APPENDICES

APPENDIX A

ABBREVIATIONS FOR BASIC KEYWORDS

As a time-saver when typing in programs and commands, Commodore 64 BASIC allows the user to abbreviate most keywords. The abbreviation for PRINT is a question mark. The abbreviations for other words are made by typing the first one or two letters of the word, followed by the SHIFTed next letter of the word. If the abbreviations are used in a program line, the keyword will LIST in the full form.

Command	Abbreviation	Looks like this on screen	Command	Abbreviation	Looks like this on screen
ABS	A SHIFT B	A	END	E SHIFT N	E
AND	A SHIFT N	A	EXP	E SHIFT X	E
ASC	A SHIFT S	A	FN	NONE	FN
ATN	A SHIFT T	A	FOR	F SHIFT O	F
CHR$	C SHIFT H	C	FRE	F SHIFT R	F
CLOSE	CL SHIFT O	CL	GET	G SHIFT E	G
CLR	C SHIFT L	C	GET#	NONE	GET#
CMD	C SHIFT M	C	GOSUB	GO SHIFT S	GO
CONT	C SHIFT O	C	GOTO	G SHIFT O	G
COS	NONE	COS	IF	NONE	IF
DATA	D SHIFT A	D	INPUT	NONE	INPUT
DEF	D SHIFT E	D	INPUT#	I SHIFT N	I
DIM	D SHIFT I	D	INT	NONE	INT

Command	Abbreviation	Looks like this on screen
LEFT$	LE SHIFT F	LE
LEN	NONE	LEN
LET	L SHIFT E	L
LIST	L SHIFT I	L
LOAD	L SHIFT O	L
LOG	NONE	LOG
MID$	M SHIFT I	M
NEW	NONE	NEW
NEXT	N SHIFT E	N
NOT	N SHIFT O	N
ON	NONE	ON
OPEN	O SHIFT P	O
OR	NONE	OR
PEEK	P SHIFT E	P
POKE	P SHIFT O	P
POS	NONE	POS
PRINT	?	?
PRINT#	P SHIFT R	P
READ	R SHIFT E	R
REM	NONE	REM
RESTORE	RE SHIFT S	RE
RETURN	RE SHIFT T	RE
RIGHT$	R SHIFT I	R
RND	R SHIFT N	R
RUN	R SHIFT U	R
SAVE	S SHIFT A	S
SGN	S SHIFT G	S
SIN	S SHIFT I	S
SPC(	S SHIFT P	S
SQR	S SHIFT Q	S
STATUS	ST	ST
STEP	ST SHIFT E	ST
STOP	S SHIFT T	S
STR$	ST SHIFT R	ST
SYS	S SHIFT Y	S
TAB(	T SHIFT A	T
TAN	NONE	TAN
THEN	T SHIFT H	T
TIME	TI	TI
TIME$	TI$	TI$
USR	U SHIFT S	U
VAL	V SHIFT A	V
VERIFY	V SHIFT E	V
WAIT	W SHIFT A	W

APPENDIX B

SCREEN DISPLAY CODES

The following chart lists all of the characters built into the Commodore 64 character sets. It shows which numbers should be POKEd into screen memory (locations 1024-2023) to get a desired character. Also shown is which character corresponds to a number PEEKed from the screen.

Two character sets are available, but only one set at a time. This means that you cannot have characters from one set on the screen at the same time you have characters from the other set displayed. The sets are switched by holding down the SHIFT and C= keys simultaneously.

From BASIC, POKE 53272,21 will switch to upper case mode and POKE 53272,23 switches to lower case.

Any number on the chart may also be displayed in REVERSE. The reverse character code may be obtained by adding 128 to the values shown.

If you want to display a solid circle at location 1504, POKE the code for the circle (81) into location 1504: POKE 1504,81.

There is a corresponding memory location to control the color of each character displayed on the screen (locations 55296-56295). To change the color of the circle to yellow (color code 7) you would POKE the corresponding memory location (55776) with the character color: POKE 55776,7.

Refer to Appendix D for the complete screen and color memory maps, along with color codes.

NOTE: The following POKEs display the same symbol in set 1 and 2: 1, 27–64, 91–93, 96–104, 106–121, 123–127.

SCREEN CODES

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
@		0	C	c	3	F	f	6
A	a	1	D	d	4	G	g	7
B	b	2	E	e	5	H	h	8

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
I	i	9	%		37		A	65
J	j	10	&		38		B	66
K	k	11	'		39		C	67
L	l	12	(		40		D	68
M	m	13	)		41		E	69
N	n	14	*		42		F	70
O	o	15	+		43		G	71
P	p	16	,		44		H	72
Q	q	17	–		45		I	73
R	r	18	.		46		J	74
S	s	19	/		47		K	75
T	t	20	0		48		L	76
U	u	21	1		49		M	77
V	v	22	2		50		N	78
W	w	23	3		51		O	79
X	x	24	4		52		P	80
Y	y	25	5		53		Q	81
Z	z	26	6		54		R	82
[		27	7		55		S	83
£		28	8		56		T	84
]		29	9		57		U	85
↑		30	:		58		V	86
←		31	;		59		W	87
SPACE		32	<		60		X	88
!		33	=		61		Y	89
"		34	>		62		Z	90
#		35	?		63			91
$		36			64			92

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
		93			105			117
		94			106			118
		95			107			119
SPACE		96			108			120
		97			109			121
		98			110			122
		99			111			123
		100			112			124
		101			113			125
		102			114			126
		103			115			127
		104			116			

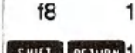
Codes from 128-255 are reversed images of codes 0-127.

APPENDIX C

ASCII AND CHR$ CODES

This appendix shows you what characters will appear if you PRINT CHR$(X), for all possible values of X. It will also show the values obtained by typing PRINT ASC("x"), where x is any character you can type. This is useful in evaluating the character received in a GET statement, converting upper/lower case, and printing character based commands (like switch to upper/lower case) that could not be enclosed in quotes.

PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
	0	CRSR ↑	17	"	34	3	51
	1	RVS ON	18	#	35	4	52
	2	CLR HOME	19	$	36	5	53
	3	INST DEL	20	%	37	6	54
	4		21	&	38	7	55
WHT	5		22	'	39	8	56
	6		23	(	40	9	57
	7		24	)	41	:	58
DISABLES SHIFT C=	8		25	*	42	;	59
ENABLES SHIFT C=	9		26	+	43	<	60
	10		27	,	44	=	61
	11	RED	28	-	45	>	62
	12	CRSR →	29	.	46	?	63
RETURN	13	GRN	30	/	47	@	64
SWITCH TO LOWER CASE	14	BLU	31	0	48	A	65
	15	SPACE	32	1	49	B	66
	16	!	33	2	50	C	67

PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
D	68		97		126	Grey 3	155
E	69		98		127		156
F	70		99		128		157
G	71		100	Orange	129		158
H	72		101		130		159
I	73		102		131		160
J	74		103		132		161
K	75		104	f1	133		162
L	76		105	f3	134		163
M	77		106	f5	135		164
N	78		107	f7	136		165
O	79		108	f2	137		166
P	80		109	f4	138		167
Q	81		110	f6	139		168
R	82		111	f8	140		169
S	83		112		141		170
T	84		113		142		171
U	85		114		143		172
V	86		115		144		173
W	87		116		145		174
X	88		117		146		175
Y	89		118		147		176
Z	90		119		148		177
[	91		120	Brown	149		178
£	92		121	Lt. Red	150		179
]	93		122	Grey 1	151		180
↑	94		123	Grey 2	152		181
←	95		124	Lt. Green	153		182
	96		125	Lt. Blue	154		183

PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
	184		186		188		190
	185		187		189		191

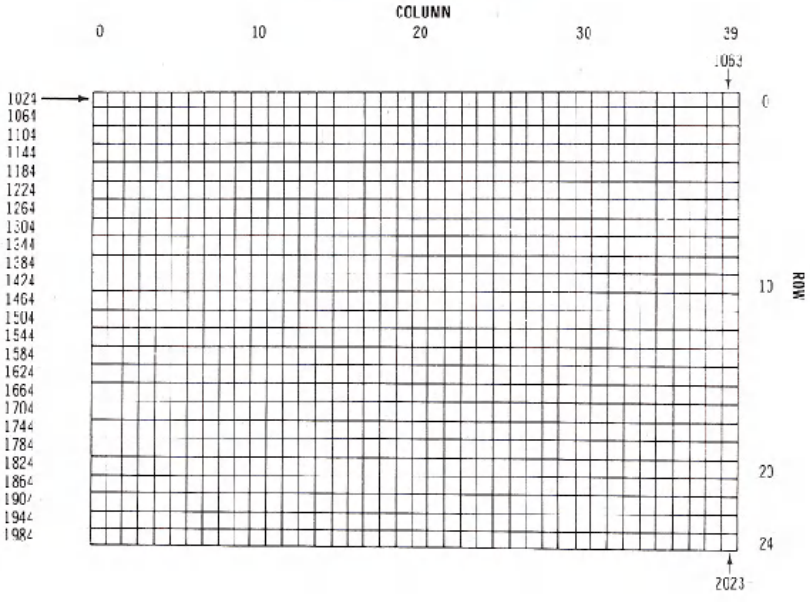
CODES 192-223 **SAME AS** 96-127
CODES 224-254 **SAME AS** 160-190
CODE 255 **SAME AS** 126

APPENDIX D

SCREEN AND COLOR MEMORY MAPS

The following charts list which memory locations control placing characters on the screen, and the locations used to change individual character colors, as well as showing character color codes.

SCREEN MEMORY MAP

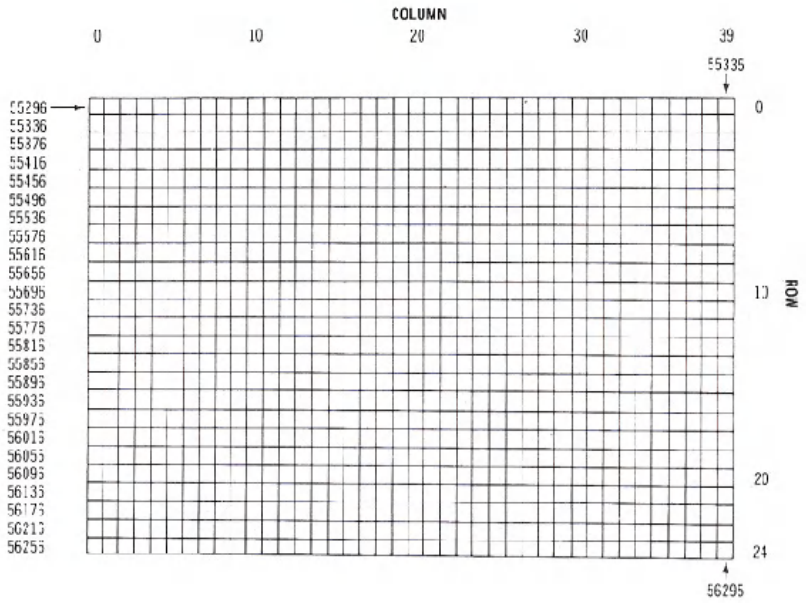


The actual values to POKE into a color memory location to change a character's color are:

Ø	BLACK	8	ORANGE
1	WHITE	9	BROWN
2	RED	1Ø	Light RED
3	CYAN	11	GRAY 1
4	PURPLE	12	GRAY 2
5	GREEN	13	Light GREEN
6	BLUE	14	Light BLUE
7	YELLOW	15	GRAY 3

For example, to change the color of a character located at the upper left-hand corner of the screen to red, type: POKE 55296,2.

COLOR MEMORY MAP



APPENDIX E

MUSIC NOTE VALUES

This appendix contains a complete list of Note#, actual note, and the values to be POKEd into the HI FREQ and LOW FREQ registers of the sound chip to produce the indicated note.

MUSICAL NOTE		OSCILLATOR FREQ		
NOTE	OCTAVE	DECIMAL	HI	LOW
0	C–0	268	1	12
1	C#–0	284	1	28
2	D–0	301	1	45
3	D#–0	318	1	62
4	E–0	337	1	81
5	F–0	358	1	102
6	F#–0	379	1	123
7	G–0	401	1	145
8	G#–0	425	1	169
9	A–0	451	1	195
10	A#–0	477	1	221
11	B–0	506	1	250
16	C–1	536	2	24
17	C#–1	568	2	56
18	D–1	602	2	90
19	D#–1	637	2	125
20	E–1	675	2	163
21	F–1	716	2	204
22	F#–1	758	2	246
23	G–1	803	3	35
24	G#–1	851	3	83
25	A–1	902	3	134
26	A#–1	955	3	187
27	B–1	1012	3	244
32	C–2	1072	4	48

MUSICAL NOTE		OSCILLATOR FREQ		
NOTE	OCTAVE	DECIMAL	HI	LOW
33	C#–2	1136	4	112
34	D–2	1204	4	180
35	D#–2	1275	4	251
36	E–2	1351	5	71
37	F–2	1432	5	152
38	F#–2	1517	5	237
39	G–2	1607	6	71
40	G#–2	1703	6	167
41	A–2	1804	7	12
42	A#–2	1911	7	119
43	B–2	2025	7	233
48	C–3	2145	8	97
49	C#–3	2273	8	225
50	D–3	2408	9	104
51	D#–3	2551	9	247
52	E–3	2703	10	143
53	F–3	2864	11	48
54	F#–3	3034	11	218
55	G–3	3215	12	143
56	G#–3	3406	13	78
57	A–3	3608	14	24
58	A#–3	3823	14	239
59	B–3	4050	15	210
64	C–4	4291	16	195
65	C#–4	4547	17	195
66	D–4	4817	18	209
67	D#–4	5103	19	239
68	E–4	5407	21	31
69	F–4	5728	22	96
70	F#–4	6069	23	181
71	G–4	6430	25	30
72	G#–4	6812	26	156
73	A–4	7217	28	49
74	A#–4	7647	29	223
75	B–4	8101	31	165
80	C–5	8583	33	135
81	C#–5	9094	35	134

MUSICAL NOTE		OSCILLATOR FREQ		
NOTE	OCTAVE	DECIMAL	HI	LOW
82	D–5	9634	37	162
83	D#–5	10207	39	223
84	E–5	10814	42	62
85	F–5	11457	44	193
86	F#–5	12139	47	107
87	G–5	12860	50	60
88	G#–5	13625	53	57
89	A–5	14435	56	99
90	A# 5	15294	59	190
91	B–5	16203	63	75
96	C–6	17167	67	15
97	C#–6	18188	71	12
98	D–6	19269	75	69
99	D#–6	20415	79	191
100	E–6	21629	84	125
101	F–6	22915	89	131
102	F#–6	24278	94	214
103	G–6	25721	100	121
104	G#–6	27251	106	115
105	A–6	28871	112	199
106	A#–6	30588	119	124
107	B–6	32407	126	151
112	C–7	34334	134	30
113	C#–7	36376	142	24
114	D–7	38539	150	139
115	D#–7	40830	159	126
116	E–7	43258	168	250
117	F–7	45830	179	6
118	F# 7	48556	189	172
119	G–7	51443	200	243
120	G#–7	54502	212	230
121	A–7	57743	225	143
122	A#–7	61176	238	248
123	B–7	64814	253	46

FILTER SETTINGS

Location	Contents
54293	Low cutoff frequency (0–7)
54294	High cutoff frequency (0–255)
54295	Resonance (bits 4–7) Filter voice 3 (bit 2) Filter voice 2 (bit 1) Filter voice 1 (bit 0)
54296	High pass (bit 6) Bandpass (bit 5) Low pass (bit 4) Volume (bits 0–3)

APPENDIX F

BIBLIOGRAPHY

Addison-Wesley	"BASIC and the Personal Computer", Dwyer and Critchfield
Compute	"Compute's First Book of PET/CBM"
Cowboy Computing	"Feed Me, I'm Your PET Computer", Carol Alexander
	"Looking Good with Your PET", Carol Alexander
	"Teacher's PET—Plans, Quizzes, and Answers"
Creative Computing	"Getting Acquainted With Your VIC 20", T. Hartnell
Dilithium Press	"BASIC Basic-English Dictionary for the PET", Larry Noonan
	"PET BASIC", Tom Rugg and Phil Feldman
Faulk Baker Associates	"MOS Programming Manual", MOS Technology
Hayden Book Co.	"BASIC From the Ground Up", David E. Simon
	"I Speak BASIC to My PET", Aubrey Jones, Jr.
	"Library of PET Subroutines", Nick Hampshire
	"PET Graphics", Nick Hampshire
	"BASIC Conversions Handbook, Apple, TRS-80, and PET", David A. Brain, Phillip R. Oviatt, Paul J. Paquin, and Chandler P. Stone

Howard W. Sams	"The Howard W. Sams Crash Course in Microcomputers", Louis E. Frenzel, Jr.
	"Mostly BASIC: Applications for Your PET", Howard Berenbon
	"PET Interfacing", James M. Downey and Steven M. Rogers
	"VIC 20 Programmer's Reference Guide", A. Finkel, P. Higginbottom, N. Harris, and M. Tomczyk
Little, Brown & Co.	"Computer Games for Businesses, Schools, and Homes", J. Victor Nagigian, and William S. Hodges
	"The Computer Tutor: Learning Activities for Homes and Schools", Gary W. Orwig, University of Central Florida, and William S. Hodges
McGraw-Hill	"Hands-On BASIC With a PET", Herbert D. Peckman
	"Home and Office Use of VisiCalc", D. Castlewitz, and L. Chisauki
Osborne/McGraw-Hill	"PET/CBM Personal Computer Guide", Carroll S. Donahue
	"PET Fun and Games", R. Jeffries and G. Fisher
	"PET and the IEEE", A. Osborne and C. Donahue
	"Some Common BASIC Programs for the PET", L. Poole, M. Borchers, and C. Donahue
	"Osborne CP/M User Guide", Thom Hogan
	"CBM Professional Computer Guide"
	"The PET Personal Guide"
	"The 8086 Book", Russell Rector and George Alexy
P. C. Publications	"Beginning Self-Teaching Computer Lessons"

Prentice-Hall	"The PET Personal Computer for Beginners", S. Dunn and V. Morgan
Reston Publishing Co.	"PET and the IEEE 488 Bus (GPIB)", Eugene Fisher and C. W. Jensen
	"PET BASIC—Training Your PET Computer", Ramon Zamora, Wm. F. Carrie, and B. Allbrecht
	"PET Games and Recreation", M. Ogelsby, L. Lindsey, and D. Kunkin
	"PET BASIC", Richard Huskell
	"VIC Games and Recreation"
Telmas Courseware Ratings	"BASIC and the Personal Computer", T. A. Dwyer, and M. Critchfield
Total Information Services	"Understanding Your PET/CBM, Vol. 1, BASIC Programming"
	"Understanding Your VIC", David Schultz

Commodore Magazines provide you with the most up-to-date information for your Commodore 64. Two of the most popular publications that you should seriously consider subscribing to are:

COMMODORE—*The Microcomputer Magazine* is published bi-monthly and is available by subscription ($15.00 per year, U.S., and $25.00 per year, worldwide).

POWER/PLAY—*The Home Computer Magazine* is published quarterly and is available by subscription ($10.00 per year, U.S., and $15.00 per year worldwide).

APPENDIX G

VIC CHIP REGISTER MAP

53248 ($D000) Starting (Base) Address

Register # Dec	Hex	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
0	0	S0X7							S0X0	SPRITE 0 X Component
1	1	S0Y7							S0Y0	SPRITE 0 Y Component
2	2	S1X7							S1X0	SPRITE 1 X
3	3	S1Y7							S1Y0	SPRITE 1 Y
4	4	S2X7							S2X0	SPRITE 2 X
5	5	S2Y7							S2Y0	SPRITE 2 Y
6	6	S3X7							S3X0	SPRITE 3 X
7	7	S3Y7							S3Y0	SPRITE 3 Y
8	8	S4X7							S4X0	SPRITE 4 X
9	9	S4Y7							S4Y0	SPRITE 4 Y
10	A	S5X7							S5X0	SPRITE 5 X
11	B	S5Y7							S5Y0	SPRITE 5 Y
12	C	S6X7							S6X0	SPRITE 6 X
13	D	S6Y7							S6Y0	SPRITE 6 Y
14	E	S7X7							S7X0	SPRITE 7 X Component
15	F	S7Y7							S7Y0	SPRITE 7 Y Component
16	10	S7X8	S6X8	S5X8	S4X8	S3X8	S2X8	S1X8	S0X8	MSB of X COORD.
17	11	RC8	ECM	BMM	BLNK	RSEL	YSCL2	YSCL1	YSCL0	Y SCROLL MODE
18	12	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	RASTER
19	13	LPX7							LPX0	LIGHT PEN X
20	14	LPY7							LPY0	LIGHT PEN Y

Register # Dec	Hex	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
21	15	SE7							SE0	SPRITE ENABLE (ON/OFF)
22	16	N.C.	N.C.	RST	MCM	CSEL	XSCL2	XSCL1	XSCL0	X SCROLL MODE
23	17	SEXY7							SEXY0	SPRITE EXPAND Y
24	18	VS13	VS12	VS11	VS10	CB13	CB12	CB11	N.C.	SCREEN and Character Memory Base Address
25	19	IRQ	N.C.	N.C.	N.C.	LPIRQ	ISSC	ISBC	RIRQ	Interrupt Request's
26	1A	N.C.	N.C.	N.C.	N.C.	MLPI	MISSC	MISBC	MRIRQ	Interrupt Request MASKS
27	1B	BSP7							BSP0	Background-Sprite PRIORITY
28	1C	SCM7							SCM0	MULTICOLOR SPRITE SELECT
29	1D	SEXX7							SEXX0	SPRITE EXPAND X
30	1E	SSC7							SSC0	Sprite-Sprite COLLISION
31	1F	SBC7							SBC0	Sprite-Background COLLISION

Register # Dec	Hex	Color
32	20	BORDER COLOR
33	21	BACKGROUND COLOR 0
34	22	BACKGROUND COLOR 1
35	23	BACKGROUND COLOR 2
36	24	BACKGROUND COLOR 3
37	25	SPRITE MULTICOLOR 0
38	26	SPRITE MULTICOLOR 1

Register # Dec	Hex	Color
39	27	SPRITE 0 COLOR
40	28	SPRITE 1 COLOR
41	29	SPRITE 2 COLOR
42	2A	SPRITE 3 COLOR
43	2B	SPRITE 4 COLOR
44	2C	SPRITE 5 COLOR
45	2D	SPRITE 6 COLOR
46	2E	SPRITE 7 COLOR

COLOR CODES

Dec	Hex	Color
0	0	BLACK
1	1	WHITE
2	2	RED
3	3	CYAN
4	4	PURPLE
5	5	GREEN
6	6	BLUE
7	7	YELLOW

Dec	Hex	Color
8	8	ORANGE
9	9	BROWN
10	A	LT RED
11	B	GRAY 1
12	C	GRAY 2
13	D	LT GREEN
14	E	LT BLUE
15	F	GRAY 3

LEGEND:
ONLY COLORS 0-7 MAY BE USED IN MULTICOLOR CHARACTER MODE

APPENDIX H

DERIVING MATHEMATICAL FUNCTIONS

Functions that are not intrinsic to Commodore 64 BASIC may be calculated as follows:

FUNCTION	BASIC EQUIVALENT
SECANT	SEC(X)=1/COS(X)
COSECANT	CSC(X)=1/SIN(X)
COTANGENT	COT(X)=1/TAN(X)
INVERSE SINE	ARCSIN(X)=ATN(X/SQR(−X*X+1))
INVERSE COSINE	ARCCOS(X)=−ATN(X/SQR (−X*X +1)) +π/2
INVERSE SECANT	ARCSEC(X)=ATN(X/SQR(X*X−1))
INVERSE COSECANT	ARCCSC(X)=ATN(X/SQR(X*X−1)) +(SGN(X)−1*π/2
INVERSE COTANGENT	ARCOT(X)=ATN(X)+π/2
HYPERBOLIC SINE	SINH(X)=(EXP(X)−EXP(−X))/2
HYPERBOLIC COSINE	COSH(X)=(EXP(X)+EXP(−X))/2
HYPERBOLIC TANGENT	TANH(X)=EXP(−X)/(EXP(x)+EXP (−X))*2+1
HYPERBOLIC SECANT	SECH(X)=2/(EXP(X)+EXP(−X))
HYPERBOLIC COSECANT	CSCH(X)=2/(EXP(X)−EXP(−X))
HYPERBOLIC COTANGENT	COTH(X)=EXP(−X)/(EXP(X) −EXP(−X))*2+1
INVERSE HYPERBOLIC SINE	ARCSINH(X)=LOG(X+SQR(X*X+1))
INVERSE HYPERBOLIC COSINE	ARCCOSH(X)=LOG(X+SQR(X*X−1))
INVERSE HYPERBOLIC TANGENT	ARCTANH(X)=LOG((1+X)/(1−X))/2
INVERSE HYPERBOLIC SECANT	ARCSECH(X)=LOG((SQR (−X*X+1)+1/X)
INVERSE HYPERBOLIC COSECANT	ARCCSCH(X)=LOG((SGN(X)*SQR (X*X+1/x)
INVERSE HYPERBOLIC COTANGENT	ARCCOTH(X)=LOG((X+1)/(X−1))/2

APPENDIX I

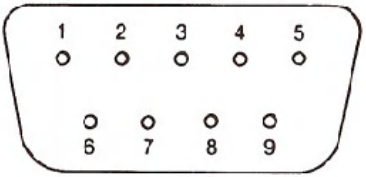
PINOUTS FOR INPUT/OUTPUT DEVICES

This appendix is designed to show you what connections may be made to the Commodore 64.

