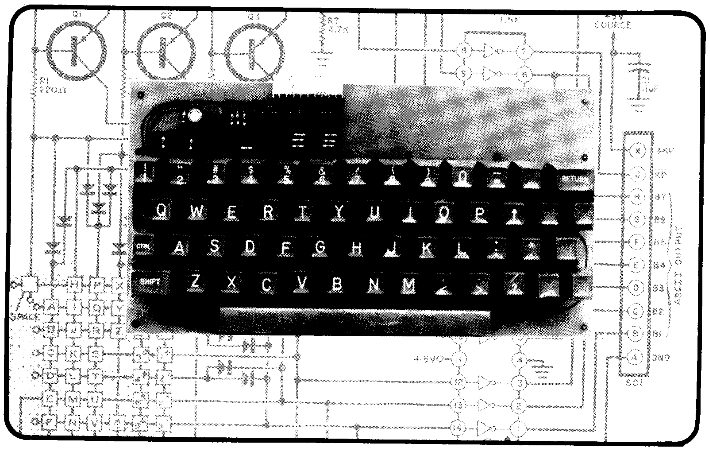


KEYBOARD
and
ENCODER

SWTPC 219 W. Rhapsody, San Antonio, Texas 78216

KBD-5 KEYBOARD

General Description

The KBD-5 keyboard and ASCII encoder utilizes a single MOS intergrated circuit to perform the bulk of all necessary operations to generate full 8 bit parallel ASCII output code complete with a parity bit for error detection. Debounce is internal and automatic and choice between positive or negative keypressed output strobe is provided. The keyboard can be programmed for upper case characters only, or upper and lower case characters. The KBD-5 has the features of n key lockout and 2 key rollover to help in reducing typing errors. Once a key is pressed pressing other keys will have no effect until the first key is released. The KBD-5 is already programmed for the control functions LINE FEED, RETURN and ESCAPE and can easily be programmed for other control functions. The keyboard also has two SPST locking keyswitches which can be used for turning the RECEIVE, TRANSMIT and ECHO functions of a terminal such as the CT-1024 on and off. The keyboard also has an automatic repeat function which causes a string of characters to be transmitted after a key has been depressed for more than about 0.5 seconds.

Assembling your Keyboard and ASCII Encoder

NOTE: MOS integrated circuits are susceptable to damage by static electricity. Although some degree of protection is provided internally within the integrated circuits, their cost demands the utmost in care. Before opening and/or installing any MOS integrated circuits you should ground your body and all metallic tools coming into contact with the leads through a 1M ohm 1/4 watt resistor (supplied with the kit). The ground must be an "earth" ground such as a water pipe, and not the circuit board ground. As for the connection to your body, attach a clip lead to your watch or metal ID bracelet. Make absolutely sure you have the 1M ohm resistor connected between you and the "earth" ground, otherwise you will be creating a dangerous shock hazard. Avoid touching the leads of the integrated circuits any more than necessary when installing them even if you are grounded. On those MOS IC's being soldered in place, the tip of the soldering iron should be grounded as well (seperately from your body ground) either with or without a 1 Meg ohm resistor. Most soldering irons having a three prong line cord already have a grounded tip. Static electricity should be an important consideration in cold, dry environments. It is less of a problem when it is warm and humid.

(✓) Install the 14 jumpers on the PC board. Use some of the light gauge wire supplied with the kit or excess resistor leads. On the longer jumpers use <u>insulated</u> wire to keep the jumper from shorting to other components. Jumpers are denoted on the circuit board by solid lines connecting two pads. As with all other component installation, unless otherwise noted, pull the jumpers down flush with the circuit board and solder.

(✓) Install the resistors and capacitors on the circuit board using the component layout on the board and the parts list. Bend the leads over on the back side of the board and trim so that 1/16" to 1/8" of lead remains. Solder.

