shift *	4		70	shift F	5		82	shift R	6		100	cmd @	7
--	----	-------	---	--	----	---------	---	--	----	---------	---	--	----	---------	---	--	----	---------	---	--	----	---------	---	--	----	---------	---	--	-----	-------	---

	32	SPC	0		99	cmd T	1		119	cmd Y	2		120	cmd U	3		226	RVS cmd I	4		119	RVS cmd O	5		239	RVS cmd P	6		228	RVS cmd @	7
--	----	-----	---	--	----	-------	---	--	-----	-------	---	--	-----	-------	---	--	-----	-----------	---	--	-----	-----------	---	--	-----	-----------	---	--	-----	-----------	---

	160	RVS SPC	0		227	RVS cmd T	1		247	RVS cmd Y	2		248	RVS cmd U	3		98	cmd I	4		121	cmd O	5		111	cmd P	6		100	cmd @	7
--	-----	---------	---	--	-----	-----------	---	--	-----	-----------	---	--	-----	-----------	---	--	----	-------	---	--	-----	-------	---	--	-----	-------	---	--	-----	-------	---

85	73
shift U	shift I
74	75
shift J	shift K

112	110
cmd A	cmd S
109	125
cmd Z	cmd X

107	115
cmd Q	cmd W
113	114
cmd E	cmd R

64	66
shift *	shift B
91	86
shift +	shift V

79	80
shift O	shift P
76	122
shift L	shift @

78	77
shift N	shift M
255	127
R cmd B	cmd B

101	103
cmd M	cmd G
99	100
cmd T	cmd @

97	225
shift K	R shift K
226	98
R shift I	shift I

233	223
R shift £	R cmd *
95	105
cmd *	shift £

108	123
cmd D	cmd F
124	126
cmd C	cmd V

236	251
R cmd D	R cmd F
252	254
R cmd C	R cmd V

65	83
shift A	shift S
88	90
shift X	shift Z

160	32
R space	space

81	87
shift Q	shift W

102	230
cmd +	R cmd +

DEC

0	@	32		64	▬	96	
1	A a	33	!	65	♣ A	97	▬
2	B b	34	"	66	▬ B	98	▬
3	C c	35	#	67	▬ C	99	▬
4	D d	36	\$	68	▬ D	100	▬
5	E e	37	%	69	▬ E	101	▬
6	F f	38	&	70	▬ F	102	▬
7	G g	39	'	71	▬ G	103	▬
8	H h	40	(72	▬ H	104	▬
9	I i	41)	73	▬ I	105	▬
10	J j	42	*	74	▬ J	106	▬
11	K k	43	+	75	▬ K	107	▬
12	L l	44	,	76	▬ L	108	▬
13	M m	45	-	77	▬ M	109	▬
14	N n	46	.	78	▬ N	110	▬
15	O o	47	/	79	▬ O	111	▬
16	P p	48	0	80	▬ P	112	▬
17	Q q	49	1	81	▬ Q	113	▬
18	R r	50	2	82	▬ R	114	▬
19	S s	51	3	83	▬ S	115	▬
20	T t	52	4	84	▬ T	116	▬
21	U u	53	5	85	▬ U	117	▬
22	V v	54	6	86	▬ V	118	▬
23	W w	55	7	87	▬ W	119	▬
24	X x	56	8	88	▬ X	120	▬
25	Y y	57	9	89	▬ Y	121	▬
26	Z z	58	:	90	▬ Z	122	▬
27	[59	;	91	▬	123	▬
28	&	60	<	92	▬	124	▬
29]	61	=	93	▬	125	▬
30	↑	62	>	94	▬	126	▬
31	←	63	?	95	▬	127	▬

DEC

128		160	rvs space	192		224	
129		161		193		225	
130		162		194		226	
131		163		195		227	
132		164		196		228	
133		165		197		229	
134		166		198		230	
135		167		199		231	
136		168		200		232	
137		169		201		233	
138		170		202		234	
139		171		203		235	
140		172		204		236	
141		173		205		237	
142		174		206		238	
143		175		207		239	
144		176		208		240	
145		177		209		241	
146		178		210		242	
147		179		211		243	
148		180		212		244	
149		181		213		245	
150		182		214		246	
151		183		215		247	
152		184		216		248	
153		185		217		249	
154		186		218		250	
155		187		219		251	
156		188		220		252	
157		189		221		253	
158		190		222		254	
159		191		223		255	

	0	1	2	3	4	5	6	7	8	9	10
1	7680	7681	7682	7683	7684	7685	7686	7687	7688	7689	7690
2	7702	7703	7704	7705	7706	7707	7708	7709	7710	7711	7712
3	7724	7725	7726	7727	7728	7729	7730	7731	7732	7733	7734
4	7746	7747	7748	7749	7750	7751	7752	7753	7754	7755	7756
5	7768	7769	7770	7771	7772	7773	7774	7775	7776	7777	7778
6	7790	7791	7792	7793	7794	7795	7796	7797	7798	7799	7800
7	7812	7813	7814	7815	7816	7817	7818	7819	7820	7821	7822
8	7834	7835	7836	7837	7838	7839	7840	7841	7842	7843	7844
9	7856	7857	7858	7859	7860	7861	7862	7863	7864	7865	7866
10	7878	7879	7880	7881	7882	7883	7884	7885	7886	7887	7888
11	7900	7901	7902	7903	7904	7905	7906	7907	7908	7909	7910
12	7922	7923	7924	7925	7926	7927	7928	7929	7930	7931	7932
13	7944	7945	7946	7947	7948	7949	7950	7951	7952	7953	7954
14	7966	7967	7968	7969	7970	7971	7972	7973	7974	7975	7976
15	7988	7989	7990	7991	7992	7993	7994	7995	7996	7997	7998
16	8010	8011	8012	8013	8014	8015	8016	8017	8018	8019	8020
17	8032	8033	8034	8035	8036	8037	8038	8039	8040	8041	8042
18	8054	8055	8056	8057	8058	8059	8060	8061	8062	8063	8064
19	8076	8077	8078	8079	8080	8081	8082	8083	8084	8085	8086
20	8098	8099	8100	8101	8102	8103	8104	8105	8106	8107	8108
21	8120	8121	8122	8123	8124	8125	8126	8127	8128	8129	8130
22	8142	8143	8144	8145	8146	8147	8148	8149	8150	8151	8152
23	8164	8165	8166	8167	8168	8169	8170	8171	8172	8173	8174
	0	1	2	3	4	5	6	7	8	9	10

11	12	13	14	15	16	17	18	19	20	21	
7691	7692	7693	7694	7695	7696	7697	7698	7699	7700	7701	22
7713	7714	7715	7716	7717	7718	7719	7720	7721	7722	7723	44
7735	7736	7737	7738	7739	7740	7741	7742	7743	7744	7745	66
7757	7758	7759	7760	7761	7762	7763	7764	7765	7766	7767	88
7779	7780	7781	7782	7783	7784	7785	7786	7787	7788	7789	110
7801	7802	7803	7804	7805	7806	7807	7808	7809	7810	7811	132
7823	7824	7825	7826	7827	7828	7829	7830	7831	7832	7833	154
7845	7846	7847	7848	7849	7850	7851	7852	7853	7854	7855	176
7867	7868	7869	7870	7871	7872	7873	7874	7875	7876	7877	198
7889	7890	7891	7892	7893	7894	7895	7896	7897	7898	7899	220
7911	7912	7913	7914	7915	7916	7917	7918	7919	7920	7921	242
7933	7934	7935	7936	7937	7938	7939	7940	7941	7942	7943	264
7955	7956	7957	7958	7959	7960	7961	7962	7963	7964	7965	286
7977	7978	7979	7980	7981	7982	7983	7984	7985	7986	7987	308
7999	8000	8001	8002	8003	8004	8005	8006	8007	8008	8009	330
8021	8022	8023	8024	8025	8026	8027	8028	8029	8030	8031	352
8043	8044	8045	8046	8047	8048	8049	8050	8051	8052	8053	374
8065	8066	8067	8068	8069	8070	8071	8072	8073	8074	8075	396
8087	8088	8089	8090	8091	8092	8093	8094	8095	8096	8097	418
8109	8110	8111	8112	8113	8114	8115	8116	8117	8118	8119	440
8131	8132	8133	8134	8135	8136	8137	8138	8139	8140	8141	462
8153	8154	8155	8156	8157	8158	8159	8160	8161	8162	8163	484
8175	8176	8177	8178	8179	8180	8181	8182	8183	8184	8185	506
11	12	13	14	15	16	17	18	19	20	21	

HEX

00	@	A 20		C 40	—	E 60	
01	À à	21	!	41	▲ À	61	■
02	Â â	22	"	42	▮ Â	62	■
03	Ã ã	23	#	43	— Ã	63	■
04	Ä ä	24	\$	44	— Ä	64	■
05	Å å	25	%	45	— Å	65	■
06	Æ æ	26	&	46	— Æ	66	■
07	Ç ç	27	'	47	▮ Ç	67	■
08	È è	28	(48	▮ È	68	■
09	É é	29)	49	▮ É	69	■
0A	Ê ê	2A	*	4A	▮ Ê	6A	■
0B	Ë ë	2B	+	4B	▮ Ë	6B	■
0C	Ï ï	2C	,	4C	▮ Ï	6C	■
0D	Ì ì	2D	-	4D	▮ Ì	6D	■
0E	Ó ó	2E	.	4E	▮ Ó	6E	■
0F	Ô ô	2F	/	4F	▮ Ô	6F	■
10	Õ õ	B 30	0	D 50	▮ Õ	F 70	■
11	Ö ö	31	1	51	▮ Ö	71	■
12	Ï ï	32	2	52	▮ Ï	72	■
13	Š š	33	3	53	▮ Š	73	■
14	Ť ť	34	4	54	▮ Ť	74	■
15	Ú ú	35	5	55	▮ Ú	75	■
16	Û û	36	6	56	▮ Û	76	■
17	Ŵ ŵ	37	7	57	▮ Ŵ	77	■
18	Ŷ ŷ	38	8	58	▮ Ŷ	78	■
19	Ÿ Ź	39	9	59	▮ Ÿ	79	■
1A	Ž ž	3A	:	5A	▮ Ž	7A	■
1B	[3B	;	5B	▮	7B	■
1C	£	3C	<	5C	▮	7C	■
1D]	3D	=	5D	▮	7D	■
1E	↑	3E	>	5E	▮	7E	■
1F	←	3F	?	5F	▮	7F	■

HEX

80		A0	rvs space	C0		E0		
81		A1		C1		61	E1	
82		A2		C2		62	E2	
83		A3		C3		63	E3	
84		A4		C4		64	E4	
85		A5		C5		65	E5	
86		A6		C6		66	E6	
87		A7		C7		67	E7	
88		A8		C8		68	E8	
89		A9		C9		69	E9	
8A		AA		CA		6A	EA	
8B		AB		CB		6B	EB	
8C		AC		CC		6C	EC	
8D		AD		CD		6D	ED	
8E		AE		CE		6E	EE	
8F		AF		CF		6F	EF	
90		B0		D0		70	F0	
91		B1		D1		71	F1	
92		B2		D2		72	F2	
93		B3		D3		73	F3	
94		B4		D4		74	F4	
95		B5		D5		75	F5	
96		B6		D6		76	F6	
97		B7		D7		77	F7	
98		B8		D8		78	F8	
99		B9		D9		79	F9	
9A		BA		DA		7A	FA	
9B		BB		DB		7B	FB	
9C		BC		DC		7C	FC	
9D		BD		DD		7D	FD	
9E		BE		DE		7E	FE	
9F		BF		DF		7F	FF	

screen	color	8K+ screen	8K+ color
1Exx	96xx	10xx	94xx
1Fxx	97xx	11xx	95xx

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15
16	17	18	19	1A	1B	1C	1D	1E	1F	20	21	22	23	24	25	26	27	28	29	2A	2B
2C	2D	2E	2F	30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F	40	41
42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53	54	55	56	57
58	59	5A	5B	5C	5D	5E	5F	60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D
6E	6F	70	71	72	73	74	75	76	77	78	79	7A	7B	7C	7D	7E	7F	80	81	82	83
84	85	86	87	88	89	8A	8B	8C	8D	8E	8F	90	91	92	93	94	95	96	97	98	99
9A	9B	9C	9D	9E	9F	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF
B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF	C0	C1	C2	C3	C4	C5
C6	C7	C8	C9	CA	CB	CC	CD	CE	CF	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB
DC	DD	DE	DF	E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF	F0	F1
F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	FD	FE	FF	00	01	02	03	04	05	06	07
08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
1E	1F	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F	30	31	32	33
34	35	36	37	38	39	3A	3B	3C	3D	3E	3F	40	41	42	43	44	45	46	47	48	49
4A	4B	4C	4D	4E	4F	50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F	70	71	72	73	74	75
76	77	78	79	7A	7B	7C	7D	7E	7F	80	81	82	83	84	85	86	87	88	89	8A	8B
8C	8D	8E	8F	90	91	92	93	94	95	96	97	98	99	9A	9B	9C	9D	9E	9F	A0	A1
A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF	B0	B1	B2	B3	B4	B5	B6	B7
B8	B9	BA	BB	BC	BD	BE	BF	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD
CE	CF	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF	EA	E1	E2	E3
E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9

screen 7680 \$1E00	+30720	color 38400 \$9600
screen 7680 \$1E00	-3584	8K+ screen 4096 \$1000
color 38400 \$9600	-30720	screen 7680 \$1E00
color 38400 \$9600	-512	8K+ color 37888 \$9400
8K+ screen 4096 \$1000	+30208	8K+ color 37888 \$9400
8K+ screen 4096 \$1000	+3584	screen 7680 \$1E00
8K+ color 37888 \$9400	-30208	8K+ screen 4096 \$1000
8K+ color 37888 \$9400	+512	color 38400 \$9600

screen	color	8K+ screen	8K+ color
7680	38400	4096	37888
1E00	9600	1100	9400

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65
66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109
110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131
132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153
154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197
198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219
220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241
242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263
264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285
286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307
308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329
330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351
352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373
374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395
396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417
418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439
440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461
462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483
484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505

-115	-114	-113	-112	-111	-110	-109	-108	-107	-106	-105
-93	-92	-91	-90	-89	-88	-87	-86	-85	-84	-85
-71	-70	-69	-68	-67	-66	-65	-64	-63	-62	-61
-49	-48	-47	-46	-45	-44	-43	-42	-41	-40	-39
-27	-26	-25	-24	-23	-22	-21	-20	-19	-18	-17
-5	-4	-3	-2	-1	0	+1	+2	+3	+4	+5
+17	+18	+19	+20	+21	+22	+23	+24	+25	+26	+27
+39	+40	+41	+42	+43	+44	+45	+46	+47	+48	+49
+61	+62	+63	+64	+65	+66	+67	+68	+69	+70	+71
+83	+84	+85	+86	+87	+88	+89	+90	+91	+92	+93
+115	+114	+113	+112	+111	+110	+109	+108	+107	+106	105

SET SCREEN LOCATION					\$0288	\$9002	\$9005
POKE 36869, (PEEK(36869) AND 15) OR X : POKE 648, Z POKE 36866, (PEEK(36866) AND 127) OR Y : SYS 58648							
DEC	HEX	X	Y	Z			
1024	0400	129	0	4			
3072	0C00	131	0	12			
3584	0E00	131	128	14			
4096	1000	132	0	16			
4608	1200	132	128	18			
5120	1400	133	0	20			
5632	1600	133	128	22			
6144	1800	134	0	24			
6656	1A00	134	128	26			
7168	1C00	135	0	28			
7680	1E00	135	128	30			
8192	2000	136	0	32			
15872	3E00	143	128	62			

RETURN SCREEN LOCATION	\$9002	\$9005
4* (PEEK(36866) AND 128) + 64*(PEEK(36869) AND 112)		

RETURN COLOR MAP LOCATION	\$9002	\$9400
37888 + 4 * (PEEK(36866) AND 128)		

SET TOP LIMIT OF MEMORY POINTER			\$0034	\$0038
POKE 52, X : POKE 56, X				
DEC	HEX	X		
5120	1400	20		
6144	1800	24		
7168	1C00	28		
7680	1E00	30		

SET BASIC PROGRAM LOCATION	\$0282 \$0284 \$E378
POKE 642, X : POKE 644, Y : SYS 58232	

UNEXPANDED		3K EXP	8K+ EXP
1024 \$0400			3K available 1024 - 4095 \$0400 - 0FFF
4096 \$1000	USER AREA 4096 - 7679 \$1000 - 1DFF	USER AREA 1024 - 7679 \$0400 - 1DFF POKE 642, 4 POKE 644, 30	SCREEN MAP 4096 - 4609 \$1000 - 11FF
	POKE 642, 16 POKE 644, 30		USER AREA 4608 - \$1200 -
7680 \$1E00	SCREEN MAP 7680 - 8185 \$1E00 - 1FFF	SCREEN MAP 7680 - 8185 \$1E00 - 1FFF	POKE 642, 18
8192 \$2000			8K BLOCK 1 - 16383 - 3FFF POKE 644, 64
			8K BLOCK 2 - 24575 - 5FFF POKE 644, 96
			8K BLOCK 3 - 32767 - 7FFF POKE 644, 128
			32768 \$7FFF
37888 \$9400			COLOR MAP 37888 - 38399 \$9400 - 95FF
38400 \$9600	COLOR MAP 38400 - 38911 \$9600 - 97FF	COLOR MAP 38400 - 38911 \$9600 - 97FF	38400 \$9600

INTERLACE MODE (NTSC only)	\$9000
POKE 36864, PEEK(36864) OR 128	[ON]
POKE 36864, PEEK(36864) AND 127	[OFF]

HORIZONTAL SCREEN ORIGIN	\$9000
POKE 36864, PEEK(36864) AND 128 OR X	

VERTICAL SCREEN ORIGIN	\$9001
POKE 36865, X	

NUMBER OF COLUMNS	\$9003
POKE 36867, PEEK(36867) AND 128 OR X	

NUMBER OF ROWS	\$9003
POKE 36867, PEEK(36867) AND 129 OR (X*2)	

8 x 16 CHARACTER MODE	\$9003
POKE 36867, PEEK(36867) OR 1	[ON]
POKE 36867, PEEK(36867) AND 254	[OFF]

CHARACTER BASE LOCATION	\$9005
POKE 36869, <see chart in memory map>	

AUXILIARY COLOR	
POKE 36878, PEEK (36878) OR X*16	

SCREEN REVERSE MODE	\$900F
POKE 36879, PEEK (36879) AND 247	[ON]
POKE 36879, PEEK (36879) OR 8	[OFF]

BORDER

SCREEN	0 BLK	1 WHT	2 RED	3 CYN	4 PUR	5 GRN	6 BLU	7 YEL
0 BLK	8	9	10	11	12	13	14	15
1 WHT	24	25	26	27	28	29	30	31
2 RED	40	41	42	43	44	45	46	47
3 CYN	56	57	58	59	60	61	62	63
4 PUR	72	73	74	75	76	77	78	79
5 GRN	88	89	90	91	92	93	94	95
6 BLU	104	105	106	107	108	109	110	111
7 YEL	120	121	122	123	124	125	126	127
8 ORG	136	137	138	139	140	141	142	143
9 ORG2	152	153	154	155	156	157	158	159
10 PNK	168	169	170	171	172	173	174	175
11 CYN2	184	185	186	187	188	189	190	191
12 PUR2	200	201	202	203	204	205	206	207
13 GRN2	216	217	218	219	220	221	222	223
14 BLU2	232	233	234	235	236	237	238	239
15 YEL2	248	249	250	251	252	253	254	255
	BLK	WHT	RED	CYN	PUR	GRN	BLU	YEL

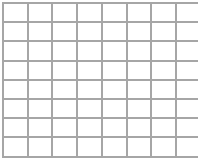
SCREEN AND BORDER COLOR	\$900F
POKE 36879, X	

BORDER COLOR	\$900F
POKE 36879, PEEK (36879) AND 248 OR X	

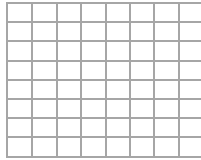
SCREEN COLOR	\$900F
POKE 36879, PEEK (36879) AND 240 OR X	

BIT	7	6	5	4	3	2	1	0
VALUE	128	64	32	16	8	4	2	1

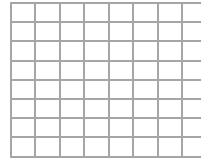
7 6 5 4 3 2 1 0



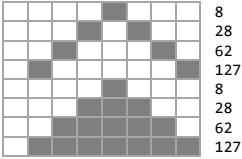
7 6 5 4 3 2 1 0



7 6 5 4 3 2 1 0

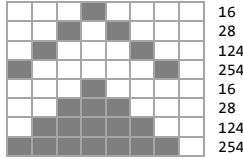


7 6 5 4 3 2 1 0



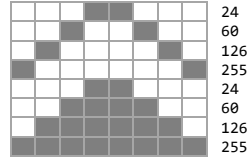
8
28
62
127
8
28
62
127

7 6 5 4 3 2 1 0



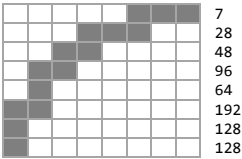
16
28
124
254
16
28
124
254

7 6 5 4 3 2 1 0



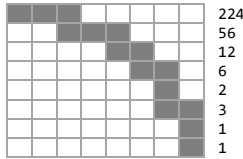
24
60
126
255
24
60
126
255

7 6 5 4 3 2 1 0



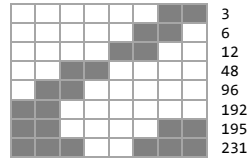
7
28
48
96
64
192
128
128

7 6 5 4 3 2 1 0



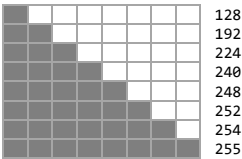
224
56
12
6
2
3
1
1

7 6 5 4 3 2 1 0



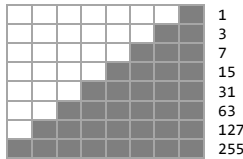
3
6
12
48
96
192
195
231

7 6 5 4 3 2 1 0



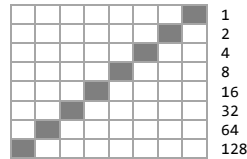
128
192
224
240
248
252
254
255

7 6 5 4 3 2 1 0



1
3
7
15
31
63
127
255

7 6 5 4 3 2 1 0



1
2
4
8
16
32
64
128

CHARACTER BASE LOCATION

\$9005

POKE 36869, X	240 upper case	251 start at 3072	12
	241 lower reverse	252 start at 4096	16
	242 lower case	253 start at 5120	20
	243 lower reverse	254 start at 6144	24
	244-250 ---	255 start at 7168	28

PROTECT USER DEFINED GRAPHICS MEMORY

\$0034 \$0038

POKE 52, Y : POKE 56, Y [see page 36]

7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
0	64	128	192
1	65	129	193
2	66	130	194
3	67	131	195
4	68	132	196
5	69	133	197
6	70	134	198
7	71	135	199
8	72	136	200
9	73	137	201
10	74	138	202
11	75	139	203
12	76	140	204
13	77	141	205
14	78	142	206
15	79	143	207
16	80	144	208
17	81	145	209
18	82	146	210
19	83	147	211
20	84	148	212
21	85	149	213
22	86	150	214
23	87	151	215
24	88	152	216
25	89	153	217
26	90	154	218
27	91	155	219
28	92	156	220
29	93	157	221
30	94	158	222
31	95	159	223
32	96	160	224
33	97	161	225
34	98	162	226
35	99	163	227
36	100	164	228
37	101	165	229
38	102	166	230
39	103	167	231
40	104	168	232
41	105	169	233
42	106	170	234
43	107	171	235
44	108	172	236
45	109	173	237
46	110	174	238
47	111	175	239
48	112	176	240
49	113	177	241
50	114	178	242
51	115	179	243
52	116	180	244
53	117	181	245
54	118	182	246
55	119	183	247
56	120	184	248
57	121	185	249
58	122	186	250
59	123	187	251
60	124	188	252
61	125	189	253
62	126	190	254
63	127	191	255
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0

BIT	7	6	5	4	3	2	1	0
VALUE	128	64	32	16	8	4	2	1

SCREEN	<input type="checkbox"/>	<input type="checkbox"/>
BORDER	<input type="checkbox"/>	<input checked="" type="checkbox"/>
CHARACTER	<input checked="" type="checkbox"/>	<input type="checkbox"/>
AUX	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

0	0	0	0
64	16	4	1
128	32	8	2
192	48	12	3

7 6 5 4 3 2 1 0

7 6 5 4 3 2 1 0

7 6 5 4 3 2 1 0

				80
				96
				112
				144
				160
				176
				208
				224
				240

				5
				10
				15
				20
				40
				60
				80
				160
				240














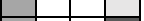
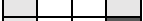
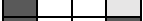




















































































































































































































































				21
				42
				63
				84
				168
				252
				85
				170
				255

	COLOR CODE	MULTICOLOR MODE	AUX
BLK	0	8	0
WHT	1	9	16
RED	2	10	32
CYN	3	11	44
PUR	4	12	64
GRN	5	13	80
BLU	6	14	96
YEL	7	15	112