1) Game I/O
2) Cartridge Slot
3) Audio/Video
4) Serial I/O (Disk/Printer)
5) Modulator Output
6) Cassette
7) User Port

Control Port 1

Pin	Type	Note
1	JOYA0	
2	JOYA1	
3	JOYA2	
4	JOYA3	
5	POT AY	
6	BUTTON A/LP	
7	+5V	MAX. 50mA
8	GND	
9	POT AX	



Control Port 2

Pin	Type	Note
1	JOYB0	
2	JOYB1	
3	JOYB2	
4	JOYB3	
5	POT BY	
6	BUTTON B	
7	+5V	MAX. 50mA
8	GND	
9	POT BX	

Cartridge Expansion Slot

Pin	Type
1	GND
2	+5V
3	+5V
4	$\overline{\text{IRQ}}$
5	R/$\overline{\text{W}}$
6	Dot Clock
7	I/O 1
8	$\overline{\text{GAME}}$
9	$\overline{\text{EXROM}}$
10	I/O 2
11	$\overline{\text{ROML}}$

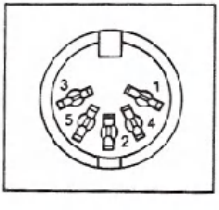
Pin	Type
12	$\overline{\text{BA}}$
13	$\overline{\text{DMA}}$
14	D7
15	D6
16	D5
17	D4
18	D3
19	D2
20	D1
21	D0
22	GND

Pin	Type
A	GND
B	$\overline{\text{ROMH}}$
C	$\overline{\text{RESET}}$
D	$\overline{\text{NMI}}$
E	S 02
F	A15
H	A14
J	A13
K	A12
L	A11
M	A10

Pin	Type
N	A9
P	A8
R	A7
S	A6
T	A5
U	A4
V	A3
W	A2
X	A1
Y	A0
Z	GND

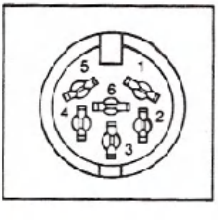
Audio/Video

Pin	Type	Note
1	LUMINANCE	
2	GND	
3	AUDIO OUT	
4	VIDEO OUT	
5	AUDIO IN	



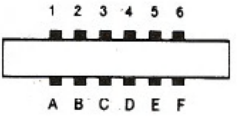
Serial I/O

Pin	Type
1	SERIAL $\overline{\text{SRQIN}}$
2	GND
3	SERIAL ATN IN/OUT
4	SERIAL CLK IN/OUT
5	SERIAL DATA IN/OUT
6	$\overline{\text{RESET}}$



Cassette

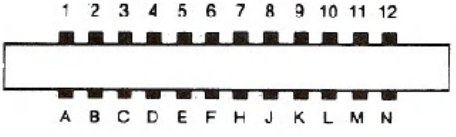
Pin	Type
A-1	GND
B-2	+5V
C-3	CASSETTE MOTOR
D-4	CASSETTE READ
E-5	CASSETTE WRITE
F-6	CASSETTE SENSE



User I/O

Pin	Type	Note
1	GND	
2	+5V	MAX. 100 mA
3	$\overline{\text{RESET}}$	
4	CNT1	
5	SP1	
6	CNT2	
7	SP2	
8	$\overline{\text{PC2}}$	
9	SER. ATN IN	
10	9 VAC	MAX. 100 mA
11	9 VAC	MAX. 100 mA
12	GND	

Pin	Type	Note
A	GND	
B	$\overline{\text{FLAG2}}$	
C	PB0	
D	PB1	
E	PB2	
F	PB3	
H	PB4	
J	PB5	
K	PB6	
L	PB7	
M	PA2	
N	GND	



APPENDIX J

CONVERTING STANDARD BASIC PROGRAMS TO COMMODORE 64 BASIC

If you have programs written in a BASIC other than Commodore BASIC, some minor adjustments may be necessary before running them on the Commodore-64. We've included some hints to make the conversion easier.

String Dimensions

Delete all statements that are used to declare the length of strings. A statement such as DIM A$(I,J), which dimensions a string array for J elements of length I, should be converted to the Commodore BASIC statement DIM A$(J).

Some BASICs use a comma or an ampersand for string concatenation. Each of these must be changed to a plus sign, which is the Commodore BASIC operator for string concatenation.

In Commodore 64 BASIC, the MID$, RIGHT$, and LEFT$ functions are used to take substrings of strings. Forms such as A$(I) to access the Ith character in A$, or A$(I,J) to take a substring of A$ from position I to J, must be changed as follows:

Other BASIC	Commodore 64 BASIC
A$(I) = X$	A$ = LEFT$(A$,I−1)+X$+MID$(A$,I+1)
A$(I,J) = X$	A$ = LEFT$(A$,I−1)+X$+MID$(A$,J+1)

Multiple Assignments

To set B and C equal to zero, some BASICs allow statements of the form:

```
10 LET B=C=0
```

Commodore 64 BASIC would interpret the second equal sign as a logical operator and set B = −1 if C = 0. Instead, convert this statement to:

```
1Ø C=Ø : B=Ø
```

Multiple Statements

Some BASICs use a backslash (\ ) to separate multiple statements on a line. With Commodore 64 BASIC, separate all statements by a colon (:).

MAT Functions

Programs using the MAT functions available on some BASICs must be rewritten using FOR. . .NEXT loops to execute properly.

APPENDIX K

ERROR MESSAGES

This appendix contains a complete list of the error messages generated by the Commodore-64, with a description of causes.

BAD DATA String data was received from an open file, but the program was expecting numeric data.

BAD SUBSCRIPT The program was trying to reference an element of an array whose number is outside of the range specified in the DIM statement.

BREAK Program execution was stopped because you hit the STOP key.

CAN'T CONTINUE The CONT command will not work, either because the program was never RUN, there has been an error, or a line has been edited.

DEVICE NOT PRESENT The required I/O device was not available for an OPEN, CLOSE, CMD, PRINT#, INPUT#, or GET#.

DIVISION BY ZERO Division by zero is a mathematical oddity and not allowed.

EXTRA IGNORED Too many items of data were typed in response to an INPUT statement. Only the first few items were accepted.

FILE NOT FOUND If you were looking for a file on tape, and END-OF-TAPE marker was found. If you were looking on disk, no file with that name exists.

FILE NOT OPEN The file specified in a CLOSE, CMD, PRINT#, INPUT#, or GET#, must first be OPENed.

FILE OPEN An attempt was made to open a file using the number of an already open file.

FORMULA TOO COMPLEX The string expression being evaluated should be split into at least two parts for the system to work with, or a formula has too many parentheses.

ILLEGAL DIRECT The INPUT statement can only be used within a program, and not in direct mode.

ILLEGAL QUANTITY A number used as the argument of a function or statement is out of the allowable range.

LOAD There is a problem with the program on tape.

NEXT WITHOUT FOR This is caused by either incorrectly nesting loops or having a variable name in a NEXT statement that doesn't correspond with one in a FOR statement.

NOT INPUT FILE An attempt was made to INPUT or GET data from a file which was specified to be for output only.

NOT OUTPUT FILE An attempt was made to PRINT data to a file which was specified as input only.

OUT OF DATA A READ statement was executed but there is no data left unREAD in a DATA statement.

OUT OF MEMORY There is no more RAM available for program or variables. This may also occur when too many FOR loops have been nested, or when there are too many GOSUBs in effect.

OVERFLOW The result of a computation is larger than the largest number allowed, which is 1.70141884E+38.

REDIM'D ARRAY An array may only be DIMensioned once. If an array variable is used before that array is DIM'd, an automatic DIM operation is performed on that array setting the number of elements to ten, and any subsequent DIMs will cause this error.

REDO FROM START Character data was typed in during an INPUT statement when numeric data was expected. Just re-type the entry so that it is correct, and the program will continue by itself.

RETURN WITHOUT GOSUB A RETURN statement was encountered, and no GOSUB command has been issued.

STRING TOO LONG A string can contain up to 255 characters.

?SYNTAX ERROR A statement is unrecognizable by the Commodore 64. A missing or extra parenthesis, misspelled keywords, etc.

TYPE MISMATCH This error occurs when a number is used in place of a string, or vice-versa.

UNDEF'D FUNCTION A user defined function was referenced, but it has never been defined using the DEF FN statement.

UNDEF'D STATEMENT An attempt was made to GOTO or GOSUB or RUN a line number that doesn't exist.

VERIFY The program on tape or disk does not match the program currently in memory.

APPENDIX L

6510 MICROPROCESSOR CHIP SPECIFICATIONS

DESCRIPTION

The 6510 is a low-cost microcomputer system capable of solving a broad range of small-systems and peripheral-control problems at minimum cost to the user.

An 8-bit Bi-Directional I/O Port is located on-chip with the Output Register at Address 0000 and the Data-Direction Register at Address 0001. The I/O Port is bit-by-bit programmable.

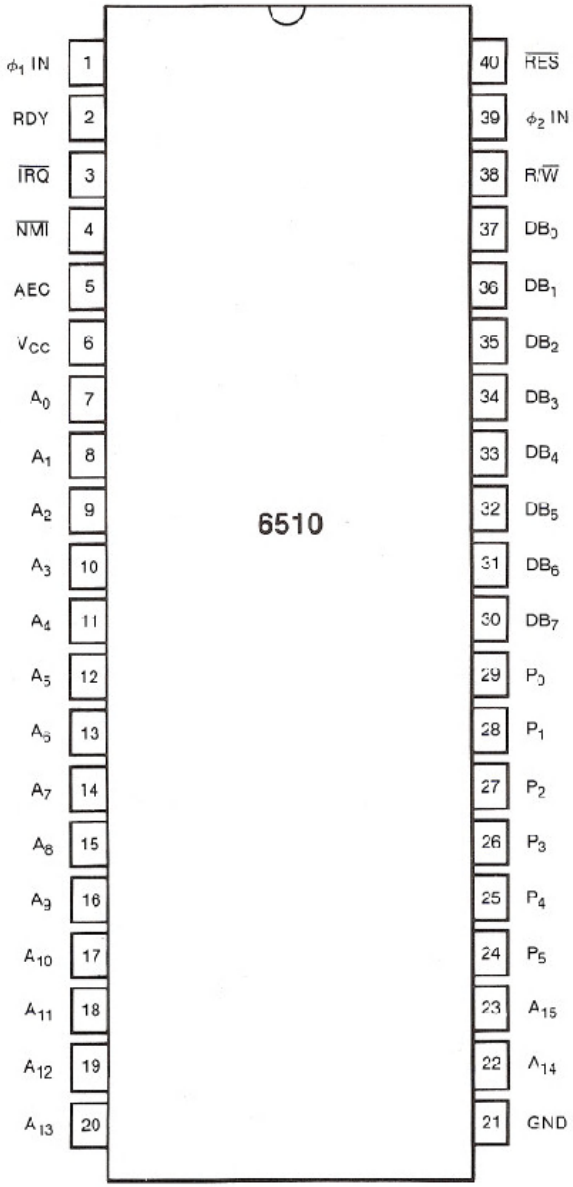
The Three-State sixteen-bit Address Bus allows Direct Memory Accessing (DMA) and multiprocessor systems sharing a common memory.

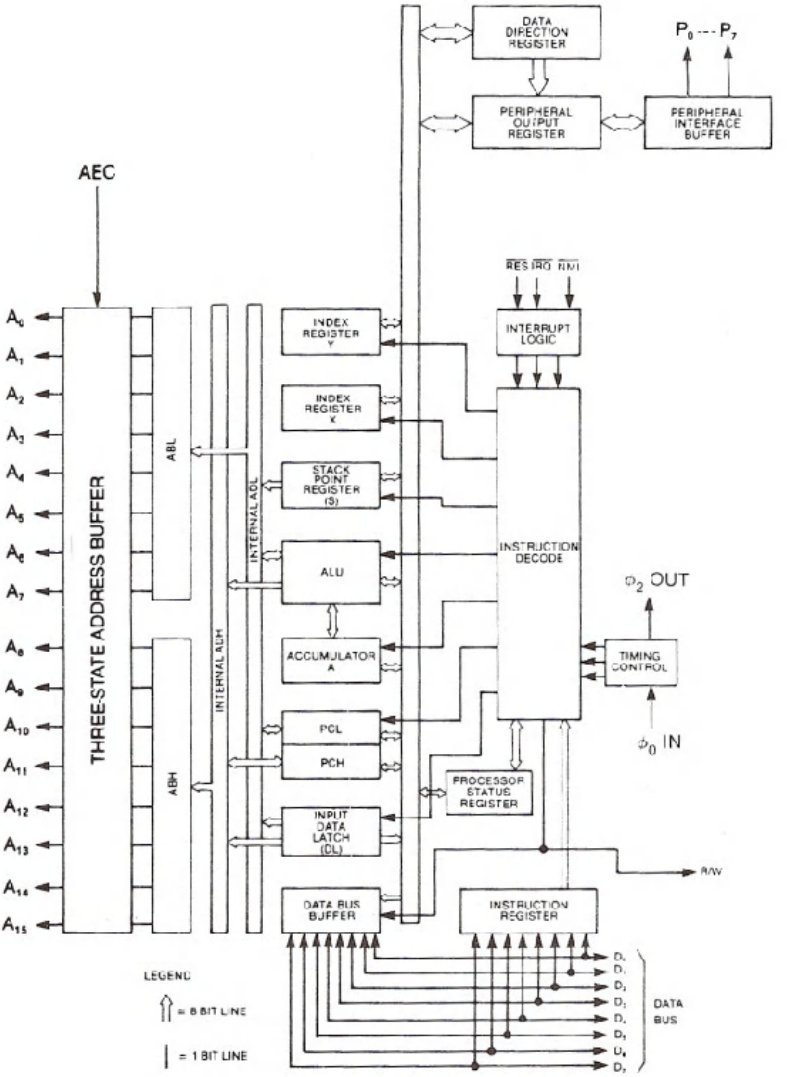
The internal processor architecture is identical to the MOS Technology 6502 to provide software compatibility.

FEATURES OF THE 6510 . . .

- Eight-Bit Bi-Directional I/O Port
- Single +5-volt supply
- N-channel, silicon gate, depletion load technology
- Eight-bit parallel processing
- 56 Instructions
- Decimal and binary arithmetic
- Thirteen addressing modes
- True indexing capability
- Programmable stack pointer
- Variable length stack
- Interrupt capability
- Eight-Bit Bi-Directional Data Bus
- Addressable memory range of up to 65K bytes
- Direct memory access capability
- Bus compatible with M6800
- Pipeline architecture
- 1-MHz and 2-MHz operation
- Use with any type or speed memory

PIN CONFIGURATION





6510 BLOCK DIAGRAM

6510 CHARACTERISTICS

MAXIMUM RATINGS

RATING	SYMBOL	VALUE	UNIT
SUPPLY VOLTAGE	V_{CC}	−0.3 to +7.0	V_{DC}
INPUT VOLTAGE	V_{in}	−0.3 to +7.0	V_{DC}
OPERATING TEMPERATURE	T_A	0 to +70	°C
STORAGE TEMPERATURE	T_{STG}	−55 to +150	°C

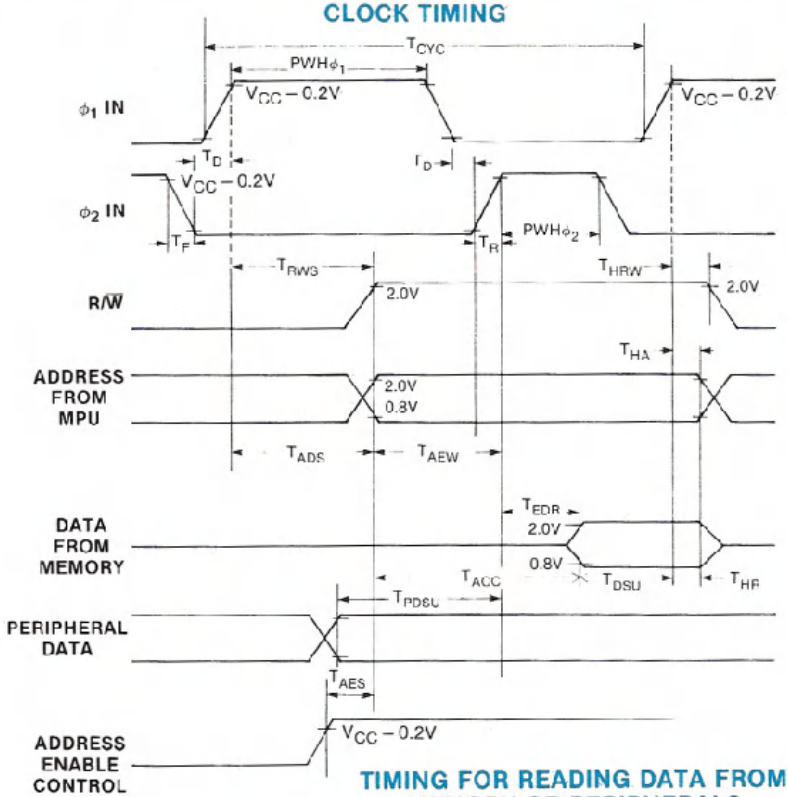
NOTE: This cevice contains input protection against damage due to high static voltages or electric fields; however, precautions should be taken to avoid application of voltages higher than the maximum rating.

ELECTRICAL CHARACTERISTICS
(VCC = 5.0 V ±5%, VSS = 0, T_A = 0° to +70°C)

CHARACTERISTIC	SYMBOL	MIN.	TYP.	MAX.	UNIT
Input High Voltage ϕ_1, $\phi_{2(in)}$	V_{IH}	V_{CC} − 0.2	—	V_{CC} + 1.0V	V_{DC}
Input High Voltage $\overline{RES}$, P_0-P_7, $\overline{IRQ}$, Data		V_{SS} + 2.0	—	—	V_{DC}
Input Low Voltage ϕ_1, $\phi_{2(in)}$	V_{IL}	V_{SS} − 0.3	—	V_{SS} + 0.2	V_{DC}
$\overline{RES}$, P_0-P_7 $\overline{IRQ}$, Data		—	—	V_{SS} + 0.8	V_{DC}
Input Leakage Current (V_{in} = 0 to 5.25V, V_{CC} = 5.25V) Logic	I_{in}	—	—	2.5	μA
ϕ_1, $\phi_{2(in)}$		—	—	100	μA
Three State (Off State) Input Current (V_{in} = 0.4 to 2.4V, V_{CC} = 5.25V) Data Lines	I_{TSI}	—	—	10	μA
Output High Voltage (I_{OH} = −100μA$_{DC}$, V_{CC} = 4.75V) Data, A0-A15, R/W, P_0-P_7	V_{OH}	V_{SS} + 2.4	—	—	V_{DC}

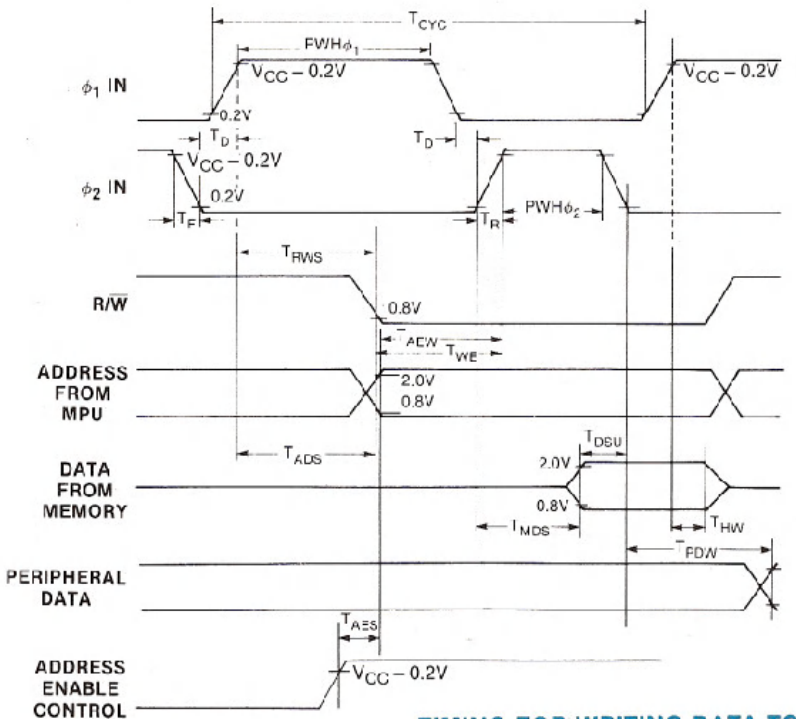
CHARACTERISTIC	SYM-BOL	MIN.	TYP.	MAX.	UNIT
Out Low Voltage (I_{OL} = 1.6mA$_{DC}$, V_{CC} = 4.75V) Data, A0-A15, R/W, P_0-P_7	V_{OL}	—	—	V_{SS} + 0.4	V_{DC}
Power Supply Current	I_{CC}	—	125		mA
Capacitance (V_{in} = 0, T_A = 25°C, f = 1MHz)	C				pF
Logic, P_0-P_7	C_{in}	—	—	10	
Data		—	—	15	
A0-A15, R/W	C_{out}	—	—	12	
ϕ_1	$C\phi_1$	—	30	50	
ϕ_2	$C\phi_2$	—	50	80	

CLOCK TIMING



TIMING FOR READING DATA FROM MEMORY OR PERIPHERALS

CLOCK TIMING



TIMING FOR WRITING DATA TO MEMORY OR PERIPHERALS

AC CHARACTERISTICS

ELECTRICAL CHARACTERISTICS (V_{CC} = 5 V ±5%, V_{SS} = 0 V, T_A = 0°–70°C)

CLOCK TIMING		1MHz TIMING			2 MHz TIMING			
CHARACTERISTIC	SYMBOL	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	UNITS
Cycle Time	T_{CYC}	1000	—	—	500	—	—	ns
Clock Pulse Width $\phi 1$	$PWH\phi 1$	430	—	—	215	—	—	ns
(Measured at V_{CC} − 0.2V) $\phi 2$	$PWH\phi 2$	470	—	—	235	—	—	ns
Fall Time, Rise Time (Measured from 0.2V to V_{CC} − 0.2V)	T_F, T_R	—	—	25	—	—	15	ns
Delay Time between Clocks (Measured at 0.2V)	T_D	0	—	—	0	—	—	ns

READ/WRITE TIMING (LOAD = 1TTL)		1 MHz TIMING			2 MHz TIMING			
CHARACTERISTIC	SYMBOL	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.	UNITS
Read/Write Setup Time from 6508	T_{RWS}	—	100	300	—	100	150	ns
Address Setup Time from 6508	T_{ADS}	—	100	300	—	100	150	ns
Memory Read Access Time	T_{ACC}	—	—	575	—	—	300	ns

Data Stability Time Period	T_{DSU}	100	—	—	50			ns
Data Hold Time-Read	T_{HR}		—	—				ns
Data Hold Time-Write	T_{HW}	10	30	—	10	30		ns
Data Setup Time from 6510	T_{MDS}	—	150	200	—	75	100	ns
Address Hold Time	T_{HA}	10	30		10	30		ns
R/W Hold Time	T_{HRW}	10	30	—	10	30		ns
Delay Time, Address valid to $\phi 2$ positive transition	T_{AEW}	180	—	—				ns
Delay Time, $\phi 2$ positive transition to Data valid on bus	T_{EDR}	—	—	395				ns
Delay Time, Data valid to $\phi 2$ negative transition	T_{DSU}	300	—	—				ns
Delay Time, R/W negative transition to $\phi 2$ positive transition	T_{WE}	130	—	—				ns
Delay Time, $\phi 2$ negative transition to Peripheral Data valid	T_{PDW}	—	—	1				μs
Peripheral Data Setup Time	T_{PDSU}	300	—	—				ns
Address Enable Setup Time	T_{AES}			60			60	ns

SIGNAL DESCRIPTION

Clocks (ϕ_1, ϕ_2)

The 6510 requires a two-phase non-overlapping clock that runs at the V_{CC} voltage level.