(✓) Install integrated circuits IC2-IC3 on the board. Be sure to orient the IC's as shown on the board and do not bend the leads on the back side of the board. Doing so makes it very difficult to remove the IC should replacement ever be necessary. Do not install IC1 at this time. Solder.

(✓) Install the transistor and diodes on the board. Leave about 1/4" clearance between the transistor body and the circuit board. Be sure to orient the components exactly as shown on the component side of the board. Solder.

(✓) Check the key switches with an ohmmeter before they are installed to be sure that they are making contact properly and to be sure that they do not stick. Solder the switches in place one at a time using as little heat as necessary to get a good solder joint. Note that the keyswitches for the RCVE/XMIT and ECHO positions are a SPST switch and are different from the rest. The switches can be recognized by pressing each keyswitch. The SPST switches will have a locking position. Install these two switches first. Be careful not to force any switch into a hole in such a way as to damage the connecting wires. Excessive heating can distort, or even melt the plastic body and ruin the keyswitch. Check the switch again with an ohmmeter for proper operation after soldering it in place. This can save much troubleshooting later. If the switch works properly heat stake the switch in place by melting over the ends of the plastic mounting bosses. This can be done with the tip of your soldering iron.

(✓) After all keyswitches have been installed and checked you are ready to install the programming strips on the bottom of the board. These strips mount vertically, at right angles to the main board. The etched finger, connection points on the programming strips will match the connection pads on the main board when the strips are in the correct position. Hold one of the strips in the correct position and turn it so that the connection fingers match the pads on both sides of the strip. Note that there are two ways the strip may be turned and only one is correct. If the strips edge is too rough to fit down solidly against the main board, file, or sand that edge flat. Hold, or clamp sides of the strip to the main board, pads. Install the other strip in the same way. Both strips are identical.

(✓) The spacebar and equalizer assembly goes together as follows. Mount a keyswitch on the board in the center of the spacebar area. Mount the two "L" shaped brackets with a notch in the top on each side of the keyswitch in the holes provided. The side of the bracket with the notch should be next to the circuit board edge. These should be fixed in place by melting and flattenting slightly the plastic pins on the back of the board with your soldering iron tip. Press one of the equalizer wire retainers into place on the end of the spacebar itself. The hole for the wire should be on the more slanted side of the spacebar that has the four casting bumps. Slip the equalizer wire into the hole in the retainer you have not yet pressed into place. Press the second retainer into place. Turn the spacebar upside down in front of the board and press the equalizer wire into the notches in the top of the mounting clips. Turn the spacebar over and position it over the keyswitch in the center of the board. Press it down into place.

( ) Install IC1 following the precautions given for MOS IC's. Solder.

(✓) To finish the keyboard assembly snap the plastic key tops on their respective keyswitch as noted on the key configuration drawing. If some of the keytops are loose you can securely fasten them to the switches by using a small amount of plastic cement. Blank keytops may be supplied for the RCVE/XMIT and ECHO positions.

The programming of the "CASE", "PARITY" and the "KEYPRESSED" jumpers depends on what your keyboard will be used for. For upper case characters only connect a jumper from the pad marked "CASE" to "U". For upper plus lower case connect the jumper from the "CASE" pad to "U + L". For a positive keypressed strobe (KP line normally low, goes high when a key is pressed) connect a jumper from the KP pad to "+", for a negative keypressed strobe (KP line normally high, goes low when a key is pressed) connect a jumper from the KP pad to "-". A jumper for "PAR" should also be installed to its respective "U" or "U + L" pad depending on whether upper case or upper plus lower case characters will be used.

Notice to CT-1024 Terminal Owners

In order for the KBD-5 to be used on the CT-1024 terminal the keyboard should be programmed for a negative keypressed strobe (KP select jumper connected to -) and for upper case characters only, ("CASE" jumper connected to U). This is the normal programming for other TV TYPEWRITERS but may need to be modified for your particular application. The parity bit is not used in the CT-1024 but the "PAR" jumper should be connected to "U". The terminals R, T and E of J1 can be connected to control the receive/transmit and echo functions of your CT-1024 if a serial interface is installed. Connect R to JS-1 pin 5, T to JS-1 pin 4 and E to JS-1 pin 8 of the serial interface.