	AUX
ORG	128
Lt. ORG	144
PNK	160
Lt. CYN	176
Lt. PUR	192
Lt. GRN	208
Lt. BLU	224
Lt. YEL	240

SET CURSOR COLOR	\$0286
POKE 646, X	

SET AUXILIARY COLOR	\$900E
POKE 36878, <aux color> + <volume 0-15>	

SCREEN 	BORDER 	CHAR 	AUX 
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			
			

SCREEN	BORDER	CHAR	AUX
\$	\$	\$	\$
00	40	80	C0
01	41	81	C1
02	42	82	C2
03	43	83	C3
04	44	84	C4
05	45	85	C5
06	46	86	C6
07	47	87	C7
08	48	88	C8
09	49	89	C9
0A	4A	8A	CA
0B	4B	8B	CB
0C	4C	8C	CC
0D	4D	8D	CD
0E	4E	8E	CE
0F	4F	8F	CF
10	50	90	D0
11	51	91	D1
12	52	92	D2
13	53	93	D3
14	54	94	D4
15	55	95	D5
16	56	96	D6
17	57	97	D7
18	58	98	D8
19	59	99	D9
1A	5A	9A	DA
1B	5B	9B	DB
1C	5C	9C	DC
1D	5D	9D	DD
1E	5E	9E	DE
1F	5F	9F	DF
20	60	A0	E0
21	61	A1	E1
22	62	A2	E2
23	63	A3	E3
24	64	A4	E4
25	65	A5	E5
26	66	A6	E6
27	67	A7	E7
28	68	A8	E8
29	69	A9	E9
2A	6A	AA	EA
2B	6B	AB	EB
2C	6C	AC	EC
2D	6D	AD	ED
2E	6E	AE	EE
2F	6F	AF	EF
30	70	B0	F0
31	71	B1	F1
32	72	B2	F2
33	73	B3	F3
34	74	B4	F4
35	75	B5	F5
36	76	B6	F6
37	77	B7	F7
38	78	B8	F8
39	79	B9	F9
3A	7A	BA	FA
3B	7B	BB	FB
3C	7C	BC	FC
3D	7D	BD	FD
3E	7E	BE	FE
3F	7F	BF	FF

DEC	SCRN	255	254	240
0	@	7168	6144	32768
1	A	7176	6152	32776
2	B	7184	6160	32784
3	C	7192	6168	32792
4	D	7200	6176	32800
5	E	7208	6184	32808
6	F	7216	6192	32816
7	G	7224	6200	32824
8	H	7232	6208	32832
9	I	7240	6216	32840
10	J	7248	6224	32848
11	K	7256	6232	32856
12	L	7264	6240	32864
13	M	7272	6248	32872
14	N	7280	6256	32880
15	O	7288	6264	32888
16	P	7296	6272	32896
17	Q	7304	6280	32904
18	R	7312	6288	32912
19	S	7320	6296	32920
20	T	7328	6304	32928
21	U	7336	6312	32936
22	V	7344	6320	32944
23	W	7352	6328	32952
24	X	7360	6336	32960
25	Y	7368	6344	32968
26	Z	7376	6352	32976
27	[7384	6360	32984
28	£	7392	6368	32992
29]	7400	6376	33000
30	↑	7408	6384	33008
31	←	7416	6392	33016

DEC	SCRN	255	254	240
32		7424	6400	33024
33	!	7432	6408	33032
34	"	7440	6416	33040
35	#	7448	6424	33048
36	\$	7456	6432	33056
37	%	7464	6440	33064
38	&	7472	6448	33072
39	'	7480	6456	33080
40	(7488	6464	33088
41)	7496	6472	33096
42	*	7504	6480	33104
43	+	7512	6488	33112
44	,	7520	6496	33120
45	-	7528	6504	33128
46	.	7536	6512	33136
47	/	7544	6520	33144
48	0	7552	6528	33152
49	1	7560	6536	33160
50	2	7568	6544	33168
51	3	7576	6552	33176
52	4	7584	6560	33184
53	5	7592	6568	33192
54	6	7600	6576	33200
55	7	7608	6584	33208
56	8	7616	6592	33216
57	9	7624	6600	33224
58	:	7632	6608	33232
59	;	7640	6616	33240
60	<	7648	6624	33248
61	=	7656	6632	33256
62	>	7664	6640	33264
63	?	7672	6648	33272

DEC	SCRN	255	254	240
64		7680	6656	33280
65		7688	6664	33288
66		7696	6672	33296
67		7704	6680	33304
68		7712	6688	33312
69		7720	6696	33320
70		7728	6704	33328
71		7736	6712	33336
72		7744	6720	33344
73		7752	6728	33352
74		7760	6736	33360
75		7768	6744	33368
76		7776	6752	33376
77		7784	6760	33384
78		7792	6768	33392
79		7800	6776	33400
80		7808	6784	33408
81		7816	6792	33416
82		7824	6800	33424
83		7832	6808	33432
84		7840	6816	33440
85		7848	6824	33448
86		7856	6832	33456
87		7864	6840	33464
88		7872	6848	33472
89		7880	6856	33480
90		7888	6864	33488
91		7896	6656	33496
92		7904	6664	33504
93		7912	6672	33512
94		7920	6680	33520
95		7928	6688	33528

DEC	SCRN	255	254	240
96		7936	6696	33536
97		7944	6704	33544
98		7952	6712	33552
99		7960	6720	33560
100		7968	6728	33568
101		7976	6736	33576
102		7984	6744	33584
103		7992	6752	33592
104		8000	6760	33600
105		8008	6768	33608
106		8016	6776	33616
107		8024	6784	33624
108		8032	6792	33632
109		8040	6800	33640
110		8048	6808	33648
111		8056	6816	33656
112		8064	6824	33664
113		8072	6832	33672
114		8080	6840	33680
115		8088	6848	33688
116		8096	6856	33696
117		8104	6864	33704
118		8112	6872	33712
119		8120	6880	33720
120		8128	6888	33728
121		8136	6896	33736
122		8144	6904	33744
123		8152	6912	33752
124		8160	6920	33760
125		8168	6928	33768
126		8176	6936	33776
127		8184	6944	33784

MEMORY MAP

57344-65535	E000-FFFF	BLK 7	KERNAL
49152-57343	C000-DFFF	BLK 6	BASIC
40960-49151	A000-BFFF	BLK 5	RAM EXPANSION
32768-40959	8000-9FFF	BLK 4	ROM VIC VIA RAM I/O
24576-32767	6000-7FFF	BLK 3	RAM EXPANSION
16384-24575	4000-5FFF	BLK 2	RAM EXPANSION
8192-16383	2000-3FFF	BLK 1	RAM EXPANSION
0-8191	0000-1FFF	BLK 0	RAM

How to LOAD multipart files:

1. LOAD the first file: LOAD"<file name>",8,1
2. Enter NEW.
3. LOAD the next file: LOAD"<file name>",8,1
4. Perform a soft reset: SYS64802

VIC 1110 MEMORY EXPANSION SETTINGS							
	HEX	DEC	DIP SWITCH				
BLK 1	\$2000-\$3FFF	8192-16383	OFF	OFF	OFF	ON	
BLK 2	\$4000-\$5FFF	16384-24575	OFF	OFF	ON	OFF	
BLK 3	\$6000-\$7FFF	24576-32767	OFF	ON	OFF	OFF	
BLK 5	\$A000-\$BFFF	40960-49151	ON	OFF	OFF	OFF	

How to Set the DIP Switches:

1. Remove the screw on the back of the RAM pack using a Phillips head screwdriver.
2. Using a flat headed screwdriver open the plastic case by inserting the blade of the screwdriver into each of the slots and levering upward. Remove the case.
3. The DIP switch bank is located at the center of the PC board Set the switches to the desired setting.
4. Snap the cover closed and replace the screw.

REMEMBER:

Never have more than one switch on at the same time!
 Never allow two cartridges to be in the same memory area.

HEX	DEC		DESCRIPTION
	0-143		BASIC Working Storage
0000	0	USRPOK	Jump for USR function default value 76
0001	1-	ADDPRC	Vector for USR function LSB (low byte) default value 72
0002	2		Vector for USR function MSB (high byte) default value 210
0003	3-	ADRAY1	Float-Fixed vector, Convert float to integer (LSB) default value 170
0004	4		Float-Fixed vector, Convert float to integer (MSB) default value 209
0005	5-	ADRAY2	Fixed-Float vector, Convert integer to float (LSB) default value 145
0006	6		Fixed-Float vector, Convert integer to float (MSB) default value 211
0007	7	CHARAC	General counter for Basic. Search colon or end line 111 temporary value for string pointer
0008	8	ENDCHR	Scan-quotes flag 0 delimiter
0009	9	TRMPOS	TAB save; Cursor column before last TAB or SPC 0 to 87 range
000A	10	VERCHK	Verify flag 0 LOAD 1 VERIFY
000B	11	COUNT	Input buffer pointer/# subscript, array dimensions AND sets location to 0 OR sets location to 255 DIM: number of dimensions in array
000C	12	DIMFLG	Default DIM flag, First character of array name
000D	13	VALTYP	Variable type, (FRMEVL routines set) 0 numeric 255 (\$FF) string
000E	14	INTFLG	Integer flag (type) when location 13 is 0. 0 floating point 128 (\$80) integer
000F	15	GARBFL	DATA scan / LIST quote/memory flag LIST and DATA: prevents detokenization
0010	16	SUBFLG	Subscript flag / FNx flag 128 when FN is defined
0011	17	INPFLG	Input flags; line number for input error message 0 INPUT 64 (\$40) GET 152 (\$98) READ ERROR message pointer
0012	18	TANSGN	TAN, SIN, and Comparison eval flag

HEX	DEC		DESCRIPTION
			bit 0: > (greater than) bit 1: = (equal) bit 2: < (less than)
0013	19	CHANNL	Current I/O device prompt flag 0 keyboard (default) 1 tape 2 RS-232 / user port 3 screen 4-5 printer 6-7 plotter 8-11 disk 4-31 any serial port
0014	20-	LINNUM	Line number integer in (LSB)
0015	21		Line number integer in (MSB) BASIC integer address (for SYS, GOTO, LIST, etc.)
0016	22	TEMPPT	Temp string descriptor stack pointer (default 25) 25 empty (default), zero descriptors used 34 results in ?FORMULA TOO COMPLEX 35 disable line numbers in LIST
0017	23-24	LASTPT	Last temporary string vector (default 22,0) 25, 0 one string descriptor being used 28, 0 two string descriptors being used 31, 0 all three string descriptors being used
0019	25-33	TEMPST	Stack of descriptors for three temporary strings Length of string, LSB address, MSB address
0022	34-35	INDEX1	Utility pointer area: Pointer for number transfer
0024	36-37	INDEX2	Utility pointer area: Pointer for number transfer
0026	38-42	RESHO	Product area for BASIC multiplication
002B	43-	TXTTAB	Pointer: Start of Basic program (LSB)
002C	44		Pointer: Start of Basic program (MSB) default values 1, 16 (4097) PEEK(43)+256*PEEK(44) detects start of BASIC
002D	45-	VARTAB	Pointer: Start of Variables (end of prog) (LSB)
002E	46		Pointer: Start of Variables (MSB) PEEK(45)+256*PEEK(46) detects end of BASIC
002F	47-	ARYTAB	Pointer: Start of Arrays (end of vars) (LSB)
0030	48		Pointer: Start of Arrays (end of vars) (MSB) One byte past the end of the BASIC program
0031	49-	STREND	Pointer: End of Arrays (LSB)
0032	50		Pointer: End of Arrays (MSB)
0033	51-	FRETOP	Pointer: String storage (down) (LSB)
0034	52		Pointer: String storage (down) (MSB)
0035	53-	FRESPEC	Utility pointer to active string (LSB)
0036	54		Utility pointer to active string (MSB)
0037	55-	MEMSIZ	Pointer: End of program memory (LSB)
0038	56		Pointer: End of program memory (MSB) default values 0, 30 (7680)
0039	57	CURLIN	Current Basic line number (LSB)
0040	58		Current Basic line number (MSB)

HEX	DEC		DESCRIPTION
			255 direct mode statement
003B	59-60	OLDLIN	Previous Basic line number
003D	61-62	OLDTXT	Pointer: Basic statement for CONT
003F	63-64	DATLIN	Current DATA line number
0041	65-66	DATPTR	Current DATA address
0043	67-68	INPPTR	Input vector; pointer to source of INPUT GET READ
0045	69-70	VARNAM	Current variable name (2 characters with type flags) If one character, 70=zero Floating point: no high bits on (bit 7 OFF OFF) Integer: both have high bits on (bit 7 ON ON) String: (bit 7 OFF ON) Function (bit 7 ON OFF)
0047	71-72	VARPNT	Current variable address
0049	73-74	FORPNT	Variable pointer for FOR / NEXT LIST uses 73 as temp save area WAIT uses 73 as second parameter, 74 third CLOSE uses 73 to save file number LOAD and SAVE uses 73 to save device number
004A	74		WAIT uses 74 as third parameter RETURN uses 74 as a flag set to 255 for GOSUB
004B	75-76	OPPTR	Y save register-new operator save
004D	77	OPMASK	Current operator, comparison symbol accumulator bit 0: > (greater than) bit 1: = (equal) bit 2: < (less than)
004E	78-79	DEFPNT	Misc. work area; FN definition pointer high-low
0050	80-81	DSCPTN	Work area; pointer to current string description
0052	82	SIZE	Length of above string (location 80)
0053	83	FOUR6	Constant, 3 or 7 for garbage collection
0054	84-86	JMPER	Jump vector for functions 84 76 jump opcode 85 Function address LSB / garbage collection 86 Function address MSB / exp, addition work area
0057	87-96	TEMPF3	Misc. numeric work area 95 Pointer to variable 7 byte descriptor 95 Decimal point flag when converting a string 95-96 LIST active pointer
0061	97-102	FAC	BASIC floating point accumulator one
0061	97		Accum#1: Exponent
	98-99		Integer stored during floating conversion
	98-101		Accum#1: Normalized mantissa
	100-101		String descriptor during string processing routines
0066	102		Accum#1: Sign 0 positive 128-255 negative

HEX	DEC		DESCRIPTION
0067	103	SGNFLG	Series evaluation constant pointer
0068	104	BITS	Accum#1 hi-order (overflow)
0069	105-110	FAC2	BASIC floating point Accum#2: Exponent, etc.
	105	ARGEXP	Exponent +128
006A	106-110		Accum#2: Mantissa
	106-109	ARGHO	Normalized mantissa of the value
	110	ARGSGN	0 positive, 128-255 negative
006F	111	ARISGN	Sign comparison, Acc#1 vs #2 0 FAC and FAC2 have same sign 225 FAC and FAC2 have different signs
0070	112	FACOV	Accum#1 lo-order (rounding)
0071	113-114	FBUFPT	Cassette buffer length/Series pointer
0073	115-138	CHRGET	Subroutine (get BASIC char)
007A	122-123		Basic pointer (within subroutine)
008B	139-143	RNDX	RND seed value, work area
0090	144-255		KERNAL working storage
0090	144	STATUS	Status word ST, status of I/O completion
0091	145	STKEY	Key switch PIA: STOP and RVS flags (bottom row) 127 (\$7F) cursor down/up key 223 (\$DF) comma key pressed 239 (\$EF) N key pressed 247 (\$F7) V key pressed 251 (\$FB) X key pressed 253 (\$FD) left shift key pressed 254 (\$FE) STOP key pressed 255 (\$FF) no key pressed
0092	146	SVXT	Timing constant for tape resets to 0 after each bit has been read
0093	147	VERCK	Verify check flag 0 LOAD 1 VERIFY
0094	148	C3PO	Serial output: deferred char flag
0095	149	BSOUR	Serial deferred character 255 (\$FF) no character is waiting for serial output
0096	150	SYNO	Tape EOT received 0 no block recognized OR data being read 16-126 has read at least 16 leader bits
0097	151	XSAV	Register save
0098	152	LDTND	Number of open files (not to exceed 10)
0099	153	DFLTN	Input device (normally 0) see value chart in location 19
009A	154	DFLTO	Output (CMD) device, (normally 3: screen) see value chart in location 19
009B	155	PRTY	Tape character parity bit 0 used to calculate parity

HEX	DEC		DESCRIPTION
009C	156	DPSW	Tape dipole switch; Byte-received flag 0 waiting or in process 1 received
009D	157	MSGFLG	KERNAL message control flag 0 output control 64 (\$40) error messages wanted 128 (\$80) control messages wanted 192 (\$C0) error and control messages wanted
009E	158	PTRI	Tape 1 error log/char buffer, header ID out byte Tape SAVE: 1 relocatable Tape SAVE: 2 user data record Tape SAVE: 3 nonrelocatable Tape SAVE: 4 user data header Tape SAVE: 5 end of tape Tape LOAD: pass 1 error equals 2 times # of errors Tape header LOAD: index for filename compare Tape write from BASIC: character for CHROUT
009F	159	PTR2	Tape Pass 2 error log corrected
00A0	160-162	TIME	Jiffy Clock (HML) 160 Incremented every 18.2044 minutes 161 Incremented every 4.26667 seconds 162 Incremented every 0.01667 second (1 jiffy)
00A3	163	PCNTR	Serial bit count/Tape input/EOI flag SERIAL: flag indicates last byte sent to device TAPE: Set to 8 and decremented each read / write
00A4	164	FIRT	Cycle count, tape dipole number, serial input byte SERIAL: the input byte read in LOAD or SAVE TAPE: 0 second half of dipole processed TAPE: 1 first half of dipole processed
00A5	165	CNTDN	Serial countdown, tape write/bit count TAPE: initializes to 9 each block, counts down SERIAL: initializes to 8, bits to be sent, counts down
00A6	166	BUFPNT	Pointer: tape buffer
00A7	167	INBIT	Tape Write leader / Read pass / input bit Tape write leader length counter Set to 0 before writing tape leader for header Set to 128 before writing leader between blocks Tape LOAD: 0 all blocks read Tape LOAD: 1 second block loading Tape LOAD: 2 first block loading RS-232: bit 0 is temp storage for input bit
00A8	168	BITCI	Tape Write new byte/Read error / in bit TAPE: 0 no errors RS-232: input byte bit count
00A9	169	RINONE	Tape dipole balance counter, RS input flag start bit

HEX	DEC		DESCRIPTION
			TAPE: 126 maximum value RS-232: 144 no start bit received
00AA	170	RIDATA	Tape input status flag, sync count, RS byte Tape LOAD: 0 waiting for first character to arrive Tape LOAD: 1-15 block countdown being read Tape LOAD: 64 valid character has arrived Tape LOAD: 128 first block loaded, search for next Tape SAVE: 128 start address is greater than end RS-232:byte assembly area
00AB	171	RIPRTY	Tape write lead length / Rd checksum / parity Set to 0 for leader between blocks Set to 20 before writing leader first block Set to 105 before writing the header leader
00AC	172-173	SAL	Pointer: tape buffer, scrolling
00AE	174-175	EAL	Tape end addresses/End of program
00B0	176	CMPO	Tape timing constant, dipole timing adjustment
00B1	177	TEMPI	Tape timing constant 2, dipole timer 2 difference
00B2	178-179	TAPEI	Pointer: start of tape buffer (LSB/MSB)
00B4	180	BITTS	Tape timer (1=enable); bit count TAPE: 0 rest between blocks TAPE: >0 ready to receive data RS-232: transmit bit count out, timer flag, parity
00B5	181	NXTBIT	Tape EOT/RS-232 next bit to send (>0 read)
00B6	182	RODATA	Read character error / out byte buffer TAPE: >0 byte read is in error
00B7	183	FNLEN	Length of current file name string
00B8	184	LA	Current logical file number
00B9	185	SA	Current secondary address, or R / W command 0-31 valid serial devices 0-127 non-serial devices
00BA	186	FA	Current device number see value chart in location 19
00BB	187-188	FNADR	Address of current file name string
00BD	189	ROPRTY	Write shift word / Read input char
00BE	190	FSBLK	Number of blocks remaining to Write / Read 0 both copies of block are done 1 last copy of block remains to save/load 2 both copies of block remain to save/load
00BF	191	MYCH	Serial word buffer
00C0	192	CASI	Tape motor interlock, manual / contr switch 0,2,4,6,10 stops the motor 8 no change 12,14 starts the motor
00C1	193-194	STAL	I/O start addresses
00C3	195-196	MEMUSS	KERNAL setup pointer

HEX	DEC		DESCRIPTION									
00C5	197	LSTX	Current key pressed (default 64)									
			0	1	12		26	X	39	F1	52	O
			1	3	13	P	27	V	41	2	53	@
			2	5	14	*	28	N	42	F	54	↑
			3	7	15	ret	29	,	43	H	55	F5
			4	9	17	A	30	/	44	K	56	2
			5	+	18	D	31	↓	45	:	57	4
			6	£	19	G	32	spc	46	=	58	6
			7	del	20	J	33	Z	47	F3	59	8
			8	←	21	L	34	C	48	Q	60	0
			9	W	22	;	35	B	49	E	61	-
			10	R	23	↔	36	M	50	T	62	ho
			11	Y	24	stp	37	.	51	U	63	F7
			64 no key pressed (default)									
00C6	198	NDX	Number of chars in keyboard buffer 0 clear, no keys (default) 10 maximum value									
00C7	199	RVS	Screen reverse flag 0 disable reverse mode 18 enable reverse mode									
00C8	200	INDX	Pointer: End-of-line for input									
00C9	201	LXSP	Input cursor log (row) range 0 to 22									
00CA	202	LXSP	Input cursor log (column) range 0 to 87									
00CB	203	SFDX	Shift mode on print flag; current key press see location 197									
00CC	204	BLNSW	Cursor blink switch 0 flash >0 disable, cursor will not flash									
00CD	205	BLNCT	Cursor timing countdown (normal 20 jiffies)									
00CE	206	GDBLN	Character under cursor in PETSCII									
00CF	207	BLNON	Cursor in blink phase 0 not reversed 1 reversed character									
00D0	208	CRSW	Input from screen/from keyboard 0 input from keyboard possible values: 21, 43, 65, 87									
00D1	209-210	PNT	Pointer to screen line address									
00D3	211	PNTR	Position of cursor on current line 0 to 87 possible range									
00D4	212	QTSW	Quote mode flag 0 not within quotes (off) 1 within quotes (on)									
00D5	213	LNMX	Current screen line length									

HEX	DEC		DESCRIPTION
			possible lengths: 21, 43, 65, 87
00D6	214	TBLX	Current screen row number
00D7	215	ASCII	ASCII value of last key press / checksum/buffer
00D8	216	INSRT	Insert mode flag (outstanding inserts remaining) 0 turn off insert more
00D9	217-241	LDTB1	Screen line link table
00F1	241		Save byte for screen link table
00F2	242		Screen row marker
00F3	243-244	USER	Pointer to current screen line color map area
00F5	245-246	KEYTAB	Pointer to current keyboard table
00F7	247-248	RIBUF	RS-232 pointer to start of receiving buffer
00F9	249-250	ROBUF	RS-232 Tx pointer to start of transmitting buffer
00FB	251-254	FREEKZP	Operating system free zero page space
00FF	255	BASZPT	Basic storage for floating point to ASCII conversion
0100	256-511	STACK	Stack area
0100	256-266		Floating to ASCII work area (62 bytes)
0100	256-318	BAD	Tape error log
0140	320-511	BASTACK	BASIC Proc stack area ?OUT OF MEMORY error if exceeded
0200	512-600	BUF	Basic input buffer (89 bytes) Allows 88 characters in input line, 0 at end of line
0259	601-610	LAT	Logical file table
0263	611-620	FAT	Open device number table
026D	621-630	SAT	Open secondary Address table
0277	631-640	KEYD	Keyboard buffer Stores 10 characters, see location197
0281	641-	MEMSTR	Start of memory pointer (LSB)
0282	642		Start of memory pointer (MSB) 16: unexpanded default value (4096 \$1000) 4: +3K expanded default value (1024 \$0400) 18: +8K+ expanded default value (4608 \$1200) POKE642,PEEK(642)+{pages up}: POKE644,PEEK(644)-{pages down}:SYS58232
0283	643-644	MEMHIGH	Top of memory, pointer to end of user RAM +1 unexpanded and 3K default values 0, 30 (7680) 64 block 1 end value (8192 \$2000) 96 block 2 end value (16384 \$4000) 128 block 3 end value (24576 \$6000)
0285	645	TIMOUT	Serial bus timeout enable flag
0286	646	COLOR	Current color code (default 6) bit 2-0 standard color 0 BLK 1 WHT 2 RED 3 CYN 4 PUR 5 GRN 6 BLU 7 YEL bit 3 multi-color mode flag
0287	647	GDCOL	Color under cursor

HEX	DEC		DESCRIPTION
0288	648	HIBASE	Screen memory page (current location MSB) unexpanded and 3K default value 30 (7680) 8K+ expanded default value 16 (4096)
0289	649	XMAX	Max size of keyboard buffer (default 10) 0 completely disables keyboard 15 maximum range for normal buffer (631-645)
028A	650	RPTFLG	Key repeat flag (default 0) 60 repeat no keys 128 repeats all keys
028B	651	KOUNT	Repeat delay speed counter Initialized to 6 after first delay, decremented to 0
028C	652	DELAY	Repeat delay first timer counter Initialized to 16, decremented to 0 key hold puts 4 in 651 for faster repeat
028D	653	SHFLAG	Shift / C= / Control key flag (default 0) bit 0 shift key (1 key down) bit 1 Commodore key (2 key down) bit 2 control key (4 key down)
028E	654	LSTSHF	Last keyboard shift pattern
028F	655-656	KEYLOG	Pointer: decode logic (table address stored in 245)
0291	657	MODE	Shift mode switch 0 enabled or CHR\$(9) 128 disabled or CHR\$(8)
0292	658	AUTODN	Auto scroll down flag 0 enable 1-255 disable
0293	659	M51CTR	Pseudo 6551 / RS-232 control register bit 7 stop bits: 0=1 stop bit, 1=2 stop bits bit 6-5 word length: 00=8 bits 01=7 bits 10=6 bits 00=5 bit 4 unused bit 3-0 baud rate: 0001=50 0010=75 0011=110 0100=134.5 0100=150 0110=300 0111=600 1000=1200 1001=1800 1010=2400 1011=3600 1100=4800 rates above 3600 unavailable on VIC
0294	660	M51CDR	Pseudo 6551 / RS-232 command register bit 7-5 parity: xx0=disabled 001=odd 011=even 101=mark 111=space bit 4 duplex: 0=full, 1=half bit 3-1 unused bit 0 handshaking: 0=3 line, 1=X line
0295	661-662	M51AJB	Nonstandard (Bit time / 2-100)
0297	663	RSSTAT	RS-232 status register