Address Bus (A_0-A_{15})

These outputs are TTL compatible, capable of driving one standard TTL load and 130 pf.

Data Bus (D_0-D_7)

Eight pins are used for the data bus. This is a Bi-Directional bus, transferring data to and from the device and peripherals. The outputs are tri-state buffers capable of driving one standard TTL load and 130 pf.

Reset

This input is used to reset or start the microprocessor from a power down condition. During the time that this line is held low, writing to or from the microprocessor is inhibited. When a positive edge is detected on the input, the microprocessor will immediately begin the reset sequence.

After a system initialization time of six clock cycles, the mask interrupt flag will be set and the microprocessor will load the program counter from the memory vector locations FFFC and FFFD. This is the start location for program control.

After V_{CC} reaches 4.75 volts in a power-up routine, reset must be held low for at least two clock cycles. At this time the R/W signal will become valid.

When the reset signal goes high following these two clock cycles, the microprocessor will proceed with the normal reset procedure detailed above.

Interrupt Request ($\overline{IRQ}$)

This TTL level input requests that an interrupt sequence begin within the microprocessor. The microprocessor will complete the current instruction being executed before recognizing the request. At that time, the interrupt mask bit in the Status Code Register will be examined. If the interrupt mask flag is not set, the microprocessor will begin an interrupt sequence. The Program Counter and Processor Status Register are stored in the stack. The microprocessor will then set the interrupt mask

flag high so that no further interrupts may occur. At the end of this cycle, the program counter low will be loaded from address FFFE, and program counter high from location FFFF, therefore transferring program control to the memory vector located at these addresses.

Address Enable Control (AEC)

The Address Bus is valid only when the Address Enable Control line is high. When low, the Address Bus is in a high-impedance state. This feature allows easy DMA and multiprocessor systems.

I/O Port (P_0-P_5)

Six pins are used for the peripheral port, which can transfer data to or from peripheral devices. The Output Register is located in RAM at Address 0001, and the Data Direction Register is at Address 0000. The outputs are capable of driving one standard TTL load and 130 pf.

Read/Write (R/W)

This signal is generated by the microprocessor to control the direction of data transfers on the Data Bus. This line is high except when the microprocessor is writing to memory or a peripheral device.

ADDRESSING MODES

ACCUMULATOR ADDRESSING—This form of addressing is represented with a one byte instruction, implying an operation on the accumulator.

IMMEDIATE ADDRESSING—In immediate addressing, the operand is contained in the second byte of the instruction, with no further memory addressing required.

ABSOLUTE ADDRESSING—In absolute addressing, the second byte of the instruction specifies the eight low order bits of the effective address while the third byte specifies the eight high order bits. Thus, the absolute addressing mode allows access to the entire 65K bytes of addressable memory.

ZERO PAGE ADDRESSING—The zero page instructions allow for shorter code and execution times by only fetching the second byte of the instruction and assuming a zero high address byte. Careful use of the zero page can result in significant increase in code efficiency.

INDEXED ZERO PAGE ADDRESSING—(X, Y indexing)—This form of addressing is used in conjunction with the index register and is referred to as "Zero Page, X" or "Zero Page, Y." The effective address is calculated by adding the second byte to the contents of the index register. Since this is a form of "Zero Page" addressing, the content of the second byte references a location in page zero. Additonally, due to the "Zero Page" addressing nature of this mode, no carry is added to the high order 8 bits of memory and crossing of page boundaries does not occur.

INDEXED ABSOLUTE ADDRESSING—(X, Y indexing)—This form of addressing is used in conjunction with X and Y index register and is referred to as "Absolute, X," and "Absolute, Y." The effective address is formed by adding the contents of X and Y to the address contained in the second and third bytes of the instruction. This mode allows the index register to contain the index or count value and the instruction to contain the base address. This type of indexing allows any location referencing and the index to modify multiple fields resulting in reduced coding and execution time.

IMPLIED ADDRESSING—In the implied addressing mode, the address containing the operand is implicitly stated in the operation code of the instruction.

RELATIVE ADDRESSING—Relative addressing is used only with branch instructions and establishes a destination for the conditional branch.

The second byte of the instruction becomes the operand which is an "Offset" added to the contents of the lower eight bits of the program counter when the counter is set at the next instruction. The range of the offset is −128 to +127 bytes from the next instruction.

INDEXED INDIRECT ADDRESSING—In indexed indirect addressing (referred to as [Indirect, X]), the second byte of the instruction is added to the contents of the X index register, discarding the carry. The result of this addition points to a memory location on page zero whose contents is the low order eight bits of the effective address. The next memory location in page zero contains the high order eight bits of the effective address. Both memory locations specifying the high and low order bytes of the effective address must be in page zero.

INDIRECT INDEXED ADDRESSING—In indirect indexed addressing (referred to as [Indirect], Y), the second byte of the instruction points to a memory location in page zero. The contents of this memory location is

added to the contents of the Y index register, the result being the low order eight bits of the effective address. The carry from this addition is added to the contents of the next page zero memory location, the result being the high order eight bits of the effective address.

ABSOLUTE INDIRECT—The second byte of the instruction contains the low order eight bits of a memory location. The high order eight bits of that memory location is contained in the third byte of the instruction. The contents of the fully specified memory location is the low order byte of the effective address. The next memory location contains the high order byte of the effective address which is loaded into the sixteen bits of the program counter.

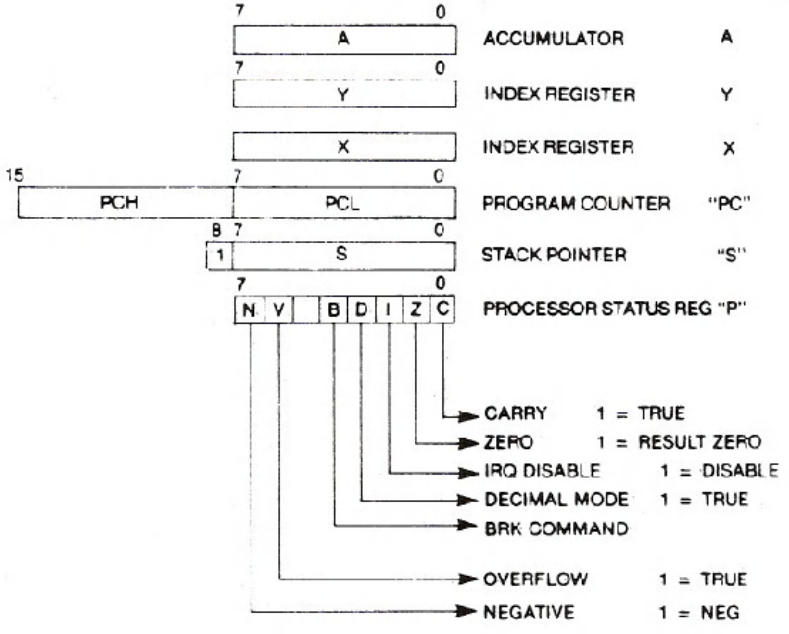
INSTRUCTION SET—ALPHABETIC SEQUENCE

ADC	Add Memory to Accumulator with Carry
AND	"AND" Memory with Accumulator
ASL	Shift Left One Bit (Memory or Accumulator)
BCC	Branch on Carry Clear
BCS	Branch on Carry Set
BEQ	Branch on Result Zero
BIT	Test Bits in Memory with Accumulator
BMI	Branch on Result Minus
BNE	Branch on Result not Zero
BPL	Branch on Result Plus
BRK	Force Break
BVC	Branch on Overflow Clear
BVS	Branch on Overflow Set
CLC	Clear Carry Flag
CLD	Clear Decimal Mode
CLI	Clear Interrupt Disable Bit
CLV	Clear Overflow Flag
CMP	Compare Memory and Accumulator
CPX	Compare Memory and Index X
CPY	Compare Memory and Index Y

DEC	Decrement Memory by One
DEX	Decrement Index X by One
DEY	Decrement Index Y by One
EOR	"Exclusive-OR" Memory with Accumulator
INC	Increment Memory by One
INX	Increment Index X by One
INY	Increment Index Y by One
JMP	Jump to New Location
JSR	Jump to New Location Saving Return Address
LDA	Load Accumulator with Memory
LDX	Load Index X with Memory
LDY	Load Index Y with Memory
LSR	Shift One Bit Right (Memory or Accumulator)
NOP	No Operation
ORA	"OR" Memory with Accumulator
PHA	Push Accumulator on Stack
PHP	Push Processor Status on Stack
PLA	Pull Accumulator from Stack
PLP	Pull Processor Status from Stack
ROL	Rotate One Bit Left (Memory or Accumulator)
ROR	Rotate One Bit Right (Memory or Accumulator)
RTI	Return from Interrupt
RTS	Return from Subroutine
SBC	Subtract Memory from Accumulator with Borrow
SEC	Set Carry Flag
SED	Set Decimal Mode
SEI	Set Interrupt Disable Status
STA	Store Accumulator in Memory
STX	Store Index X in Memory
STY	Store Index Y in Memory

TAX	Transfer Accumulator to Index X
TAY	Transfer Accumulator to Index Y
TSX	Transfer Stack Pointer to Index X
TXA	Transfer Index X to Accumulator
TXS	Transfer Index X to Stack Register
TYA	Transfer Index Y to Accumulator

PROGRAMMING MODEL



INSTRUCTIONS		Immediate			Absolute			Zero Page			Accum.			Implied			(Ind.) X			(Ind.) Y			Z. Page, X			Abs. X			Abs. Y			Relative			Indirect			Z. Page, Y			CONDITION CODES						
Mnemonic	Operation	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	N	Z	C	I	D	V
ADC	A + M + C→A (4) (1)	69	2	2	6D	4	3	65	3	2							61	6	2	71	5	2	75	4	2	7D	4	3	79	4	3										✓	✓	✓	—	—	✓
AND	A∧M→A (1)	29	2	2	2D	4	3	25	3	2							21	6	2	31	5	2	35	4	2	3D	4	3	39	4	3										✓	✓	—	—	—	—
ASL	C←[7 … 0]←0				0E	6	3	06	5	2	0A	2	1										16	6	2	1E	7	3													✓	✓	✓	—	—	—
BCC	BRANCH ON C = 0 (2)																															90	2	2							—	—	—	—	—	—
BCS	BRANCH ON C = 1 (2)																															B0	2	2							—	—	—	—	—	—
BEQ	BRANCH ON Z = 1 (2)																															F0	2	2							—	—	—	—	—	—
BIT	A∧M				2C	4	3	24	3	2																															M_7	✓	—	—	—	M_6
BMI	BRANCH ON N = 1 (2)																															30	2	2							—	—	—	—	—	—
BNE	BRANCH ON Z = 0 (2)																															D0	2	2							—	—	—	—	—	—
BPL	BRANCH ON N = 0 (2)																															10	2	2							—	—	—	—	—	—
BRK	(See Fig. 1)													00	7	1																									—	—	—	1	—	—
BVC	BRANCH ON V = 0 (2)																															50	2	2							—	—	—	—	—	—
BVS	BRANCH ON V = 1 (2)																															70	2	2							—	—	—	—	—	—
CLC	0→C													18	2	1																									—	—	0	—	—	—
CLD	0→D													D8	2	1																									—	—	—	—	0	—
CLI	0→I													58	2	1																									—	—	—	0	—	—
CLV	0→V													B8	2	1																									—	—	—	—	—	0
CMP	A − M (1)	C9	2	2	CD	4	3	C5	3	2							C1	6	2	D1	5	2	D5	4	2	DD	4	3	D9	4	3										✓	✓	✓	—	—	—
CPX	X − M	E0	2	2	EC	4	3	E4	3	2																															✓	✓	✓	—	—	—
CPY	Y − M	C0	2	2	CC	4	3	C4	3	2																															✓	✓	✓	—	—	—
DEC	M − 1→M				CE	6	3	C6	5	2													D6	6	2	DE	7	3													✓	✓	—	—	—	—
DEX	X − 1→X													CA	2	1																									✓	✓	—	—	—	—
DEY	Y − 1→Y													88	2	1																									✓	✓	—	—	—	—
EOR	A∨M→A (1)	49	2	2	4D	4	3	45	3	2							41	6	2	51	5	2	55	4	2	5D	4	3	59	4	3										✓	✓	—	—	—	—
INC	M + 1→M				EE	6	3	E6	5	2													F6	6	2	FE	7	3													✓	✓	—	—	—	—
INX	X + 1→X													E8	2	1																									✓	✓	—	—	—	—
INY	Y + 1→Y													C8	2	1																									✓	✓	—	—	—	—
JMP	JUMP TO NEW LOC.				4C	3	3																													6C	5	3				—	—	—	—	—	—
JSR	(See Fig. 2) JUMP SUB.				20	6	3																																			—	—	—	—	—	—
LDA	M→A (1)	A9	2	2	AD	4	3	A5	3	2							A1	6	2	B1	5	2	B5	4	2	BD	4	3	B9	4	3										✓	✓	—	—	—	—

INSTRUCTIONS		Immediate			Absolute			Zero Page			Accum			Implied			(Ind.,X			(Ind.)Y			Z. Page, X			Abs. X			Abs. Y			Relative			Indirect			Z. Page, Y			CONDITION CODES
Mnemonic	Operation	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	OP	N	#	N Z C I D V
LDX	M→X (1)	A2	2	2	AE	4	3	A6	3	2																			BE	4	3							B6	4	2	↓ ↓ — — — —
LDY	M→Y (1)	A0	2	2	AC	4	3	A4	3	2													B4	4	2	BC	4	3												0 ↓ ↓ — — —
LSR	0→[7 0]→C				4E	6	3	46	5	2	4A	2	1										56	6	2	5E	7	3												↓ ↓ — — —
NOP	NO OPERATION													EA	2	1																								— — — — —
ORA	AVM→A	09	2	2	0D	4	3	05	3	2							01	6	2	11	5	2	15	4	2	1D	4	3	19	4	3										↓ ↓ — — —
PHA	A→M_S S−1→S													48	3	1																								— — — — —
PHP	P→M_S S−1→S													08	3	1																								— — — — —
PLA	S+1→S M_S→A													68	4	1																								↓ ↓ — — —
PLP	S+1→S M_S→P													28	4	1																								(RESTORED)
ROL	[7 0]←[C]				2E	6	3	26	5	2	2A	2	1										36	6	2	3E	7	3												↓ ↓ ↓ — —
ROR	[C]→[7 0]				6E	6	3	66	5	2	6A	2	1										76	6	2	7E	7	3												↓ ↓ ↓ — —
RTI	(See Fig. 1) RTRN INT													4D	6	1																								(RESTORED)
RTS	(See Fig. 2) RTRN SUB													60	6	1																								— — — — —
SBC	A−M−$\overline{C}$→A (1)	E9	2	2	ED	4	3	E5	3	2							E1	6	2	F1	5	2	F5	4	2	FD	4	3	F9	4	3										↓ ↓ (3) — — ↓
SEC	1→C													38	2	1																								— — 1 — —
SED	1→D													F8	2	1																								— — — — 1 —
SEI	1→I													78	2	1																								— — — 1 — —
STA	A→M				8D	4	3	85	3	2							81	6	2	91	6	2	95	4	2	9D	5	3	99	5	3										— — — — —
STX	X→M				8E	4	3	86	3	2																													96	4	2	— — — — —
STY	Y→M				8C	4	3	84	3	2													94	4	2																— — — — —
TAX	A→X													AA	2	1																								↓ ↓ — — —
TAY	A→Y													A8	2	1																								↓ ↓ — — —
TSX	S→X													BA	2	1																								↓ ↓ — — —
TXA	X→A													8A	2	1																								↓ ↓ — — —
TXS	X→S													9A	2	1																								— — — — —
TYA	Y→A													98	2	1																								↓ ↓ — — —

(1) ADD 1 TO "N" IF PAGE BOUNDARY IS CROSSED.
(2) ADD 1 TO "N" IF BRANCH OCCURS TO SAME PAGE.
ADD 2 TO "N" IF BRANCH OCCURS TO DIFFERENT PAGE.
(3) CARRY NOT = BORROW
(4) IF IN DECIMAL MODE Z FLAG IS INVALID.
ACCUMULATOR MUST BE CHECKED FOR ZERO RESULT.

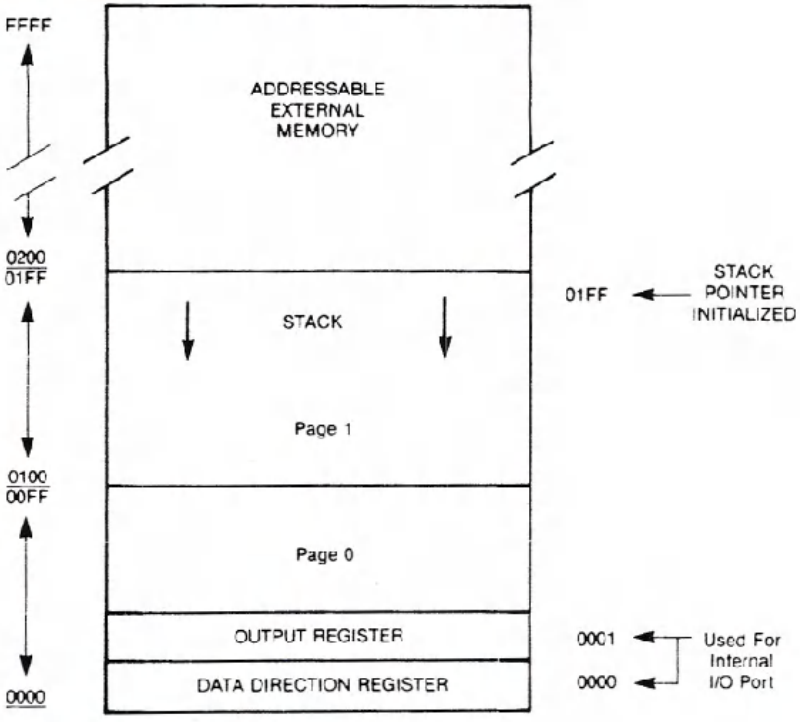
X INDEX X
Y INDEX Y
A ACCUMULATOR
M MEMORY PER EFFECTIVE ADDRESS
M_S MEMORY PER STACK POINTER

\+ ADD
− SUBTRACT
∧ AND
∨ OR
∀ EXCLUSIVE OR

↓ MODIFIED
— NOT MODIFIED
M_7 MEMORY BIT 7
M_6 MEMORY BIT 6
N NO. CYCLES
NO. BYTES

NOTE: **COMMODORE SEMICONDUCTOR GROUP** cannot assume liability for the use of undefined OP CODES.

6510 MEMORY MAP



APPLICATIONS NOTES

Locating the Output Register at the internal I/O Port in Page Zero enhances the powerful Zero Page Addressing instructions of the 6510.

By assigning the I/O Pins as inputs (using the Data Direction Register) the user has the ability to change the contents of address 0001 (the Output Register) using peripheral devices. The ability to change these contents using peripheral inputs, together with Zero Page Indirect Addressing instructions, allows novel and versatile programming techniques not possible earlier.

APPENDIX M

6526 COMPLEX INTERFACE ADAPTER (CIA) CHIP SPECIFICATIONS

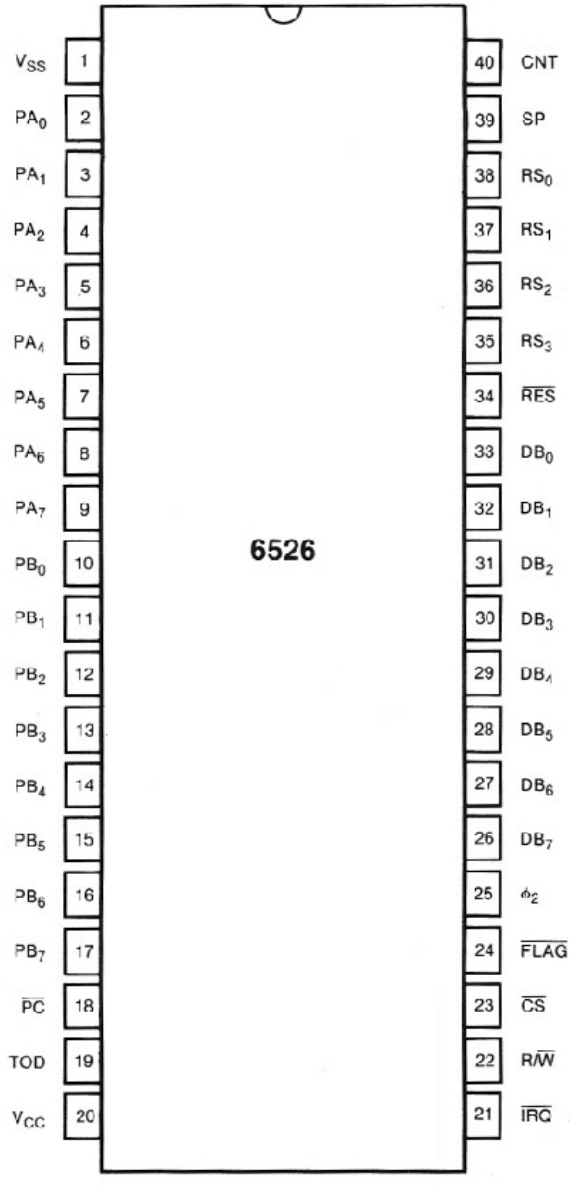
DESCRIPTION

The 6526 Complex Interface Adapter (CIA) is a 65XX bus compatible peripheral interface device with extremely flexible timing and I/O capabilities.