Checkout

The only equipment needed to check the operation of your keyboard is a DC milliammeter and a DC voltmeter. Apply +5 volts, ground, and -12 volts to the proper pins and check for currents of about 4 mA on the -12 line and about 8mA on the +5 line (no keys pressed). If the currents are much higher than this stop immediately and check for parts inserted incorrectly or solder bridges. If the currents check OK check the keypressed output. If the KP select jumper is set for (-) the KP output should be high (4.5V nominal) and go low (0.1 volts nominal) when a key is pressed. If the KP output is held low check for possible solder bridges or a shorted key switch. If the KP select jumper is set for (+) the KP output will be normally low and will go high when a kev is pressed. If all seems well refer to the ASCII code tables supplied and check the outputs of bits 1-7 with your voltmeter for each individual character. Be sure to use the correct table depending on how the "CASE" selector is wired on your keyboard. With the "CASE" selector set for the upper case only the SHIFT control will have no effect on the output code when any letter is pressed but will make bit 5 a 0 when any character of column 3 of Table 1 is pressed. When in the upper case only mode the SHIFT key will not affect any of the characters in columns 4 or 5. Pressing the control key forces bit 7 to be a 0 therefore allowing you to select the operators in columns 0 and 1. When in the upper plus lower case mode the SHIFT key makes bit 6 a 1 when letters of columns

4 and 5 of table 2 are pressed and makes bit 5 a 0 when any character of column 3 is selected. The control key operates as before.

This keyboard uses EVEN parity in both the upper case only and the upper plus lower case modes. The parity bit should be low when the number of zeros in the output code is odd and should be high when the number of zeros is even. For example the ASCII code for X, 0001101, should make the parity bit high. The repeat function should be checked by holding any one key down for more than about 1 sec. The keypressed strobe should switch on and off until the key is released. If a faster or slower repeat rate is desired the value of C6 can be changed slightly.

Circuit Description

As shown in figure 1 the keyswitches are arranged into a matrix with the X lines from IC-1 outputing test pulses in time sequence and the Y lines sensing their presence. If no key is pressed, IC-1 continuously scans the keys, at a rate of 50,000 keys per second. Whenever a key is closed, the scanning action stops and that location is held by the internal circuitry of the encoder. This location is called an address. This address is routed to an internal fixed memory called a read only memory. In exchange for an address and some information on the status of the shift and control keys the read only memory gives the proper 8 bit ASCII output code, complete with parity.

As soon as the scanning stops, a time delay is started and controlled by R1 and C1. This takes into account any switch closure bounce or noise and makes sure the key is firmly down and not just brushed on the way by. After the delay time, an output "keypressed" strobe is made available. This tells whatever you attach to your keyboard that the code is ready for use and valid.

If a key is released the scanning action starts up again and goes on till a new key is pressed. If two keys are pressed at nearly the same time the first key pressed provides its output code after a debounce delay. When the first key is released, the scanner starts up but only goes around till it hits the other key's location. After a new debounce delay, the second key's code is output. Keys can continue to be pressed in sequence two down at a time forever, with always the right code being output in sequence, and nothing missed or out of order. This most handy feature takes care of sloppy typing and "burst" rate typing where keys are hit fast and furious in sequence.

IC2 and IC3 are used for the automatic repeat function. IC3 is setup as an astable multivibrator that continuously runs at a frequency of approximately 8.5 Hz. When a key is first pressed the KP line of IC1 goes high causing an immediate low output of IC2-C. If the key is immediately released no repeat function is initiated. If the key is held down for more than about 1 second C7 will charge to a point where IC2A is triggered. At this time the output of IC3 is NAND'ed with the KP strobe therefore causing the KP signal to be chopped just as if the key was continously pushed and released. This action will continue until the key is released.