HEX	DEC		DESCRIPTION
			bit 7 BREAK detected; interrupt has occurred
			bit 6 data set not ready; modem not free
			bit 5 unused
			bit 4 CTS signal missing
			bit 3 unused
			bit 2 receiver buffer overrun
			bit 1 framing error
			bit 0 parity error
0298	664	BITNUM	Number of bits to send or receive
0299	665-666	BAUDOF	Baud rate (full) bit time divided by baud rate
029B	667	RIDBE	RS-232 receive end pointer
029C	668	RIDBS	RS-232 input start pointer
029D	669	RODBS	RS-232 transmit start pointer
029E	670	RODBE	RS-232 output end pointer
029F	671-672	IRQTMP	Holds IRQ during tape operations
02A1	673-767	USRVCTR	User Program indirect vectors
0300	768-778	BVECTOR	BASIC vectors table
0300	768-769	IERROR	Error message link
0302	770-771	IMAIN	Basic warm start link
0304	772-773	ICRNCH	Crunch Basic tokens link
0306	774-775	IQPLOP	Print tokens, vectors to BASIC routines POKE 774, 200 disables RUN STOP key POKE 775, 171 crashes if LIST attempted
0308	776-777	IGONE	Start new Basic code link
030A	778-779	IEVAL	Get arithmetic element link (evaluate variables)
030C	780-783		Register Save Area
030C	780	SAREG	Storage for 6502 .A register
030D	781	SXREG	Storage for 6502 .X register
030E	782	SYREG	Storage for 6502 .Y register
030F	783	SPREG	Storage for 6502 .P register (see page 64 values)
0310	784-787	PG3FREE	Unused space
0314	788-819	KVECTOR	Table of 16 KERNAL indirect vectors
0314	788-789	CINV	Hardware (IRQ) interrupt vector (EABF) POKE 788,194 disable timer and STOP key POKE 788,181 enable timer and STOP key
0316	790-791	CBINV	Break interrupt vector
0318	792-793	NMINV	NMI interrupt vector
031A	794-795	IOPEN	OPEN vector
031C	796-797	ICLOSE	CLOSE vector
031E	798-799	ICHKIN	Set-input vector CHKIN
0320	800-801	ICHKOUT	Set-output vector CHKOUT
0322	802-803	ICLRCH	Restore I/O vector CLRCHN
0324	804-805	IBASIN	INPUT vector CHRIN
0326	806-807	IBSOUT	Output vector CHROUT
0328	808	ISTOP	Test-STOP key vector 1

HEX	DEC		DESCRIPTION
			100 disables RUN/STOP and RESTORE keys 112 enables RUN/STOP and RESTORE keys
0329	809	ISTOP	Test-STOP key vector 2
032A	810-811	IGETIN	GET vector
032C	812-813	ICLALL	Abort I/O vector, close all files
032E	814-815	USRCMD	User vector (default BRK)
0330	816-817	ILOAD	Link to load RAM
0332	818-819	ISAVE	Link to save RAM
0334	820-827	USRCMDS	Unknown, likely unused (eight user bytes?)
033C	828-1019	TBUFFR	Cassette buffer (192 bytes)
033C	828	TPHDRID	Tape date block identifier 1 relocatable 2 BASIC program data block 3 nonrelocatable 4 BASIC program data header 5 logical end of tape
033D	829-830	TPHBGN	Starting address for tape LOAD
033D	829-1019	TPBLOCK	Block of 191 bytes for user data
033F	831-832	TPHNAME	Ending address of tape LOAD +1
03FC	1020-1023		Unused 4 bytes (possible user storage area)
0400	1024-4095	RAMBLK0	3K expansion RAM area (3072 bytes) 3072- User-defined character set when pointer is 251
1000	4096-7679	USRPGM3	User Basic area; start of screen memory (8K+)
1000	4096-4607	SCREENX	Screen map visible when pointer is 18 (8K+) 4096- User-defined character set when pointer is 252
1200	4608-8191	USRPGM8	First 3583 bytes of BASIC area on 8K+ 5120- User-defined character set when pointer is 253 6144- User-defined character set when pointer is 254 7168- User-defined character set when pointer is 255
1E00	7680-8185	SCREEN	Screen map visible for unexpanded and +3K
1FFA	8186-8191		Screen memory normally outside view by default
2000	8192-	RAMBLK1	8K expansion RAM/ROM block 1
4000	16384-	RAMBLK2	8K expansion RAM/ROM block 2
6000	24576-	RAMBLK3	8K expansion RAM/ROM block 3 32592 SYS for Scott Adams adventure games
8000	32768-36863		4K ROM Character Maps
8000	32768-	CASEU	Character ROM: Upper case and graphics 32768 start of alphabet (A-Z) 33024 space 33152 start of numbers(0-9) 33280 start of symbol graphics
8400	33792-	CASEURV	Character ROM: Reversed upper case and graphics 33792 start of alphabet (A-Z) 34048 space 34176 start of numbers(0-9)

HEX	DEC		DESCRIPTION
			34304 start of symbol graphics
8800	34816-	CASEL	Character ROM: Upper and lower case 34816 start of alphabet (A-Z) 35090 space 35200 start of numbers(0-9) 35328 start of symbol graphics
8C00	35840-	CASELRV	Character ROM: Reversed upper and lower case 35840 start of alphabet (A-Z) 36096 space 36224 start of numbers(0-9) 36352 start of symbol graphics
9000	36864 - 37125		6560 Video Interface Chip
9000	36864	VICCR0	Horizontal centering and interlace scan bit 7 enable interlace scan bit (default 0) bit 6–0 horizontal screen position (default 5)
9001	36865	VICCR1	Vertical picture origin default value 25
9002	36866	VICCR2	Number of columns bit 7 is bit 9 of screen memory location (see 36869) bit 7=1, color map begins at 38400 bit 7=0, color map begins at 37888 bit 6–0 number of screen columns default value 150
9003	36867	VICCR3	Number of rows and 8x16 mode bit 7 raster value (light pen sync with TV, 36868) bit 6-1 number of visible rows on screen x2 (0 to 23) bit 0 character size switch 0 8x8 mode (default) 1, 8x16 mode (default 46 or 176)
9004	36868	VICCR4	TV raster beam line low order bit combined with high order in 36867
9005	36869	VICCR5	bits 7 must be 1, video matrix bits 7-4 are bits 10-12 of screen location (bit 13-10) 1100 (192) begin at 4096 when bit 7 of 36866=0 1100 (192) begin at 4608 when bit 7 of 36866=1 1101 (208) begin at 5120 when bit 7 of 36866=0 1101 (208) begin at 5632 when bit 7 of 36866=1 1110 (224) begin at 6144 when bit 7 of 36866=0 1110 (224) begin at 6656 when bit 7 of 36866=1 1111 (240) begin at 7168 when bit 7 of 36866=0 1111 (240) begin at 7680 when bit 7 of 36866=1 bits 0-3 start of character location (bit 13-10) 0000 (240) begin at 32768 (\$8000) Upper (default) 0001 (241) begin at 33792 (\$8400) Upper rev

HEX	DEC		DESCRIPTION
			0010 (242) begin at 34816 (\$8800) Lower case 0011 (243) begin at 35840 (\$8C00) Lower rev 0100 – 1011 outside of normal use 1100 (252) begin at 4096 (\$1000) RAM 1101 (253) begin at 5120 (\$1400) RAM 1110 (254) begin at 6144 (\$1800) RAM 1111 (255) begin at 7168 (\$1C00) RAM
9006	36870	VICCR6	Horizontal position of light pen default value 0
9007	36871	VICCR7	Vertical position of light pen default value 0
9008	36872	VICCR8	Digitized value of paddle X 0 maximum RIGHT position 255 maximum LEFT position (default)
9009	36873	VICCR9	Digitized value of paddle Y 0 maximum RIGHT position 255 maximum LEFT position (default)
900A	36874	VICCR A	Frequency for oscillator 1 (low) bit 7 enable switch BASS bit 6–0 oscillator range
900B	36875	VICCR B	Frequency for oscillator 2 (medium) bit 7 enable switch ALTO bit 6–0 oscillator range
900C	36876	VICCR C	Frequency for oscillator 3 (high) bit 7 enable switch SOPRANO bit 6–0 oscillator range
900D	36877	VICCR D	Frequency of noise source bit 7 enable switch NOISE bit 6–0 oscillator range
900E	36878	VICCR E	Volume and auxiliary color bit 7–4 auxiliary color bit 3–0 volume
900F	36879	VICCR F	Screen and border color register (default 27) bit 7–4 background color bit 3 inverse switch bit 2–0 border color default value 27 (white screen, cyan border)
9010	36880–37125		Unused; Unreliable reflections of 36864-36868 (-16)
9110	37136–37151		6522 Versatile Interface Adapter 1
9110	37136	VIA1PB	Port B input / output register bit 7 Data Set Ready (DSR) IN X-line bit 6 Clear to Send (CTS) IN X-line bit 5 unused bit 4 Data Carrier Detect (DCD) IN X-line bit 3 Ring Indicator (RI) IN

HEX	DEC		DESCRIPTION
			bit 2 Data Terminal Ready (DTR) OUT X-line bit 1 Request To Send (RTS) OUT X-line bit 0 Received Data Signal (Sin) IN X-line, 3 line
9111	37137	VIA1PA1	Port A input / output register bit 7 serial attention out bit 6 tape button down flag bit 5 light pen / joystick FIRE bit 4 joystick LEFT off bit 3 joystick DOWN off bit 2 joystick UP off bit 1 serial data IN bit 0 serial clock IN
9112	37138	VIA1DDRB	Data direction register B
9113	37139	VIAIDDRA	Data direction register A
9114	37140	VIA1T1CL	Timer 1 low byte (LSB)
9115	37141	VIA1T1CH	Timer 1 high byte & counter (MSB) setting this byte for timer 1 starts timer
9116	37142	VIA1T1LL	Timer 1 low byte (LSB)
9117	37143	VIA1T1LH	Timer 1 high byte (MSB)
9118	37144	VIA1T2CL	Timer 2 low byte (LSB)
9119	37145	VIA1T2CH	Timer 2 high byte (MSB)
911A	37146	VIA1SR	Shift register for parallel and serial conversion NOTE: the KERNAL does not use this shift register bit 7 shift out onto CB2 line bit 4-2 aux control register at 37147 select shift reg bit 2 interrupt enable and interrupt flag corresponds
911B	37147	VIA1ACR	Auxiliary control register bit 7-6 timer 1 options: PB7 output (neg, square) 00 single interval more, no PB7 output pulse 01 free run more, no PB7 output pulse 10 single interval more, PB7 negative pulsed 11 single interval more, PB7 negative pulsed bit 5 timer 2 options 0 single interval timing 1 countdown incoming PB6 pulses bit 4-2 shift register options 000 shift register disabled 001 input CB2 shift bit 0, timer 2, output clock CB1 010 input CB2 shift bit 0, system, output clock CB1 011 output CB2 bit 7, timer 2, output clock CB1 100 output CB2 bit 7, timer 2 as delay clock 110 output CB2 bit 7, system, output clock CB1 111 output CB2 bit 7, output external clock, in CB1 bit 1 Port B (location 37136 \$9110) 0 port B reflects changing values on pins

HEX	DEC		DESCRIPTION
			<p>1 port in latch mode (input CB1 interrupt occurred)</p> <p>bit 0 Port A (location 37137 \$9111)</p> <p>0 port A reflects changing values on pins</p> <p>1 port in latch mode (input CA1 interrupt occurred)</p>
911C	37148	VIA1PCR	<p>Peripheral control register (handshake)</p> <p>bit 7 CB2 line control input output mode</p> <p>0 input mode</p> <p>1 output mode</p> <p>bit 6-5 CB2 line control</p> <p>00 IFR bit 3 high to low CB2, clears IFR</p> <p>01 IFR bit 3 high to low CB2, does not clear IFR</p> <p>10 IFR bit 3 low to high CB2, clears IFR</p> <p>11 IFR bit 3 low to high CB2, does not clear IFR</p> <p>bit 4 CB1 line control low (0) or high (1)</p> <p>0 IFR bit 4 high to low transition of CB1 (default)</p> <p>1 IFR bit 4 low to high transition of CB1</p> <p>bit 3-1 CA2 line control</p> <p>000 IFR bit 0 high to low CA2, clears CA2</p> <p>001 IFR bit 0 high to low CA2, does not clear CA2</p> <p>010 IFR bit 0 low to high CA2, clears CA2</p> <p>011 IFR bit 0 low to high CA2, does not clear CA2</p> <p>100 output (handshake) CA2 low</p> <p>101 output mode (pulse)</p> <p>110 output mode (manual), CA2 low</p> <p>111 output mode (manual), CA2 high</p> <p>bit 0 CA1 line control low (0) or high (1)</p> <p>0 IFR bit 4 high to low transition of CA1</p> <p>1 IFR bit 4 low to high transition of CA1</p>
911D	37149	VIA11FR	<p>Interrupt flag register (IFR) (service order 1,6,5,4)</p> <p>bit 7 IRQ status (NMI)</p> <p>bit 6 timer 1 interrupt flag, RS-232 send</p> <p>bit 5 timer 2 interrupt flag, RS-232 receive</p> <p>bit 4 CB1 interrupt flag, RS-232 receive</p> <p>bit 3 CB2 interrupt flag</p> <p>bit 2 shift register flag</p> <p>bit 1 CA1 interrupt flag</p> <p>bit 0 CA2 interrupt flag, tape motor, if STOP, BRK</p>
911E	37150	VIA11ER	<p>Interrupt enable register (IER)</p> <p>bit 7 IER set / clear control (1 enable IFR)</p> <p>bit 6 timer 1 interrupt enable</p> <p>bit 5 timer 2 interrupt enable</p> <p>bit 4 CB1 interrupt enable</p> <p>bit 3 CB2 interrupt enable</p> <p>bit 2 shift register enable</p> <p>bit 1 CA1 interrupt enable</p>

HEX	DEC		DESCRIPTION																																																																																	
			bit 0 CA2 interrupt enable																																																																																	
911F	37151	VIA1PA2	Port A (sense tape switch); mirror of 37137																																																																																	
911F	37151 - 37167		6522 Versatile Interface Adapter 2																																																																																	
9120	37152	VIA2PB	Port B input / output register; Key column scan default value 247 except when SCNKEY or UDTIM bit 7 (127) key column 7; Joy 3, pin 4 game port bit 6 (191) key column 6 bit 5 (223) key column 5 bit 4 (239) key column 4 bit 3 (247) key column 3; tape write, port E-5 bit 2 (251) key column 2 bit 1 (253) key column 1 bit 0 (254) key column 0 see value chart in 37153																																																																																	
9121	37153	VIA2PA1	Port A input / output register; Keyboard row scan bit 7 keyboard row 7 bit 6 keyboard row 6 bit 5 keyboard row 5 bit 4 keyboard row 4 bit 3 keyboard row 3 bit 2 keyboard row 2 bit 1 keyboard row 1 bit 0 keyboard row 0																																																																																	
			<table border="1"> <thead> <tr> <th></th> <th>7</th> <th>6</th> <th>5</th> <th>4</th> <th>3</th> <th>2</th> <th>1</th> <th>0</th> </tr> </thead> <tbody> <tr> <td>7</td> <td>F7</td> <td>F5</td> <td>F3</td> <td>F1</td> <td>⌘</td> <td>⇄</td> <td>Rt</td> <td>De</td> </tr> <tr> <td>6</td> <td>H0</td> <td>↑</td> <td>=</td> <td>Ri</td> <td>/</td> <td>;</td> <td>*</td> <td>£</td> </tr> <tr> <td>5</td> <td>-</td> <td>@</td> <td>:</td> <td>.</td> <td>,</td> <td>L</td> <td>P</td> <td>+</td> </tr> <tr> <td>4</td> <td>0</td> <td>O</td> <td>K</td> <td>M</td> <td>N</td> <td>J</td> <td>I</td> <td>9</td> </tr> <tr> <td>3</td> <td>8</td> <td>U</td> <td>H</td> <td>B</td> <td>V</td> <td>G</td> <td>Y</td> <td>7</td> </tr> <tr> <td>2</td> <td>6</td> <td>T</td> <td>F</td> <td>C</td> <td>X</td> <td>D</td> <td>R</td> <td>5</td> </tr> <tr> <td>1</td> <td>4</td> <td>E</td> <td>S</td> <td>Z</td> <td>Lf</td> <td>A</td> <td>W</td> <td>3</td> </tr> <tr> <td>0</td> <td>2</td> <td>Q</td> <td>C=</td> <td>SPC</td> <td>St</td> <td>Ct</td> <td>←</td> <td>1</td> </tr> </tbody> </table>		7	6	5	4	3	2	1	0	7	F7	F5	F3	F1	⌘	⇄	Rt	De	6	H0	↑	=	Ri	/	;	*	£	5	-	@	:	.	,	L	P	+	4	0	O	K	M	N	J	I	9	3	8	U	H	B	V	G	Y	7	2	6	T	F	C	X	D	R	5	1	4	E	S	Z	Lf	A	W	3	0	2	Q	C=	SPC	St	Ct	←	1
	7	6	5	4	3	2	1	0																																																																												
7	F7	F5	F3	F1	⌘	⇄	Rt	De																																																																												
6	H0	↑	=	Ri	/	;	*	£																																																																												
5	-	@	:	.	,	L	P	+																																																																												
4	0	O	K	M	N	J	I	9																																																																												
3	8	U	H	B	V	G	Y	7																																																																												
2	6	T	F	C	X	D	R	5																																																																												
1	4	E	S	Z	Lf	A	W	3																																																																												
0	2	Q	C=	SPC	St	Ct	←	1																																																																												
			column stored in 37152 (\$9120)																																																																																	
9122	37154	VIA2DDR B	Data direction register B when a bit is set to 0, port B but is used for input 127 set direction for controller, some keys ignored 255 all lines as output (default and restore value)																																																																																	
9123	37155	VIA2DDRA	Data direction register A when a bit is set to 0, port A but is used for input 0 (default and restore value) Power-on/reset, RUN/STOP-RESTORE sets to 0																																																																																	
9124	37156	VIA2T1CL	Timer 1, low byte latch (LSB) used to generate the IRQ 60 times per second																																																																																	
9125	37157	VIA2T1CH	Timer 1, high byte latch (MSB) setting this byte of timer to 1 starts timer running																																																																																	

HEX	DEC		DESCRIPTION
9126	37158	VIA2T1LL	Timer 1, low byte counter (LSB)
9127	37159	VIA2T1HL	Timer 1, high byte counter (MSB)
9128	37160	VIA2T2CL	Timer 2, low byte latch (LSB) timer used by the KERNAL for detecting timeouts
9129	37161	VIA2T2CH	Timer 2, high byte latch (MSB)
912A	37162	VIA2SR	Shift register for parallel and serial conversion
912B	37163	VIA2ACR	Auxiliary control register bit 7-6 timer 1 options (single/free, PB7) bit 5 timer 2 (0 single interval, 1 count PB6 pulse) bit 4-2 shift register options 000 shift disabled 001 input on CB2 using timer 2 010 input on CB2 using system clock 011 output on CBs using timer 2 100 output on CB2 using timer 2 delay 110 output on CB2 using system clock 111 output on CB2 using external clock bit 1 Port B latch enable options bit 0 Port A latch enable options
912C	37164	VIA2PCR	Peripheral control register (handshake) bit 7-5 CB2 line control, data out 000 input mode IFR bit 3 hi and clear CB2 001 input mode IFR bit 3 hi and do not clear 010 input mode IFR bit 3 lo and clear CB2 011 input mode IFR bit 3 lo and do not clear 100 output mode (handshake) CB2 low 110 output mode (pulse) CB2 low 111 output mode (manual) CB2 high 000 shift disabled bit 4 CB1 SRQ IN (0 high to low, 1 low to high) bit 3-1 CA2 line control, serial clock out 000 input mode IFR bit 0 hi and clear CA2 001 input mode IFR bit 3 hi and do not clear 010 input mode IFR bit 3 lo and clear CA2 011 input mode IFR bit 3 lo and do not clear 100 output mode (handshake) CA2 low 110 output mode (pulse) CA2 low 111 output mode (manual) CA2 high 000 shift disabled bit 0 CA1 line control, TAPE read line
912D	37165	VIA2IFR	Interrupt flag register (IFR) bit 7 IRQ occurred (0 clears all interrupts) bit 6 timer 1 interrupt bit 5 timer 2 interrupt bit 4 CB1 transition interrupt

HEX	DEC		DESCRIPTION
			bit 3 CB2 transition interrupt bit 2 shift register interrupt bit 1 CA1 transition interrupt bit 0 CA2 transition interrupt
912E	37166	VIA2IER	Interrupt enable register (IER) bit 7 IFR enable (1) disable (1) bit 6 timer 1 interrupt enable (1) bit 5 timer 2 interrupt bit 4 CB1 interrupt bit 3 CB2 interrupt bit 2 shift register interrupt bit 1 CA1 (TAPE I/O) interrupt bit 0 CA2 interrupt
912F	37167	VIA2PA2	Port A output register; mirror 37153
9130	37168-		Unused I/O for future expansion block1
9400	37888-38399	COLORMA	Color RAM map for 8k+ expansion bit 7-4 not used bit 3 multicolor enable (1) disable default (0) bit 2-0 foreground color (default value 1)
9600	38400-38911	COLORMA	Color RAM map for unexp. and +3K expansion bit 7-4 not used bit 3 multicolor enable (1) disable default (0) bit 2-0 foreground color (default value 1)
97FA	38906-38911		Screen color memory outside view by default
9800	38912-39935		I/O block 2
9C00	39936-40959		I/O block 3
A000	40960-	RAMBLK5	8K decoded for exp ROM block 4; auto start carts
A000	40960-40961		ROM cart: Vector for power-on/reset routines Auto start "hard start" location (lo-hi)
A002	40962-40963		ROM cart: Vector for NMI (RESTORE key) routines Auto start "warm start" location (lo-hi)
A004	40964-40968		Auto start code data 65,48,195,194,205 ("A0CBM") initialization routines to JSR when autostarted: \$FD8D RAM test \$FD52 set vectors \$FDF9 initialize I/O, CLI \$E518 initialize screen
C000	49152	COLDST	Vector to routine for cold start of BASIC (58232)
C002	49154	WARMST	Vector to routine for warm start of BASIC (58471)
CC04	49156-	CBMBASI	CBMBASIC characters in ASCII
C00C	49164-	STMDSP	Keyword operation pointer directory in token order
C052	49234-	FUNDSP	Function operation pointer directory in token order
C080	49280-	OPTAB	Math operation pointer directory in token order
C080	49280		addition

HEX	DEC		DESCRIPTION
C083	49283		subtraction
C086	49286		multiplication
C089	49289		division
C08C	49292		up arrow
C08F	49295		AND
C092	49298		OR
C095	49301		mondatic
C098	49304		NOT
C09B	49307		lless than/ equal to / greater than
C09E	49310-	RESLST	BASIC keywords in token order and ASCII
C19D	49565-	ERRTAB	BASIC error messages in ASCII
C328	49960-	BMSGs	Pointers for 30 error messages
C364	50020-	MISCMSG	Miscellaneous error messages in ASCII
C38A	50058	SCNSTK	Routine: Find FOR and GOSUB stack entries
C3B8	50104	MAKSPC	Open space in mem for new BASIC line / variable
C3BF	50111	MOVEBL	Routine: Move a block of memory
C3FB	50171	STKSPC	Routine: Check stack requested space available
C408	50184	RAMSPC	Routine: Check space available for dynamic request
C435	50229	MEMERR	Set OUT OF MEMORY error message code
C437	50231	ERROR	Routine: Look up BASIC error message
C469	50281	PRDY	Display ERROR or BREAK message
C474	50292	READY	Routine: Display READY message
C480	50304	MAIN	Routine: Receive and execute or store BASIC line
C49C	50332	NEWLIN	Routine: Store or replace a BASIC program line
C533	50483	LNKPRG	Routine: Rechain BASIC program lines
C560	50528	GETLIN	Receive input from device
C579	50553	CRNCH	Tokenize the BASIC line in BASIC text buffer
C613	50707	FINLIN	Routine: Find BASIC program line from line number
C642	50754	NEW	Routine: BASIC NEW
C65E	50782	CLR	Routine: BASIC CLR (C659?)
C68E	50830	STXTPT	Routine (SYS will branch to first program line)
C69C	50844	LIST	Routine: BASIC LIST
C71A	50970	QPLOP	Routine: List detokenized BASIC keywords
C742	51010	FOR	Routine: BASIC FOR
C7AE	51118	NEWSTT	Routine: Find next BASIC statement for execution
C7E4	51172	GONE	Routine: Execute current BASIC statement
C81D	51229	RESTORE	Routine: BASIC RESTORE
C82C	51244	TSTSTOP	Test for STOP key
C82F	51247	BSTOP	Routine: BASIC STOP
C831	51249	END	Routine: BASIC END
C857	51287	CONT	Routine: BASIC CONT
C871	51313	RUN	Routine: BASIC RUN
C883	51331	GOSUB	Routine: BASIC GOSUB
C8A0	51360	GOTO	Routine: BASIC GOTO
C8D2	51410	RETURN	Routine: BASIC RETURN