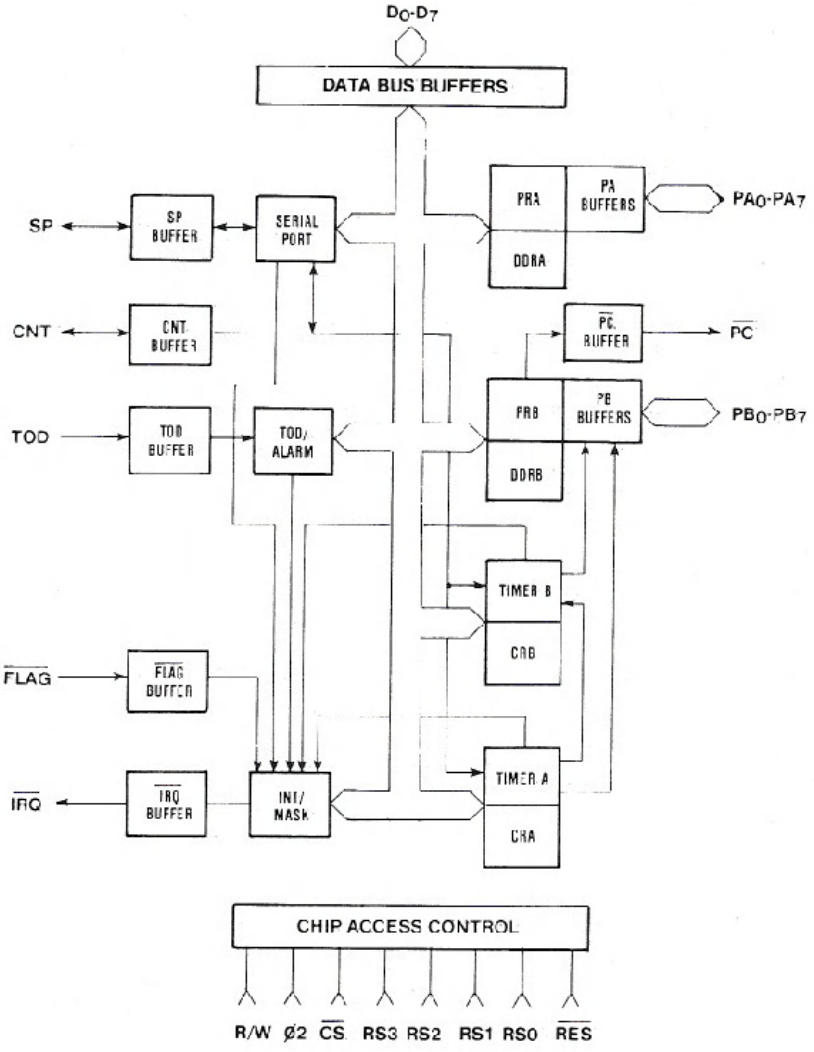
FEATURES

- 16 Individually programmable I/O lines
- 8 or 16-Bit handshaking on read or write
- 2 independent, linkable 16-Bit interval timers
- 24-hour (AM/PM) time of day clock with programmable alarm
- 8-Bit shift register for serial I/O
- 2TTL Load capability
- CMOS compatible I/O lines
- 1 or 2 MHz operation available

PIN CONFIGURATION



6526
BLOCK DIAGRAM



MAXIMUM RATINGS

Supply Voltage, V_{CC}	−0.3V to +7.0V
Input/Output Voltage, V_{IN}	−0.3V to +7.0V
Operating Temperature, T_{OP}	0° C to 70° C
Storage Temperature, T_{STG}	−55° C to 150° C

All inputs contain protection circuitry to prevent damage due to high static discharges. Care should be exercised to prevent unnecessary application of voltages in excess of the allowable limits.

COMMENT

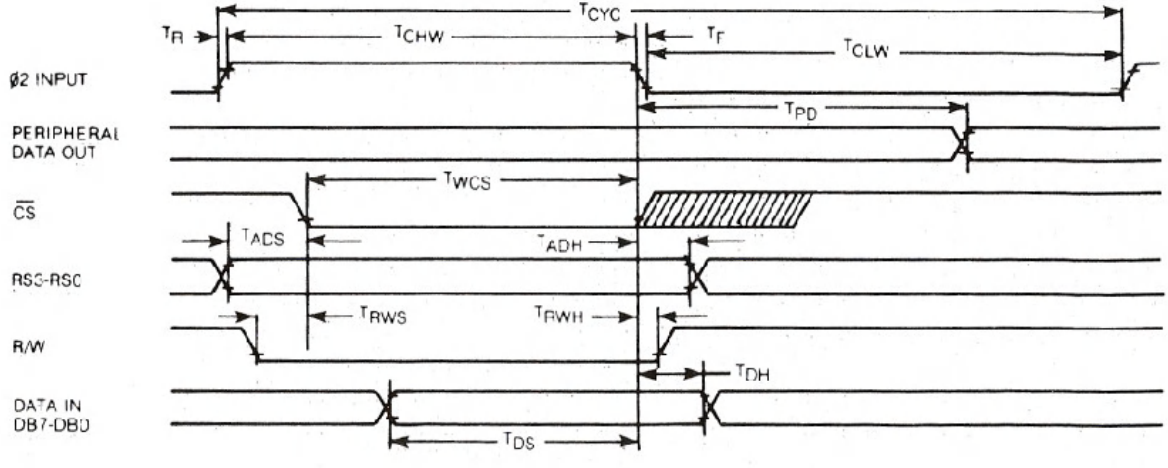
Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only. Functional operation of this device at these or any other conditions above those indicated in the operational sections of this specification is not implied and exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS (V_{CC} ± 5%, V_{SS} = 0 V, T_A = 0–70°C)

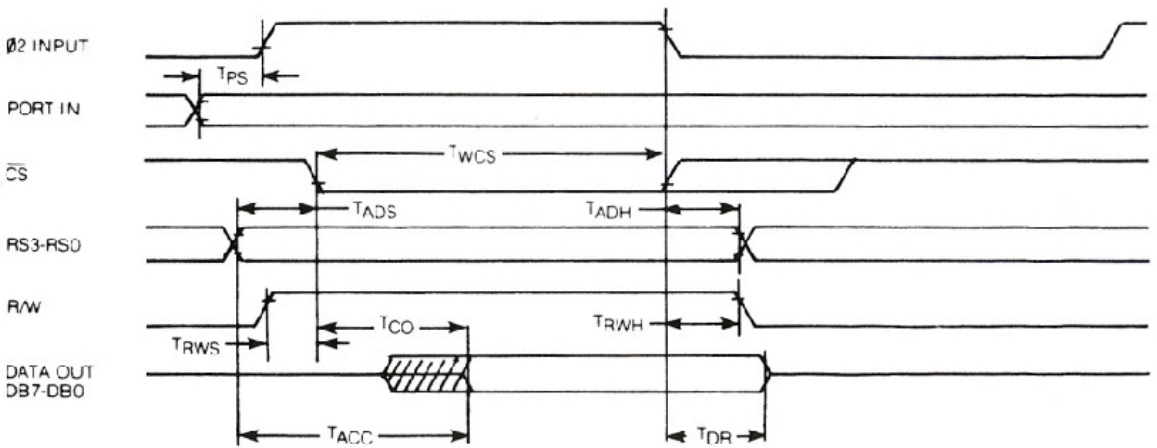
CHARACTERISTIC	SYMBOL	MIN.	TYP.	MAX.	UNIT
Input High Voltage	V_{IH}	+2.4	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	—	V
Input Leakage Current, $V_{IN}=V_{SS}$ +5V ($\overline{TOD}$, R/W, $\overline{FLAG}$, $\phi 2$, $\overline{RES}$, RS0-RS3, $\overline{CS}$)	I_{IN}	—	1.0	2.5	μA

CHARACTERISTIC	SYMBOL	MIN.	TYP.	MAX.	UNIT
Port Input Pull-up Resistance	R_{PI}	3.1	5.0	—	KΩ
Output Leakage Current for High Impedance State (Three State); V_{IN} = 4V to 2.4V; (DB0–DB7, SP, CNT, $\overline{IRQ}$)	I_{TSI}	—	±1.0	±10.0	μA
Output High Voltage V_{CC}=MIN, I_{LOAD} < −200μA (PA0–PA7, $\overline{PC}$ PB0–PB7, DB0–DB7)	V_{OH}	+2.4	—	V_{CC}	V
Output Low Voltage V_{CC}= MIN, I_{LOAD} < 3.2 mA	V_{OL}	—	—	+0.40	V
Output High Current (Sourcing); V_{OH} > 2.4V (PA0–PA7, PB0–PB7, PC, DB0–DB7	I_{OH}	−200	−1000	—	μA
Output Low Current (Sinking); V_{OL} < .4V (PA0–PA7, $\overline{PC}$, PB0–PB7, DB0–DB7)	I_{OL}	3.2	—	—	mA
Input Capacitance	C_{IN}	—	7	10	pf
Output Capacitance	C_{OUT}	—	7	10	pf
Power Supply Current	I_{CC}	—	70	100	mA

6526 WRITE TIMING DIAGRAM



6526 READ TIMING DIAGRAM



6526 INTERFACE SIGNALS

$\phi 2$—Clock Input

The $\phi 2$ clock is a TTL compatible input used for internal device operation and as a timing reference for communicating with the system data bus.

$\overline{CS}$—Chip Select Input

The $\overline{CS}$ input controls the activity of the 6526. A low level on $\overline{CS}$ while $\phi 2$ is high causes the device to respond to signals on the R/W and address (RS) lines. A high on $\overline{CS}$ prevents these lines from controlling the 6526. The $\overline{CS}$ line is normally activated (low) at $\phi 2$ by the appropriate address combination.

R/W—Read/Write Input

The R/W signal is normally supplied by the microprocessor and controls the direction of data transfers of the 6526. A high on R/W indicates a read (data transfer out of the 6526), while a low indicates a write (data transfer into the 6526).

RS3–RS0—Address Inputs

The address inputs select the internal registers as described by the Register Map.

DB7–BD0—Data Bus Inputs/Outputs

The eight data bus pins transfer information between the 6526 and the system data bus. These pins are high impedance inputs unless $\overline{CS}$ is low and R/W and $\phi 2$ are high to read the device. During this read, the data bus output buffers are enabled, driving the data from the selected register onto the system data bus.

$\overline{IRQ}$—Interrupt Request Output

$\overline{IRQ}$ is an open drain output normally connected to the processor interrupt input. An external pullup resistor holds the signal high, allowing multiple $\overline{IRQ}$ outputs to be connected together. The $\overline{IRQ}$ output is normally off (high impedance) and is activated low as indicated in the functional description.

$\overline{RES}$—Reset Input

A low on the $\overline{RES}$ pin resets all internal registers. The port pins are set as inputs and port registers to zero (although a read of the ports will return all highs because of passive pullups). The timer control registers are set to zero and the timer latches to all ones. All other registers are reset to zero.

6526 TIMING CHARACTERISTICS

Symbol	Characteristic	1MHz MIN	1MHz MAX	2MHz MIN	2MHz MAX	Unit
	ϕ2 Clock					
T_{CYC}	Cycle Time	1000	20,000	500	20,000	ns
T_R, T_F	Rise and Fall Time	—	25	—	25	ns
T_{CHW}	Clock Pulse Width (High)	420	10,000	200	10,000	ns
T_{CLW}	Clock Pulse Width (Low)	420	10,000	200	10,000	ns
	Write Cycle					
T_{PD}	Output Delay From ϕ2	—	1000	—	500	ns
T_{WCS}	$\overline{CS}$ low while ϕ2 high	420	—	200	—	ns
T_{ADS}	Address Setup Time	0	—	0	—	ns
T_{ADH}	Address Hold Time	10	—	5	—	ns
T_{RWS}	R/W Setup Time	0	—	0	—	ns
T_{RWH}	R/W Hold Time	0	—	0	—	ns
T_{DS}	Data Bus Setup Time	150	—	75	—	ns
T_{DH}	Data Bus Hold Time	0	—	0	—	ns
	Read Cycle					
T_{PS}	Port Setup Time	300	—	150	—	ns
T_{WCS}(2)	$\overline{CS}$ low while ϕ2 high	420	—	20	—	ns
T_{ADS}	Address Setup Time	0	—	0	—	ns
T_{ADH}	Address Hold Time	10	—	5	—	ns
T_{RWS}	R/W Setup Time	0	—	0	—	ns
T_{RWH}	R/W Hold Time	0	—	0	—	ns

Symbol	Characteristic	1MHz		2MHz		Unit
		MIN	MAX	MIN	MAX	
T_{ACC}	Data Access from RS3-RS0	—	550	—	275	ns
T_{CO}(3)	Data Access from $\overline{CS}$	—	320	—	150	ns
T_{DR}	Data Release Time	50	—	25	—	ns

NOTES: 1—All timings are referenced from V_{IL} max and V_{IH} min on inputs and V_{OL} max and V_{OH} min on outputs.

2—T_{WCS} is measured from the later of $\phi 2$ high or $\overline{CS}$ low. $\overline{CS}$ must be low at least until the end of $\phi 2$ high.

3—T_{CO} is measured from the later of $\phi 2$ high or $\overline{CS}$ low. Valid data is available only after the later of T_{ACC} or T_{CO}.

REGISTER MAP

RS3	RS2	RS1	RS0	REG	NAME	
0	0	0	0	0	PRA	PERIPHERAL DATA REG A
0	0	0	1	1	PRB	PERIPHERAL DATA REG B
0	0	1	0	2	DDRA	DATA DIRECTION REG A
0	0	1	1	3	DDRB	DATA DIRECTION REG B
0	1	0	0	4	TA LO	TIMER A LOW REGISTER
0	1	0	1	5	TA HI	TIMER A HIGH REGISTER
0	1	1	0	6	TB LO	TIMER B LOW REGISTER
0	1	1	1	7	TB HI	TIMER B HIGH REGISTER
1	0	0	0	8	TOD 10THS	10THS OF SECONDS REGISTER
1	0	0	1	9	TOD SEC	SECONDS REGISTER
1	0	1	0	A	TOD MIN	MINUTES REGISTER
1	0	1	1	B	TOD HR	HOURS—AM/PM REGISTER
1	1	0	0	C	SDR	SERIAL DATA REGISTER
1	1	0	1	D	ICR	INTERRUPT CONTROL REGISTER
1	1	1	0	E	CRA	CONTROL REG A
1	1	1	1	F	CRB	CONTROL REG B

6526 FUNCTIONAL DESCRIPTION

I/O Ports (PRA, PRB, DDRA, DDRB).

Ports A and B each consist of an 8-bit Peripheral Data Register (PR) and an 8-bit Data Direction Register (DDR). If a bit in the DDR is set to a one, the corresponding bit in the PR is an output; if a DDR bit is set to a zero, the corresponding PR bit is defined as an input. On a READ, the PR reflects the information present on the actual port pins (PA0–PA7, PB0–PB7) for both input and output bits. Port A and Port B have passive pull-up devices as well as active pull-ups, providing both CMOS and TTL compatibility. Both ports have two TTL load drive capability. In addition to normal I/O operation, PB6 and PB7 also provide timer output functions.

Handshaking

Handshaking on data transfers can be accomplished using the $\overline{PC}$ output pin and the $\overline{FLAG}$ input pin. $\overline{PC}$ will go low for one cycle following a read or write of PORT B. This signal can be used to indicate "data ready" at PORT B or "data accepted" from PORT B. Handshaking on 16-bit data transfers (using both PORT A and PORT B) is possible by always reading or writing PORT A first. $\overline{FLAG}$ is a negative edge sensitive input which can be used for receiving the $\overline{PC}$ output from another 6526, or as a general purpose interrupt input. Any negative transition of $\overline{FLAG}$ will set the $\overline{FLAG}$ interrupt bit.

REG	NAME	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	PRA	PA_7	PA_6	PA_5	PA_4	PA_3	PA_2	PA_1	PA_0
1	PRB	PB_7	PB_6	PB_5	PB_4	PB_3	PB_2	PB_1	PB_0
2	DDRA	DPA_7	DPA_6	DPA_5	DPA_4	DPA_3	DPA_2	DPA_1	DPA_0
3	DDRB	DPB_7	DPB_6	DPB_5	DPB_4	DPB_3	DPB_2	DPB_1	DPB_0

Interval Timers (Timer A, Timer B)

Each interval timer consists of a 16-bit read-only Timer Counter and a 16-bit write-only Timer Latch. Data written to the timer are latched in the Timer Latch, while data read from the timer are the present contents of the Time Counter. The timers can be used independently or linked for extended operations. The various timer modes allow generation of long time delays, variable width pulses, pulse trains and variable frequency

waveforms. Utilizing the CNT input, the timers can count external pulses or measure frequency, pulse width and delay times of external signals. Each timer has an associated control register, providing independent control of the following functions:

Start/Stop

A control bit allows the timer to be started or stopped by the microprocessor at any time.

PB On/Off:

A control bit allows the timer output to appear on a PORT B output line (PB6 for TIMER A and PB7 for TIMER B). This function overrides the DDRB control bit and forces the appropriate PB line to an output.

Toggle/Pulse

A control bit selects the output applied to PORT B. On every timer underflow the output can either toggle or generate a single positive pulse of one cycle duration. The Toggle output is set high whenever the timer is started and is set low by $\overline{\text{RES}}$.

One-Shot/Continuous

A control bit selects either timer mode. In one-shot mode, the timer will count down from the latched value to zero, generate an interrupt, reload the latched value, then stop. In continuous mode, the timer will count from the latched value to zero, generate an interrupt, reload the latched value and repeat the procedure continuously.

Force Load

A strobe bit allows the timer latch to be loaded into the timer counter at any time, whether the timer is running or not.

Input Mode:

Control bits allow selection of the clock used to decrement the timer. TIMER A can count $\phi 2$ clock pulses or external pulses applied to the CNT pin. TIMER B can count $\phi 2$ pulses, external CNT pulses, TIMER A underflow pulses or TIMER A underflow pulses while the CNT pin is held high.

The timer latch is loaded into the timer on any timer underflow, on a force load or following a write to the high byte of the prescaler while the timer is stopped. If the timer is running, a write to the high byte will load the timer latch, but not reload the counter.

READ (TIMER)

REG	NAME								
4	TA LO	TAL_7	TAL_6	TAL_5	TAL_4	TAL_3	TAL_2	TAL_1	TAL_0
5	TA HI	TAH_7	TAH_6	TAH_5	TAH_4	TAH_3	TAH_2	TAH_1	TAH_0
6	TB LO	TBL_7	TBL_6	TBL_5	TBL_4	TBL_3	TBL_2	TBL_1	TBL_0
7	TB HI	TBH_7	TBH_6	TBH_5	TBH_4	TBH_3	TBH_2	TBH_1	TBH_0

WRITE (PRESCALER)

REG	NAME								
4	TA LO	PAL_7	PAL_6	PAL_5	PAL_4	PAL_3	PAL_2	PAL_1	PAL_0
5	TA HI	PAH_7	PAH_6	PAH_5	PAH_4	PAH_3	PAH_2	PAH_1	PAH_0
6	TB LO	PBL_7	PBL_6	PBL_5	PBL_4	PBL_3	PBL_2	PBL_1	PBL_0
7	TB HI	PBH_7	PBH_6	PBH_5	PBH_4	PBH_3	PBH_2	PBH_1	PBH_0

Time of Day Clock (TOD)

The TOD clock is a special purpose timer for real-time applications. TOD consists of a 24-hour (AM/PM) clock with 1/10th second resolution. It is organized into 4 registers: 10ths of seconds, Seconds, Minutes and Hours. The AM/PM flag is in the MSB of the Hours register for easy bit testing. Each register reads out in BCD format to simplify conversion for driving displays, etc. The clock requires an external 60 Hz or 50 Hz (programmable) TTL level input on the TOD pin for accurate time-keeping. In addition to time-keeping, a programmable ALARM is provided for generating an interrupt at a desired time. The ALARM registers are located at the same addresses as the corresponding TOD registers. Access to the ALARM is governed by a Control Register bit. The ALARM is write-only; any read of a TOD address will read time regardless of the state of the ALARM access bit.

A specific sequence of events must be followed for proper setting and reading of TOD. TOD is automatically stopped whenever a write to the Hours register occurs. The clock will not start again until after a write to the 10ths of seconds register. This assures TOD will always start at the desired time. Since a carry from one stage to the next can occur at any time with respect to a read operation, a latching function is included to keep all Time Of Day information constant during a read sequence. All four TOD registers latch on a read of Hours and remain latched until after a read of 10ths of seconds. The TOD clock continues to count when

the output registers are latched. If only one register is to be read, there is no carry problem and the register can be read "on the fly," provided that any read of Hours is followed by a read of 10ths of seconds to disable the latching.

READ

REG	NAME								
8	TOD 10THS	0	0	0	0	T_8	T_4	T_2	T_1
9	TOD SEC	0	SH_4	SH_2	SH_1	SL_8	SL_4	SL_2	SL_1
A	TOD MIN	0	MH_4	MH_2	MH_1	ML_8	ML_4	ML_2	ML_1
B	TOD HR	PM	0	0	HH	HL_8	HL_4	HL_2	HL_1

WRITE

CRB_7=0 TOD
CRB_7=1 ALARM
(SAME FORMAT AS READ)

Serial Port (SDR)

The serial port is a buffered, 8-bit synchronous shift register system. A control bit selects input or output mode. In input mode, data on the SP pin is shifted into the shift register on the rising edge of the signal applied to the CNT pin. After 8 CNT pulses, the data in the shift register is dumped into the Serial Data Register and an interrupt is generated. In the output mode, TIMER A is used for the baud rate generator. Data is shifted out on the SP pin at ½ the underflow rate of TIMER A. The maximum baud rate possible is $\phi 2$ divided by 4, but the maximum useable baud rate will be determined by line loading and the speed at which the receiver responds to input data. Transmission will start following a write to the Serial Data Register (provided TIMER A is running and in continuous mode). The clock signal derived from TIMER A appears as an output on the CNT pin. The data in the Serial Data Register will be loaded into the shift register then shift out to the SP pin when a CNT pulse occurs. Data shifted out becomes valid on the falling edge of CNT and remains valid until the next falling edge. After 8 CNT pulses, an interrupt is generated to indicate more data can be sent. If the Serial Data Register was loaded with new information prior to this interrupt, the new data will automatically be loaded into the shift register and transmission will continue. If the microprocessor stays one byte ahead of the shift register, transmission will be continuous. If no further data is to be transmitted, after the 8th CNT pulse, CNT will return high and SP will

remain at the level of the last data bit transmitted. SDR data is shifted out MSB first and serial input data should also appear in this format.

The bidirectional capability of the Serial Port and CNT clock allows many 6526 devices to be connected to a common serial communication bus on which one 6526 acts as a master, sourcing data and shift clock, while all other 6526 chips act as slaves. Both CNT and SP outputs are open drain to allow such a common bus. Protocol for master/slave selection can be transmitted over the serial bus, or via dedicated handshaking lines.

REG	NAME								
C	SDR	S_7	S_6	S_5	S_4	S_3	S_2	S_1	S_0

Interrupt Control (ICR)

There are five sources of interrupts on the 6526: underflow from TIMER A, underflow from TIMER B, TOD ALARM, Serial Port full/empty and $\overline{\text{FLAG}}$. A single register provides masking and interrupt information. The interrupt Control Register consists of a write-only MASK register and a read-only DATA register. Any interrupt will set the corresponding bit in the DATA register. Any interrupt which is enabled by the MASK register will set the IR bit (MSB) of the DATA register and bring the $\overline{\text{IRQ}}$ pin low. In a multi-chip system, the IR bit can be polled to detect which chip has generated an interrupt request. The interrupt DATA register is cleared and the $\overline{\text{IRQ}}$ line returns high following a read of the DATA register. Since each interrupt sets an interrupt bit regardless of the MASK, and each interrupt bit can be selectively masked to prevent the generation of a processor interrupt, it is possible to intermix polled interrupts with true interrupts. However, polling the IR bit will cause the DATA register to clear, therefore, it is up to the user to preserve the information contained in the DATA register if any polled interrupts were present.

The MASK register provides convenient control of individual mask bits. When writing to the MASK register, if bit 7 (SET/$\overline{\text{CLEAR}}$) of the data written is a ZERO, any mask bit written with a one will be cleared, while those mask bits written with a zero will be unaffected. If bit 7 of the data written is a ONE, any mask bit written with a one will be set, while those mask bits written with a zero will be unaffected. In order for an interrupt flag to set IR and generate an Interrupt Request, the corresponding MASK bit must be set.

READ (INT DATA)

REG	NAME								
D	ICR	IR	0	0	FLG	SP	ALRM	TB	TA

WRITE (INT MASK)

REG	NAME								
D	ICR	S/$\overline{C}$	X	X	FLG	SP	ALRM	TB	TA

CONTROL REGISTERS

There are two control registers in the 6526, CRA and CRB. CRA is associated with TIMER A and CRB is associated with TIMER B. The register format is as follows:

CRA:

Bit	Name	Function
0	START	1=START TIMER A, 0=STOP TIMER A. This bit is automatically reset when underflow occurs during one-shot mode.
1	PBON	1=TIMER A output appears on PB6, 0=PB6 normal operation.
2	OUTMODE	1=TOGGLE, 0=PULSE
3	RUNMODE	1=ONE-SHOT, 0=CONTINUOUS
4	LOAD	1=FORCE LOAD (this is a STROBE input, there is no data storage, bit 4 will always read back a zero and writing a zero has no effect).
5	INMODE	1=TIMER A counts positive CNT transitions, 0=TIMER A counts $\phi 2$ pulses.
6	SPMODE	1=SERIAL PORT output (CNT sources shift clock), 0=SERIAL PORT input (external shift clock required).
7	TODIN	1=50 Hz clock required on TOD pin for accurate time, 0=60 Hz clock required on TOD pin for accurate time.

CRB:

Bit	Name	Function
		(Bits CRB0-CRB4 are identical to CRA0-CRA4 for TIMER B with the exception that bit 1 controls the output of TIMER B on PB7).
5,6	INMODE	Bits CRB5 and CRB6 select one of four input modes for TIMER B as:

CRB6	CRB5	
0	0	TIMER B counts $\phi 2$ pulses.
0	1	TIMER B counts positive CNT transistions.
1	0	TIMER B counts TIMER A underflow pulses.
1	1	TIMER B counts TIMER A underflow pulses while CNT is high.

7	ALARM	1=writing to TOD registers sets ALARM, 0=writing to TOD registers sets TOD clock.

REG	NAME	TOD IN	SP MODE	IN MODE	LOAD	RUN MODE	OUT MODE	PB ON	START
E	CRA	0=60Hz	0=INPUT	0=$\phi 2$	1=FORCE LOAD	0=CONT.	0=PULSE	0=PB_6OFF	0=STOP
		1=50Hz	1=OUTPUT	1=CNT	(STROBE)	1=O.S.	1=TOGGLE	1=PB_6 ON	1=START

TA

REG	NAME	ALARM	IN	MODE	LOAD	RUN MODE	OUT MODE	PB ON	START
F	CRB	0=TOD	0	0=$\phi 2$	1=FORCE	0=CONT.	0=PULSE	0=PB_7 OFF	0=STOP
			1	1=CNT	LOAD				
			1	0=TA					
		1=ALARM	1	1=CNT·TA	(STROBE)	1=O.S.	1=TOGGLE	1=PB_7 ON	1=START

TB

All unused register bits are unaffected by a write and are forced to zero on a read.

APPENDIX N

6566/6567 (VIC-II) CHIP SPECIFICATIONS

The 6566/6567 are multi-purpose color video controller devices for use in both computer video terminals and video game applications. Both devices contain 47 control registers which are accessed via a standard 8-bit microprocessor bus (65XX) and will access up to 16K of memory for display information. The various operating modes and options within each mode are described.

CHARACTER DISPLAY MODE

In the character display mode, the 6566/6567 fetches CHARACTER POINTERs from the VIDEO MATRIX area of memory and translates the pointers to character dot location addresses in the 2048 byte CHARACTER BASE area of memory. The video matrix is comprised of 1000 consecutive locations in memory which each contain an eight-bit character pointer. The location of the video matrix within memory is defined by VM13–VM10 in register 24 ($18) which are used as the 4 MSB of the video matrix address. The lower order 10 bits are provided by an internal counter (VC3–VC0) which steps through the 1000 character locations. Note that the 6566/6567 provides 14 address outputs; therefore, additional system hardware may be required for complete system memory decodes.

CHARACTER POINTER ADDRESS

A13	A12	A11	A10	A09	A08	A07	A06	A05	A04	A03	A02	A01	A00
VM13	VM12	VM11	VM10	VC9	VC8	VC7	VC6	VC5	VC4	VC3	VC2	VC1	VC0

The eight-bit character pointer permits up to 256 different character definitions to be available simultaneously. Each character is an 8×8 dot matrix stored in the character base as eight consecutive bytes. The location of the character base is defined by CB13–CB11 also in register 24 ($18) which are used for the 3 most significant bits (MSB) of the character base address. The 11 lower order addresses are formed by the 8-bit character pointer from the video matrix (D7–D0) which selects a particular character, and a 3-bit raster counter (RC2–RC0) which selects one of the eight character bytes. The resulting characters are formatted as 25 rows of 40 characters each. In addition to the 8-bit character pointer, a 4-bit COLOR NYBBLE is associated with each video matrix location (the video matrix memory must be 12 bits wide) which defines one of sixteen colors for each character.

CHARACTER DATA ADDRESS

A13	A12	A11	A10	A09	A08	A07	A06	A05	A04	A03	A02	A01	A00
CB13	CB12	CB11	D7	D6	D5	D4	D3	D2	D1	D0	RC2	RC1	RC0

STANDARD CHARACTER MODE (MCM = BMM = ECM = 0)

In the standard character mode, the 8 sequential bytes from the character base are displayed directly on the 8 lines in each character region. A "0" bit causes the background #0 color (from register 33 ($21)) to be displayed while the color selected by the color nybble (foreground) is displayed for a "1" bit (see Color Code Table).

FUNCTION	CHARACTER BIT	COLOR DISPLAYED
Background	0	Background #0 color (register 33 ($21))
Foreground	1	Color selected by 4-bit color nybble

Therefore, each character has a unique color determined by the 4-bit color nybble (1 of 16) and all characters share the common background color.

MULTI-COLOR CHARACTER MODE (MCM = 1, BMM = ECM = 0

Multi-color mode provides additional color flexibility allowing up to four colors within each character but with reduced resolution. The multi-color mode is selected by setting the MCM bit in register 22 ($16) to "1," which causes the dot data stored in the character base to be interpreted in a different manner. If the MSB of the color nybble is a "0," the character will be displayed as described in standard character mode, allowing the two modes to be inter-mixed (however, only the lower order 8 colors are available). When the MSB of the color nybble is a "1" (if MCM:MSB(CM) — 1) the character bits are interpreted in the multi-color mode:

FUNCTION	CHARACTER BIT PAIR	COLOR DISPLAYED
Background	00	Background #0 Color (register 33 ($21))
Background	01	Background #1 Color (register 34 ($22))
Foreground	10	Background #2 Color (register 35 ($23))
Foreground	11	Color specified by 3 LSB of color nybble

Since two bits are required to specify one dot color, the character is now displayed as a 4 × 8 matrix with each dot twice the horizontal size as in standard mode. Note, however, that each character region can now contain 4 different colors, two as foreground and two as background (see MOB priority).

EXTENDED COLOR MODE (ECM = 1, BMM = MCM = 0)

The extended color mode allows the selection of individual background colors for each character region with the normal 8 × 8 character resolution. This mode is selected by setting the ECM bit of register 17 ($11) to "1." The character dot data is displayed as in the standard mode (foreground color determined by the color nybble is displayed for

a "1" data bit), but the 2 MSB of the character pointer are used to select the background color for each character region as follows:

CHAR. POINTER MS BIT PAIR	BACKGROUND COLOR DISPLAYED FOR 0 BIT
00	Background #0 color (register 33 ($21))
01	Background #1 color (register 34 ($22))
10	Background #2 color (register 35 ($23))
11	Background #3 color (register 36 ($24))

Since the two MSB of the character pointers are used for color information, only 64 different character definitions are available. The 6566/6567 will force CB10 and CB9 to "0" regardless of the original pointer values, so that only the first 64 character definitions will be accessed. With extended color mode each character has one of sixteen individually defined foreground colors and one of the four available background colors.

NOTE: Extended color mode and multi-color mode should not be enabled simultaneously.

BIT MAP MODE

In bit map mode, the 6566/6567 fetches data from memory in a different fashion, so that a one-to-one correspondence exists between each displayed dot and a memory bit. The bit map mode provides a screen resolution of 320H × 200V individually controlled display dots. Bit map mode is selected by setting the BMM bit in register 17 ($11) to a "1." The VIDEO MATRIX is still accessed as in character mode, but the video matrix data is no longer interpreted as character pointers, but rather as color data. The VIDEO MATRIX COUNTER is then also used as an address to fetch the dot data for display from the 8000-byte DISPLAY BASE. The display base address is formed as follows:

A13	A12	A11	A10	A09	A08	A07	A06	A05	A04	A03	A02	A01	A00
CB13	VC9	VC8	VC7	VC6	VC5	VC4	VC3	VC2	VC1	VC0	RC2	RC1	RC0

VCx denotes the video matrix counter outputs, RCx denotes the 3-bit raster line counter and CB13 is from register 24 ($18). The video matrix counter steps through the same 40 locations for eight raster lines, continuing to the next 40 locations every eighth line, while the raster counter increments once for each horizontal video line (raster line). This addressing results in each eight sequential memory locations being formatted as an 8 × 8 dot block on the video display.

STANDARD BIT MAP MODE (BMM =1, MCM = 0)

When standard bit map mode is in use, the color information is derived only from the data stored in the video matrix (the color nybble is disregarded). The 8 bits are divided into two 4-bit nybbles which allow two colors to be independently selected in each 8 × 8 dot block. When a bit in the display memory is a "0" the color of the output dot is set by the least significant (lower) nybble (LSN). Similarly, a display memory bit of "1" selects the output color determined by the MSN (upper nybble).

BIT	DISPLAY COLOR
0	Lower nybble of video matrix pointer
1	Upper nybble of video matrix pointer

MULTI-COLOR BIT MAP MODE (BMM = MCM = 1)

Multi-colored bit map mode is selected by setting the MCM bit in register 22 ($16) to a "1" in conjunction with the BMM bit. Multi-color mode uses the same memory access sequences as standard bit map mode, but interprets the dot data as follows:

BIT PAIR	DISPLAY COLOR
00	Background #0 color (register 33 ($21))
01	Upper nybble of video matrix pointer
10	Lower nybble of video matrix pointer
11	Video matrix color nybble

Note that the color nybble (DB11–DB8) IS used for the multi-color bit map mode. Again, as two bits are used to select one dot color, the

horizontal dot size is doubled, resulting in a screen resolution of 160H × 200V. Utilizing multi-color bit map mode, three independently selected colors can be displayed in each 8 × 8 block in addition to the background color.

MOVABLE OBJECT BLOCKS

The movable object block (MOB) is a special type of character which can be displayed at any one position on the screen without the block constraints inherent in character and bit map mode. Up to 8 unique MOBs can be displayed simultaneously, each defined by 63 bytes in memory which are displayed as a 24×21 dot array (shown below). A number of special features make MOBs especially suited for video graphics and game applications.

MOB DISPLAY BLOCK

BYTE	BYTE	BYTE
00	01	02
03	04	05
.	.	.
.	.	.
.	.	.
57	58	59
60	61	62

ENABLE

Each MOB can be selectively enabled for display by setting its corresponding enable bit (MnE) to "1" in register 21 ($15). If the MnE bit is "0," no MOB operations will occur involving the disabled MOB.

POSITION

Each MOB is positioned via its X and Y position register (see register map) with a resolution of 512 horizontal and 256 vertical positions. The

position of a MOB is determined by the upper-left corner of the array. X locations 23 to 347 ($17–$157) and Y locations 50 to 249 ($32–$F9) are visible. Since not all available MOB positions are entirely visible on the screen, MOBs may be moved smoothly on and off the display screen.

COLOR

Each MOB has a separate 4-bit register to determine the MOB color. The two MOB color modes are:

STANDARD MOB (MnMC = 0)

In the standard mode, a "0" bit of MOB data allows any background data to show through (transparent) and a "1" bit is displayed as the MOB color determined by the corresponding MOB Color register.

MULTI-COLOR MOB (MnMC = 1)

Each MOB can be individually selected as a multi-color MOB via MnMC bits in the MOB Multi-color register 28 ($1C). When the MnMC bit is "1," the corresponding MOB is displayed in the multi-color mode. In the multi-color mode, the MOB data is interpreted in pairs (similar to the other multi-color modes) as follows:

BIT PAIR	COLOR DISPLAYED
00	Transparent
01	MOB Multi-color #0 (register 37 ($25))
10	MOB Color (registers 39-46 ($27-$2E))
11	MOB Multi-color #1 (register 38 ($26))

Since two bits of data are required for each color, the resolution of the MOB is reduced to 12×21, with each horizontal dot expanded to twice standard size so that the overall MOB size does not change. Note that up to 3 colors can be displayed in each MOB (in addition to transparent) but that two of the colors are shared among all the MOBs in the multi-color mode.

MAGNIFICATION

Each MOB can be selectively expanded (2×) in both the horizontal and vertical directions. Two registers contain the control bits (MnXE,MnYE) for the magnification control:

REGISTER	FUNCTION
23 ($17)	Horizontal expand MnXE—"1"=expand; "0"=normal
29 ($1D)	Vertical expand MnYE—"1"=expand; "0"=normal

When MOBs are expanded, no increase in resolution is realized. The same 24×21 array (12×21 if multi-colored) is displayed, but the overall MOB dimension is doubled in the desired direction (the smallest MOB dot may be up to 4× standard dot dimension if a MOB is both multi-colored and expanded).

PRIORITY

The priority of each MOB may be individually controlled with respect to the other displayed information from character or bit map modes. The priority of each MOB is set by the corresponding bit (MnDP) of register 27 ($1B) as follows:

REG BIT	PRIORITY TO CHARACTER OR BIT MAP DATA
0	Non-transparent MOB data will be displayed (MOB in front)
1	Non-transparent MOB data will be displayed only instead of Bkgd #0 or multi-color bit pair 01 (MOB behind)

MOB—DISPLAY DATA PRIORITY

MnDP = 1	MnDP = 0
MOBn	Foreground
Foreground	MOBn
Background	Background

MOB data bits of "0" ("00" in multi-color mode) are transparent, always permitting any other information to be displayed.

The MOBs have a fixed priority with respect to each other, with MOB 0 having the highest priority and MOB 7 the lowest. When MOB data (except transparent data) of two MOBs are coincident, the data from the lower number MOB will be displayed. MOB vs. MOB data is prioritized before priority resolution with character or bit map data.

COLLISION DETECTION

Two types of MOB collision (coincidence) are detected, MOB to MOB collision and MOB to display data collision:

1) A collision between two MOBs occurs when non-transparent output data of two MOBs are coincident. Coincidence of MOB transparent areas will not generate a collision. When a collision occurs, the MOB bits (MnM) in the MOB–MOB COLLISION register 30 ($1E) will be set to "1" for both colliding MOBs. As a collision between two (or more) MOBs occurs, the MOB–MOB collision bit for each collided MOB will be set. The collision bits remain set until a read of the collision register, when all bits are automatically cleared. MOBs collisions are detected even if positioned off-screen.
2) The second type of collision is a MOB–DATA collision between a MOB and foreground display data from the character or bit map modes. The MOB-DATA COLLISION register 31 ($1F) has a bit (MnD) for each MOB which is set to "1" when both the MOB and non-background display data are coincident. Again, the coincidence of only transparent data does not generate a collision. For special applications, the display data from the 0–1 multicolor bit pair also does not cause a collision. This feature permits their use as background display data without interfering with true MOB collisions. A MOB–DATA collision can occur off-screen in the horizontal direction if actual display data has been scrolled to an off-screen position (see scrolling). The MOB–DATA COLLISION register also automatically clears when read.

The collision interrupt latches are set whenever the first bit of either register is set to "1." Once any collision bit within a register is set high, subsequent collisions will not set the interrupt latch until that collision register has been cleared to all "0s" by a read.

MOB MEMORY ACCESS

The data for each MOB is stored in 63 consecutive bytes of memory. Each block of MOB data is defined by a MOB pointer, located at the end of the VIDEO MATRIX. Only 1000 bytes of the video matrix are used in the normal display modes, allowing the video matrix locations 1016–1023 (VM base+$3F8 to VM base–$3FF) to be used for MOB pointers 0–7, respectively. The eight-bit MOB pointer from the video matrix together with the six bits from the MOB byte counter (to address 63 bytes) define the entire 14–bit address field:

A13	A12	A11	A10	A09	A08	A07	A06	A05	A04	A03	A02	A01	A00
MP7	MP6	MP5	MP4	MP3	MP2	MP1	MP0	MC5	MC4	MC3	MC2	MC1	MC0

Where MPx are the MOB pointer bits from the video matrix and MCx are the internally generated MOB counter bits. The MOB pointers are read from the video matrix at the end of every raster line. When the Y position register of a MOB matches the current raster line count, the actual fetches of MOB data begin. Internal counters automatically step through the 63 bytes of MOB data, displaying three bytes on each raster line.

OTHER FEATURES

SCREEN BLANKING

The display screen may be blanked by setting the DEN bit in register 17 ($11) to a "0." When the screen is blanked, the entire screen will be filled with the exterior color as set in register 32 ($20). When blanking is active, only transparent (Phase 1) memory accesses are required, permitting full processor utilization of the system bus. MOB data, however, will be accessed if the MOBs are not also disabled. The DEN bit must be set to "1" for normal video display.

ROW/COLUMN SELECT

The normal display consists of 25 rows of 40 characters (or character regions) per row. For special display purposes, the display window may be reduced to 24 rows and 38 characters. There is no change in the format of the displayed information, except that characters (bits) adjacent to the exterior border area will now be covered by the border. The select bits operate as follows:

RSEL	NUMBER OF ROWS	CSEL	NUMBER OF COLUMNS
0	24 rows	0	38 columns
1	25 rows	1	40 columns

The RSEL bit is in register 17 ($11) and the CSEL bit is in register 22 ($16). For standard display the larger display window is normally used, while the smaller display window is normally used in conjunction with scrolling.

SCROLLING

The display data may be scrolled up to one entire character space in both the horizontal and vertical direction. When used in conjunction with the smaller display window (above), scrolling can be used to create a smooth panning motion of display data while updating the system memory only when a new character row (or column) is required. Scrolling is also used to center a fixed display within the display window.

BITS	REGISTER	FUNCTION
X2,X1,X0	22 ($16)	Horizontal Position
Y2,Y1,Y0	17 ($11)	Vertical Position

LIGHT PEN

The light pen input latches the current screen position into a pair of registers (LPX,LPY) on a low-going edge. The X position register 19 ($13) will contain the 8 MSB of the X position at the time of transition. Since the X position is defined by a 512-state counter (9 bits) resolution to 2 horizontal dots is provided. Similarly, the Y position is latched to its reg-

ister 20 ($14) but here 8 bits provide single raster resolution within the visible display. The light pen latch may be triggered only once per frame, and subsequent triggers within the same frame will have no effect. Therefore, you must take several samples before turning the light pen to the screen (3 or more samples, average), depending upon the characteristics of your light pen.

RASTER REGISTER

The raster register is a dual-function register. A read of the raster register 18 ($12) returns the lower 8 bits of the current raster position (the MSB—RC8 is located in register 17 ($11)). The raster register can be interrogated to implement display changes outside the visible area to prevent display flicker. The visible display window is from raster 51 through raster 251 ($033—$0FB). A write to the raster bits (including RC8) is latched for use in an internal raster compare. When the current raster matches the written value, the raster interrupt latch is set.

INTERRUPT REGISTER

The interrupt register shows the status of the four sources of interrupt. An interrupt latch in register 25 ($19) is set to "1" when an interrupt source has generated an interrupt request. The four sources of interrupt are:

LATCH BIT	ENABLE BIT	WHEN SET
IRST	ERST	Set when (raster count) = (stored raster count)
IMDC	EMDC	Set by MOB—DATA collision register (first collision only)
IMMC	EMMC	Set by MOB—MOB collision register (first collision only)
ILP	ELP	Set by negative transition of LP input (once per frame)
IRQ		Set high by latch set and enabled (invert of IRQ/ output)

To enable an interrupt request to set the IRQ/ output to "0," the corresponding interrupt enable bit in register 26 ($1A) must be set to "1." Once an interrupt latch has been set, the latch may be cleared only by writing a "1" to the desired latch in the interrupt register. This feature allows selective handling of video interrupts without software required to "remember" active interrupts.

DYNAMIC RAM REFRESH

A dynamic ram refresh controller is built in to the 6566/6567 devices. Five 8-bit row addresses are refreshed every raster line. This rate guarantees a maximum delay of 2.02 ms between the refresh of any single row address in a 128 refresh scheme. (The maximum delay is 3.66 ms in a 256 address refresh scheme.) This refresh is totally transparent to the system, since the refresh occurs during Phase 1 of the system clock. The 6567 generates both RAS/ and CAS/ which are normally connected directly to the dynamic rams. RAS/ and CAS/ are generated for every Phase 2 and every video data access (including refresh) so that external clock generation is not required.

THEORY OF OPERATION

SYSTEM INTERFACE

The 6566/6567 video controller devices interact with the system data bus in a special way. A 65XX system requires the system buses only during the Phase 2 (clock high) portion of the cycle. The 6566/6567 devices take advantage of this feature by normally accessing system memory during the Phase 1 (clock low) portion of the clock cycle. Therefore, operations such as character data fetches and memory refresh are totally transparent to the processor and do not reduce the processor throughput. The video chips provide the interface control signals required to maintain this bus sharing.

The video devices provide the signal AEC (address enable control) which is used to disable the processor address bus drivers allowing the video device to access the address bus. AEC is active low which permits direct connection to the AEC input of the 65XX family. The AEC signal is

normally activated during Phase 1 so that processor operation is not affected. Because of this bus "sharing," all memory accesses must be completed in 1/2 cycle. Since the video chips provide a 1-MHz clock (which must be used as system Phase 2), a memory cycle is 500 ns including address setup, data access and, data setup to the reading device.