HEX	DEC		DESCRIPTION
C8F8	51448	SKIPST	Routine: BASIC DATA
C8FB	51451	BUMPTP	Increment TXTPTR by amount in .Y
C906	51462	FIND2	Scan for the next BASIC statement
C928	51496	IF	Routine: BASIC IF
C93B	51515	REM	Routine: BASIC REM
C94B	51531	ON	Routine: BASIC ON
C96B	51563	DECBIN	Convert decimal line no. into LSB-HSB format
C9A5	51621	LET	Routine: BASIC LET
C9C2	51650	LET2	Routine: Assign integer variable (LET)
C9DA	51674	LET5	Routine: Assign TI\$ (LET)
C9E3	51683		Set TI\$ from string
CA2C	51756	LET9	Routine: Assign string variable (LET)
CA80	51840	PRINTN	Routine: BASIC PRINT#
CA86	51846	CMD	Routine: BASIC CMD
CA9A	51866 - 51871	PRT1	Instructions for PRINT routine
CAA0	51872	PRINT	Routine: BASIC PRINT
CAE8	51944	PRT6	Tabs to column when comma is used
CAF8	51960	PRT7	Routine: BASIC TAB and SPC
CB1E	51998	PRTSTR	Part of print routine outputs string with cR return
CB3B	52027	PRTOS	PRINT routine for special chars (space, right, ?)
CB4D	52045	IGRERR	Error msg format routine for GET, INPUT, READ
CB7B	52091	GET	Routine: BASIC GET
CBA5	52133	INPUTN	Routine: BASIC INPUT#
CBBF	52159	INPUT	Routine: BASIC INPUT
CC06	52230	READ	Routine: Scan for next DATA item for READ
CCFC	52476 - 52509	EXTRA	INPUT error messages in ASCII
CD1E	52510	NEXT	Routine: BASIC NEXT
CD8A	52618 -	TYPCHK	Four routines to check variable type
CD9E	52638	FRMEVL	Master routine for formula and expression
CE83	52867	EVAL	Evaluate a single term of an expression
CEA8	52904 - 52908	PIVAL	Value of PI
CED4	52948		Routine: BASIC NOT
CEF1	52977	PAREXP	Evaluation within parenthesis
CEF7	52983	RPACHK	Check for close parenthesis
CEFA	52986	LPACHK	Check for open parenthesis
CEFD	52989	COMCHK	Check for comma in syntax
CEFF	52991	SYNCHR	Syntax check for specific character from CHRGET
CF08	53000	SYNERR	Cause SYNTAX ERROR message
CF0D	53005	FACT10	Set index for nomadic minus, FACT10
CF14	53012	VARRANG	Check range of variable
CF28	53032	FACT12	Get variable name and type, check for null string
CFA7	53159	FACT17	Invoke function FACT17
CFE6	53222	ORR	Routine: BASIC OR
CFE9	53225	ANDD	Routine: BASIC AND
D016	53270	COMPAR	Routine: Compare numbers or strings: < = >

HEX	DEC		DESCRIPTION
D02E	53294	CMPST	Compare strings
D081	53377	DIM	Routine: BASIC DIM
D08B	53387	EVLVAR	Evaluate / create a variable
D0E7	53479	FNDVAR	Locate variable, jump to make var if not found
D113	53523	CHRTST	Check if ASCII character is alphabetic
D11D	53533	MAKVAR	Routine: Create a new variable
D185	53637	RETVP	Return the address of found or created variable
D194	53652	ARYHED	Calculate the length of an array
D1A5	53669	MAXINT	Max length of integer value in floating point: 32768
D1AA	53674	INTIDX	Convert float point to 2-byte fixed point in A and Y
D1B2	53682	GETSUB	Convert an expression to an int number if positive
D1BF	53695	MAKINT	Convert floating point to signed integer
D1D1	53713	ARY	Locate an array, create if not found
D245	53829	BADSUB	Display BAD SUBSCRIPT message
D248	53832	ILQUAN	Display ILLEGAL QUANTITY message
D24D	53837	ARY2	Checks if subscript is within the range
D261	53857	ARY6	Routine: Create an array
D2EA	53994	ARY14	Locate a specific array element
D34C	54092	M16	Calculate multidimensional array size
D37D	54141	FRE	Routine: BASIC FRE (free memory calculation)
D391	54161	MAKFP	Convert Y and A integer to floating point
D39E	54174	POS	Routine: BASIC POS
D3A6	54182	NODIRM	Check if statement is entered in direct mode
D3AE	54190	UNDEF	Display UNDEF'D FUNCTION message
D3B3	54195	DEF	Routine: BASIC DEF
D3E1	54241	FN	Check DEF FN and FN syntax
D3F4	54260	EVALFN	Routine: BASIC FN
D44F	54351	EVFN3	Store DEF FN values into stack
D465	54373	STR	Routine: BASIC STR\$
D475	54389	ALC1	Calculate new string length and vector
D487	54407	MAKSTR	Scan and set up a string
D4F4	54516	ALCSPC	Allocate memory for a string
D526	54566	GRBCOL	Garbage collection for new or altrd string GRBCOL
D5BD	54717 - 54844	GCOL13	String garbage routines
E092	57490	COLECT	Garbage collection for string
D63D	54845	ADDSTR	BASIC add to string, check for string too long
D67A	54906	XFERSTR	Routine: Move string in memory
D6A3	54947	DELST	Discard a temporary string
D6DB	55003	DELTSO	Routine to delete temporary string stack
D6EC	55020	CHR	Routine: BASIC CHR\$
D700	55040	LEFT	Routine: BASIC LEFT\$
D72C	55084	RIGHT	Routine: BASIC RIGHT\$
D737	55095	MID	Routine: BASIC MID\$
D761	55137	FINLMR	Obtain string param for LEFT\$, MID\$, RIGHT\$
D77C	55164	LEN	Routine: BASIC LEN, get the length param of string

HEX	DEC		DESCRIPTION
D782	55170	GSINFO	Get string information
D78B	55179	ASC	Routine: BASIC ASC
D79B	55195	GETBYT	Get a one-byte parameter 0 to 255
D7AD	55213	VAL	Routine: BASIC VAL
D7EB	55275	GETAD	Get two-byte address for POKE and WAIT
D7F7	55287	MAKADR	Convert floating point into two-byte address integer
D80D	55309	PEEK	Routine: BASIC PEEK
D824	55332	POKE	Routine: BASIC POKE
D82D	55341	WAIT	Routine: BASIC WAIT
D849	55369	ADD05	Routine: round number by adding .5
D850	55376	LAMIN	Subtract mem contents from floating point accum.
D853	55379	SUB	BASIC - (subtract)
D862	55394	PLUS1	Routine for processing exponents
D867	55399	LAPLUS	Add memory contents to floating point accum.
D86A	55402	PLUS	BASIC + (add)
D8A7	55463	PLUS6	Adjust exponent to negative
D8F7	55543	ZERFAC	Set FAC to zero and sign to positive
D8FE	55550	NORMLZ	Add fractions and normalize result
D947	55623	COMFAC	Complement FAC by reversing bits with EOR +1
D97E	55678	OVERFL	Display OVERFLOW message
D983	55683	ASRRES	Shift exponents to equal prior to math function
D9BC	55740	FPC1	Constant of one for floating point accumulator
D9C1	55745	LOGCON	Constants for LOG function
D9EA	55786	LOG	Routine: BASIC LOG
DA28	55848	TIMES	BASIC * (multiply)
DA59	55897	TIMES3	Multiply a byte subroutine
DA8C	55948	LODARG	Move floating point mem loc to FAC2
DAB7	55991	MULDIV	Add exponents pf FAC to FAC2
DAE2	56034	MULTEN	Multiply FAC by 10
DAF9	56057	FPCTEN	+10 floating point constant
DAFE	56062	DIVTEN	Divide FAC by 10
DB07	56071		Divide (AFAC) by MFLPT (A/Y), sign (X), ans (FAC)
DB0F	56079	LADIV	Move floating point in memory to FAC2
DB12	56082	DIVIDE	BASIC / (divide)
DBA2	56226	LODFAC	Move floating point in memory to FAC
DBC7	56263	FACTF2	Move FAC2 to memory
DBC4	56266	FACTF1	Move FAC to memory
DBD0	56272	FACTFP	Move FAC to memory
DBD4	56276	STORFAC	Perform move of FAC to memory
DBFC	56316	ATOF	Transfer FAC2 to FAC
DC0C	56332	RFTOA	Move FAC to FAC2 with rounding
DC0F	56335	FTOA	Move FAC to FAC2 without rounding
DC1B	56347	ROUND	Round FAC by adjusting the rounding byte
DC2B	56363	SGNFAC	Test the sign of FAC
DC39	56377	SGN	Routine: BASIC SGN

HEX	DEC		DESCRIPTION
DC3C	56380	INTFP	Convert sign to 0 or -1
DC44	56388	INTFP1	Convert 2byte integer to floating point in FAC
DC58	56408	ABS	Routine: BASIC ABS
DC5B	56411	CMPFAC	Compare FAC to memory
DC9B	56475	FPINT	Convert FAC floating point to signed integer
DCCC	56524	INT	Routine: BASIC INT
DCE9	56553	FILFAC	Convert an ASCII string to a floating point number
DCF3	56563	ASCFLT	Convert ASCII string to floating point no. in FAC
DD7E	56702	ASC18	Add .A to FAC
DDB3	56755	FPC12	String to floating point conversion constants
DDC2	56770	PRTIN	Issue message IN
DDCD	56781	PRTFIX	Routine: Display decimal number
DDDD	56797	FLTASC	Routine: Convert to TI\$ or ASCII string
DF11	57105	FLP05	Constant 0.5 for rounding and SQR
DF16	57110	FLTCON	Powers of 10 table in 4byte fixed integer
DF3A	57146	HMSCON	Constants for TI\$ division conversion
DF52	57170-57200		Unused area?
DF71	57201	SQR	Routine: BASIC SQR
DF7B	57211	EXPONT	BASIC power (up arrow),exponent
DFB4	57268	NEGFAC	BASIC monadic
DFBF	57279	EXPCON	Table for EXP in floating point format
DFED	57325	EXP	Routine: BASIC EXP
E000	57344-65535		8K KERNAL ROM (sys location)
E000	57344-58527		BASIC spillover into Kernal ROM
E040	57408	SEREVL	Series evaluation subroutine
E056	57430	SER2	Math series evaluation routine
E08A	57482	RNDC1	Table of constants for RND
E094	57492	RND	Routine: BASIC RND
E0F6	57590	PATCHBA	BASIC patch routines
E127	57639	SYSTEM	BASIC SYS
E153	57683	SAVE	Routine: BASIC SAVE
E162	57698	BVERIF	Routine: BASIC VERIFY
E165	57701	BLOAD	Routine: BASIC LOAD
E1BB	57787	FOPEN	Routine: BASIC OPEN
E1C4	57796	FCLOSE	Routine: BASIC CLOSE
E1D1	57809	PARSL	Set parameters for SAVE, VERIFY, and LOAD
E203	57859	IFCHRG	Check for more characters in current statement
E20B	57867	SKPCOM	Skip any comma in parameters being scanned
E20E	57870	CHRERR	Check for parameter after comma
E216	57878	PAROC	Routine: set parameters for OPEN and CLOSE
E261	57953	COS	Routine: BASIC COS
E268	57960	SIN	Routine: BASIC SIN
E2B1	58033	TAN	Routine: BASIC TAN
E2DD	58077	FPC20	Trigonometric constants for COS, SIN, and TAN
E30B	58123	ATN	Routine: BASIC ATN

HEX	DEC		DESCRIPTION
E33B	58171	ATNCON	Constants for ATN evaluation
E378	58232	COLDBA	Routine: Cold start BASIC
E387	58247	CGIMAG	CHRGET rout, RND seed copied to zero RAM
E3A4	58276	INITBA	Initialize BASIC: restore page zero pointers
E404	58372	FREMSG	Display routine for startup; calculates free memory
E429	58409-	CBMMSG	Vic BASIC start up screen in ASCII
E44F	58447	BASVCTR	6 BASIC vectors copied to 768
E45B	58459	INITVCTR	Copy BASIC vectors from ROM to RAM
E467	58471	WARMBAS	Routine: Warm start BASIC
E476	58486	PATCHER	Program patch area
E4A0	58528-65535		KERNAL ROM
E4A0	58528	SEROUT1	Serial output a 1
E4A9	58537	SEROUT0	Serial output a 0
E4B2	58546	SERGET	Serial get input bit from VIA1 and stabilize
E4BC	58556	PATCHES	Program patch area
E4C1	58561		Transfer 195-196 and display LOAD/VERIFY msg
E4CF	58575		Addendum to close tape
E4CF	58624	IOBASE	Retrieve address of I/O memory page
E505	58629	SCRN	Max screen columns and rows; comp check
E50A	58634	PLOT	Read or set current cursor column and line
E518	58648	INITSK	Initialize 6550 VIC chip, screen, and pointers
E55F	58719	CLSR	Clear the screen
E581	58753	HOME	Move cursor to home position
E587	58759	SETSLINK	Reset screen line link table pointers
E5B5	58805	UNUSDNM	NMI entry for restore key
E5BB	58811	SETIODEF	Reset device numbers to defaults
E5C3	58819	INITVIC	Reset VIC chip registers from table at 60900
E5CF	58831	LP2	Get a char from keyboard buff and shift queue
E5E5	58853	GETQUE	Wait for char in key buffer
E619	58905	GET2RTN	Purge the keyboard buffer by routine at 58959
E64F	58959	GETSCRN	Get INPUT from screen
E6B8	59064	QUOTECK	Test for quotes and set flag
E6C5	59077	SETCHAR	Set up display of character on screen
E6EA	59114	SCROLL	Move cursor on screen, add line, scroll
E72D	59181	RETREAT	Mover cursor to previous logical line and column
E742	59202	SCRNOUT	Handle characters going to the screen
E8C3	59587	NEXTLINE	Mover cursor to next screen line
E8D8	59608	RTRN	Carriage return, turns off quote and revers, etc.
E8E8	59624	BACKUP	Move cursor to end of previous line
E8FA	59642	FORWARD	Mover cursor to start of next line
E912	59666	COLORSE	Set foreground color
E921	59681	COLORTB	Color code key table
E929	59689	CNVRTCD	Code conversion table
E975	59765	SCRL	Routine: Scroll the screen
E9EE	59886	OPENLIN	Routine: Open a blank line

HEX	DEC		DESCRIPTION
EA56	59990	MOVLINE	Routine: Move screen line
EA6E	60014	SETADDR	Address of screen line and color set in memory
EA7E	60030	LINPTR	Set pointer to address of start of screen line
EA8D	60045	CLRALINE	Routine: Blank out a screen line
EAA1	60065	SYNPRT	Sync color to byte and store char on screen
AAA	60074	PUTSCRN	Store character on screen
EAB2	60082	COLORSY	Address of color map byte for screen map byte
EABF	60095	IRQ	IRQ interrupt handler
EB1E	60190	SCNKEY	Scan keyboard for press using 6522 VIA2
EBDC	60380	SETKEYS	Set keyboard decode table address in 245-6
EC46	60486	KEYVCTR	Keyboard decode table addresses
EC5E	60510	NORMKEY	Table for decoding unshifted keys into ASCII
EC9F	60575	SHFTKEY	Table for decoding shifted keys into ASCII
ECE0	60640	LOGOKEY	Table for decoding C= shifted keys into ASCII
ED21	60705	CHARSET	Set uppercase / graphics character set
ED30	60720	GRAPHMO	Set graphic mode specified by certain ASCII chars
ED5B	60763	WRAPLIN	Wrap line on screen
ED69	60777	WHATKEY	Keyboard decoding table (unused?)
EDA3	60835	CTRLKEY	Table for decoding control shifted keys into ASCII
EDE4	60900	VICINIT	Initial values for video interface chip registers
EDF4	60916	RUNTB	Shift RUN keys put LOAD and RUN in buffer
EDFD	60925	LDTB2	Screen line link table LSB of lines in screen map
EE14	60948	TALK	Instruct the serial device to talk to computer
EE17	60951	LISTEN	Instruct the serial device to listen to computer
EE1C	60956	LISTI	Prepare to send serial command
EE49	61001	SRSEND	Send command or data to serial devices
EEB4	61108	SRBAD	Set ST for timeout or DEVICE NOT PRESENT
EEC0	61220	SECOND	Send secondary address after listen command
EEC5	61125	SCATN	Serial: clear attention
EECE	61134	TKSA	Serial: send secondary address after talk command
EEE4	61156	CIOUT	Send byte on serial line
EEE4	61156	UNTLK	Serial: send untalk command to serial devices
EF04	61188	UNLSN	Send unlisten command to serial devices
EF19	61209	ACPTR	Receive byte from serial device
EF84	61316	SRCLKHI	Serial: set clock line high
EF84	61325	SRCLKLO	Serial: set clock line low
EF96	61334	WAITABIT	Delay one millisecond
EFA3	61347	RSNXTBIT	RS-232: send next bit
EFBF	61375	RSPRTY	RS-232: calculate parity and stop bits value
EFE8	61416	RSSTOPS	RS-232: transmit stop bits
EFEE	61422	RSNXTBY	RS-232: prepare next byte to be sent from buffer
F016	61462	RSMISSN	RS-232: clear to send OR data set ready missing
F027	61479	RSCPTBIT	RS-232: compute desired word length bit count
F036	61494	RSINBIT	RS-232: receive an input bit NMI
F04B	61515	RSSTPBIT	RS-232: determine if all stop bits received

HEX	DEC		DESCRIPTION
F05B	61531	RSPREPIN	RS-232: prepare to receive the next input byte
F068	61544	RSSTRBIT	RS-232: check for start bit in receive mode
F06F	61551	RSINBYTE	RS-232: put constructed byte into receive buffer
F09B	61579	RSINPRTY	RS-232: parity check for the input byte
F09D	61597	RSPRTYE	RS-232: parity error on input byte
F0A2	61602	RSOVERU	RS-232: buffer overrun on input byte
F0A5	61605	RSBREAK	RS-232: break detected on input
F0A8	61608	RSFRAME	RS-232: framing error on input
F0AA	61610	RSINERR	RS-232: set input error status and continue
F0B9	61625	RSDVCER	RS-232: ILLEGAL DEVICE message
F0BC	61628	RSOPNOU	RS-232: open channel for output
FOED	61677	RSOUTSA	RS-232: store character in transmit buffer
F102	61698	RSPREPO	RS-232: set up NMI for transmission
F116	61718	RSOPNIN	RS-232: open channel for input
F14F	61775	RSNXTIN	RS-232: receive next character from buffer
F160	61792	RSPAUSE	RS-232: check that serial and tape are idle
F174	61812	KMSGTBL	Table of Kernal messages
F1E2	61922	SPMSG	Display LOADING or VERIFYING messages
F1E6	61926	KMSGSH	Print Kernal control messages
F1F5	61941	GETIN	Routing routine for obtaining character input data
F20E	61966	CHRIN	Input characters from current input device
F230	62000	CHRINTP	Obtain byte from tape buffer
F250	62032	CHRINTP2	Load .A with next tape character
F264	62052	CHRINSR	Obtain byte from the serial line
F26F	62063	CHRINRS	Obtain byte from RS-232
F27A	62074	CHROUT	Output character to current output device
F290	62096	CHROUTT	Output a character to tape
F2C7	62151	CHKIN	Open .X file number channel for input
F309	62217	CHKOUT	Open .X file number channel for output
F34A	62282	CLOSE	Close logical file number in .A
F3CF	62415	FNDFLNO	Find file number (.X) in file table at 601(\$259)
F3DF	62431	SETFLCH	Set file characteristics of file (.X) into 184-186
F3EF	62447	CLALL	Abort all open files with no close
F3F3	62451	CLRCHN	Abort all open channels
F40A	62474	OPEN	Open logical file, file number in 184
F495	62613	SERNAME	Send secondary address and filename to serial
F4C7	62663	OPENRS	Open RS-232 device
F542	62786	LOAD	Load or verify to RAM from device (number in 186)
F55C	62812	LOADSER	Load and verify RAM from a serial device
F5D1	62929	LOADTP	Load or verify RAM from tape
F647	63047	SRCHING	Display SEARCHING message for tape
F659	63065	FILENAME	Routine: Display filename
F66A	63082	LDVMSG	Display LOADING or VERIFYING message
F675	63093	SAVE	Save RAM to device number specified in 186
F692	63122	SAVESER	Save RAM to serial device

HEX	DEC		DESCRIPTION
F6F1	63217	SAVETP	Save RAM to tape
F728	63272	SAVING	Display SAVING message
F734	63284	UDTIM	Increment jiffy clock (160-162)
F760	63328	RDTIM	Put jiffy clock from 160-2 into .Y, .X, and .A
F767	63335	SETTIM	Set time into jiffy clock 160-2 from .Y, .X, and .A
F770	63344	STOP	Check for the STOP key, purge key buffer
F77E	63358	FILEMSG	I/O error file message handler
F7AF	53407	FAH	Find next tape header, .X (header ID number)
F7E7	63463	TAPEH	Build an output tape header in tape buffer
F84D	63565	TPBUFA	Load tape buffer address from 178-179 into .X, .Y
F854	63572	LDAD1	Set load/save start and end pointers to tape buffer
F867	63591	FNDHDR	Find the tape header for specified filename or next
F88A	63626	JTP20	Increment tape buffer character counter
F894	63636	CSTEL	Display PRESS PLAY ON TAPE message
F8AB	63659	CSIO	Check tape play/rewind/forward button status
F8B7	63671	CSTE2	Display PRESS RECORD & PLAY ON TAPE
F8C0	63680	RDTPBLK	Initiate tape header read
F8C9	63689	RBLK	Read blocks from tape
F8E3	63715	WBLK	Write blocks to tape
F8F4	63732	TAPE	Common tape read/write, start tape operations
F94B	63819	TSTOP	Check for tape STOP key
F95D	63837	STTI	Set time limit for tape dipole
F98E	63886	READT	Read tape data bits into location 191 (\$BF)
FAAD	64173	TPSTORE	Determine to store the input character from tape
FBD2	64466	RD300	Reset tape read pointer
FBDB	64475	NEWCH	New tape character setup
FBEA	64490	TPTOGLE	Toggle tape write line to invert output signal
FC06	64518	BLKEND	End of block write processing
FC08	64523	WRITE	IRQ driven tape data write
FC95	64661	WRTNI	block leader write
FCA8	64680	WRTZ	IRQ driven tape leader write
FCCF	64719	TNIF	Restore IRQ vector, call TNOFF
FCF6	64758	BSIV	Reset current IRQ vector
FD08	64776	TNOFF	Routine: Stop tape motor
FD11	64776	VPRTY	Compare current to end of load/save pointers
FD1B	64785	WRT62	Increment current load/save pointer
FD22	64802	START	Power on / reset routine; check for cartridge
FD3F	64831	CHKAUTO	Check for auto starting program at 40960(A000)
FD4D	64845	AOCBM	\$41,30,C3,C2,CD ("A0CBM")
FD52	64850	RESTOR	Reset RAM vectors to defaults
FD57	64855	VECTOR	Read or set system RAM vectors
FD6D	64877	VECTORS	Default system vector address storage table
FD8D	65909	INITMEM	Initialize system memory
PDF1	65009	IRQVCTRS	IRQ vectors table
PDF9	65017	INITVIA	Initialize the 6522 VIA registers