Certain operations of the 6566/6567 require data at a faster rate than available by reading only during the Phase 1 time; specifically, the access of character pointers from the video matrix and the fetch of MOB data. Therefore, the processor must be disabled and the data accessed during the Phase 2 clock. This is accomplished via the BA (bus available) signal. The BA line is normally high but is brought low during Phase 1 to indicate that the video chip will require a Phase 2 data access. Three Phase-2 times are allowed after BA low for the processor to complete any current memory accesses. On the fourth Phase 2 after BA low, the AEC signal will remain low during Phase 2 as the video chip fetches data. The BA line is normally connected to the RDY input of a 65XX processor. The character pointer fetches occur every eighth raster line during the display window and require 40 consecutive Phase 2 accesses to fetch the video matrix pointers. The MOB data fetches require 4 memory accesses as follows:

PHASE	DATA	CONDITION
1	MOB Pointer	Every raster
2	MOB Byte 1	Each raster while MOB is displayed
1	MOB Byte 2	Each raster while MOB is displayed
2	MOB Byte 3	Each raster while MOB is displayed

The MOB pointers are fetched every other Phase 1 at the end of each raster line. As required, the additional cycles are used for MOB data fetches. Again, all necessary bus control is provided by the 6566/6567 devices.

MEMORY INTERFACE

The two versions of the video interface chip, 6566 and 6567, differ in address output configurations. The 6566 has thirteen fully decoded ad-

dresses for direct connection to the system address bus. The 6567 has multiplexed addresses for direct connection to 64K dynamic RAMs. The least significant address bits, A06–A00, are present on A06–A00 while RAS/ is brought low, while the most significant bits, A13–A08, are present on A05–A00 while CAS/ is brought low. The pins A11–A07 on the 6567 are static address outputs to allow direct connection of these bits to a conventional 16K (2K×8) ROM. (The lower order addresses require external latching.)

PROCESSOR INTERFACE

Aside from the special memory accesses described above, the 6566/6567 registers can be accessed similar to any other peripheral device. The following processor interface signals are provided:

DATA BUS (DB7–DB0)

The eight data bus pins are the bi-directional data port, controlled by CS/, RW, and Phase 0. The data bus can only be accessed while AEC and Phase 0 are high and CS/ is low.

CHIP SELECT (CS/)

The chip select pin, CS/, is brought low to enable access to the device registers in conjunction with the address and RW pins. CS/ low is recognized only while AEC and Phase 0 are high.

READ/WRITE (R/W)

The read/write input, R/W, is used to determine the direction of data transfer on the data bus, in conjunction with CS/. When R/W is high ("1") data is transferred from the selected register to the data bus output. When R/W is low ("0") data presented on the data bus pins is loaded into the selected register.

ADDRESS BUS (A05–A00)

The lower six address pins, A5–A0, are bi-directional. During a processor read or write of the video device, these address pins are inputs. The data on the address inputs selects the register for read or write as defined in the register map.

CLOCK OUT (PH0)

The clock output, Phase 0, is the 1-MHz clock used as the 65XX processor Phase 0 in. All system bus activity is referenced to this clock. The clock frequency is generated by dividing the 8-MHz video input clock by eight.

INTERRUPTS (IRQ/)

The interrupt output, IRQ/, is brought low when an enabled source of interrupt occurs within the device. The IRQ/ output is open drain, requiring an external pull-up resistor.

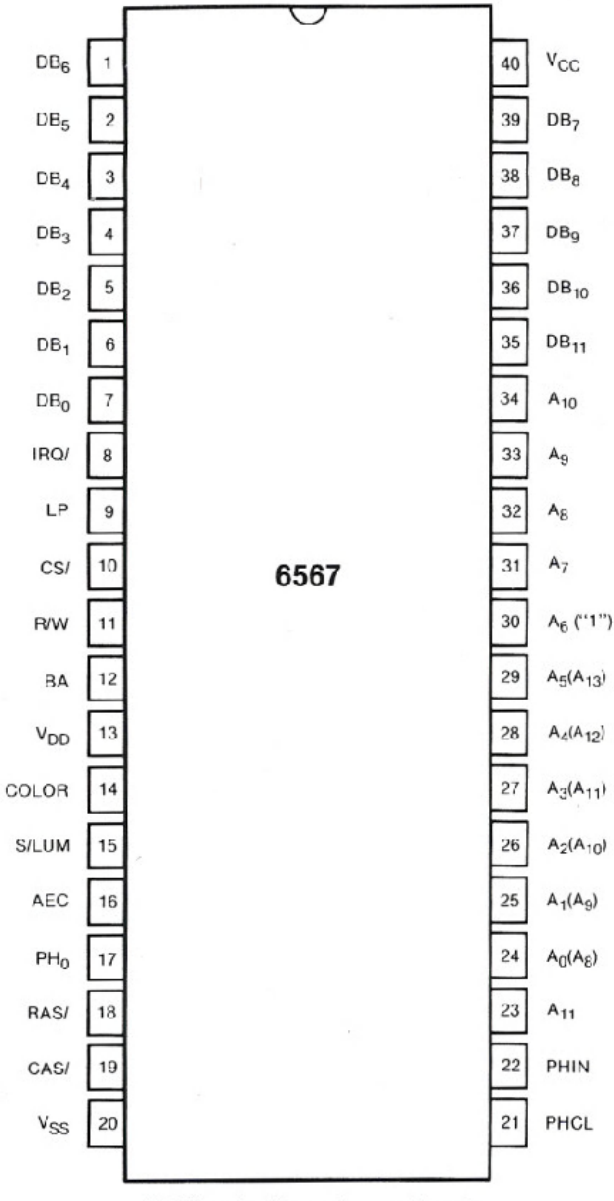
VIDEO INTERFACE

The video output signal from the 6566/6567 consists of two signals which must be externally mixed together. SYNC/LUM output contains all the video data, including horizontal and vertical syncs, as well as the luminance information of the video display. SYNC/LUM is open drain, requiring an external pull-up of 500 ohms. The COLOR output contains all the chrominance information, including the color reference burst and the color of all display data. The COLOR output is open source and should be terminated with 1000 ohms to ground. After appropriate mixing of these two signals, the resulting signal can directly drive a video monitor or be fed to a modulator for use with a standard television.

SUMMARY OF 6566/6567 BUS ACTIVITY

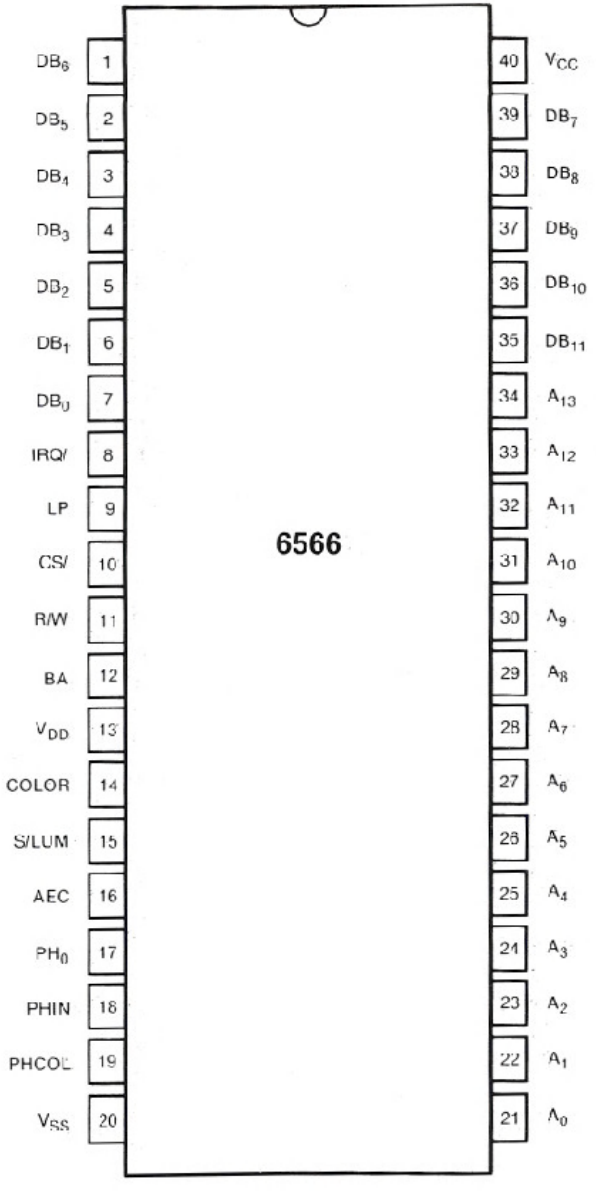
AEC	PH0	CS/	R/W	ACTION
0	0	X	X	PHASE 1 FETCH, REFRESH
0	1	X	X	PHASE 2 FETCH (PROCESSOR OFF)
1	0	X	X	NO ACTION
1	1	0	0	WRITE TO SELECTED REGISTER
1	1	0	1	READ FROM SELECTED REGISTER
1	1	1	X	NO ACTION

PIN CONFIGURATION



(Multiplexed addresses in parentheses)

PIN CONFIGURATION



REGISTER MAP

ADDRESS		DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	DESCRIPTION
00	($00)	M0X7	M0X6	M0X5	M0X4	M0X3	M0X2	M0X1	M0X0	MOB 0 X-position
01	($01)	M0Y7	M0Y6	M0Y5	M0Y4	M0Y3	M0Y2	M0Y1	M0Y0	MOB 0 Y-position
02	($02)	M1X7	M1X6	M1X5	M1X4	M1X3	M1X2	M1X1	M1X0	MOB 1 X-position
03	($03)	M1Y7	M1Y6	M1Y5	M1Y4	M1Y3	M1Y2	M1Y1	M1Y0	MOB 1 Y-position
04	($04)	M2X7	M2X6	M2X5	M2X4	M2X3	M2X2	M2X1	M2X0	MOB 2 X-position
05	($05)	M2Y7	M2Y6	M2Y5	M2Y4	M2Y3	M2Y2	M2Y1	M2Y0	MOB 2 Y-position
06	($06)	M3X7	M3X6	M3X5	M3X4	M3X3	M3X2	M3X1	M3X0	MOB 3 X-position
07	($07)	M3Y7	M3Y6	M3Y5	M3Y4	M3Y3	M3Y2	M3Y1	M3Y0	MOB 3 Y-position
08	($08)	M4X7	M4X6	M4X5	M4X4	M4X3	M4X2	M4X1	M4X0	MOB 4 X-position
09	($09)	M4Y7	M4Y6	M4Y5	M4Y4	M4Y3	M4Y2	M4Y1	M4Y0	MOB 4 Y-position
10	($0A)	M5X7	M5X6	M5X5	M5X4	M5X3	M5X2	M5X1	M5X0	MOB 5 X-position
11	($0B)	M5Y7	M5Y6	M5Y5	M5Y4	M5Y3	M5Y2	M5Y1	M5Y0	MOB 5 Y-position
12	($0C)	M6X7	M6X6	M6X5	M6X4	M6X3	M6X2	M6X1	M6X0	MOB 6 X-position
13	($0D)	M6Y7	M6Y6	M6Y5	M6Y4	M6Y3	M6Y2	M6Y1	M6Y0	MOB 6 Y-position
14	($0E)	M7X7	M7X6	M7X5	M7X4	M7X3	M7X2	M7X1	M7X0	MOB 7 X-position
15	($0F)	M7Y7	M7Y6	M7Y5	M7Y4	M7Y3	M7Y2	M7Y1	M6Y0	MOB 7 Y-position
16	($10)	M7X8	M6X8	M5X8	M4X8	M3X8	M2X8	M1X8	M0X8	MSB of X-position
17	($11)	RC8	ECM	BMM	DEN	RSEL	Y2	Y1	Y0	See text
18	($12)	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	Raster register
19	($13)	LPX8	LPX7	LPX6	LPX5	LPX4	LPX3	LPX2	LPX1	Light Pen X
20	($14)	LPY7	LPY6	LPY5	LPY4	LPY3	LPY2	LPY1	LPY0	Light Pen Y
21	($15)	M7E	M6E	M5E	M4E	M3E	M2E	M1E	M0E	MOB Enable
22	($16)	—	—	RES	MCM	CSEL	X2	X1	X0	See text
23	($17)	M7YE	M6YE	M5YE	M4YE	M3YE	M2YE	M1YE	M0YE	MOB Y-expand

24	($18)	VM13	VM12	VM11	VM10	CB13	CB12	CB11	—	Memory Pointers
25	($19)	IRQ	—	—	—	ILP	IMMC	IMBC	IRST	Interrupt Register
26	($1A)	—	—	—	—	ELP	EMMC	EMBC	ERST	Enable Interrupt
27	($1B)	M7DP	M6DP	M5DP	M4DP	M3DP	M2DP	M1DP	M0DP	MOB-DATA Priority
28	($1C)	M7MC	M6MC	M5MC	M4MC	M3MC	M2MC	M1MC	M0MC	MOB Multicolor Sel
29	($1D)	M7XE	M6XE	M5XE	M4XE	M3XE	M2XE	M1XE	M0XE	MOB X-expand
30	($1E)	M7M	M6M	M5M	M4M	M3M	M2M	M1M	M0M	MOB-MOB Collision
31	($1F)	M7D	M6D	M5D	M4D	M3D	M2D	M1D	M0D	MOB-DATA Collision
32	($20)	—	—	—	—	EC3	EC2	EC1	EC0	Exterior Color
33	($21)	—	—	—	—	B0C3	B0C2	B0C1	B0C0	Bkgd #0 Color
34	($22)	—	—	—	—	B1C3	B1C2	B1C1	B1C0	Bkgd #1 Color
35	($23)	—	—	—	—	B2C3	B2C2	B2C1	B2C0	Bkgd #2 Color
36	($24)	—	—	—	—	B3C3	B3C2	B3C1	B3C0	Bkgd #3 Color
37	($25)	—	—	—	—	MM03	MM02	MM01	MM00	MOB Multicolor #0
38	($26)	—	—	—	—	MM13	MM12	MM11	MM10	MOB Multicolor #1
39	($27)	—	—	—	—	M0C3	M0C2	M0C1	M0C0	MOB 0 Color
40	($28)	—	—	—	—	M1C3	M1C2	M1C1	M1C0	MOB 1 Color
41	($29)	—	—	—	—	M2C3	M2C2	M2C1	M2C0	MOB 2 Color
42	($2A)	—	—	—	—	M3C3	M3C2	M3C1	M3C0	MOB 3 Color
43	($2B)	—	—	—	—	M4C3	M4C2	M4C1	M4C0	MOB 4 Color
44	($2C)	—	—	—	—	M5C3	M5C2	M5C1	M5C0	MOB 5 Color
45	($2D)	—	—	—	—	M6C3	M6C2	M6C1	M6C0	MOB 6 Color
46	($2E)	—	—	—	—	M7C3	M7C2	M7C1	M7C0	MOB 7 Color

NOTE: A dash indicates a no connect. All no connects are read as a "1."

COLOR CODES

D4	D3	D1	D0	HEX	DEC	COLOR
0	0	0	0	0	0	BLACK
0	0	0	1	1	1	WHITE
0	0	1	0	2	2	RED
0	0	1	1	3	3	CYAN
0	1	0	0	4	4	PURPLE
0	1	0	1	5	5	GREEN
0	1	1	0	6	6	BLUE
0	1	1	1	7	7	YELLOW
1	0	0	0	8	8	ORANGE
1	0	0	1	9	9	BROWN
1	0	1	0	A	10	LT RED
1	0	1	1	B	11	DARK GREY
1	1	0	0	C	12	MED GREY
1	1	0	1	D	13	LT GREEN
1	1	1	0	E	14	LT BLUE
1	1	1	1	F	15	LT GREY

APPENDIX O

6581 SOUND INTERFACE DEVICE (SID) CHIP SPECIFICATIONS

CONCEPT

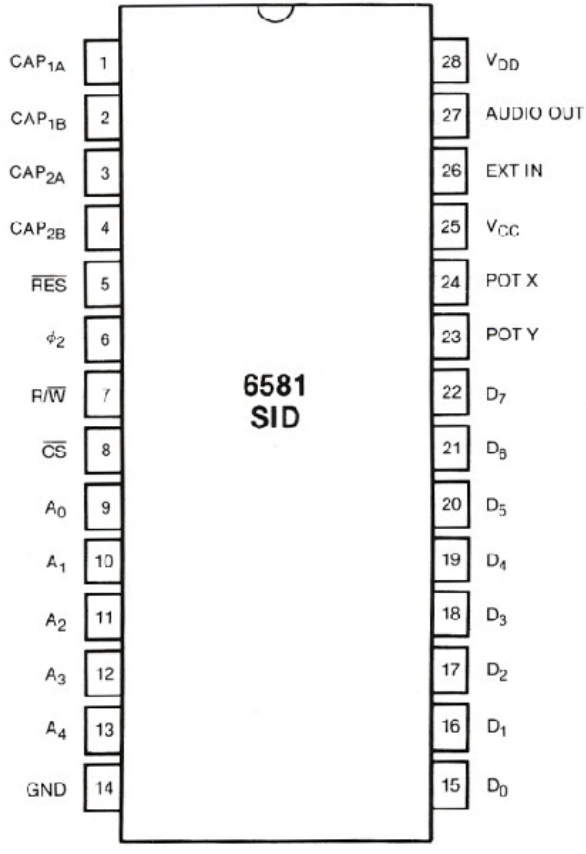
The 6581 Sound Interface Device (SID) is a single-chip, 3-voice electronic music synthesizer/sound effects generator compatible with the 65XX and similar microprocessor families. SID provides wide-range, high-resolution control of pitch (frequency), tone color (harmonic content), and dynamics (volume). Specialized control circuitry minimizes software overhead, facilitating use in arcade/home video games and low-cost musical instruments.

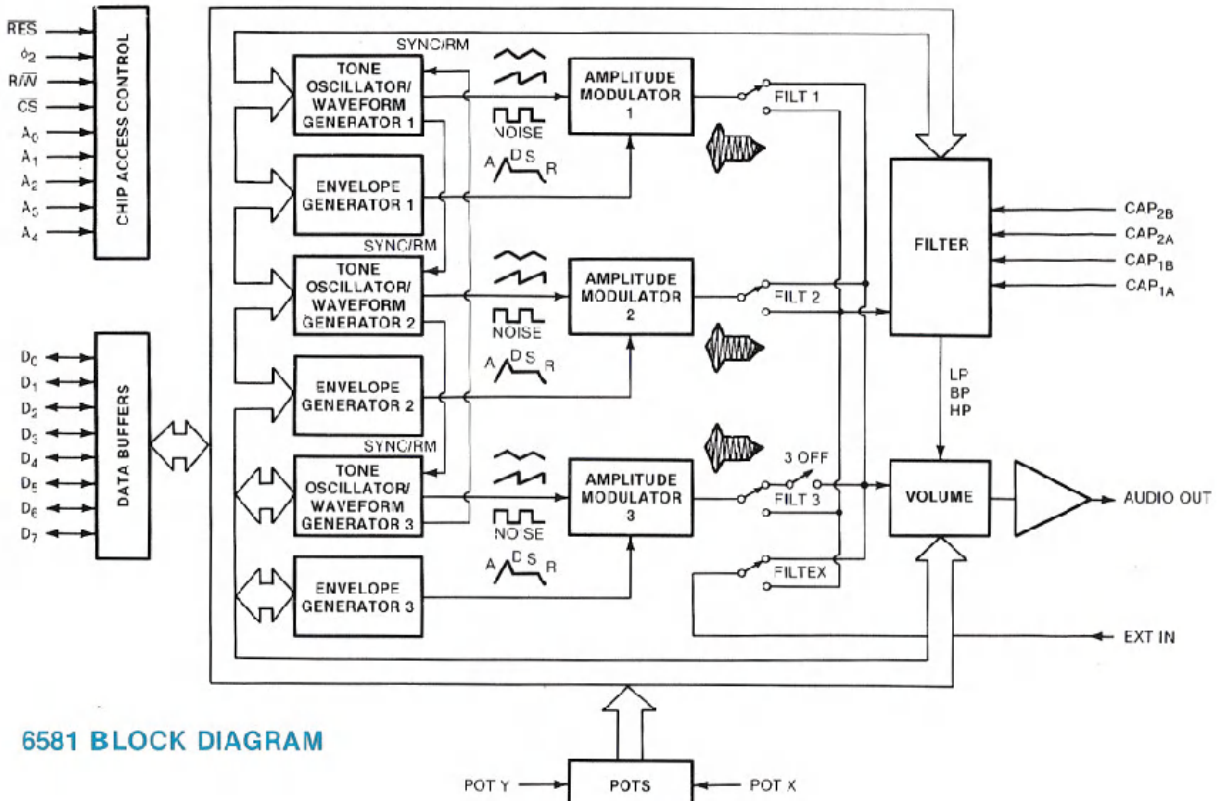
FEATURES

- 3 TONE OSCILLATORS
  - Range: 0–4 kHz
- 4 WAVEFORMS PER OSCILLATOR
  - Triangle, Sawtooth,
  - Variable Pulse, Noise
- 3 AMPLITUDE MODULATORS
  - Range: 48 dB
- 3 ENVELOPE GENERATORS
  - Exponential response
  - Attack Rate: 2 ms–8 s
  - Decay Rate: 6 ms–24 s
  - Sustain Level: 0–peak volume
  - Release Rate: 6 ms–24 s
- OSCILLATOR SYNCHRONIZATION
- RING MODULATION

- PROGRAMMABLE FILTER
  - Cutoff range: 30 Hz–12 kHz
  - 12 dB/octave Rolloff
  - Low pass, Bandpass,
  - High pass, Notch outputs
  - Variable Resonance
- MASTER VOLUME CONTROL
- 2 A/D POT INTERFACES
- RANDOM NUMBER/MODULATION GENERATOR
- EXTERNAL AUDIO INPUT

PIN CONFIGURATION





6581 BLOCK DIAGRAM

DESCRIPTION

The 6581 consists of three synthesizer "voices" which can be used independently or in conjunction with each other (or external audio sources) to create complex sounds. Each voice consists of a Tone Oscillator/Waveform Generator, an Envelope Generator and an Amplitude Modulator. The Tone Oscillator controls the pitch of the voice over a wide range. The Oscillator produces four waveforms at the selected frequency, with the unique harmonic content of each waveform providing simple control of tone color. The volume dynamics of the oscillator are controlled by the Amplitude Modulator under the direction of the Envelope Generator. When triggered, the Envelope Generator creates an amplitude envelope with programmable rates of increasing and decreasing volume. In addition to the three voices, a programmable Filter is provided for generating complex, dynamic tone colors via subtractive synthesis.

SID allows the microprocessor to read the changing output of the third Oscillator and third Envelope Generator. These outputs can be used as a source of modulation information for creating vibrato, frequency/filter sweeps and similar effects. The third oscillator can also act as a random number generator for games. Two A/D converters are provided for interfacing SID with potentiometers. These can be used for "paddles" in a game environment or as front panel controls in a music synthesizer. SID can process external audio signals, allowing multiple SID chips to be daisy-chained or mixed in complex polyphonic systems.

SID CONTROL REGISTERS

There are 29 eight-bit registers in SID which control the generation of sound. These registers are either WRITE-only or READ-only and are listed below in Table 1.

Table 1. SID Register Map

	ADDRESS					REG #	DATA								REG NAME	REG TYPE
	A_4	A_3	A_2	A_1	A_0	(HEX)	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0		
															Voice 1	
0	0	0	0	0	0	00	F_7	F_6	F_5	F_4	F_3	F_2	F_1	F_0	FREQ LO	WRITE-ONLY
1	0	0	0	0	1	01	F_{15}	F_{14}	F_{13}	F_{12}	F_{11}	F_{10}	F_9	F_8	FREQ HI	WRITE-ONLY
2	0	0	0	1	0	02	PW_7	PW_6	PW_5	PW_4	PW_3	PW_2	PW_1	PW_0	PW LO	WRITE-ONLY
3	0	0	0	1	1	03					PW_{11}	PW_{10}	PW_9	PW_8	PW HI	WRITE-ONLY
4	0	0	1	0	0	04	NOISE	PULSE	SAW	TRIANGLE	TEST	RING MOD	SYNC	GATE	CONTROL REG	WRITE-ONLY
5	0	0	1	0	1	05	ATK_3	ATK_2	ATK_1	ATK_0	DCY_3	DCY_2	DCY_1	DCY_0	ATTACK/DECAY	WRITE-ONLY
6	0	0	1	1	0	06	STN_3	STN_2	STN_1	STN_0	RLS_3	RLS_2	RLS_1	RLS_0	SUSTAIN/RELEASE	WRITE-ONLY
															Voice 2	
7	0	0	1	1	1	07	F_7	F_6	F_5	F_4	F_3	F_2	F_1	F_0	FREQ LO	WRITE-ONLY
8	0	1	0	0	0	08	F_{15}	F_{14}	F_{13}	F_{12}	F_{11}	F_{10}	F_9	F_8	FREQ HI	WRITE-ONLY
9	0	1	0	0	1	09	PW_7	PW_6	PW_5	PW_4	PW_3	PW_2	PW_1	PW_0	PW LO	WRITE-ONLY
10	0	1	0	1	0	0A	—	—	—	—	PW_{11}	PW_{10}	PW_9	PW_8	PW HI	WRITE-ONLY
11	0	1	0	1	1	0B	NOISE	PULSE	SAW	TRIANGLE	TEST	RING MOD	SYNC	GATE	CONTROL REG	WRITE-ONLY
12	0	1	1	0	0	0C	ATK_3	ATK_2	ATK_1	ATK_0	DCY_3	DCY_2	DCY_1	DCY_0	ATTACK/DECAY	WRITE-ONLY
13	0	1	1	0	1	0D	STN_3	STN_2	STN_1	STN_0	RLS_3	RLS_2	RLS_1	RLS_0	SUSTAIN/RELEASE	WRITE-ONLY
															Voice 3	
14	0	1	1	1	0	0E	F_7	F_6	F_5	F_4	F_3	F_2	F_1	F_0	FREQ LO	WRITE-ONLY
15	0	1	1	1	1	0F	F_{15}	F_{14}	F_{13}	F_{12}	F_{11}	F_{10}	F_9	F_8	FREQ HI	WRITE-ONLY
16	1	0	0	0	0	10	PW_7	PW_6	PW_5	PW_4	PW_3	PW_2	PW_1	PW_0	PW LO	WRITE-ONLY
17	1	0	0	0	1	11	—	—	—	—	PW_{11}	PW_{10}	PW_9	PW_8	PW HI	WRITE-ONLY
18	1	0	0	1	0	12	NOISE	PULSE	SAW	TRIANGLE	TEST	RING MOD	SYNC	GATE	CONTROL REG	WRITE-ONLY
19	1	0	0	1	1	13	ATK_3	ATK_2	ATK_1	ATK_0	DCY_3	DCY_2	DCY_1	DCY_0	ATTACK/DECAY	WRITE-ONLY
20	1	0	1	0	0	14	STN_3	STN_2	STN_1	STN_0	RLS_3	RLS_2	RLS_1	RLS_0	SUSTAIN/RELEASE	WRITE-ONLY
															Filter	
21	1	0	1	0	1	15	—	—	—	—	—	FC_2	FC_1	FC_0	FC LO	WRITE-ONLY
22	1	0	1	1	0	16	FC_{10}	FC_9	FC_8	FC_7	FC_6	FC_5	FC_4	FC_3	FC HI	WRITE-ONLY
23	1	0	1	1	1	17	RES_3	RES_2	RES_1	RES_0	FILTEX	FILT 3	FILT 2	FILT 1	RES/FILT	WRITE-ONLY
24	1	1	0	0	0	18	3 OFF	HP	BP	LP	VOL_3	VOL_2	VOL_1	VOL_0	MODE/VOL	WRITE ONLY
															Misc.	
25	1	1	0	0	1	19	PX_7	PX_6	PX_5	PX_4	PX_3	PX_2	PX_1	PX_0	POT X	READ-ONLY
26	1	1	0	1	0	1A	PY_7	PY_6	PY_5	PY_4	PY_3	PY_2	PY_1	PY_0	POT Y	READ-ONLY
27	1	1	0	1	1	1B	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0	OSC_3/RANDOM	READ-ONLY
28	1	1	1	0	0	1C	E_7	E_6	E_5	E_4	E_3	E_2	E_1	E_0	ENV_3	READ-ONLY

SID REGISTER DESCRIPTION

VOICE 1

FREQ LO/FREQ HI (Registers 00,01)

Together these registers form a 16-bit number which linearly controls the frequency of Oscillator 1. The frequency is determined by the following equation:

$$F_{out} = (F_n \times F_{clk}/16777216) \text{ Hz}$$

Where F_n is the 16-bit number in the Frequency registers and F_{clk} is the system clock applied to the $\phi 2$ input (pin 6). For a standard 1.0-MHz clock, the frequency is given by:

$$F_{out} = (F_n \times 0.059604645) \text{ Hz}$$

A complete table of values for generating 8 octaves of the equally tempered musical scale with concert A (440 Hz) tuning is provided in Appendix E. It should be noted that the frequency resolution of SID is sufficient for any tuning scale and allows sweeping from note to note (portamento) with no discernable frequency steps.

PW LO/PW HI (Registers 02,03)

Together these registers form a 12-bit number (bits 4–7 of PW HI are not used) which linearly controls the Pulse Width (duty cycle) of the Pulse waveform on Oscillator 1. The pulse width is determined by the following equation:

$$PW_{out} = (PW_n/40.95) \%$$

Where PWn is the 12-bit number in the Pulse Width registers.

The pulse width resolution allows the width to be smoothly swept with no discernable stepping. Note that the Pulse waveform on Oscillator 1 must be selected in order for the Pulse Width registers to have any audible effect. A value of 0 or 4095 ($FF) in the Pulse Width registers will produce a constant DC output, while a value of 2048 ($800) will produce a square wave.

CONTROL REGISTER (Register 04)

This register contains eight control bits which select various options on Oscillator 1.

GATE (Bit 0): The GATE bit controls the Envelope Generator for Voice 1. When this bit is set to a one, the Envelope Generator is Gated (triggered) and the ATTACK/DECAY/SUSTAIN cycle is initiated. When the bit is reset to a zero, the RELEASE cycle begins. The Envelope Generator controls the amplitude of Oscillator 1 appearing at the audio output, therefore, the GATE bit must be set (along with suitable envelope parameters) for the selected output of Oscillator 1 to be audible. A detailed discussion of the Envelope Generator can be found at the end of this Appendix.

SYNC (Bit 1): The SYNC bit, when set to a one, synchronizes the fundamental frequency of Oscillator 1 with the fundamental frequency of Oscillator 3, producing "Hard Sync" effects.

Varying the frequency of Oscillator 1 with respect to Oscillator 3 produces a wide range of complex harmonic structures from Voice 1 at the frequency of Oscillator 3. In order for sync to occur, Oscillator 3 must be set to some frequency other than zero but preferably lower than the frequency of Oscillator 1. No other parameters of Voice 3 have any effect on sync.

RING MOD (Bit 2): The RING MOD bit, when set to a one, replaces the Triangle waveform output of Oscillator 1 with a "Ring Modulated" combination of Oscillators 1 and 3. Varying the frequency of Oscillator 1 with respect to Oscillator 3 produces a wide range of non-harmonic overtone structures for creating bell or gong sounds and for special effects. In order for ring modulation to be audible, the Triangle waveform of Oscillator 1 must be selected and Oscillator 3 must be set to some frequency other than zero. No other parameters of Voice 3 have any effect on ring modulation.

TEST (Bit 3): The TEST bit, when set to a one, resets and locks Oscillator 1 at zero until the TEST bit is cleared. The Noise waveform output of Oscillator 1 is also reset and the Pulse waveform output is held at a DC level. Normally this bit is used for testing purposes, however, it can be used to synchronize Oscillator 1 to external events, allowing the generation of highly complex waveforms under real-time software control.

(Bit 4): When set to a one, the Triangle waveform output of Oscillator 1 is selected. The Triangle waveform is low in harmonics and has a mellow, flute-like quality.

(Bit 5): When set to a one, the Sawtooth waveform output of Oscillator 1 is selected. The Sawtooth waveform is rich in even and odd harmonics and has a bright, brassy quality.

(Bit 6): When set to a one, the Pulse waveform output of Oscillator 1 is selected. The harmonic content of this waveform can be adjusted by the Pulse Width registers, producing tone qualities ranging from a bright, hollow square wave to a nasal, reedy pulse. Sweeping the pulse width in real-time produces a dynamic "phasing" effect which adds a sense of motion to the sound. Rapidly jumping between different pulse widths can produce interesting harmonic sequences.

NOISE (Bit 7): When set to a one, the Noise output waveform of Oscillator 1 is selected. This output is a random signal which changes at the frequency of Oscillator 1. The sound quality can be varied from a low rumbling to hissing white noise via the Oscillator 1 Frequency registers. Noise is useful in creating explosions, gunshots, jet engines, wind, surf and other unpitched sounds, as well as snare drums and cymbals. Sweeping the oscillator frequency with Noise selected produces a dramatic rushing effect.

One of the output waveforms must be selected for Oscillator 1 to be audible, however, it is NOT necessary to de-select waveforms to silence the output of Voice 1. The amplitude of Voice 1 at the final output is a function of the Envelope Generator only.

NOTE: The oscillator output waveforms are NOT additive. If more than one output waveform is selected simultaneously, the result will be a logical ANDing of the waveforms. Although this technique can be used to generate additional waveforms beyond the four listed above, it must be used with care. If any other waveform is selected while Noise is on, the Noise output can "lock up." If this occurs, the Noise output will remain silent until reset by the TEST bit or by bringing RES (pin 5) low.

ATTACK/DECAY (Register 05)

Bits 4–7 of this register (ATK0–ATK3) select 1 of 16 ATTACK rates for the Voice 1 Envelope Generator. The ATTACK rate determines how rapidly the output of Voice 1 rises from zero to peak amplitude when the Envelope Generator is Gated. The 16 ATTACK rates are listed in Table 2.

Bits 0–3 (DCY0–DCY3) select 1 of 16 DECAY rates for the Envelope Generator. The DECAY cycle follows the ATTACK cycle and the DECAY rate determines how rapidly the output falls from the peak amplitude to the selected SUSTAIN level. The 16 DECAY rates are listed in Table 2.

SUSTAIN/RELEASE (Register 06)

Bits 4–7 of this register (STN0–STN3) select 1 of 16 SUSTAIN levels for the Envelope Generator. The SUSTAIN cycle follows the DECAY cycle and the output of Voice 1 will remain at the selected SUSTAIN amplitude as long as the Gate bit remains set. The SUSTAIN levels range from zero to peak amplitude in 16 linear steps, with a SUSTAIN value of 0 selecting zero amplitude and a SUSTAIN value of 15 ($F) selecting the peak amplitude. A SUSTAIN value of 8 would cause Voice 1 to SUSTAIN at an amplitude one-half the peak amplitude reached by the ATTACK cycle.

Bits 0–3 (RLS0–RLS3) select 1 of 16 RELEASE rates for the Envelope Generator. The RELEASE cycle follows the SUSTAIN cycle when the Gate bit is reset to zero. At this time, the output of Voice 1 will fall from the SUSTAIN amplitude to zero amplitude at the selected RELEASE rate. The 16 RELEASE rates are identical to the DECAY rates.

NOTE: The cycling of the Envelope Generator can be altered at any point via the Gate bit. The Envelope Generator can be Gated and Released without restriction. For example, if the Gate bit is reset before the envelope has finished the ATTACK cycle, the RELEASE cycle will immediately begin, starting from whatever amplitude had been reached. If the envelope is then Gated again (before the RELEASE cycle has reached zero amplitude), another ATTACK cycle will begin, starting from whatever amplitude had been reached. This technique can be used to generate complex amplitude envelopes via real-time software control.

Table 2. Envelope Rates

VALUE		ATTACK RATE	DECAY/RELEASE RATE
DEC	(HEX)	(Time/Cycle)	(Time/Cycle)
0	(0)	2 ms	6 ms
1	(1)	8 ms	24 ms
2	(2)	16 ms	48 ms
3	(3)	24 ms	72 ms
4	(4)	38 ms	114 ms
5	(5)	56 ms	168 ms
6	(6)	68 ms	204 ms
7	(7)	80 ms	240 ms
8	(8)	100 ms	300 ms
9	(9)	250 ms	750 ms
10	(A)	500 ms	1.5 s
11	(B)	800 ms	2.4 s
12	(C)	1 s	3 s
13	(D)	3 s	9 s
14	(E)	5 s	15 s
15	(F)	8 s	24 s

NOTE Envelope rates are based on a 1.0-MHz $\phi 2$ clock. For other $\phi 2$ frequencies, multiply the given rate by 1 MHz/$\phi 2$. The rates refer to the amount of time per cycle. For example, given an ATTACK value of 2, the ATTACK cycle would take 16 ms to rise from zero to peak amplitude. The DECAY/RELEASE rates refer to the amount of time these cycles would take to fall from peak amplitude to zero.

VOICE 2

Registers 07–$0D control Voice 2 and are functionally identical to registers 00–06 with these exceptions:

1) When selected, SYNC synchronizes Oscillator 2 with Oscillator 1.
2) When selected, RING MOD replaces the Triangle output of Oscillator 2 with the ring modulated combination of Oscillators 2 and 1.

VOICE 3

Registers $0E–$14 control Voice 3 and are functionally identical to registers 00–06 with these exceptions:

1) When selected, SYNC synchronizes Oscillator 3 with Oscillator 2.
2) When selected, RING MOD replaces the Triangle output of Oscillator 3 with the ring modulated combination of Oscillators 3 and 2.

Typical operation of a voice consists of selecting the desired parameters: frequency, waveform, effects (SYNC, RING MOD) and envelope rates, then gating the voice whenever the sound is desired. The sound can be sustained for any length of time and terminated by clearing the Gate bit. Each voice can be used separately, with independent parameters and gating, or in unison to create a single, powerful voice. When used in unison, a slight detuning of each oscillator or tuning to musical intervals creates a rich, animated sound.

FILTER

FC LO/FC HI (Registers $15,$16)

Together these registers form an 11-bit number (bits 3–7 of FC LO are not used) which linearly controls the Cutoff (or Center) Frequency of the programmable Filter. The approximate Cutoff Frequency ranges from 30 Hz to 12 KHz.

RES/FILT (Register $17)

Bits 4–7 of this register (RES0–RES3) control the resonance of the filter. Resonance is a peaking effect which emphasizes frequency components at the Cutoff Frequency of the Filter, causing a sharper sound. There are 16 resonance settings ranging linearly from no resonance (0) to maximum resonance (15 or $F). Bits 0–3 determine which signals will be routed through the Filter:

FILT 1 (Bit 0): When set to a zero, Voice 1 appears directly at the audio output and the Filter has no effect on it. When set to a one, Voice 1 will be processed through the Filter and the harmonic content of Voice 1 will be altered according to the selected Filter parameters.

FILT 2 (Bit 1): Same as bit 0 for Voice 2.

FILT 3 (Bit 2): Same as bit 0 for Voice 3.

FILTEX (Bit 3): Same as bit 0 for External audio input (pin 26).

MODE/VOL (Register $18)

Bits 4–7 of this register select various Filter mode and output options:

LP (Bit 4): When set to a one, the Low-Pass output of the Filter is selected and sent to the audio output. For a given Filter input signal, all frequency components below the Filter Cutoff Frequency are passed unaltered, while all frequency components above the Cutoff are attenuated at a rate of 12 dB/Octave. The Low-Pass mode produces full-bodied sounds.

BP (Bit 5): Same as bit 4 for the Bandpass output. All frequency components above and below the Cutoff are attenuated at a rate of 6 dB/Octave. The Bandpass mode produces thin, open sounds.

HP (Bit 6): Same as bit 4 for the High-Pass output. All frequency components above the Cutoff are passed unaltered, while all frequency components below the Cutoff are attenuated at a rate of 12 dB/Octave. The High-Pass mode produces tinny, buzzy sounds.

3 OFF (Bit 7): When set to a one, the output of Voice 3 is disconnected from the direct audio path. Setting Voice 3 to bypass the Filter (FILT 3 = 0) and setting 3 OFF to a one prevents Voice 3 from reaching the audio output. This allows Voice 3 to be used for modulation purposes without any undesirable output.

> **NOTE:** The Filter output modes ARE additive and multiple Filter modes may be selected simultaneously. For example, both LP and HP modes can be selected to produce a Notch (or Band Reject) Filter response. In order for the Filter to have any audible effect, at least one Filter output must be selected and at least one Voice must be routed through the Filter. The Filter is, perhaps, the most important element in SID as it allows the generation of complex tone colors via subtractive synthesis (the Filter is used to eliminate specific frequency components from a harmonically rich input signal). The best results are achieved by varying the Cutoff Frequency in real-time.

Bits 0–3 (VOL0–VOL3) select 1 of 16 overall Volume levels for the final composite audio output. The output volume levels range from no output (0) to maximum volume (15 or $F) in 16 linear steps. This control can be used as a static volume control for balancing levels in multi-chip systems or for creating dynamic volume effects, such as Tremolo. Some Volume level other than zero must be selected in order for SID to produce any sound.

MISCELLANEOUS

POTX (Register $19)

This register allows the microprocessor to read the position of the potentiometer tied to POTX (pin 24), with values ranging from 0 at minimum resistance, to 255 ($FF) at maximum resistance. The value is always valid and is updated every 512 $\phi 2$ clock cycles. See the Pin Description section for information on pot and capacitor values.

POTY (Register $1A)

Same as POTX for the pot tied to POTY (pin 23).

OSC 3/RANDOM (Register $1B)

This register allows the microprocessor to read the upper 8 output bits of Oscillator 3. The character of the numbers generated is directly related to the waveform selected. If the Sawtooth waveform of Oscillator 3 is selected, this register will present a series of numbers incrementing from 0 to 255 ($FF) at a rate determined by the frequency of Oscillator 3. If the Triangle waveform is selected, the output will increment from 0 up to 255, then decrement down to 0. If the Pulse waveform is selected, the output will jump between 0 and 255. Selecting the Noise waveform will produce a series of random numbers, therefore, this register can be used as a random number generator for games. There are numerous timing and sequencing applications for the OSC 3 register, however, the chief function is probably that of a modulation generator. The numbers generated by this register can be added, via software, to the Oscillator or Filter Frequency registers or the Pulse Width registers in real-time. Many dynamic effects can be generated in this manner. Siren-like sounds can be created by adding the OSC 3 Sawtooth output to the frequency control of another oscillator. Synthesizer "Sample and Hold" effects can be produced by adding the OSC 3 Noise output to the Filter Frequency control registers. Vibrato can be produced by setting Oscillator 3 to a frequency around 7 Hz and adding the OSC 3 Triangle output (with proper scaling) to the Frequency control of another oscillator. An unlimited range of effects are available by altering the frequency of Oscillator 3 and scaling the OSC 3 output. Normally, when Oscillator 3 is used for modulation, the audio output of Voice 3 should be eliminated (3 OFF – 1).

ENV 3 (Register S1C)

Same as OSC 3, but this register allows the microprocessor to read the output of the Voice 3 Envelope Generator. This output can be added to the Filter Frequency to produce harmonic envelopes, WAH-WAH, and similar effects. "Phaser" sounds can be created by adding this output to the frequency control registers of an oscillator. The Voice 3 Envelope Generator must be Gated in order to produce any output from this register. The OSC 3 register, however, always reflects the changing output of the oscillator and is not affected in any way by the Envelope Generator.

SID PIN DESCRIPTION

CAP1A,CAP1B, (Pins 1,2)/ CAP2A,CAP2B (Pins 3,4)

These pins are used to connect the two integrating capacitors required by the programmable Filter. C1 connects between pins 1 and 2, C2 between pins 3 and 4. Both capacitors should be the same value. Normal operation of the Filter over the audio range (approximately 30 Hz–12 kHz) is accomplished with a value of 2200 pF for C1 and C2. Polystyrene capacitors are preferred and in complex polyphonic systems, where many SID chips must track each other, matched capacitors are recommended.

The frequency range of the Filter can be tailored to specific applications by the choice of capacitor values. For example, a low-cost game may not require full high-frequency response. In this case, larger values for C1 and C2 could be chosen to provide more control over the bass frequencies of the Filter. The maximum Cutoff Frequency of the Filter is given by:

$$FC_{max} = 2.6E-5/C$$

Where C is the capacitor value. The range of the Filter extends 9 octaves below the maximum Cutoff Frequency.

RES (Pin 5)

This TTL-level input is the reset control for SID. When brought low for at least ten $\phi 2$ cycles, all internal registers are reset to zero and the audio output is silenced. This pin is normally connected to the reset line of the microprocessor or a power-on-clear circuit.

$\phi 2$ (Pin 6)

This TTL-level input is the master clock for SID. All oscillator frequencies and envelope rates are referenced to this clock. $\phi 2$ also controls data transfers between SID and the microprocessor. Data can only be transferred when $\phi 2$ is high. Essentially, $\phi 2$ acts as a high-active chip select as far as data transfers are concerned. This pin is normally connected to the system clock, with a nominal operating frequency of 1.0 MHz.

R/W (Pin 7)

This TTL-level input controls the direction of data transfers between SID and the microprocessor. If the chip select conditions have been met, a high on this line allows the microprocessor to Read data from the selected SID register and a low allows the microprocessor to Write data into the selected SID register. This pin is normally connected to the system Read/Write line.

CS (Pin 8)

This TTL-level input is a low active chip select which controls data transfers between SID and the microprocessor. CS must be low for any transfer. A Read from the selected SID register can only occur if CS is low, $\phi 2$ is high and R/W is high. A Write to the selected SID register can only occur if CS is low, $\phi 2$ is high and R/W is low. This pin is normally connected to address decoding circuitry, allowing SID to reside in the memory map of a system.

A0–A4 (Pins 9–13)

These TTL-level inputs are used to select one of the 29 SID registers. Although enough addresses are provided to select 1 of 32 registers, the remaining three register locations are not used. A Write to any of these three locations is ignored and a Read returns invalid data. These pins are normally connected to the corresponding address lines of the microprocessor so that SID may be addressed in the same manner as memory.

GND (Pin 14)

For best results, the ground line between SID and the power supply should be separate from ground lines to other digital circuitry. This will minimize digital noise at the audio output.

D0–D7 (Pins 15–22)

These bidirectional lines are used to transfer data between SID and the microprocessor. They are TTL compatible in the input mode and capable of driving 2 TTL loads in the output mode. The data buffers are usually in the high-impedance off state. During a Write operation, the data buffers remain in the off (input) state and the microprocessor supplies data to SID over these lines. During a Read operation, the data buffers turn on and SID supplies data to the microprocessor over these lines. The pins are normally connected to the corresponding data lines of the microprocessor.