HEX	DEC		DESCRIPTION
FE49	65097	SETNAM	Filename pointer and length stored to .X, .Y, .A
FE50	65104	SETLFS	Set current file number, device, and secondary add
FE57	65111	READST	Reset RS-232 status. Branch to 65128 for non
FE66	65126	SETMSG	Set byte to enable/disable Kernal message display
FE68	65128	READIOST	Load .A with the non-RS-232 I/O status ST
FE6A	65130	ORIOST	OR .A with contents of 144 (\$90) ST and store
FE6F	65135	SETTMO	Set timeout value for IEEE-488
FE73	65139	MEMTOP	Read or set the top of memory pointer
FE82	65154	MEMBOT	Read or set the bottom of memory pointer
FE91	65169	TSTMEM	Test a memory location
FEA9	65193	NMI	NMI handler routine
FED2	65234	BREAK	Break interrupt entry
FEDE	65246	RSNMI	RS-232: NMI sequences
FF56	65366	RTI	Restore 6502 registers from stack, return interrupt
FF5C	65372	BAUDTBL	RS-232 VIA timer 2 values for baud rate table
FF72	65394	IRQROUT	IRQ routine initial 6502 entry point
FF84	65412	VECTOR	Read / set vectored I/O
FF85	65413	C4FFS	Five unused bytes (255)
FF87	65415	RESTOR	Restore default I/O vectors
FF8A	65418-		ROM JMP OPCODE FOLLOWED BY VECTORS
FF8A	65418	CRESTOR	Jump to 64850 (\$5D52) RESTOR
FF8D	65421	CVECTOR	Jump to 64855 (\$FF8D) VECTOR
FF90	65424	CSETMSG	Jump to 65126 (\$FE66) SETMSG
FF93	65427	CSECOND	Send secondary address after listen
FF96	65430	CTKSA	Send secondary address after talk
FF99	65433	CMEMTOP	Read / set the top of memory
FF9C	65436	CMEMBOT	Read / set the bottom of memory
FF9F	65439	CSCNKEY	Scan keyboard
FFA2	65442	CSETTMO	Set timeout on serial bus
FFA5	65445	CACPTR	Input byte from serial port
FFA8	65448	CCIOUT	Output byte to serial port
FFAB	65451	CUNTALK	Command serial bus to untalk
FFAE	65454	CUNLSN	Command serial bus to unlisten
FFB1	65457	CLISTEN	Listen to serial bus
FFB4	65460	CTALK	Command serial bus device to talk
FFB7	65463	CREADST	Read I/O status word
FFBA	65466	CSETLFS	Set logical, first, second addresses
FFBD	65469	CSETNAM	Set filename
FFC0	65472	COPEN	Open a logical file
FFC3	65475	CCLOSE	Close specified logical file
FFC6	65478	CHKIN	Set Input channel
FFC9	65481	CHKOUT	Set Output channel
FFCC	65484	CLRCHN	Close/clear input and output channels
FFCF	65487	CHRIN	Input character from channel
FFD2	65490	CHROUT	Output character to channel

HEX	DEC		DESCRIPTION
FFD5	65493	LOAD	Load RAM from a device
FFD8	65496	SAVE	Save RAM to device
FFDB	65499	SETTIM	Set real time clock
FFDE	65502	RDTIM	Read real time clock
FFE1	65505	STOP	Test Stop key
FFE4	65508	IGETIN	Indirect to get input from keyboard
FFE7	65511	CLALL	Close all channels and files
FFE8	65512	GETIN	Get character from keyboard buffer
FFEA	65514	UDTIM	Increment real time clock
FFED	65517	SCREEN	Return XY organization of screen
FFF0	65520	PLOT	Read / set XY cursor position
FFF3	65523	IOBASE	Returns base address of I/O devices
FFF6	65526		Four unused bytes (value 255)?
FFFA	65530	VCTRNM	B NMI
FFFC	65532	VCTRRST	D RESET
FFFE	65534	VCTRIRQ	F IRQ
FFFF	65535		Top of memory, Top of 8K KERNAL ROM

ALPHABETICAL CROSS REFERENCE TO MEMORY MAP LABELS

ABS	56408	BREAK	65234	CLALL	62447	CTKSA	65430
ACPTR	61209	BSIV	64758	CLALL	65511	CTRLKEY	60835
ADD05	55369	BSOUR	149	CLISTEN	65457	CUNLSN	65454
ADDDRC	1	BSTOP	51247	CLOSE	62282	CUNTALK	65451
ADDSTR	54845	BUF	512	CLR	50782	CURLIN	57
ADRAY1	3	BUFPNT	166	CLRALINE	60045	CVECTOR	65421
ADRAY2	5	BUMPTP	51451	CLRCHN	62451	DATLIN	63
ALC1	54389	BVECTOR	768	CLRCHN	65484	DATPTR	65
ALCSPC	54516	BVERIF	57698	CLSR	58719	DECBIN	51563
ANDD	53225	C3PO	148	CMD	51846	DEF	54195
AOCBM	64845	C4FFS	65413	CMEBOT	65436	DEFPNT	78
ARISGN	111	CACPTR	65445	CMEMTOP	65433	DELAY	652
ARY	53713	CASELRV	35840	CMPFAC	56411	DELST	54947
ARY14	53994	CASEU	32768	CMPO	176	DELTS	55003
ARY2	53837	CASEURV	33792	CMPST	53294	DFLTIN	153
ARY6	53857	CASI	192	CNTDN	165	DFLT	154
ARYHED	53652	CBINV	790	CNVRTCD	59689	DIM	53377
ARYTAB	47	CBMBASI	49156	COLDBA	58232	DIMFLG	12
ASC	55179	CBMMMSG	58409	COLDST	49152	DIVIDE	56082
ASC18	56702	CCIOUT	65448	COLECT	57490	DIVTEN	56062
ASCFLT	56563	CCLOSE	65475	COLOR	646	DPSW	156
ASCII	215	CGIMAG	58247	COLORMA	37888	DSCPTN	80
ASRRES	55683	CHANNL	19	COLORSE	59666	EAL	174
ATN	58123	CHARAC	7	COLORSY	60082	END	51249
ATNCON	58171	CHARSET	60705	COMCHK	52989	ENDCHR	8
ATOF	56316	CHKAUTO	64831	COMFAC	55623	ERROR	50231
AUTODN	658	CHKIN	62151	COMPAR	53270	ERRTAB	49565
BACKUP	59624	CHKIN	65478	CONT	51287	EVAL	52867
BAD	256	CHKOUT	62217	COPEN	65472	EVALFN	54260
BADSUB	53829	CHKOUT	65481	COS	57953	EVFN3	54351
BASTACK	320	CHR	55020	COUNT	11	EVLVAR	53387
BASVCTR	58447	CHRERR	57870	CREADST	65463	EXP	57325
BASZPT	255	CHRGET	115	CRESTOR	65418	EXPCON	57279
BAUDOF	665	CHRIN	61966	CRNCH	50553	EXPONT	57211
BAUDTBL	65372	CHRIN	65487	CRSW	208	EXTRA	52476
BITCI	168	CHRINRS	62063	CSCNKEY	65439	FA	186
BITNUM	664	CHRINSR	62052	CSECOND	65427	FAC2	105
BITS	104	CHRINTP	62000	CSETLFS	65466	FACOV	112
BITTS	180	CHRINTP2	62032	CSETMSG	65424	FACT10	53005
BLKEND	64518	CHROUT	62074	CSETNAM	65469	FACT12	53032
BLNCT	205	CHROUT	65490	CSETTMO	65442	FACT17	53159
BLNON	207	CHROUTT	62096	CSIO	63659	FACTF1	56266
BLNSW	204	CHRTST	53523	CSTE2	63671	FACTF2	56263
BLOOD	57701	CINV	788	CSTEL	63636	FACTFP	56272
BMSG	49960	CIOUT	61156	CTALK	65460	FAH	53407

ALPHABETICAL CROSS REFERENCE TO MEMORY MAP LABELS

FAT	611	GETBYT	55195	INPPTR	67	LINNUM	20
FBUFPT	113	GETIN	61941	INPUT	52159	LINPTR	60030
FCLOSE	57796	GETIN	65512	INPUTN	52133	LIST	50844
FILEMSG	63358	GETLIN	50528	INSRT	216	LISTEN	60951
FILENAME	63065	GETQUE	58853	INT	56524	LISTI	60956
FILFAC	56553	GETSCRN	58959	INTFLG	14	LNKPRG	50483
FIND2	51462	GETSUB	53682	INTFP	56380	LNMX	213
FINLIN	50707	GONE	51172	INTFP1	56388	LOAD	62786
FINLMR	55137	GOSUB	51331	INTIDX	53674	LOAD	65493
FIRT	164	GOTO	51360	IOBASE	58624	LOADSER	62812
FLP05	57105	GRAPHMO	60720	IOBASE	65523	LOADTP	62929
FLTASC	56797	GRBCOL	54566	IOPEN	794	LODARG	55948
FLTCON	57110	GSINFO	55170	IQPLOP	774	LODFAC	56226
FN	54241	HIBASE	648	IRQ	60095	LOG	55786
FNADR	187	HMSCON	57146	IRQROUT	65394	LOGCON	55745
FNDFLNO	62415	HOME	58753	IRQTMP	671	LOGOKEY	60640
FNDHDR	63591	IBASIN	804	IRQVCTRS	65009	LP2	58831
FNDVAR	53479	IBSOUT	806	ISAVE	818	LPACK	52986
FNLEN	183	ICKIN	798	ISTOP	808	LSTSHF	654
FOPEN	57787	ICKOUT	800	JMPER	84	LSTX	197
FOR	51010	ICLALL	812	JTP20	63626	LXSP	201
FORPNT	73	ICLOSE	796	KEYD	631	M16	54092
FORWARD	59642	ICLRCH	802	KEYLOG	655	M51AJB	661
FOUR6	83	ICRNCH	772	KEYTAB	245	M51CDR	660
FPC1	55740	IERROR	768	KEYVCTR	60486	M51CTR	659
FPC12	56755	IEVAL	778	KMSGSH	61926	MAIN	50304
FPC20	58077	IF	51496	KMSGTBL	61812	MAKADR	55287
FPCTEN	56057	IFCHR	57859	KOUNT	651	MAKFP	54161
FPINT	56475	IGETIN	810	LA	184	MAKINT	53695
FRE	54141	IGETIN	65508	LADIV	56079	MAKSPC	50104
FREEKZP	251	IGONE	776	LAMIN	55376	MAKSTR	54407
FREMSG	58372	IGRERR	52045	LAPLUS	55399	MAKVAR	53533
FRESPEC	53	ILOAD	816	LASTPT	23	MAXINT	53669
FRETOP	51	ILQUAN	53832	LAT	601	MEMBOT	65154
FRMEVL	52638	IMAIN	770	LDAD1	63572	MEMERR	50229
FSBLK	190	INBIT	167	LDTB1	217	MEMHIGH	643
FTOA	56335	INDEX	34	LDTB2	60925	MEMSIZ	55
FUNDSP	49234	INDX	200	LDTND	152	MEMSTR	641
GARBFL	15	INITBA	58276	LDVRMSG	63082	MEMTOP	65139
GCOL13	54717	INITMEM	65909	LEFT	55040	MEMUSS	195
GDBLN	206	INITSK	58648	LEN	55164	MID	55095
GDCOL	647	INITVCTR	58459	LET	51621	MISCMMSG	50020
GET	52091	INITVIA	65017	LET2	51650	MODE	657
GET2RTN	58905	INITVIC	58819	LET5	51674	MOVEBL	50111
GETAD	55275	INPFLG	17	LET9	51756	MOVLINE	59990

ALPHABETICAL CROSS REFERENCE TO MEMORY MAP LABELS

MSGFLG	157	PNT	209	RETVP	53637	RSPREPO	61698
MULDIV	55991	PNTR	211	RFTOA	56332	RSPRTY	61375
MULTEN	56034	POKE	55332	RIBUF	247	RSPRTYE	61597
MYCH	191	POS	54174	RIDATA	170	RSSTAT	663
NDX	198	PRDY	50281	RIDBE	667	RSSTOPS	61416
NEGFAC	57268	PRINT	51872	RIDBS	668	RSSTPBIT	61515
NEW	50754	PRINTN	51840	RIGHT	55084	RSSTRBIT	61544
NEWCH	64475	PRT1	51866	RINONE	169	RTI	65366
NEWLIN	50332	PRT6	51944	RIPRTY	171	RTRN	59608
NEWSTT	51118	PRT7	51960	RND	57492	RUN	51313
NEXT	52510	PRTFIX	56781	RNDC1	57482	RUNTB	60916
NEXTLINE	59587	PRTIN	56770	RNDX	139	RVS	199
NMI	65193	PRTOS	52027	ROBUF	249	SA	185
NMINV	792	PRTSTR	51998	RODATA	182	SAL	172
NODIRM	54182	PRTY	155	RODBE	670	SAREG	780
NORMKEY	60510	PTR2	159	RODBS	669	SAT	621
NORMLZ	55550	PTRI	158	ROPRTY	189	SAVE	57683
NXTBIT	181	PUTSCRN	60074	ROUND	56347	SAVE	63093
OLDLIN	59	QPLOP	50970	RETVP	53637	SAVE	65496
OLDTXT	61	QTSW	212	RFTOA	56332	SAVESER	63122
ON	51531	QUOTECK	59064	RIBUF	247	SAVETP	63217
OPEN	62474	RAMBLK0	1024	RIDATA	170	SAVING	63272
OPENLIN	59886	RAMBLK1	8192	RIDBE	667	SCATN	61125
OPENRS	62663	RAMBLK2	16384	RIDBS	668	SCNKEY	60190
OPMASK	77	RAMBLK3	24576	RPACHK	52983	SCNSTK	50058
OPPTR	75	RAMBLK4	40960	RPTFLG	650	SCREEN	7680
OPTAB	49280	RAMSPC	50184	RSBREAK	61605	SCREEN	65517
ORIOST	65130	RBLK	63689	RSCPTBIT	61479	SCREENX	4096
ORR	53222	RD300	64466	RSDVCR	61625	SCRL	59765
OVERFL	55678	RDTIM	63328	RSFRAME	61608	SCRN	58629
PAREXP	52977	RDTIM	65502	RSINBIT	61494	SCRNOUT	59202
PAROC	57878	RDTPBLK	63680	RSINBYTE	61551	SCROLL	59114
PARSL	57809	READ	52230	RSINERR	61610	SECOND	61220
PATCHBA	57590	READIOST	65128	RSINPRTY	61579	SER2	57430
PATCHER	58486	READST	65111	RSMISSN	61462	SEREVL	57408
PATCHES	58556	READT	63886	RSNMI	65246	SERGET	58546
PCNTR	163	READY	50292	RSNXTBIT	61347	SERNAME	62613
PEEK	55309	REM	51515	RSNXTBY	61422	SEROUT0	58537
PG3FREE	784	RESHO	38	RSNXTIN	61775	SEROUT1	58528
PIVAL	52904	RESLST	49310	RSOPNIN	61718	SETADDR	60014
PLOT	58634	RESTOR	64850	RSOPNOU	61628	SETCHAR	59077
PLOT	65520	RESTOR	65415	RSOUTSA	61677	SETFLCH	62431
PLUS	55402	RESTORE	51229	RSOVERU	61602	SETIODEF	58811
PLUS1	55394	RETREAT	59181	RSPAUSE	61792	SETKEYS	60380
PLUS6	55463	RETURN	51410	RSPREPIN	61531	SETLFS	65104

ALPHABETICAL CROSS REFERENCE TO MEMORY MAP LABELS

SETMSG	65126	SYNPRT	60065	USRPGM8	4608	VIA2T1LL	37158
SETNAM	65097	SYREG	782	USRPOK	0	VIA2T2CH	37161
SETSLINK	58759	SYSTEM	57639	USRVCTR	673	VIA2T2CL	37160
SETTIM	63335	TALK	60948	VAL	55213	VIAIDDRA	37139
SETTIM	65499	TAN	58033	VALTYP	13	VICCR0	36864
SETTMO	65135	TANSNGN	18	VARNAM	69	VICCR1	36865
SFDX	203	TAPE	63732	VARPNT	71	VICCR2	36866
SGN	56377	TAPEH	63463	VARRANG	53012	VICCR3	36867
SGNFAC	56363	TAPEI	178	VARTAB	45	VICCR4	36868
SGNFLG	103	TBLX	214	VCTRIRQ	65534	VICCR5	36869
SHFLAG	653	TBUFFER	828	VCTRNM1	65530	VICCR6	36870
SHFTKEY	60575	TEMPF3	87	VCTRST	65532	VICCR7	36871
SIN	57960	TEMPI	177	VECTOR	64855	VICCR8	36872
SIZE	82	TEMPPT	22	VECTOR	65412	VICCR9	36873
SKIPST	51448	TEMPST	25	VECTORS	64877	VICCRA	36874
SKPCOM	57867	TIME	160	VERCHK	10	VICCRB	36875
SPMSG	61922	TIMES	55848	VERCK	147	VICCRC	36876
SPREG	783	TIMES3	55897	VIA11ER	37150	VICCRD	36877
SQR	57201	TIMOUT	645	VIA11FR	37149	VICCRE	36878
SRBAD	61108	TKSA	61134	VIA1ACR	37147	VICCRF	36879
SRCHING	63047	TNIF	64719	VIA1DDRB	37138	VICINIT	60900
SRCLKHI	61316	TNOFF	64776	VIA1PA1	37137	VPRTY	64776
SRCLKLO	61325	TPBLOCK	829	VIA1PA2	37151	WAIT	55341
SRESEND	61001	TPBUFA	63565	VIA1PB	37136	WAITABIT	61334
STACK	256	TPHBGN	829	VIA1PCR	37148	WARMBAS	58471
STAL	193	TPHDRID	828	VIA1SR	37146	WARMST	49154
START	64802	TPHNAME	831	VIA1T1CL	37140	WBLK	63715
STATUS	144	TPSTORE	64173	VIA1T1LH	37143	WHATKEY	60777
STKEY	145	TPTOGLE	64490	VIA1T1LL	37142	WRAPLIN	60763
STKSPC	50171	TRMPOS	9	VIA1T2CH	37145	WRITE	64523
STMDSP	49164	TSTMEM	65169	VIA1T2CL	37144	WRT62	64785
STOP	63344	TSTOP	63819	VIA1TICH	37141	WRTNI	64661
STOP	65505	TSTSTOP	51244	VIA2ACR	37163	WRTZ	64680
STORFAC	56276	TXTTAB	43	VIA2DDRA	37155	XFERSTR	54906
STR	54373	TYPCHK	52618	VIA2DDRB	37154	XMAX	649
STREND	49	UDTIM	63284	VIA2IER	37166	XSAV	151
STTI	63837	UDTIM	65514	VIA2IFR	37165	ZERFAC	55543
STXTPT	50830	UNDEF	54190	VIA2PA1	37153		
SUB	55379	UNLSN	61188	VIA2PA2	37167		
SUBFLG	16	UNTLK	61156	VIA2PB	37152		
SVXT	146	UNUSDNM	58805	VIA2PCR	37164	PROG AID	28681
SXREG	781	USER	243	VIA2SR	37162	RESET	64802
SYNCHR	52991	USRCMD	814	VIA2T1CH	37157	SCOTT AD	32592
SYNERR	53000	USRCMDS	820	VIA2T1CL	37156	VICMON	24576
SYNO	150	USRPGM3	4096	VIA2T1HL	37159	VICMON	64096

CYCLES AND REGISTERS

	N	V	B	D	I	Z	C	AC	#	Z	ZX	ZY	AB	AX	AY	I	R	,X)),Y	AI
ADC	N	V	-	-	-	Z	C		2	3	4		4	4*	4*			6	5*	
AND	N	-	-	-	-	Z	-		2	3	4		4	4*	4*			6	5*	
ASL	N	-	-	-	-	Z	C	2		5	6		6	7						
BCC	-	-	-	-	-	-	-										2*			
BCS	-	-	-	-	-	-	-										2*			
BEQ	-	-	-	-	-	-	-										2*			
BIT	M7	M6	-	-	-	Z	-			3			4							
BMI	-	-	-	-	-	-	-										2*			
BNE	-	-	-	-	-	-	-										2*			
BPL	-	-	-	-	-	-	-										2*			
BRK	-	-	-	-	1	-	-													
BVC	-	-	-	-	-	-	-										2*			
BVS	-	-	-	-	-	-	-										2*			
CLC	-	-	-	-	-	-	0									2				
CLD	-	-	-	0	-	-	-									2				
CLI	-	-	-	-	0	-	-									2				
CLV	-	0	-	-	-	-	-									2				
CMP	N	-	-	-	-	Z	C		2	3	4		4	4*	4*			6	5*	
CPX	N	-	-	-	-	Z	C		2	3			4							
CPY	N	-	-	-	-	Z	C		2	3			4							
DEC	N	-	-	-	-	Z	-			5	6		6	7						
DEX	N	-	-	-	-	Z	-									2				
DEY	N	-	-	-	-	Z	-									2				
EOR	N	-	-	-	-	Z	-		2	3	4		4	4*	4*			6	5	
INC	N	-	-	-	-	Z	-			5	6		6	7						
INX	N	-	-	-	-	Z	-									2				
INY	N	-	-	-	-	Z	-									2				
JMP	-	-	-	-	-	-	-	(5)					3							
	N	V	B	D	I	Z	C	AC	#	Z	ZX	ZY	AB	AX	AY	I	R	,X)),Y	AI
	BYTES							1	2	2	2	2	3	3	3	1	2	2	2	

	N	V	B	D	I	Z	C	AC	#	Z	ZX	ZY	AB	AX	AY	I	R	,X)),Y	AI	
JSR	-	-	-	-	-	-	-						6								
LDA	N	-	-	-	-	Z	-		2	3	4		4	4*	4*			6	5*		
LDX	N	-	-	-	-	Z	-		2	3		4	4		4*						
LDY	N	-	-	-	-	Z	-		2	3	4		4	4*							
LSR	Ø	-	-	-	-	Z	C	2		5	6		6	7							
NOP	-	-	-	-	-	-	-									2					
ORA	N	-	-	-	-	Z	-		2	3	4		4	4*	4*			6	5*		
PHA	-	-	-	-	-	-	-									3					
PHP	-	-	-	-	-	-	-									3					
PLA	N	-	-	-	-	Z	-									4					
PLP	<i>from stack</i>																4				
ROL	N	-	-	-	-	Z	C	2		5	6		6	7							
ROR	N	-	-	-	-	Z	C	2		5	6		6	7							
RTI	<i>from stack</i>																6				
RTS	-	-	-	-	-	-	-									6					
SBC	N	V	-	-	-	Z	C		2	3	4		4	4*	4*			6	5*		
SEC	-	-	-	-	-	-	1									2					
SED	-	-	-	-	1	-	-									2					
SEI	-	-	-	-	1	-	-									2					
STA	-	-	-	-	-	-	-			3	4		4	5	5			6	6		
STX	-	-	-	-	-	-	-			3		4	4								
STY	-	-	-	-	-	-	-			3	4		4								
TAX	N	-	-	-	-	Z	-									2					
TAY	N	-	-	-	-	Z	-									2					
TSX	N	-	-	-	-	Z	-									2					
TXA	N	-	-	-	-	Z	-									2					
TXS	-	-	-	-	-	-	-									2					
TYA	N	-	-	-	-	Z	-									2					
	N	V	B	D	I	Z	C	AC	#	Z	ZX	ZY	AB	AX	AY	I	R	,X)),Y	AI	
	BYTES							1	2	2	2	2	2	3	3	3	1	2	2	2	

DEC	A	#	ZP	AB	ABX	ABY	ZPX	ZPY	,X)),Y
ADC		105	101	109	125	121	117		97	113
AND		41	37	45	61	57	53		33	50
ASL	10		6	14	30		22			
BIT			36	44						
CMP		201	197	205	221	217	213		193	209
CPX		224	228	236						
CPY		192	196	204						
DEC			198	206	221		214			
EOR		73	69	77	93	89	85		65	81
INC			230	238	253		246			
LDA		169	165	173	189	185	181		161	177
LDX		162	166	174		190		182		
LDY		160	164	172	188		180			
LSR	74		70	78	94		86			
ORA		9	5	13	29	25	21		1	17
ROL	42		38	46	62		54			
ROR	106		102	110	126		118			
SBC		233	229	237	253	249	245		225	241
STA			133	141	157	153	149		129	145
STX			134	142				150		
STY			132	140			148			
	A	#	ZP	AB	ABX	ABY	ZPX	ZPY	,X)),Y
bytes	1	2	2	3	3	3	2	2	2	2

BPL BMI BVC BVS BCC BCS BNE BEO
16 48 80 112 144 176 208 240

TXA TAX TYA TAY TSX TXS
138 170 152 168 186 154

PHP PLP PHA PLA
8 40 72 104

BRK JSR RTI RTS JMP JMP() NOP
0 32 64 96 76 108 234

CLC SEC CLI SEI CLV CLD SED
24 56 88 120 184 216 248

DEY INY DEX INX
136 200 202 232

HEX	A	#	ZP	AB	ABX	ABY	ZPX	ZPY	,X)),Y
ADC		69	65	6D	7D	79	75		61	71
AND		29	25	2D	3D	39	35		21	32
ASL	0A		06	0E	1E		16			
BIT			24	2C						
CMP		C9	C5	CD	DD	D9	D5		C1	D1
CPX		E0	E4	EC						
CPY		C0	C4	CC						
DEC			C6	CE	DE		D6			
EOR		49	45	4D	5D	59	55		41	51
INC			E6	EE	FE		F6			
LDA		A9	A5	AD	BD	B9	B5		A1	B1
LDX		A2	A6	AE		BE		B6		
LDY		A0	A4	AC	BC		B4			
LSR	4A		46	4E	5E		56			
ORA		09	05	0D	1D	19	15		01	11
ROL	2A		26	2E	3E		36			
ROR	6A		66	6E	7E		76			
SBC		E9	E5	ED	FD	F9	F5		E1	F1
STA			85	8D	9D	99	95		81	91
STX			86	8E				96		
STY			84	8C			94			
	A	#	ZP	AB	ABX	ABY	ZPX	ZPY	,X)),Y
bytes	1	2	2	3	3	3	2	2	2	2

BPL 10	BMI 30	BVC 50	BVS 70	BCC 90	BCS B0	BNE D0	BEO F0
TXA 8A	TAX AA	TYA 98	TAY A8	TSX BA	TXS 9A		
PHP 08	PLP 28	PHA 48	PLA 68				
BRK 00	JSR 20	RTI 40	RTS 60	JMP 4C	JMP() 6C	NOP EA	
CLC 18	SEC 38	CLI 58	SEI 78	CLV B8	CLD D8	SED F8	
DEY 88	INY C8	DEX CA	INX E8				