POTX,POTY (Pins 24,23)

These pins are inputs to the A/D converters used to digitize the position of potentiometers. The conversion process is based on the time constant of a capacitor tied from the POT pin to ground, charged by a potentiometer tied from the POT pin to +5 volts. The component values are determined by:

$$RC = 4.7E{-}4$$

Where R is the maximum resistance of the pot and C is the capacitor.

The larger the capacitor, the smaller the POT value jitter. The recommended values for R and C are 470 kΩ and 1000 pF. Note that a separate pot and cap are required for each POT pin.

V_{CC} (Pin 25)

As with the GND line, a separate +5 VDC line should be run between SID V_{CC} and the power supply in order to minimize noise. A bypass capacitor should be located close to the pin.

EXT IN (Pin 26)

This analog input allows external audio signals to be mixed with the audio output of SID or processed through the Filter. Typical sources include voice, guitar, and organ. The input impedance of this pin is on the order of 100 kΩ. Any signal applied directly to the pin should ride at a DC level of 6 volts and should not exceed 3 volts p–p. In order to pre-

vent any interference caused by DC level differences, external signals should be AC-coupled to EXT IN by an electrolytic capacitor in the 1–10 μF range. As the direct audio path (FILTEX=0) has unity gain, EXT IN can be used to mix outputs of many SID chips by daisy-chaining. The number of chips that can be chained in this manner is determined by the amount of noise and distortion allowable at the final output. Note that the output Volume control will affect not only the three SID voices, but also any external inputs.

AUDIO OUT (Pin 27)

This open-source buffer is the final audio output of SID, comprised of the three SID voices, the Filter and any external input. The output level is set by the output Volume control and reaches a maximum of 2 volts p–p at a DC level of 6 volts. A source resistor from AUDIO OUT to ground is required for proper operation. The recommended resistance is 1 kΩ for a standard output impedance.

As the output of SID rides at a 6-volt DC level, it should be AC-coupled to any audio amplifier with an electrolytic capacitor in the 1–10 μF range.

V_{DD} (Pin 28)

As with V_{CC}, a separate +12 VDC line should be run to SID V_{DD} and a bypass capacitor should be used.

6581 SID CHARACTERISTICS

ABSOLUTE MAXIMUM RATINGS

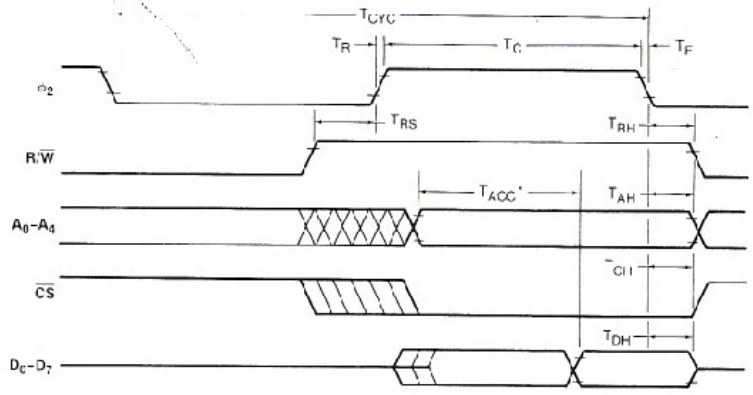
RATING	SYMBOL	VALUE	UNITS
Supply Voltage	V_{DD}	−0.3 to +17	VDC
Supply Voltage	V_{CC}	−0.3 to +7	VDC
Input Voltage (analog)	V_{ina}	−0.3 to +17	VDC
Input Voltage (digital)	V_{inc}	−0.3 to +7	VDC
Operating Temperature	T_A	0 to +70	°C
Storage Temperature	T_{STG}	−55 to +150	°C

ELECTRICAL CHARACTERISTICS (V_{DD}=12 VDC±5%, V_{CC}=5 VDC±5%, T_A=0 to 70° C)

CHARACTERISTIC		SYMBOL	MIN	TYP	MAX	UNITS
Input High Voltage Input Low Voltage	(RES, ϕ2, R/W, CS, A0–A4, D0–D7)	V_{IH} V_{IL}	2 −0.3	— —	V_{CC} 0.8	VDC VDC
Input Leakage Current	(RES, ϕ2, R/W, CS, A0–A4; V_{in}=0–5 VDC)	I_{in}	—	—	2.5	μA
Three-State (Off)	(D0–D7; V_{CC}=max)	I_{TSI}	—	—	10	μA
Input Leakage Current	V_{in}=0.4–2.4 VDC					
Output High Voltage	(D0–D7; V_{CC}=min, I load=200 μA)	V_{OH}	2.4	—	V_{CC}−0.7	VDC
Output Low Voltage	(D0–D7; V_{CC}=max, I load=3.2 mA)	V_{OL}	GND	—	0.4	VDC
Output High Current	(D0–D7; Sourcing, V_{OH}=2.4 VDC)	I_{OH}	200	—	—	μA

Output Low Current	(D0–D7; Sinking, V_{OL}=0.4 VDC)	I_{OL}	3.2	—	—	mA
Input Capacitance	(RES, ϕ2, R/W, CS, A0–A4, D0–D7)	C_{in}	—	—	10	pF
Pot Trigger Voltage	(POTX, POTY)	V_{pot}	—	$V_{CC}/2$	—	VDC
Pot Sink Current	(POTX, POTY)	I_{pot}	500	—	—	μA
Input Impedance	(EXT IN)	R_{in}	100	150	—	kΩ
Audio Input Voltage	(EXT IN)	V_{in}	5.7 —	6 0.5	6.3 3	VDC VAC
Audio Output Voltage	(AUDIO OUT; 1 kΩ load, volume=max) One Voice on: All Voices on:	V_{out}	5.7 0.4 1.0	6 0.5 1.5	6.3 0.6 2.0	VDC VAC VAC
Power Supply Current	(V_{DD})	I_{DD}	—	20	25	mA
Power Supply Current	(V_{CC})	I_{CC}	—	70	100	mA
Power Dissipation	(Total)	P_D	—	600	1000	mW

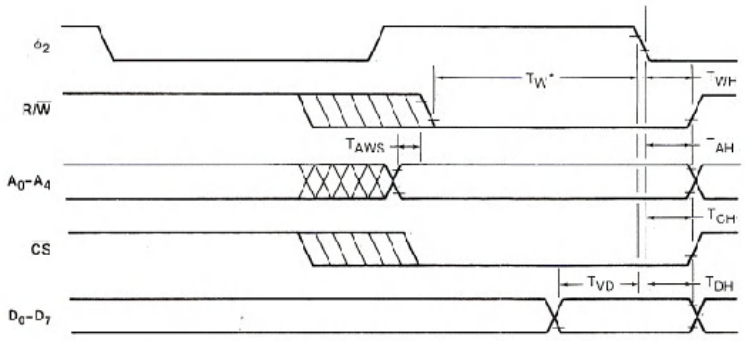
6581 SID TIMING



*T_{ACC} is measured from the latest occurring of ϕ_2, CS, A_0-A_4

READ CYCLE

SYMBOL	NAME	MIN	TYP	MAX	UNITS
T_{CYC}	Clock Cycle Time	1	—	20	μs
T_C	Clock High Pulse Width	450	500	10,000	ns
T_R, T_F	Clock Rise/Fall Time	—	—	25	ns
T_{RS}	Read Set-Up Time	0	—	—	ns
T_{RH}	Read Hold Time	0	—	—	ns
T_{ACC}	Access Time	—	—	300	ns
T_{AH}	Address Hold Time	10	—	—	ns
T_{CH}	Chip Select Hold Time	0	—	—	ns
T_{DH}	Data Hold Time	20	—	—	ns



*T_W is measured from the latest occurring of ϕ_2, $\overline{CS}$, R/$\overline{W}$.

WRITE CYCLE

SYMBOL	NAME	MIN	TYP	MAX	UNITS
T_W	Write Pulse Width	300	—	—	ns
T_{WH}	Write Hold Time	0	—	—	ns
T_{AWS}	Address Set-up Time	0	—	—	ns
T_{AH}	Address Hold Time	10	—	—	ns
T_{CH}	Chip Select Hold Time	0	—	—	ns
T_{VD}	Valid Data	80	—	—	ns
T_{DH}	Data Hold Time	10	—	—	ns

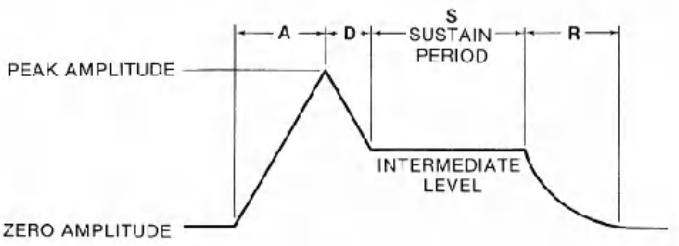
EQUAL-TEMPERED MUSICAL SCALE VALUES

The table in Appendix E lists the numerical values which must be stored in the SID Oscillator frequency control registers to produce the notes of the equal-tempered musical scale. The equal-tempered scale consists of an octave containing 12 semitones (notes): C,D,E,F,G,A,B and C#,D#,F#,G#,A#. The frequency of each semitone is exactly the 12th root of 2 ($\sqrt[12]{2}$) times the frequency of the previous semitone. The table is based on a $\phi 2$ clock of 1.02 MHz. Refer to the equation given in the Register Description for use of other master clock frequencies. The scale selected is concert pitch, in which A–4 = 440 Hz. Transpositions of this scale and scales other than the equal-tempered scale are also possible.

Although the table in Appendix E provides a simple and quick method for generating the equal-tempered scale, it is very memory inefficient as it requires 192 bytes for the table alone. Memory efficiency can be improved by determining the note value algorithmically. Using the fact that each note in an octave is exactly half the frequency of that note in the next octave, the note look-up table can be reduced from 96 entries to 12 entries, as there are 12 notes per octave. If the 12 entries (24 bytes) consist of the 16-bit values for the eighth octave (C-7 through B-7), then notes in lower octaves can be derived by choosing the appropriate note in the eighth octave and dividing the 16-bit value by two for each octave of difference. As division by two is nothing more than a right-shift of the value, the calculation can easily be accomplished by a simple software routine. Although note B-7 is beyond the range of the oscillators, this value should still be included in the table for calculation purposes (the MSB of B-7 would require a special software case, such as generating this bit in the CARRY before shifting). Each note must be specified in a form which indicates which of the 12 semitones is desired, and which of the eight octaves the semitone is in. Since four bits are necessary to select 1 of 12 semitones and three bits are necessary to select 1 of 8 octaves, the information can fit in one byte, with the lower nybble selecting the semitone (by addressing the look-up table) and the upper nybble being used by the division routine to determine how many times the table value must be right-shifted.

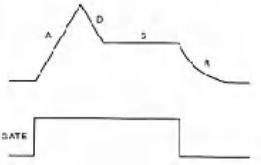
SID ENVELOPE GENERATORS

The four-part ADSR (ATTACK, DECAY, SUSTAIN, RELEASE) envelope generator has been proven in electronic music to provide the optimum trade-off between flexibility and ease of amplitude control. Appropriate selection of envelope parameters allows the simulation of a wide range of percussion and sustained instruments. The violin is a good example of a sustained instrument. The violinist controls the volume by bowing the instrument. Typically, the volume builds slowly, reaches a peak, then drops to an intermediate level. The violinist can maintain this level for as long as desired, then the volume is allowed to slowly die away. A "snapshot" of this envelope is shown below:



This volume envelope can be easily reproduced by the ADSR as shown below, with typical envelope rates:

ATTACK:	10 ($A)	500 ms
DECAY:	8	300 ms
SUSTAIN:	10 ($A)	
RELEASE:	9	750 ms

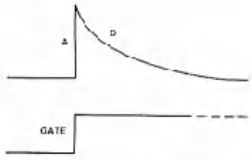


Note that the tone can be held at the intermediate SUSTAIN level for as long as desired. The tone will not begin to die away until GATE is cleared. With minor alterations, this basic envelope can be used for brass and woodwinds as well as strings.

An entirely different form of envelope is produced by percussion instruments such as drums, cymbals and gongs, as well as certain keyboards such as pianos and harpsichords. The percussion envelope is characterized by a nearly instantaneous attack, immediately followed by a decay to zero volume. Percussion instruments cannot be sustained

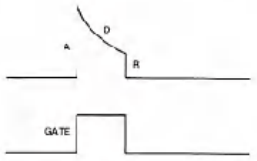
at a constant amplitude. For example, the instant a drum is struck, the sound reaches full volume and decays rapidly regardless of how it was struck. A typical cymbal envelope is shown below:

ATTACK:	0	2ms
DECAY:	9	750ms
SUSTAIN:	0	
RELEASE:	9	750ms



Note that the tone immediately begins to decay to zero amplitude after the peak is reached, regardless of when GATE is cleared. The amplitude envelope of pianos and harpsichords is somewhat more complicated, but can be generated quite easily with the ADSR. These instruments reach full volume when a key is first struck. The amplitude immediately begins to die away slowly as long as the key remains depressed. If the key is released before the sound has fully died away, the amplitude will immediately drop to zero. This envelope is shown below:

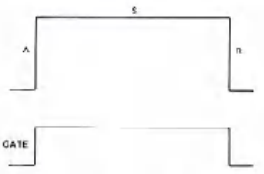
ATTACK:	0	2 ms
DECAY:	9	750 ms
SUSTAIN:	0	
RELEASE:	0	6 ms



Note that the tone decays slowly until GATE is cleared, at which point the amplitude drops rapidly to zero.

The most simple envelope is that of the organ. When a key is pressed, the tone immediately reaches full volume and remains there. When the key is released, the tone drops immediately to zero volume. This envelope is shown below:

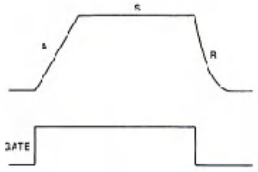
ATTACK:	0	2 ms
DECAY:	0	6 ms
SUSTAIN:	15 ($F)	
RELEASE:	0	6 ms



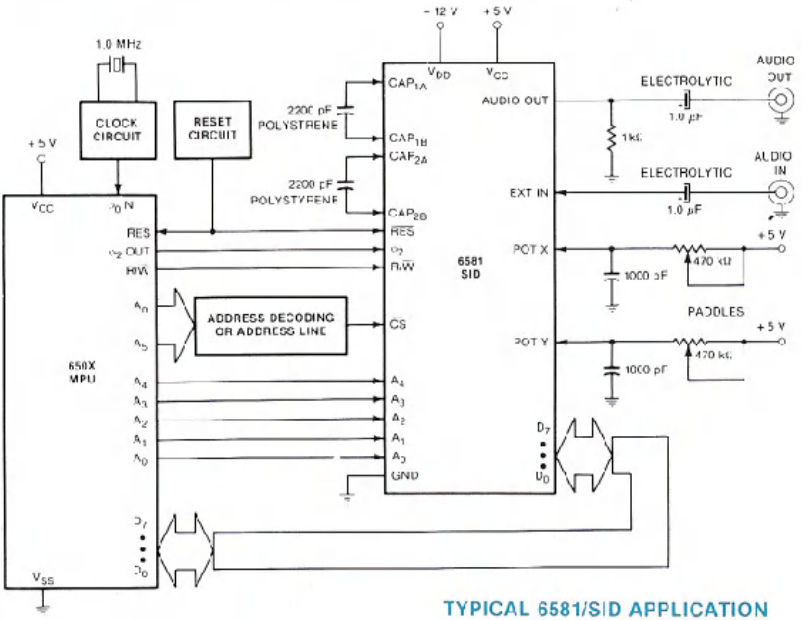
The real power of SID lies in the ability to create original sounds rather than simulations of acoustic instruments. The ADSR is capable of creating envelopes which do not correspond to any "real" instruments. A good example would be the "backwards" envelope. This envelope is characterized by a slow attack and rapid decay which sounds very

much like an instrument that has been recorded on tape then played backwards. This envelope is shown below:

ATTACK:	10 ($A)	500 ms
DECAY:	0	6 ms
SUSTAIN:	15 ($F)	
RELEASE:	3	72 ms



Many unique sounds can be created by applying the amplitude envelope of one instrument to the harmonic structure of another. This produces sounds similar to familiar acoustic instruments, yet notably different. In general, sound is quite subjective and experimentation with various envelope rates and harmonic contents will be necessary in order to achieve the desired sound.



TYPICAL 6581/SID APPLICATION

APPENDIX P

GLOSSARY

ADSR	Attack/Decay/Sustain/Release envelope.
attack	Rate at which musical note reaches peak volume.
binary	Base-2 number system.
Boolean operators	Logical operators.
byte	Memory location.
CHROMA noise	Color distortion.
CIA	Complex Interface Adapter.
DDR	Data Direction Register.
decay	Rate at which musical note falls from peak volume to sustain volume.
decimal	Base-10 number system.
e	Mathematical constant (approx. 2.71828183).
envelope	Shape of the volume of a note over time.
FIFO	First-In/First-Out.
hexadecimal	Base-16 number system.
integer	Whole number (without decimal point).
jiffy clock	Hardware interval timer.
NMI	Non-Maskable Interrupt.
octal	Base-8 number system.
operand	Parameter.
OS	Operating System.
pixel	Dot of resolution on the screen.
queue	Single-file line.
register	Special memory storage location.
release	Rate at which a musical note falls from sustain volume to no volume.
ROM	Read-Only Memory.
SID	Sound Interface Device.
signed numbers	Plus or minus numbers.
subscript	Index variable.
sustain	Volume level for sustain of musical note.
syntax	Programming sentence structure.
truncated	Cut off, eliminated (*not* rounded).
VIC-II	Video Interface Chip.
video screen	Television set.

INDEX

COMMODORE 64 QUICK REFERENCE CARD

SIMPLE VARIABLES

Type	Name	Range
Real	XY	=1.70141163E+38
		=2.93873588E-39
Integer	XY%	=32767
String	XY$	0 to 255 characters

X is a letter (A-Z), Y is a letter or number (0-9). Variable names can be more than 2 characters, but only the first two are recognized.

ARRAY VARIABLES

Type	Name
Single Dimension	XY(5)
Two-Dimension	XY(5,5)
Three-Dimension	XY(5,5,5)

Arrays of up to eleven elements (subscripts 0-10) can be used where needed. Arrays with more than eleven elements need to be DIMensioned.

ALGEBRAIC OPERATORS

= Assigns value to variable
- Negation
↑ Exponentiation
* Multiplication
/ Division
+ Addition
- Subtraction

RELATIONAL AND LOGICAL OPERATORS

= Equal
<> Not Equal To
< Less Than
> Greater Than
<= Less Than or Equal To
>= Greater Than or Equal To
NOT Logical "Not"
AND Logical "And"
OR Logical "Or"
Expression equals 1 if true, 0 if false.

SYSTEM COMMANDS

LOAD "NAME"	Loads a program from tape
SAVE "NAME"	Saves a program on tape
LOAD "NAME",8	Loads a program from disk
SAVE "NAME",8	Saves a program to disk
VERIFY "NAME"	Verifies that program was SAVEd without errors
RUN	Executes a program
RUN xxx	Executes program starting at line xxx
STOP	Halts execution
END	Ends execution
CONT	Continues program execution from line where program was halted
PEEK(X)	Returns contents of memory location X
POKE X,Y	Changes contents of location X to value Y
SYS xxxxx	Jumps to execute a machine language program, starting at xxxxx
WAIT X,Y,Z	Program waits until contents of location X, when EORed with Z and ANDed with Y, is nonzero.
USR(X)	Passes value of X to a machine language subroutine

EDITING AND FORMATTING COMMANDS

LIST	Lists entire program
LIST A-B	Lists from line A to line B
REM Message	Comment message can be listed but is ignored during program execution
TAB(X)	Used in PRINT statements. Spaces X positions on screen
SPC(X)	PRINTs X blanks on line
POS(X)	Returns current cursor position
CLR/HOME	Positions cursor to left corner of screen
SHIFT CLR/HOME	Clears screen and places cursor in "Home" position
SHIFT INST/DEL	Inserts space at current cursor position
INST/DEL	Deletes character at current cursor position
CTRL	When used with numeric color key selects text color. May be used in PRINT statement.
CRSR Keys	Moves cursor up, down, left, right on screen
Commodore Key	When used with SHIFT selects between upper/lower case and graphic display mode. When used with numeric color key, selects optional text color

ARRAYS AND STRINGS

DIM A(X,Y,Z)	Sets maximum subscripts for A; reserves space for (X+1)*(Y+1)*(Z+1) elements starting at A(0,0,0)
LEN (X$)	Returns number of characters in X$
STR$(X)	Returns numeric value of X, converted to a string
VAL(X$)	Returns numeric value of A$, up to first nonnumeric character
CHR$(X)	Returns ASCII character whose code is X
ASC(X$)	Returns ASCII code for first character of X$
LEFT$(A$,X)	Returns leftmost X characters of A$
RIGHT$(A$,X)	Returns rightmost X characters of A$
MID$(A$,X,Y)	Returns Y characters of A$ starting at character X

INPUT/OUTPUT COMMANDS

INPUT A$ OR A	PRINTs '?' on screen and waits for user to enter a string or value
INPUT "ABC";A	PRINTs message and waits for user to enter value. Can also INPUT A$
GET A$ or A	Waits for user to type one-character value; no RETURN needed
DATA A,"B",C	Initializes a set of values that can be used by READ statement
READ A$ or A	Assigns next DATA value to A$ or A
RESTORE	Resets data pointer to start READing the DATA list again
PRINT "A=";A	PRINTs string 'A=' and value of A. ';' suppresses spaces - ',' tabs data to next field

PROGRAM FLOW

GOTO X	Branches to line X
IF A=3 THEN 10	IF assertion is true THEN execute following part of statement. IF false, execute next line number
FOR A=1 TO 10 STEP 2 : NEXT	Executes all statements between FOR and corresponding NEXT, with A going from 1 to 10 by 2. Step size is 1 unless specified
NEXT A	Defines end of loop. A is optional
GOSUB 2000	Branches to subroutine starting at line 2000
RETURN	Marks end of subroutine. Returns to statement following most recent GOSUB
ON X GOTO A,B	Branches to Xth line number on list. If X = 1 branches to A, etc.
ON X GOSUB A,B	Branches to subroutine at Xth line number in list

ABOUT THE COMMODORE 64 PROGRAMMER'S REFERENCE GUIDE . . .

Game cartridge compatibility . . . spectacular sound . . . arcade style graphics . . . and high caliber computing capabilities make the Commodore 64 the most advanced personal computer in its class for home, business and educational use.

The **COMMODORE 64 PROGRAMMER'S REFERENCE GUIDE** tells you everything you need to know about your Commodore 64. The perfect companion to your Commodore 64 User's Guide, this manual presents detailed information on everything from graphics and sound to advanced machine language techniques. This book is a must for everyone from the beginner to the advanced programmer.

For the beginner, the most complicated topics are explained with many sample programs and an easy-to-read writing style. For the advanced programmer, this book has been subjected to heavy pre-testing with your needs in mind. And it's designed so that you can easily get the most out of your Commodore 64's extensive capabilities.

commodore COMPUTER

Commodore Business Machines, Inc.—Computer Systems Division,
487 Devon Park Drive, Wayne, PA 19087.

DISTRIBUTED BY

Howard W. Sams & Co., Inc.
4300 W. 62nd Street, Indianapolis, Indiana 46268 USA

$19.95/22056

ISBN 0-672-22056-3