7	6	5	4	3	2	1	0
N	V	1	B	D	I	Z	C
negative	overflow	-	break	decimal	interrupt	zero	carry

ADC

A + M + C → A, C add mem to acc with carry

N	V	D	I	Z	C
√	√	-	-	√	√

#	ADC #oper	69	105	2B	2
ZP	ADC addr	65	101	2B	3
ZPX	ADC addr, X	75	117	2B	4
AB	ADC ADDR	6D	109	3B	4
ABX	ADC ADDR, X	7D	125	3B	4*
ABY	ADC ADDR, Y	79	121	3B	4*
,X)	ADC (addr, X)	61	97	2B	6
,Y	ADC (addr), Y	71	113	2B	5*

*add 1 if page boundary is crossed

2.2.1

AND

A AND M → A and mem with accumulator

N	V	D	I	Z	C
√	-	-	-	√	-

#	AND #oper	29	41	2B	2
ZP	AND addr	25	37	2B	3
ZPX	AND addr, X	35	53	2B	4
AB	AND ADDR	2D	45	3B	4
ABX	AND ADDR, X	3D	61	3B	4*
ABY	AND ADDR, Y	79	121	3B	4*
,X)	AND (addr, X)	61	97	2B	6
,Y	AND (addr), Y	71	113	2B	5*

*add 1 if page boundary is crossed

2.2.3.0

ASL

C ← [76543210] ← 0 arithmetic shift left

N	V	D	I	Z	C
√	-	-	-	√	√

ACC	ASL #oper	0A	10	1B	2
ZP	ASL addr	06	6	2B	5
ZPX	ASL addr, X	16	22	2B	6
AB	ASL ADDR	0E	14	3B	6
ABX	ASL ADDR, X	1E	30	3B	7

10.2

BCC

Branch on C=0 branch on carry clear

N	V	D	I	Z	C
-	-	-	-	-	-

REL	BCC Oper	90	144	2B	2*
-----	----------	----	-----	----	----

*add 1 if branch is to same page

4.1.1.3

*add 2 if branch is to different page

BCS

Branch on C=1 branch on carry set

N	V	D	I	Z	C
-	-	-	-	-	-

REL	BCS Oper	B0	176	2B	2*
-----	----------	----	-----	----	----

*add 1 if branch is to same page

4.1.1.4

*add 2 if branch is to next page

BEQ

Branch on Z=1 branch on result zero

N	V	D	I	Z	C
-	-	-	-	-	-

REL	BEQ Oper	F0	240	2B	2*
*add 1 if branch occurs to same page					4.1.1.5
*add 2 if branch occurs to next page					

BIT

A AND M, M7 → N, M6 → V bit test

N	V	D	I	Z	C
M7	M6	-	-	√	-

ZP	BIT addr	24	36	2B	3
AB	BIT ADDR	2C	44	3B	4
					4.2.1.1

BMI

Branch on N=1 branch on result minus

N	V	D	I	Z	C
-	-	-	-	-	-

REL	BMI Oper	30	48	2B	2*
*add 1 if branch occurs to same page					4.1.1.1
*add 2 if branch occurs to different page					

BNE

Branch on Z=0 branch on result not zero

N	V	D	I	Z	C
-	-	-	-	-	-

REL	BNE Oper	D0	206	2B	2*
*add 1 if branch occurs to same page					4.1.1.6
*add 2 if branch occurs to different page					

BPL

Branch on N=0 branch on result plus

N	V	D	I	Z	C
-	-	-	-	-	-

REL	BPL Oper	10	16	2B	2*
*add 1 if branch occurs to same page					4.1.1.2
*add 2 if branch occurs to different page					

BRK

interrupt PC+2 ↓ SR ↓ force break

N	V	D	I	Z	C
-	-	-	1	-	-

IMPL	BRK	00	0	1B	7
A BRK command cannot be masked by setting 1					9.11

BVC

Branch on V=0 branch on overflow clear

N	V	D	I	Z	C
-	-	-	-	-	-

REL	BVC Oper	50	80	2B	2*
*add 1 if branch occurs to same page					4.1.1.8
*add 2 if branch occurs to different page					

BVS

Branch on V=1 branch on overflow set

N	V	D	I	Z	C
-	-	-	-	-	-

REL	BVS Oper	70	112	2B	2*
*add 1 if branch occurs to same page					4.1.1.7
*add 2 if branch occurs to different page					

CLC

0 → C clear carry flag

N	V	D	I	Z	C
-	-	-	-	-	0

IMPL	CLC	18	24	1B	2
					3.0.2

CLD $\emptyset \rightarrow D$ clear decimal mode

N	V	D	I	Z	C
-	-	\emptyset	-	-	-

IMPL	CLD	D8	216	1B	2
					3.3.2

CLI $\emptyset \rightarrow I$ clear interrupt disable bit

N	V	D	I	Z	C
-	-	-	\emptyset	-	-

IMPL	CLI	58	88	1B	2
					3.2.2

CLV $\emptyset \rightarrow V$ clear overflow flag

N	V	D	I	Z	C
-	\emptyset	-	-	-	-

IMPL	CLV	B8	184	1B	2
					3.6.1

CMP

A - M compare with accumulator

N	V	D	I	Z	C
\checkmark	-	-	-	\checkmark	\checkmark

#	CMP #oper	C9	201	2B	2
ZP	CMP addr	C5	197	2B	3
ZPX	CMP addr, X	D5	213	2B	4
AB	CMP ADDR	CD	205	3B	4
ABX	CMP ADDR, X	DD	221	3B	4*
ABY	CMP ADDR, Y	D9	217	3B	4*
,X)	CMP (addr, X)	C1	193	2B	6
,Y	CMP (addr), Y	D1	209	2B	5*

*add 1 if page boundary is crossed

4.2.1

CPX

X - M compare with X

N	V	D	I	Z	C
\checkmark	-	-	-	\checkmark	\checkmark

#	CPX #oper	E0	224	2B	2
ZP	CPX addr	E4	228	2B	3
AB	CPX ADDR	EC	236	3B	4
					7.8

CPY

Y - M compare with Y

N	V	D	I	Z	C
\checkmark	-	-	-	\checkmark	\checkmark

#	CPY #oper	C0	192	2B	2
ZP	CPY addr	C4	196	2B	3
AB	CPY ADDR	CC	204	3B	4
					7.9

DECM - 1 \rightarrow M decrement memory by one

N	V	D	I	Z	C
\checkmark	-	-	-	\checkmark	-

ZP	CMP addr	C6	198	2B	5
ZPX	CMP addr, X	D6	214	2B	6
AB	CMP ADDR	CE	206	3B	6
ABX	CMP ADDR, X	DE	222	3B	7
					10.7

DEX

X - 1 → X

decrement index X by one

N	V	D	I	Z	C
√	-	-	-	√	-

IMPL	DEX	CA	202	1B	2
					7.6

DEY

Y - 1 → Y

decrement index Y by one

N	V	D	I	Z	C
√	-	-	-	√	-

IMPL	DEY	88	136	1B	2
					7.7

EOR

A EOR M → A

exclusive or with accumulator

N	V	D	I	Z	C
√	-	-	-	√	-

#	EOR #oper	49	73	2B	2
ZP	EOR addr	45	69	2B	3
ZPX	EOR addr, X	55	85	2B	4
AB	EOR ADDR	4D	77	3B	4
ABX	EOR ADDR, X	5D	93	3B	4*
ABY	EOR ADDR, Y	59	89	3B	4*
,X)	EOR (addr, X)	41	86	2B	6
,Y	EOR (addr), Y	51	81	2B	5*

*add 1 if page boundary is crossed

2.2.3.2

INC

M + 1 → M

increment memory by one

N	V	D	I	Z	C
√	-	-	-	√	-

ZP	INC addr	E6	230	2B	5
ZPX	INC addr, X	F6	246	2B	6
AB	INC ADDR	EE	238	3B	6
ABX	INC ADDR, X	FE	254	3B	7

10.6

INX

X + 1 → X

increment index X by one

N	V	D	I	Z	C
√	-	-	-	√	-

IMPL	INX	E8	232	1B	2
					7.4

INY

Y + 1 → Y

increment index Y by one

N	V	D	I	Z	C
√	-	-	-	√	-

IMPL	INY	C8	200	1B	2
					7.5

JMP

(PC+1) → PCL, (PC+2) → PCH

jump to new location

N	V	D	I	Z	C
-	-	-	-	-	-

AB	JMP ADDR	4C	76	3B	3
INDRCT	JMP (ADDR)	6C	108	3B	5

4.0.2

9.8.1

JSR

(PC+2) ↓, (PC+1)→PCL, (PC+2)→PCH jump save return

N	V	D	I	Z	C
-	-	-	-	-	-

AB	JSR ADDR	20	32	3B	6
					8.1

LDA

M → A load accumulator with memory

N	V	D	I	Z	C
√	-	-	-	√	-

#	LDA #oper	A9	169	2B	2
ZP	LDA addr	A5	165	2B	3
ZPX	LDA addr, X	B5	181	2B	4
AB	LDA ADDR	AD	173	3B	4
ABX	LDA ADDR, X	BD	189	3B	4*
ABY	LDA ADDR, Y	B9	185	3B	4*
,X)	LDA (addr, X)	A1	161	2B	6
,Y	LDA (addr), Y	B1	177	2B	5*

*add 1 if page boundary is crossed

2.1.1

LDX

M → X load index X with memory

N	V	D	I	Z	C
√	-	-	-	√	-

#	LDX #oper	A2	162	2B	2
ZP	LDX addr	A6	166	2B	3
ZPX	LDX addr, X	B6	182	2B	4
AB	LDX ADDR	AE	174	3B	4
ABX	LDX ADDR, X	BE	190	3B	4*

*add 1 if page boundary is crossed

7.0

LDY

M → Y load index Y with memory

N	V	D	I	Z	C
√	-	-	-	√	-

#	LDY #oper	A0	160	2B	2
ZP	LDY addr	A4	164	2B	3
ZPX	LDY addr, X	B4	180	2B	4
AB	LDY ADDR	AC	172	3B	4
ABX	LDY ADDR, X	BC	188	3B	4*

*add 1 if page boundary is crossed

7.1

LSR

0 → [76543210] → C logical shift right

N	V	D	I	Z	C
0	-	-	-	√	√

ACC	LSR #oper	4A	74	1B	2
ZP	LSR addr	46	70	2B	5
ZPX	LSR addr, X	56	86	2B	6
AB	LSR ADDR	4E	78	3B	6
ABX	LSR ADDR, X	5E	94	3B	7

10.1

NOP

NO OPERATION no operation

N	V	D	I	Z	C
-	-	-	-	-	-

IMPL	NOP	EA	234	1B	2
2 cycles					---

ORA

N	V	D	I	Z	C
√	-	-	-	√	-

A OR M → A

OR with accumulator

#	ORA #oper	09	9	2B	2
ZP	ORA addr	05	5	2B	3
ZPX	ORA addr, X	15	21	2B	4
AB	ORA ADDR	0D	13	3B	4
ABX	ORA ADDR, X	1D	29	3B	4*
ABY	ORA ADDR, Y	19	25	3B	4*
,X)	ORA (addr, X)	01	1	2B	6
,Y	ORA (addr), Y	11	17	2B	5*

*add 1 on page crossing

2.2.3.1

PHA

N	V	D	I	Z	C
-	-	-	-	-	-

A ↓

push proc status on stack

IMPL	PHA	48	72	1B	3
					8.5

PHP

N	V	D	I	Z	C
-	-	-	-	-	-

P ↓

push accumulator from stack

IMPL	PHP	08	8	1B	3
					8.11

PLA

N	V	D	I	Z	C
√	-	-	-	√	-

A ↑

pull accumulator on stack

IMPL	PLA	68	104	1B	4
					8.6

PLP

N	V	D	I	Z	C
<i>from stack</i>					

P ↑

pull proc status from stack

IMPL	PLP	28	40	1B	4
					8.12

ROL

N	V	D	I	Z	C
√	-	-	-	√	√

C ← [76543210] ← C

rotate one bit left

ACC	ROL #oper	2A	42	1B	2
ZP	ROL addr	26	38	2B	5
ZPX	ROL addr, X	36	54	2B	6
AB	ROL ADDR	2E	46	3B	6
ABX	ROL ADDR, X	3E	62	3B	7

10.3

ROR

N	V	D	I	Z	C
√	-	-	-	√	√

C → [76543210] → C

rotate one bit right

ACC	ROR #oper	6A	106	1B	2
ZP	ROR addr	66	102	2B	5
ZPX	ROR addr, X	76	118	2B	6
AB	ROR ADDR	6E	110	3B	6
ABX	ROR ADDR, X	7E	126	3B	7

10.4

RTI

N	V	D	I	Z	C
from stack					

P ↑ PC ↑ return from interrupt

IMPL	RTI	40	64	1B	6
					9.6

RTS

N	V	D	I	Z	C
-	-	-	-	-	-

PC ↑, PC + 1 → PC return from subroutine

IMPL	RTS	60	96	1B	6
					8.2

SBC

N	V	D	I	Z	C
√	√	-	-	√	√

A - M - \bar{C} → A subtract mem from acc w/ borrow

#	SBC #oper	E9	233	2B	2
ZP	SBC addr	E5	229	2B	3
ZPX	SBC addr, X	F5	245	2B	4
AB	SBC ADDR	ED	237	3B	4
ABX	SBC ADDR, X	FD	253	3B	4*
ABY	SBC ADDR, Y	F9	249	3B	4*
,X)	SBC (addr, X)	E1	225	2B	6
,Y	SBC (addr), Y	F1	241	2B	5*
					2.2.2

SEC

N	V	D	I	Z	C
-	-	-	-	-	1

1 → C set carry flag

IMPL	SEC	38	56	1B	2
					3.0.1

SED

N	V	D	I	Z	C
-	-	1	-	-	-

1 → D set decimal mode

IMPL	SEC	F8	248	1B	2
					3.3.1

SEI

N	V	D	I	Z	C
-	-	-	1	-	-

1 → I set interrupt disable status

IMPL	SEC	78	120	1B	2
					3.2.1

STA

N	V	D	I	Z	C
-	-	-	-	-	-

A → M store accumulator in memory

ZP	STA addr	85	133	2B	3
ZPX	STA addr, X	95	149	2B	4
AB	STA ADDR	8D	141	3B	4
ABX	STA ADDR, X	9D	157	3B	5
ABY	STA ADDR, Y	99	153	3B	5
,X)	STA (addr, X)	81	129	2B	6
,Y	STA (addr), Y	91	145	2B	6
					2.1.2

STX

X → M

store index X in memory

N	V	D	I	Z	C
-	-	-	-	-	-

ZP	STX addr	86	134	2B	3
ZPX	STX addr, X	96	150	2B	4
AB	STX ADDR	8E	142	3B	4

7.2

STY

Y → M

store index Y in memory

N	V	D	I	Z	C
-	-	-	-	-	-

ZP	STY addr	84	132	2B	3
ZPX	STY addr, X	94	148	2B	4
AB	STY ADDR	8C	140	3B	4

7.3

TAX

A → X

transfer accumulator to X

N	V	D	I	Z	C
√	-	-	-	√	-

IMPL	TAX	AA	170	1B	2
------	-----	----	-----	----	---

7.11

TAY

A → Y

transfer accumulator to Y

N	V	D	I	Z	C
√	-	-	-	√	-

IMPL	TAY	A8	168	1B	2
------	-----	----	-----	----	---

7.13

TSX

S → X

transfer stack pointer to X

N	V	D	I	Z	C
√	-	-	-	√	-

IMPL	TSX	BA	186	1B	2
------	-----	----	-----	----	---

8.9

TXA

X → A

transfer X to accumulator

N	V	D	I	Z	C
√	-	-	-	√	-

IMPL	TXA	8A	138	1B	2
------	-----	----	-----	----	---

7.12

TXS

X → S

transfer X to stack pointer

N	V	D	I	Z	C
-	-	-	-	-	-

IMPL	TXS	98	152	1B	2
------	-----	----	-----	----	---

8.8

TYA

Y → A

transfer Y to accumulator

N	V	D	I	Z	C
√	-	-	-	√	-

IMPL	TYA	9A	154	1B	2
------	-----	----	-----	----	---

7.14

UNDOCUMENTED OPCODES

ALR

AND op + LSR also known as ASR

N	V	D	I	Z	C
√	-	-	-	√	√

#	ALR #oper	4B	75	2B	2
---	-----------	----	----	----	---

ANC

AND op + set C as ASL

N	V	D	I	Z	C
√	-	-	-	√	√

#	ANC #oper	0B	11	2B	2
---	-----------	----	----	----	---

ANC2

AND op + set C as ROL

N	V	D	I	Z	C
√	-	-	-	√	√

#	ANC #oper	2B	43	2B	2
---	-----------	----	----	----	---

ANE

AND X + AND op also known as XAA

N	V	D	I	Z	C
√	-	-	-	√	√

#	ANE #oper	8B	139	2B	2
---	-----------	----	-----	----	---

ARR

AND op + ROR

N	V	D	I	Z	C
√	-	-	-	√	√

#	ARR #oper	6B	107	2B	2
---	-----------	----	-----	----	---

DCP

DEC op + CMP op also known as DCM

N	V	D	I	Z	C
√	-	-	-	√	√

ZP	DCP addr	C7	119	2B	5
ZPX	DCP addr, X	D7	215	2B	6
AB	DCP ADDR	CF	207	3B	6
ABX	DCP ADDR, X	DF	223	3B	7
ABY	DCP ADDR, Y	DB	219	3B	7
,X)	DCP (addr, X)	C3	195	2B	8
,Y	DCP (addr), Y	D3	211	2B	8

ISC

INC op + SBC op also known as ISB, INS

N	V	D	I	Z	C
√	-	-	-	√	√

ZP	ISC addr	E7	231	2B	5
ZPX	ISC addr, X	F7	247	2B	6
AB	ISC ADDR	EF	239	3B	6
ABX	ISC ADDR, X	FF	255	3B	7
ABY	ISC ADDR, Y	FB	251	3B	7
,X)	ISC (addr, X)	E3	227	2B	8
,Y	ISC (addr), Y	F3	243	2B	4

LAS

LDA / TSX op

also known as LAR

N	V	D	I	Z	C
√	-	-	-	√	-

ABY	LAS ADDR, Y	BB	187	3B	4*
-----	-------------	----	-----	----	----

LAX

LDA op + LDX op

N	V	D	I	Z	C
√	-	-	-	√	-

ZP	LAX addr	A7	167	2B	3
ZPX	LAX addr, X	B7	183	2B	4
AB	LAX ADDR	AF	175	3B	4
ABY	LAX ADDR, Y	BF	191	3B	4*
,X)	LAX (addr, X)	A3	163	2B	6
,Y)	LAX (addr), Y	B3	179	2B	5*

LXA

AND op, LSR

also known as LAX immediate

N	V	D	I	Z	C
√	-	-	-	√	√

#	LXA #oper	AB	171	2B	2
---	-----------	----	-----	----	---

RLA

ROL op + AND op

N	V	D	I	Z	C
√	-	-	-	√	√

ZP	RLA addr	27	39	2B	5
ZPX	RLA addr, X	37	55	2B	6
AB	RLA ADDR	2F	47	3B	6
ABX	RLA ADDR, X	3F	63	3B	7
ABY	RLA ADDR, Y	3B	59	3B	7
,X)	RLA (addr, X)	23	35	2B	8
,Y)	RLA (addr), Y	33	51	2B	8

RRA

ROR op + ADC op

N	V	D	I	Z	C
√	√	-	-	√	√

ZP	RRA addr	67	103	2B	5
ZPX	RRA addr, X	77	119	2B	6
AB	RRA ADDR	6F	111	3B	6
ABX	RRA ADDR, X	7F	127	3B	7
ABY	RRA ADDR, Y	7B	123	3B	7
,X)	RRA (addr, X)	63	99	2B	8
,Y)	RRA (addr), Y	73	115	2B	8

SAX

A AND X → M

also known as AXS, AAX

N	V	D	I	Z	C
-	-	-	-	-	-

ZP	SAX addr	87	135	2B	3
ZPX	SAX addr, X	97	151	2B	4
AB	SAX ADDR	8F	143	3B	4
ABX	SAX ADDR, X	83	131	2B	6

SBX

CMP and DEX at once, set flags known as AXS, SAX

N	V	D	I	Z	C
√	-	-	-	√	√

#	SBX #oper	CB	203	2B	2
---	-----------	----	-----	----	---

SHA

A AND X AND (H + 1) → M A known as AHX, AXA

N	V	D	I	Z	C
-	-	-	-	-	-

ABY	SHA ADDR, Y	9F	159	3B	5
), Y	SHA (addr), X	93	147	2B	6

SHX

X AND (H + 1) → M also known as All, SXA, XAS

N	V	D	I	Z	C
-	-	-	-	-	-

ABY	SHX ADDR, Y	9E	158	3B	5
-----	-------------	----	-----	----	---

SHY

Y AND (H + 1) → M also known as All, SYA, SAY

N	V	D	I	Z	C
-	-	-	-	-	-

ABX	SHY ADDR, X	9C	156	3B	5
-----	-------------	----	-----	----	---

SLO

ASL op + ORA op also known as ASO

N	V	D	I	Z	C
√	-	-	-	√	√

ZP	SLO addr	07	7	2B	5
ZPX	SLO addr, X	17	23	2B	6
AB	SLO ADDR	0F	15	3B	6
ABX	SLO ADDR, X	1F	31	3B	7
ABY	SLO ADDR, Y	1B	27	3B	7
,X)	SLO (addr, X)	03	3	2B	8
), Y	SLO (addr), Y	13	19	2B	8

SRE

LSR op + EOR op also known as LSE

N	V	D	I	Z	C
√	-	-	-	√	√

ZP	SRE addr	47	71	2B	5
ZPX	SRE addr, X	57	87	2B	6
AB	SRE ADDR	4F	79	3B	6
ABX	SRE ADDR, X	5F	95	3B	7
ABY	SRE ADDR, Y	5B	91	3B	7
,X)	SRE (addr, X)	43	67	2B	8
), Y	SRE (addr), Y	53	83	2B	8

TAS

A AND X → SP, A AND X AND (H + 1) → M XAS, SHS

N	V	D	I	Z	C
-	-	-	-	-	-

ABY	TAS ADDR, Y	9B	155	3B	5
-----	-------------	----	-----	----	---

USBC

SBC oper + NOP also known as SBC

N	V	D	I	Z	C
√	-	-	-	√	√

#	USBC #oper	EB	235	2B	2
---	------------	----	-----	----	---

NOP

NO OPERATION

also known as DOP, TOP

N	V	D	I	Z	C
-	-	-	-	-	-

IMPL		1A	26	1B	2
	3A 5A 7A DA FA				
#		80	128	2B	2
	82 89 C2 E2				
ZP		04	4	2B	3
	44 64				
ZPX		14	14	2B	4
	34 54 74 D4 F4				
AB		0C	12	3B	4
ABX		1C	28	3B	4*
	3C 5C 7C DC FC				

JAM

freezes CPU

also known as KIL, HLT

	02	2
	12	18
	22	34
	32	50
	42	66
	52	82
	62	98
	72	114
	92	146
	B2	178
	D2	210
	FE	254

RELATIVE BRANCH BACKWARD

8_	128	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113
9_	112	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97
A_	96	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81
B_	80	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65
C_	64	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49
D_	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33
E_	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
F_	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

00	0	0	20	32	8192	40	64	16384	60	96	24576
01	1	256	21	33	8448	41	65	16640	61	97	24832
02	2	512	22	34	8704	42	66	16896	62	98	25088
03	3	768	23	35	8960	43	67	17152	63	99	25344
04	4	1024	24	36	9216	44	68	17408	64	100	25600
05	5	1280	25	37	9472	45	69	17664	65	101	25856
06	6	1536	26	38	9728	46	70	17920	66	102	26112
07	7	1792	27	39	9984	47	71	18176	67	103	26368
08	8	2048	28	40	10240	48	72	18432	68	104	26624
09	9	2304	29	41	10496	49	73	18688	69	105	26880
0A	10	2560	2A	42	10752	4A	74	18944	6A	106	27136
0B	11	2816	2B	43	11008	4B	75	19200	6B	107	27392
0C	12	3072	2C	44	11264	4C	76	19456	6C	108	27648
0D	13	3328	2D	45	11520	4D	77	19712	6D	109	27904
0E	14	3584	2E	46	11776	4E	78	19968	6E	110	28160
0F	15	3840	2F	47	12032	4F	79	20224	6F	111	28416
10	16	4096	30	48	12288	50	80	20480	70	112	28672
11	17	4352	31	49	12544	51	81	20736	71	113	28928
12	18	4608	32	50	12800	52	82	20992	72	114	29184
13	19	4864	33	51	13056	53	83	21248	73	115	29440
14	20	5120	34	52	13312	54	84	21504	74	116	29696
15	21	5376	35	53	13568	55	85	21760	75	117	29952
16	22	5632	36	54	13824	56	86	22016	76	118	30208
17	23	5888	37	55	14080	57	87	22272	77	119	30464
18	24	6144	38	56	14336	58	88	22528	78	120	30720
19	25	6400	39	57	14592	59	89	22784	79	121	30976
1A	26	6656	3A	58	14848	5A	90	23040	7A	122	31232
1B	27	6912	3B	59	15104	5B	91	23296	7B	123	31488
1C	28	7168	3C	60	15360	5C	92	23552	7C	124	31744
1D	29	7424	3D	61	15616	5D	93	23808	7D	125	32000
1E	30	7680	3E	62	15872	5E	94	24064	7E	126	32256
1F	31	7936	3F	63	16128	5F	95	24320	7F	127	32512

80	128	32768	A0	160	40960	C0	192	49152	E0	224	57344
81	129	33024	A1	161	41216	C1	193	49408	E1	225	57600
82	130	33280	A2	162	41472	C2	194	49664	E2	226	57856
83	131	33536	A3	163	41728	C3	195	49920	E3	227	58112
84	132	33792	A4	164	41984	C4	196	50176	E4	228	58368
85	133	34048	A5	165	42240	C5	197	50432	E5	229	58624
86	134	34304	A6	166	42496	C6	198	50688	E6	230	58880
87	135	34560	A7	167	42752	C7	199	50944	E7	231	59136
88	136	34816	A8	168	43008	C8	200	51200	E8	232	59392
89	137	35072	A9	169	43264	C9	201	51456	E9	233	59648
8A	138	35328	AA	170	43520	CA	202	51712	EA	234	59904
8B	139	35584	AB	171	43776	CB	203	51968	EB	235	60160
8C	140	35840	AC	172	44032	CC	204	52224	EC	236	60416
8D	141	36096	AD	173	44288	CD	205	52480	ED	237	60672
8E	142	36352	AE	174	44544	CE	206	52736	EE	238	60928
8F	143	36608	AF	175	44800	CF	207	52992	EF	239	61184
90	144	36864	B0	176	45056	D0	208	53248	F0	240	61440
91	145	37120	B1	177	45312	D1	209	53504	F1	241	61696
92	146	37376	B2	178	45568	D2	210	53760	F2	242	61952
93	147	37632	B3	179	45824	D3	211	54016	F3	243	62208
94	148	37888	B4	180	46080	D4	212	54272	F4	244	62464
95	149	38144	B5	181	46336	D5	213	54528	F5	245	62720
96	150	38400	B6	182	46592	D6	214	54784	F6	246	62976
97	151	38656	B7	183	46848	D7	215	55040	F7	247	63232
98	152	38912	B8	184	47104	D8	216	55296	F8	248	63488
99	153	39168	B9	185	47360	D9	217	55552	F9	249	63744
9A	154	39424	BA	186	47616	DA	218	55808	FA	250	64000
9B	155	39680	BB	187	47872	DB	219	56064	FB	251	64256
9C	156	39936	BC	188	48128	DC	220	56320	FC	252	64512
9D	157	40192	BD	188	48384	DD	221	56576	FD	253	64768
9E	158	40448	BE	190	48640	DE	222	56832	FE	254	65024
9F	159	40704	BF	191	48896	DF	223	57088	FF	255	65536

				stack	error
ACPTR	FFA5	65445	Input byte from serial port	13	READST
CHKIN	FFC6	65478	Open channel for input	-	3,5,6
CHKOUT	FFC9	65481	Open channel for output	-	3,5,7
CHRIN	FFCF	65487	Input character from channel	-	READST
CHROUT	FFD2	65490	Output character to channel	-	READST
CIOUT	FFA8	65448	Output byte to serial port	-	READST
CLALL	FFE7	65511	Close all channels and files	11	none
CLOSE	FFC3	65475	Close a specified logical file	-	none
CLRCHN	FFCC	65484	Restore default I/O devices	9	none
GETIN	FFE4	65512	Get char from current input dev	-	none
IOBASE	FFF3	65523	Return base address of I/O device	2	none
LISTEN	FFB1	65457	Command devices on serial to listen	-	READST
LOAD	FFD5	65493	Load (A=0) Verify (A=1) ram	-	0,4,5,8,9
MEMBOT	FF9C	65436	Read (C=1) Set(C=0) bottom of memory	-	none
MEMTOP	FF99	65433	Read (C=1) Set(C=0) top of memory	2	none
OPEN	FFC0	65472	Open a logical file	-	1,2,4,5,6
PLOT	FFF0	65520	Read (C=1) Set(C=0) x y cursor pos	2	none
RAMTAS	FF87	65415	Init RAM, allocate tape buff, set screen	2	none
RDTIM	FFDE	65502	Read real time clock	2	none
READST	FF87	65463	Read I/O status word	2	none
RESTOR	FF8A	65415	Restore default I/O vectors	2	none
SAVE	FFD8	65496	Save RAM to device from \$2B to .X, .Y	-	5,8,9
SCNKEY	FF9F	65439	Scan keyboard	-	none
SCREEN	FFED	65517	Return screen size in rows columns	2	none
SECOND	FF93	65427	Send secondary address after LISTEN	-	READST
SETLFS	FFBA	65466	Set logical, first, and second address	2	none
SETMSG	FF90	65424	Enable/disable KERNAL messages	2	none
SETNAM	FFBD	65469	Set file name	-	none
SETTIM	FFD8	65499	Set real time clock	2	none
SETTMO	FFA2	65442	Set (A<#128) Reset (A>#127) timeout	2	none
STOP	FFE1	65505	Scan STOP key	-	none
TALK	FFB4	65460	Command serial to TALK	-	READST
TKSA	FF96	65430	Send secondary address after TALK	-	READST
UDTIM	FFEA	65514	Update (increment) real time clock	2	none
UNLSN	FFAE	65454	Command serial to UNLISTEN	-	READST
UNTLK	FFAB	65451	Command serial to UNTALK	-	READST
VECTOR	FF8D	65412	Store (C=1) Restore (C=0) vectors	2	none

		IN			OUT		
		.A	.X	.Y	.A	.X	.Y
ACPTR	FFA5				data	alt	
CHKIN	FFC6		LF#		alt		
CHKOUT	FFC9		LF#		alt		
CHRIN	FFCF				data	alt	
CHROUT	FFD2	data					
CIOUT	FFA8	data					
CLALL	FFE7				alt	alt	
CLOSE	FFC3	LF#			alt	alt	alt
CLRCHN	FFCC				alt	alt	
GETIN	FFE4				data	alt	alt
IOBASE	FFF3					addr lo	addr hi
LISTEN	FFB1	DEV#					
LOAD	FFD5	load/ver	start lo	start hi		end lo +1	end hi
MEMBOT	FF9C	C=0	bot lo	bot hi	C=1	bot lo	bot hi
MEMTOP	FF99	C=0	top lo	top hi	C=1	top lo	top hi
OPEN	FFC0				alt	alt	alt
PLOT	FFF0	C=0	row	col	C=1	row	col
RAMTAS	FF87				alt	alt	alt
RDTIM	FFDE				msb	msb2	lsb
READST	FF87				ST		
RESTOR	FF8A				alt	alt	alt
SAVE	FFD8	#	end lo	end hi		end lo +1	end hi
SCNKEY	FF9F				alt	alt	alt
SCREEN	FFED					#rows	#cols
SECOND	FF93	SA/\$60					
SETLFS	FFBA	LF#	DEV#	SA			
SETMSG	FF90	.A val \$40 control msgs on, \$80 error msgs on.					
SETNAM	FFBD	len	addr lo	addr hi			
SETTIM	FFD8	msb	msb2	lsb			
SETTMO	FFA2						
STOP	FFE1	yes: .Z = 1 no: .A = last row keyboard scan					
TALK	FFB4	DEV#					
TKSA	FF96	SA/\$60					
UDTIM	FFE4				alt	alt	
UNLSN	FFAE				alt		
UNTLK	FFAB				alt		
VECTOR	FF8D	C=1	table lo	table hi	C=0	table lo	table hi

KEYBOARD		\$00C5	\$028D
KEY	PEEK(197)	PEEK(203)	
SHIFT C= CTRL	PEEK(653)	1 SHIFT	2 C= 4 CTRL

0	1	12	I	26	X	39	F1	52	O
1	3	13	P	27	V	41	S	53	@
2	5	14	*	28	N	42	F	54	↑
3	7	15	Rt	29	,	43	H	55	F5
4	9	17	A	30	/	44	K	56	2
5	+	18	D	31	↕	45	:	57	4
6	£	19	G	32	Sp	46	=	58	6
7	De	20	J	33	Z	47	F3	59	8
8	←	21	L	34	C	48	Q	60	0
9	W	22	;	35	B	49	E	61	-
10	R	23	↔	36	M	50	T	62	Ho
11	Y	24	St	37	.	51	U	63	F7

←	1	2	3	4	5	6	7	8	9	0	+	-	£	cl	de	F1
8	0	56	1	57	2	58	3	59	4	60	5	61	6	62	7	39

Q	W	E	R	T	Y	U	I	O	P	@	*	↑	F3
48	9	49	10	50	11	51	12	52	13	53	14	54	47

RUN STOP	A	S	D	F	G	H	J	K	L	:	;	=	return	F5
24	17	41	18	42	19	43	20	44	21	45	22	46	15	55

Z	X	C	V	B	N	M	,	.	/	↕	↔	F7
33	26	34	27	35	28	36	29	37	30	31	23	63

space
32

no key
64

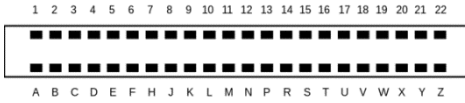
KEYBOARD MATRIX			\$9120	\$9121
column	write to Port B	37152	\$9120	
row	read from Port A	37153	\$9121	

	7	6	5	4	3	2	1	0
7	F7	F5	F3	F1	↕	↔	RET	DEL
6	HOME	↑	=	RIGHT SHIFT	/	;	*	£
5	-	@	:	.	,	L	P	+
4	0	O	K	M	N	J	I	9
3	8	U	H	B	V	G	Y	7
2	6	T	F	C	X	D	R	5
1	4	E	S	Z	LEFT SHIFT	A	W	3
0	2	Q	C=	SPC	STOP	CTRL	←	1

USEFUL KEYBOARD RAM LOCATIONS			
145	bottom keyboard row scan	649	size of keyboard buffer
197	current key pressed	650	keyboard repeater flag
198	number of chars in buffer	651	repeat delay after first key
203	current key pressed	652	delay before first repeat
204	cursor blink switch	653	current SHIFT key pattern
208	flag: screen or keyboard	654	previous SHIFT key pattern
212	flag: quote marks	655	pointer to key table setup
216	numb of outstanding inserts	657	flag: enable SHIFT CMD key
245	pointer to keyboard table		

USEFUL WAITS			
WAIT 197,64,64	any key press	WAIT 653,1	shift key press
WAIT 197,64	all keys released	WAIT 653,1,1	shift key release
WAIT 198,8	8 keys in buffer	WAIT 653,2	C= key press
WAIT 37151,4,4	joystick UP	WAIT 653,2,2	C= key release
WAIT 37151,4	UP release	WAIT 653,4	CTRL key press
WAIT 37151,32,32	joystick FIRE	WAIT 653,4,4	CTRL key release
WAIT 37151,32	FIRE release		

EXPANSION PORT



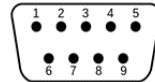
1	GND	system ground	A	GND	system ground
2	CD0	data line 0	B	CA0	address line 0
3	CD1	data line 1	C	CA1	address line 1
4	CD2	data line 2	D	CA2	address line 2
5	CD3	data line 3	E	CA3	address line 3
6	CD4	data line 4	F	CA4	address line 4
7	CD5	data line 5	H	CA5	address line 5
8	CD6	data line 6	J	CA6	address line 6
9	CD7	data line 7	K	CA7	address line 7
10	/BLK 1	Memory block 1 - \$2000	L	CA8	address line 8
11	/BLK 2	Memory block 2 - \$4000	M	CA9	address line 9
12	/BLK 3	Memory block 3 - \$6000	N	CA10	address line 10
13	/BLK 5	Memory block 5 - \$A000	P	CA11	address line 11
14	/RAM 1	1K RAM 1 - \$0400	R	CA12	address line 12
15	/RAM 2	1K RAM 2 - \$0800	S	CA13	address line 13
16	/RAM 3	1K RAM 3 - \$0C00	T	/I/O2	input/output 2 - \$9800
17	V R/W	VIC HI read - / LO write	U	/I/O3	input/output 3 - \$9C00
18	C R/W	CPU HI read - / LO write	V	SØ2	S Phi 2
19	/IRQ	interrupt request	W	/NMI	non maskable interrupt
20	NC	not connected	X	/RESET	reset pin
21	+5V DC	supply voltage +5V DC	Y	NC	not connected
22	GND	system ground	Z	GND	system ground

CASSETTE PORT



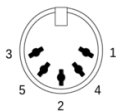
A-1	GND	Ground
B-2	+ 5V	5 Volt DC
C-3	MOTOR	6 Volt motor CA2
D-4	READ	Data Input, Read CA1
E-5	WRITE	Data Output, Write PB3
F-6	SENSE	Key press detection PA6

GAME PORT



1	VIA #1 PA2 (JOY 0 up)
2	VIA #1 PA3 (JOY 1 down)
3	VIA #1 PA4 (2 left), paddle Left fire
4	VIA #2 PB7 (3 right), paddle Right fire
5	POT Y
6	VIA #2 PA7 (joy button), (light pen)
7	+5V (Max. 100mA)
8	GND
9	POT X

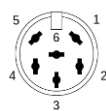
AUDIO / VIDEO



1	+5V REG (euro +6V DC 10ma)	RED
2	GND	
3	Audio	GREY
4	Video Low (composite)	BLACK
5	Video High (composite)	WHITE

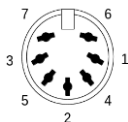
Radio Shack Part 42-2394

SERIAL I/O 6 pin DIN



1	SRQ IN
2	GND
3	VIA #1 PA7, ATN out
4	VIA #2 CA2, Clock in/out
5	VIA #2 CB2, DATA in/out
6	NC (Reset)

CR MODEL POWER



1	GND (5 Volt)
2	GND (5 Volt)
3	GND (5 Volt)
4	+5V DC or NC
5	+5V DC
6	9V AC1
7	9V AC2

USER PORT



1	GND	Ground	A	GND	(AA)	GND		3	X
2	+5V	(100mA max)	B	CB1	(BB)	Rx Sin	IN	3	X
3	RESET	COLD start	C	PB0	(BB)	Rx Sin	IN	3	X
4	PA2	Joy UP	D	PB1	(CA)	RTS	OUT	3*	X
5	PA3	Joy DOWN	E	PB2	(CD)	DTR	OUT	3*	X
6	PA4	Joy LEFT, Pad X fire	F	PB3	(CE)	DRI	IN		U
7	PA 5	pen and JOY fire	H	PB4	(CF)	DCD	IN		X
8	PA6	Tape sense switch	J	PB5	()	XXX	IN		U
9	PA7	Serial ATN out	K	PB6	(CB)	CTS	IN		X
10	9VAC	VIC transformer	L	PB7	(CC)	DSR	IN		X
11	9VAC	VIC transformer	M	CB2	(BA)	Sout	OUT	3	X
12	GND	Ground	N	GND	(AB)	GND		3	X U

Modes: 3 line, X line, User defined, *line held high

DEVICE NUMBER	
0	VIC Keyboard
1	Cassette Tape
2	RS-232 Interface Device
3	Screen Display
4	Printer
5	Printer 2
6-7	Plotter
8-30	Disk
4-127	Serial-bus Device (carriage return only)
128-255	Serial-bus Device (carriage return / line feed)

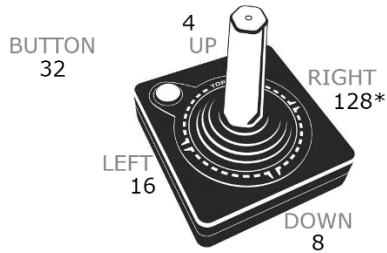
RS-232	\$293 \$294
OPEN <file id>,"<control register><command register>"	

RS-232 CONTROL

BIT		7	6	5	4	3	2	1	0	BAUD	
0: 1 stop bit										50	
1: 2 stop bits										75	
WORD LENGTH										110	
8 bits		0	0			0	1	0	0	134.5	
7 bits		0	1			0	1	0	1	150	
6 bits		1	0			0	1	1	0	300	
5 bits		1	1			0	1	1	1	600	
						1	0	0	0	1200	
						1	0	0	1	1800	
						1	0	1	0	2400	
						1	0	1	1	3600	

RS-232 COMMAND

BIT		7	6	5	4	3	2	1	0		
PARITY DISABLED		-	-	0		X	X	X		0	3 LINE
ODD PARITY		0	0	1						1	X LINE
EVEN PARITY		0	1	1							
MARK TRANSMIT		1	0	1		0	FULL DUPLEX				
SPACE TRANSMIT		1	1	1		1	HALF DUPLEX				



JOYSTICK		\$911F \$9120 \$9122
UP	$(\text{PEEK}(37151)\text{AND}4)=0$	4
DOWN	$(\text{PEEK}(37151)\text{AND}8)=0$	8
LEFT	$(\text{PEEK}(37151)\text{AND}16)=0$	16
RIGHT *	$(\text{PEEK}(37152)\text{AND}128)=0$	128
BUTTON	$(\text{PEEK}(37151)\text{AND}32)=0$	32

PADDLES		\$9008 \$9009
PADDLE A	$\text{PEEK}(36872)$	
PADDLE B	$\text{PEEK}(36873)$	
BUTTON A	$(\text{PEEK}(37151)\text{AND}16)=0$	
BUTTON B *	$(\text{PEEK}(37152)\text{AND}128)=0$	

JITTER REDUCTION A	$\text{XF}=\text{A}*\text{XF}+\text{B}*\text{PEEK}(36872) : \text{X}=\text{INT}(\text{XF}+.5)$
JITTER REDUCTION B	$\text{YF}=\text{A}*\text{YF}+\text{B}*\text{PEEK}(36873) : \text{Y}=\text{INT}(\text{YF}+.5)$

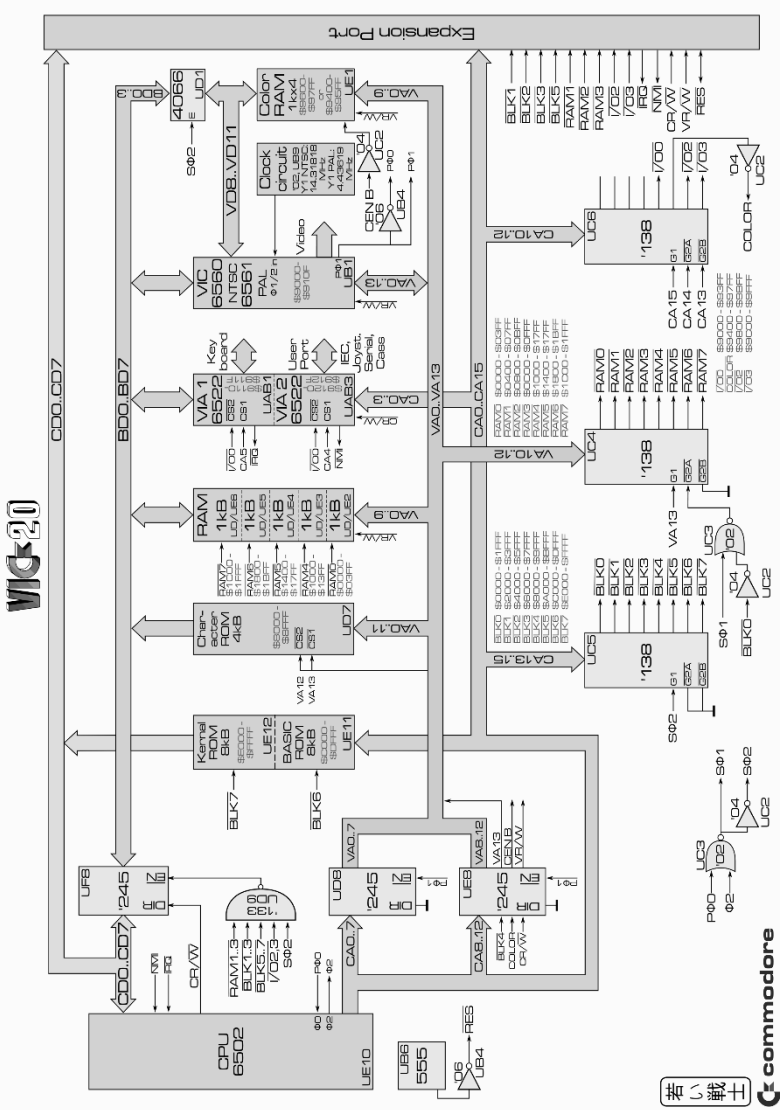
* enclose between POKE37154,127 and POKE37154,255

LIGHT PEN		\$9006 \$9007
Pen X axis	$\text{INT}((\text{PEEK}(36870)-28)/4)$	
Pen Y axis	$\text{INT}((\text{PEEK}(36871)-24)/4)$	

VIC-20 ASSY 324003 block diagram

courtesy of Sven's Techsite (Sven Petersen)

http://tech.guitarsite.de/vic-20_block.html

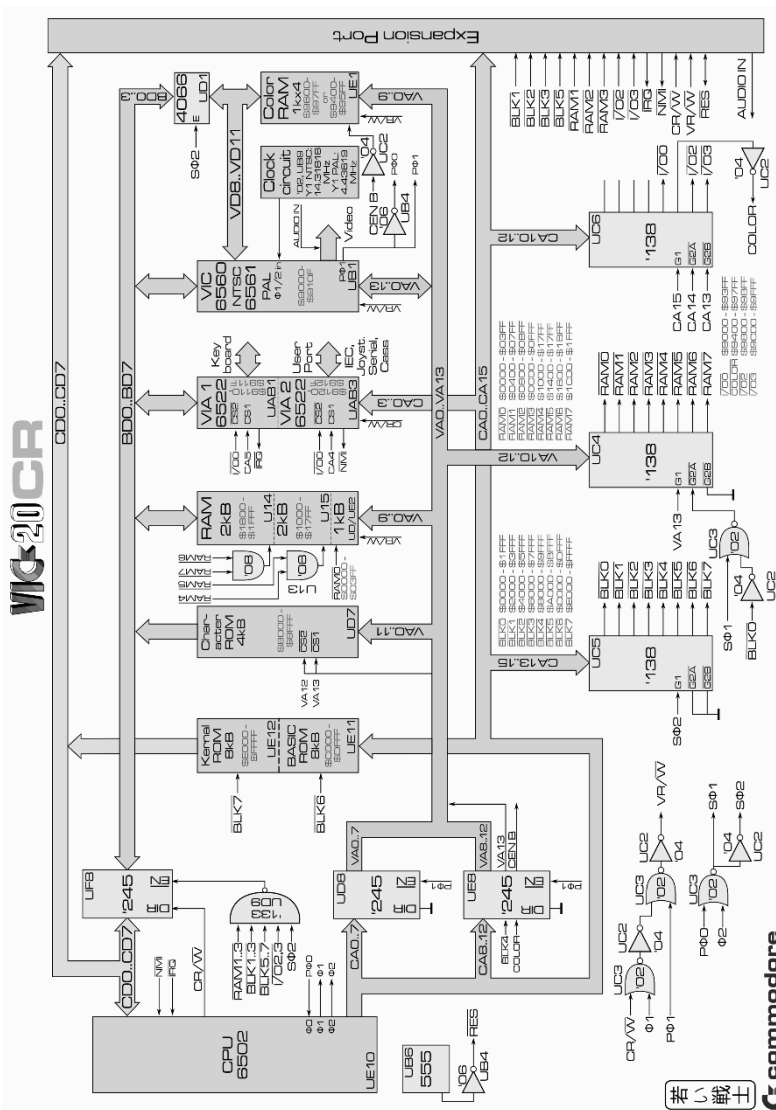


2114 (3) 1K x 4 static RAM	6522 (2) VIA
2114 (8) 1K x 4 static RAM (VIC-20)	6560 VIC (NTSC)
6116 (2) 2K x 8 static RAM (VIC-20CR)	6561 VIC (PAL)
6502A CPU	

VIC-20CR ASSY 250403 block diagram

courtesy of Sven's Techsite (Sven Petersen)

http://tech.guitarsite.de/vic-20_block.html



901460-02	2332-177	ROM character Japanese
901460-03	2332-177A	ROM character
901486-01	2364-063	ROM Basic 2 C000-DFFF
901486-02	2364-064	ROM KERNAL E000-FFFF Japanese
901486-06	2364-094	ROM KERNAL E000-FFFF (NTSC)
901486-07	2364-095	ROM KERNAL E000-FFFF (PAL)

VICMON		SYS 24576 or SYS 64096
A	ASSEMBLE	.A <address><opcode><operand>
B	BREAKPOINT	.B <address>[,<times>]
D	DISASSEMBLE	.D <address>[,<address>]
E	ENABLE VIRTUAL ZPAGE	.E <address>
F	FILL MEMORY	.F <address>,<address>,<value>
G	GO	.G [<address>]
H	HUNT	.H <address>,<address>,<data>
I	INTERPRET	.I <address>[,<address>]
J	JUMP	.J
L	LOAD	.L "<filename>",<device>
M	MEMORY DISPLAY	.M <address>[,<address>]
N	NUMBER	.N <address>,<address>,<offset>,<low limit>,<high limit>[,W <i>(optional command for word table)</i>]
Q	QUICK TRACE	.Q [<address>]
R	REGISTERS	.R
RB	REMOVE BREAKPOINT	.RB
S	SAVE	.S "<filename>",<device>,<address>,<address>
T	TRANSFER	.T <address>,<address>,<destination address>
W	WALK	.W [<address>]
X	EXIT TO BASIC	.X

\$	HEX CONVERT
#	DECIMAL CONVERT
%	BINARY CONVERT
"	ASCII CONVERT

Z	SWITCH CASE MODE
+	ADD
-	SUBTRACT
&	CHECKSUM

SUPER EXPANDER	
CHAR	<row>,<column>,"<text>"
CIRCLE	<color>,x,y,rx <width>,ry <height>[,<arc start>,<arc end>]
COLOR	<screen>,<border>,<character>,<auxiliary>
DRAW	<color>,x1,y1 TO x2,y2 [TO ... xn, yn]
GRAPHIC	<mode> (0 text, 1 multi, 2 hi-res, 3 mixed, 4 text)
KEY	<key number>,"<string>"
PAINT	<color>,x,y
POINT	<color>,x1,y1 x2,y2, ...
REGION	<color>
SCNCLR	
SOUND	<voice 1>,<voice 2>,<voice 3>,<voice 4>,<volume>

RCOLR(x)	color register	RPEN(x)	light pen
RDOT(x,y)	point color	RPOT(x)	paddle
RGR(x)	graphic mode	RSND(x)	sound register
RJOY(x)	joystick		

CTRL ←	music mode	
C D E F G A B	music notes	O select music octave
P	display all music chars	T set tempo duration
Q	cancel display of P	R play a musical rest
V	set volume	# play a sharp note
S	select music voice	\$ play a flat note

PROGRAMMER'S AID		SYS 28681
AUTO	<start line>,<interval>	
CHANGE	<old code/string>,<new code/string>,<start> - <finish>	
DELETE	<start line> - <finish line>	
DUMP	<i>(display all variables)</i>	
EDIT	<i>(change to EDIT mode)</i>	
FIND	<code>,<start> - <finish>	
HELP	<i>(highlight error)</i>	
KEY	<key number>,"<code>"	
KILL	<i>(stop AID cartridge)</i>	
MERGE	"<program>",<device>	
OFF	<i>(cancel TRACE or STEP mode)</i>	
PROG	<i>(change to PROGRAM mode)</i>	
RENUMBER	<start line>,<interval>	
STEP	<i>(halt after each line)</i>	
TRACE	<i>(display line being executed)</i>	

Ctrl-A	Scroll up	Ctrl-N	Erase all
Ctrl-E	Cancel quote	Ctrl-Q	Scroll down
Ctrl-L	Erase after cursor	Ctrl-U	Erase line

		EDIT MODE			EDIT MODE
F1	LIST	LIST	F2	MID\$(AUTO
F3	RUN	RUN	F4	LEFT\$(DELETE
F5	GOTO	FIND	F6	RIGHT\$(CHANGE
F7	INPUT	TRACE	F8	CHR\$(STEP

Hold CONTROL key for F9 – F12

F9	EDIT	PROG	F10	GOSUB	RENUMBER
F11	RETURN	MERGE	F12	STR\$(OFF

JiffyDOS		SYS 58492
@	read the disk drive error channel	
@C:newfile=file	copy a file on the same diskette	
@I	initialize the disk drive	
@N:diskname, ID	format a new diskette	
@N:diskname	short new	
@Q	disable the JiffyDOS commands	
@R:[newname]=[oldname]	rename a file	
@S:[file1],[file2]...	scratch a file (files)	
`@UJ	reset the disk drive	
@V	validate a disk	
@\$	display a disk directory	
@#device	set the default device number	
/[filename]	load a BASIC program	
↑[filename]	load and run a BASIC program	
%[filename]	load a machine language program	
←[filename]	save a BASIC program	
@B	disable the 1541 head rattle	
@D:[filename]	list a BASIC program from disk	
@F	disable the function keys	
@G	set interleave gap size	
@L:[filename]	lock/unlock a file	
@O	un-new a BASIC program	
@P	toggle printer output	
@T:[filename]	list an ASCII file from disk	
@X	set destination device number	
*"filename" type	copy a file	
f[filename]	load and run machine language file	
'[filename]	verify a file	
CTRL + S	stop / freeze listing	
CTRL + A	toggle all files for copy	
CTRL + D	default drive toggle	
CTRL + P	screen dump	
CTRL + W	toggle single file for copy	
SHIFT RUN/STOP	load and run 1st program on disk	
SYS 58492	re-enable JiffyDOS commands	

SECONDARY ADDRESS (COMMAND)		
Device	Command	Description
Tape	0	Read tape file
Tape	1	Write tape file
Tape	2	Write file w/ END marker
Disk	0	Load from disk
Disk	1	Save to disk
Disk	2-14	Open data channel
Disk	15	Open command channel
Printer	0	Upper case / graphics
Printer	7	Upper case / lower case

1541 DIP SWITCH		
DEVICE #	DIP-1/A	DIP-2/B
DEVICE 8	ON	ON
DEVICE 9	OFF	ON
DEVICE 10	ON	OFF
DEVICE 11	OFF	OFF

1541 BLOCKS BY TRACK		
TRACK	RANGE	SECTORS
1-17	0-20	21
18-24	0-18	19
25-30	0-17	18
31-35	0-16	17

CHANGE DISK DEVICE NUMBER (TEMPORARY)

```
OPEN1,8,15:PRINT#15,"M-W";CHR$(119);CHR$(0);CHR$(2);
CHR$(<device number+64>):CLOSE1
```

NAME EXTENSIONS		
PRG	program file	
SEQ	sequential file	max. file size 168.656 bytes
USR	user file	
REL	relative file	max. file size 167.132 bytes with max. 65.535 data sets
DEL	deleted file	hidden in directory
*	"splat" file	improperly closed file
⌂	protected file	write-protected software
DIR	directory	max. files in directory 144

DISK COMMANDS	
LOAD"<filename>",<device>,[<command>]	Load a BASIC program
SAVE"<filename>",<device>,[<command>]	Save a BASIC program
VERIFY"<filename>",<device>,[<command>]	Verify a file
LOAD"\$",<device>	Display a disk directory
LOAD"\$0:<text>",<device>	Display a directory wildcard
LOAD"*",<device>	Load first or next program
LOAD"\$\$",<device>	Display a disk name

COMMAND CHANNEL	
OPEN1,8,15, ... :CLOSE1	
... "UI-"	Enter VIC 20 mode
... "UI+"	Exit VIC 20 mode (64 mode)
... "C0:<folder>"	Create folder
... "CD:←"	Go to parent folder
... "I0"	Initialize the disk drive
... "N0:<diskname>,<2 char. ID>"	Format a new diskette
... "V0"	Validate a diskette
... "C0:<new name>=<old name>"	Copy a file on the same diskette
... "S0:<filename>"	Delete / Scratch a file (files)
... "R0:<new name>=<old name>"	Rename a file

MERGE OR COMBINE FILES

```
PRINT#15,"C0:<newname>=0:<file1>,[0:<file2>,
0:<file3>,0:<file4>],"D0=D1"
```

DISK DEVICE "KNOCK" REDUCTION

```
OPEN1,8,15:PRINT#15,"M-W";CHR$(106);CHR$(0);CHR$(1);
CHR$(113):CLOSE1
```

READ ERROR CHANNEL

see page 110

OPEN 1,8,15 : INPUT#15,A,A\$,B,C
 PRINT A,A\$,B,C: CLOSE 1

DEVICE STATUS

BIT	VALUE	TAPE	SERIAL	RS-232
0	0	OK	OK	OK
0	1		Write time out	Parity error
1	2		Read time out	Framing error
2	4	Data block too short		Rec buffer overrun
3	8	Data block too long		
4	16	Verify read error		CTS signal missing
5	32	Checksum error		
6	64	End of file (EOI)	End of file (EOI)	RTS signal missing
7	128		Device not present	Break detected

DISK DIRECT COMMANDS

PRINT#15, ...

Block-Allocate	"B-A";<channel>;<drive>;<track>;<sector>
Block-Execute	"B-E";<channel>;<drive>;<track>;<sector>
Block-Free	"B-F";<channel>;<track>;<sector>
Buffer-Pointer	"B-P";<channel>;<byte>
Block-Read	"B-R";<channel>;<drive>;<track>;<sector>
Block-Write	"B-W";<channel>;<drive>;<track>;<sector>
Mem-Execute	"M-E";CHR\$(<addr lo>)CHR\$(<addr hi>)
Memory-Read	"M-R";CHR\$(<addr lo>)CHR\$(<addr hi>)CHR\$(#)
Memory-Write	"M-W";CHR\$(<ad lo>)CHR\$(<ad hi>)CHR\$(#); CHR\$(<data>)
User	"U<x>";<channel>;<drive>;<track>;<sector>

SAVE MEMORY		\$2B \$2E	
POKE43,0: POKE44,X: POKE45,0: POKE46,Y: SAVE "<filename>",8			
	HEX address	POKE 44, X	POKE 46, Y
BLK 1	2000-3FFF	32	64
BLK 2	4000-5FFF	64	96
BLK 3	6000-7FFF	96	128
BLK 5	A000-BFFF	160	192

SAVEing and LOADING Memory Blocks

Courtesy of Mike via Denial (March 2014).

SAVE memory block:

```
SYS57809(N$),<device>:POKE193,<start_lo>:POKE194,<start_hi>
POKE780,193:POKE781,<end_lo>:POKE782,<end_hi>:SYS65496
```

SAVE memory block (and force load address on tape):

```
SYS57809(N$),<device>,1:POKE193,<strt_lo>:POKE194,<strt_hi>
POKE780,193:POKE781,<end_lo>:POKE782,<end_hi>:SYS65496
```

LOAD memory block ('relative' or 'relocating' load):

```
SYS57809(N$),<device>:POKE780,0:POKE781,<start_lo>:POKE782,
<start_hi>:SYS65493
```

LOAD memory block ('absolute' load):

```
SYS57809(N$),<device>,1:POKE780,0:SYS65493
```

LIST TO PRINTER (program or directory)

```
OPEN4,4:CMD4:LIST
```

```
PRINT#4:CLOSE4
```


DUMP

NEW after RUNning program. Activate with SYS828 to show all stored variables. Based on the work of Michael Krause, Andreas Dripke, wimoos, and Mike.

```
1 FORT=828T0978:READA:POKET,A:S=S+A:NEXT:IFS<>16561
  THENPRINT"CHECKSUM ERROR"
2 DATA234,234,165,45,164,46,196,48,208,2,197,47,176,
  86,133,20,132,21,105,2,144,1,200
3 DATA133,34,132,35,32,126,3,48,9,32,199,3,32,113,3,
  32,203,3,165,20,164,21,24,105,7
4 DATA144,212,200,208,209,138,48,45,152,48,57,32,166,
  219,200,76,215,221,160,0,177,20
5 DATA200,170,16,7,81,20,48,22,138,41,127,32,210,255,
  177,20,168,41,127,240,3,32,210
6 DATA255,138,48,31,152,48,31,96,160,0,177,34,133,98,
  200,177,34,168,32,151,211,80,202
7 DATA32,185,3,32,177,214,32,36,203,169,34,44,169,37,
  44,169,36,76,210,255,32,210,255
8 DATA169,61,208,246,72,169,13,32,210,255,104,96
```

CHARACTER EDITOR

Use natural on-screen editor (cursor). Change the @ character to the character you would like to edit. Empty spaces are interpreted as BIT OFF. Execute RUN2 command on screen. HOLD return to display character. Release to show DATA.

```
1 PRINT"{CLR}{RVS ON}76543210{RVS OFF}"SPC(22)"@",
  "RUN2
2 PRINT"{HOME}":FORL=0T07:R=0:I=128:FORT=0T07:R=R-I*
  (PEEK(7702+L*22+T)>>32):I=I/2:NEXT
3 POKEPEEK(7910)*8+7168+L,R:PRINT,R"{LEFT}{2 SPACES}"
  :NEXT:POKE36869,255:WAIT197,64:POKE36869,240
```

UNNEW

Restore deleted programs. Courtesy of Mike.

```
PRINT"{CLR,RVS ON}{RVS OFF}A{RVS ON}(Q{RVS OFF}+
{SPACE}3{RVS ON,SHIFT-E}J{RVS OFF,SHIFT-POUND}B
{RVS ON}E{RVS OFF}-{RVS ON}%{RVS OFF}#{SPACE,SHIFT-U,
RVS ON,SHIFT-F,RVS OFF,SHIFT-L,C=-H,RVS ON,SHIFT-D}":
SYS256*PEEK(648)
```

GAME: UFO RUSH

Navigate outer space in a U.F.O. Use the cursor keys to move left and right.

```
1 S=36876:POKES+3,14:POKES+1,0:PRINT"{CLR}{CYN}SCORE
  {WHT}"A,"{PUR}UFO RUSH{DOWN}{YEL}":A=A+1:C=8196
2 FORT=0T017:PRINT"***"SPC(RND(1)*28):NEXT:POKES+2,9
3 I=PEEK(197):B=C-22+(I=31)-(I=23):D=PEEK(B):POKEB,65:
  POKEC,32:POKES,0:IFC<7724THEN1
4 POKES,190:C=B:FORT=0T040-A*2:NEXT:IFD=32THEN3
5 POKES+1,220:FORI=1T015STEP.1:POKES+2,15-I:
  POKES+3,8+(IAND1):NEXT:WAIT197,64,64:RUN
```

GAME: CASANOVA

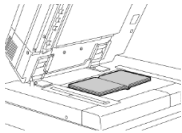
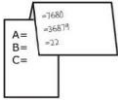
Press any key when you estimate the target number has been reached.

```
1 S=36876:POKES+3,170:POKES+2,9:B=INT(RND(1)*30)+65:
  PRINT"{CLR}{BLK}TARGET{WHT}"B
2 A=197:PRINTSPC(247)SPC(A){RVS OFF}{BLU}PRESS ANY
  KEY":C=64:WAITA,C,C:WAITA,C
3 I=7724+RND(1)*394:IFPEEK(I)=32THENPOKES,A+(IAND31):
  POKEI,83:F=F+1:POKES,0
4 IFF<99ANDPEEK(A)=CTHEN3
5 POKES+3,174-(F=B):PRINT"{home}",F:WAITA,C:WAITA,C,C:
  RUN
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21		
7680																							22	
7702																								44
7724																								66
7746																								88
7768																								110
7790																								132
7812																								154
7834																								176
7856																								198
7878																								220
7900																								242
7922																								264
7944																								286
7966																								308
7988																								330
8010																								352
8032																								374
8054																								396
8076																								418
8098																								440
8120																								462
8142																								484
8164																								506

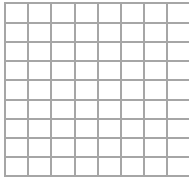
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0																																
1																																
2																																
3																																
4																																
5																																
6																																
7																																
0																																
1																																
2																																
3																																
4																																
5																																
6																																
7																																
0																																
1																																
2																																
3																																
4																																
5																																
6																																
7																																

VARIABLE LOG

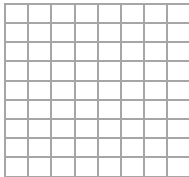


2 PG WORKSHEET
FOR PHOTOCOPY

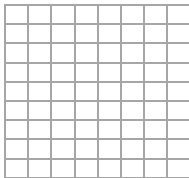
7 6 5 4 3 2 1 0



7 6 5 4 3 2 1 0



7 6 5 4 3 2 1 0



A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

BASIC MEMORY OPTIMIZATION CHECKLIST

- Remove REM and LET statements
- Remove extra spaces in lines
- Shorten line numbers (1 instead of 100)
- Combine multiple statements in each line
- Use abbreviations to exceed line limit
- Compare TAB and SPC to cursor positioning
- Use GOSUB routines for repeated task
- Streamline math operations
- Employ variables and strings for repeat use
- Use single character variables
- Use integers when possible
- Use arrays for management of larger data
- Use integer arrays to save memory
- Use READ and DATA to manage larger data
- Use a line 0 (zero)
- Remove semicolons in PRINT when possible
- Use ON to combine GOTO statements
- Remove GOTO after THEN statements
- Do not use this book as a projectile.
- Recycle variables after use

BASIC SPEED OPTIMIZATION CHECKLIST

- Follow nonconflicting memory reduction tips
- Put frequent routines early in program
- False IF statements are faster
- PRINTing direct is faster than string variables
- FOR loops are faster than GOTO
- Predefined variables are faster than numbers
- Replace isolated zeros with periods
- Streamline math operations
- IF A THEN IF B is faster than IF A AND B
- Numeric variables are faster than integers

ML OPTIMIZATION CHECKLIST *(courtesy of wimoos)*

- Change JSRs, followed by RTS into JMPs
- Remove NOPs
- Change JMPs into branches on known conditions
- A BEQ-branch taken after a CMP, CPY or CPX, will have the carry set
- A BEQ-branch taken after an EOR will have 0 in A
- A comparison using EOR affects A and Z, but not the carry
- A subroutine that is called from only one location, may be moved inline to the calling code
- An ASL/ROL may help to save b7 (negative) a little longer in the carry
- An LSR/ROR may help to save b0 a little longer in the carry

Treble Clef

A B C D E F G A B C D E F G A B C

179 187 191 198 204 207 212 217 221 223 226 230 231 234 236 238 239

do re mi fa sol la ti do re mi fa sol la ti

Bass Clef

C D E F G A B C D E F G A B C D E

255 141 153 159 170 179 187 191 198 204 207 212 217 221 223

do re mi fa sol la ti do re mi fa sol la ti

134 C#		147 D#		164 F#		174 G#		183 A#		195 C#	
255 C	141 D	153 E	159 F	170 G	179 A	187 B	191 C	198 D			

195 C#		201 D#		210 F#		215 G#		219 A#		225 C#	
191 C	198 D	204 E	207 F	212 G	217 A	221 B	223 C	226 D			

225 C#		228 D#		232 F#		235 G#		237 A#			
223 C	226 D	230 E	231 F	234 G	236 A	238 B	239 C	D			

DISK ERROR CODES

1	files scratched response.	39	command file not found
2-19	unused error messages.	50	record not present
20	block header not found on disk	51	overflow in record
21	sync character not found.	52	file too large
22	data block not present	60	file open for write
23	checksum error in data	61	file not open
24	byte decoding error	62	file not found
25	write-verify error	63	file exists
26	attempt with write protect on	64	file type mismatch
27	checksum error in header	65	no block
28	data extends into next block	66	illegal track or sector
29	disk id mismatch	67	illegal system track or sector
30	general syntax error	70	no channels available
31	invalid command	71	directory error
32	long line	72	disk full or directory full
33	invalid filename	73	power up, or write w/ DOS mismatch
34	no file given	74	drive not ready. (8050 only)

KERNAL ERROR CODES *

0	routine terminated by STOP key
1	too many open files
2	file already open
3	file not open
4	file not found
5	device not present
6	file is not an input file
7	file is not an output file
8	file name is missing
9	illegal device number

**carry bit is set, number of error message returned in accumulator*

BASIC ERROR CODES	
BAD SUBSCRIPT	attempt outside of array DIM range
CAN'T CONTINUE	cannot recall spot to resume program
DEVICE NOT PRESENT	attempt to access unavailable device
DIVISION BY ZERO	disallowed math
EXTRA IGNORED	too many items of data entered
FILE DATA	received string data when numeric expected
FILE NOT FOUND	failed to find file on device
FILE NOT OPEN	command sent to device not open
FILE OPEN	attempt to open file already open
FORMULA TOO COMPLEX	expression too long to process
ILLEGAL DIRECT	statement used in program mode
ILLEGAL DEVICE NUMBER	attempted file operation on unsuitable device
ILLEGAL QUANTITY	a number falls outside of range allowed
LOAD	device failed to LOAD
NEXT WITHOUT FOR	cannot find corresponding FOR
NOT INPUT FILE	file specified for output only
NOT OUTPUT FILE	file specified for input only
OUT OF DATA	could not find corresponding DATA
OUT OF MEMORY	no more RAM available (stack overflow)
OVERFLOW	number exceeds allowable size
REDIM'D ARRAY	array may only have one defining DIM
REDO FROM START	INPUT does not match expected value
RETURN W/O GOSUB	cannot find corresponding GOSUB
STRING TOO LONG	string exceeds 255 characters
SYNTAX	unrecognized command
TYPE MISMATCH	expected string or number but got other
UNDEF'D FUNCTION	FN employed without DEF
UNDEF'D STATEMENT	attempted GOTO or GOSUB line not found
VERIFY	program does not match file on device

INDEX

AND 21
ASCII 4, 22
Assembly 82
AV port 105
Auxiliary color 40, 44

BASIC abbreviations 12
BASIC keywords 12, 16
BASIC tokens 12, 16
Binary 4, 43, 46
Block diagrams 108, 110
Border color 41

Cartridges 112
Cassette port 104
Change case 22, 24
Character colors 44
Character set 28, 32
Clock 15, 19
Custom characters 42

Datacassette 104
Disk drive 116
Display 40
Dump 120

Errors 126
Expansions 50, 112

Flags 86

Game port 104, 107
Graphic printer 119
Graphics 42

Hardware 3, 108
Hexadecimal 4, 98
Horizontal video 40

Interlace 40
I/O 104

Jiffydos 115
Joystick 104, 107

KERNAL routines 100
Keyboard 102

Light pen 104, 107
LOAD memory blocks 119
Logical operations 21
Lower case 21

Math functions 20
Memory blocks 50
Memory labels 76
Memory map 50
Merge files 117
ML instructions 82
Most significant byte 98
Music notes 125

NOT 21

Opcodes 4, 84
Optimization tips 124
OR 21

Paddle 104, 107
PETSCII 4, 22
Ports 104
Printer 119
Programmer's Aid 114
Program location 38

Quick start 2
Quote mode 24

Relative branch 97
Reset system 2
ROM images 2
RS-232 106

SAVE memory blocks 119
Scrap paper 122-123
Screen color 41
Screen columns 40
Screen location 38
Screen map 30, 34
Screen rows 40
Sound 62
Specifications 3
Storage 115
Super Expander 113

Tape 104
Tokens 12, 16
Type-in programs 120

Undocumented codes 94
Unnew 121
Upper case 21
User port 105

Variables 18
Vertical video 40
Vicmon 112
Video 40

WAIT 102
Warm start 3

XOR 20

Zero page 51

1 Select	U undo / del	G song / tone	F1 char / col
H horizontal	P paste / mrg	£ music / rec	F3 screen
V vertical	T trade / tr+	= reset	F7 toggle
F flip	R rom / all	* isolate	RT enter
Z spin	8 828 / 3k	+ double	CL clear
9 r90 / inv	E exit / browse		Cu cursor / +

	Unexpanded	with 3K RAM	with 8K+ RAM	
\$0 0	BASIC RAM 1K	BASIC RAM 1K	BASIC RAM 1K	\$0 0
\$400 1024	unused 3K	BASIC storage 6.5K	unused 3K	\$400 1024
\$1000 4096	BASIC storage 3.5K		Screen MAP 0.5K	\$1000 4096
\$1200 4608			BASIC storage 3.5K	\$1200 4608
\$1E00 7680	Screen MAP 0.5K	Screen MAP 0.5K		\$1E00 7680
\$2000 8192	unused 8K	unused 8K	BLOCK 1 RAM 8K	\$2000 8192
\$4000 16384	unused 8K	unused 8K	BLOCK 2 RAM 8K	\$4000 16384
\$6000 24576	unused 8K	unused 8K	BLOCK 3 RAM 8K	\$6000 24576
\$8000 32768	Character ROM 4K	Character ROM 4K	Character ROM 4K	\$8000 32768
\$9000 36864	VIC CHIP	VIC CHIP	VIC CHIP	\$9000 36864
\$9100 37120	VIA CHIPS	VIA CHIPS	VIA CHIPS	\$9100 37120
\$9400 37888	unused 0.5K	unused 0.5K	Screen Color MAP 0.5K	\$9400 37888
\$9600 38400	Screen Color MAP 0.5K	Screen Color MAP 0.5K	unused 0.5K	\$9600 38400
\$9800 38912	I/O 2K	I/O 2K	I/O 2K	\$9800 38912
\$A000 40960	unused 8K	unused 8K	BLOCK 5 RAM 8K	\$A000 40960
\$C000 49152	BASIC ROM 8K	BASIC ROM 8K	BASIC ROM 8K	\$C000 49152
\$E000 57344	KERNAL ROM 8K	KERNAL ROM 8K	KERNAL ROM 8K	\$E000 57344
\$FFFF 65535	Unexpanded	with 3K RAM	with 8K+ RAM	\$FFFF 65535

H E X															
0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

B I N							
7	6	5	4	3	2	1	0
128	64	32	16	8	4	2	1
CHAR	BORD	CHAR	BORD	CHAR	BORD	CHAR	BORD
192		48		12		3	

#		AUX	MULTI
	0	BLK	8
	1	WHT	9
	2	RED	10
	3	CYN	11
	4	PUR	12
	5	GRN	13
	6	BLU	14
	7	YEL	15
		ORG	128 ---
		Lt. ORG	144 ---
		PNK	160 ---
		Lt. CYN	176 ---
		Lt. PUR	192 ---
		Lt. GRN	208 ---
		Lt. BLU	224 ---
		Lt. YEL	240 ---

	7	6	5	4	NIBBLE	3	2	1	0
0					□□□□				0
16					□□□■				1
32					□□■□				2
48					□□■■				3
64					□■□□				4
80					□■□■				5
96					□■■□				6
112					□■■■				7
128					■□□□				8
144					■□□■				9
160					■□■□				10
176					■□■■				11
192					■■□□				12
208					■■□■				13
224					■■■□				14
240					■■■■				15

ORDER OF OPERATIONS	1 st	()	6 th	< = >
	2 nd	↑	7 th	NOT
	3 rd	negative	8 th	AND
	4 th	* /	9 th	OR
	5 th	+ -		