In Case of Problems

The tests called for in the check-out procedure are designed to warn you of problems before they can cause damage. If any abnormal results are obtained during the tests, or if there is obvious overheating of any part discontinue use until the problem is located and corrected. Experience has shown that most problems are caused by minor wiring, component installation and soldering technique errors which can be prevented by careful assembly.

If you do have problems with your keyboard there are several tests you can perform which may help you locate the problem. You will need an ohmmeter, an oscilloscope, and a DC voltmeter for these tests. First go back and check once more for proper component installation and for solder bridges or cold solder joints. Also be sure that the correct jumpers are in place. Next check each keyswitch with an ohmmeter to be sure none are shorted. If you have an oscilloscope apply power to the keyboard and check for a 50 KHz signal on pin 40 of IC-1. If you obtain no signal check for 5 volts on pin 1, -12 on pin 18 and ground on pin 17.

Repair Procedure

Repairs will be made on a basic rate charge plus parts. The basic rate for the KBD-5 is $10.00. If you must return your KBD-5 please send the $10.00 in the form of a money order or cashiers check along with the keyboard. When repairs are finished the keyboard will be returned to you COD for parts charges, if any. Do Not send personal checks for repair work.

Pack all parts to be returned carefully and insure. We will not accept delivery on any parcels that arrive in damaged condition. Make check or money order payable to Southwest Technical Products Corporation.

Parts List - KBD-5 Keyboard

Resistors

R1 ✓	680K ohm 1/4 watt resistor
R2 ✓	100K " " " "
R3 ✓	150K " " " "
R4 ✓	47K " " " "
R5 ✓	10K " " " "
R6 ✓	680 " " " "
R7 ✓	330 " " " "

Capacitors

C1, C5 ✓ ✓	0.01 mfd disc capacitor
C2 ✓	47 pf polystyrene capacitor
C3, C4 ✓ ✓	0.1 mfd @16 volt disc capacitor
C6	0.47 mfd tantalum capacitor
C7 ✓	220 mfd @6.3 volt electrolytic capacitor

Semiconductors

IC1	2376 Keyboard Encoder (MOS)
IC2 ✓	74LS00 quad NAND gate
IC3 ✓	555 timer
Q1 ✓	2N5210 NPN silicon transistor
D1 - D3 ✓	1N4148/1N914 silicon diode

Misc.

J1	15 pin Molex edge connector
S1-S2	Keyboard switch, SPST locking
S3-S56	Keyboard switch

PC board, programming jumpers (2),
Keytops

BIT NUMBERS								0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1
b_7 ↓	b_6 ↓	b_5 ↓	b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	COLUMN → ROW↓	0	1	2	3	4	5
			0	0	0	0	0	NUL	DLE	SP	0	@	P
			0	0	0	1	1	SOH	DC1	'	1	A	Q
			0	0	1	0	2	STX	DC2	"	2	B	R
			0	0	1	1	3	ETX	DC3	#	3	C	S
			0	1	0	0	4	EOT	DC4	$	4	D	T
			0	1	0	1	5	ENQ	NAK	%	5	E	U
			0	1	1	0	6	ACK	SYN	&	6	F	V
			0	1	1	1	7	BEL	ETB	'	7	G	W
			1	0	0	0	8	BS	CAN	(	8	H	X
			1	0	0	1	9	HT	EM	)	9	I	Y
			1	0	1	0	10	LF	SUB	*		J	Z
			1	0	1	1	11	VT	ESC	+	,	K	[
			1	1	0	0	12	FF	FS	,	<	L	\
			1	1	0	1	13	CR	GS	-	=	M	]
			1	1	1	0	14	SO	RS		>	N	∧
			1	1	1	1	15	SI	US	/	?	O	-

Available characters and commands for the UPPER CASE mode

AY-5-2376 Keyboard Encoder

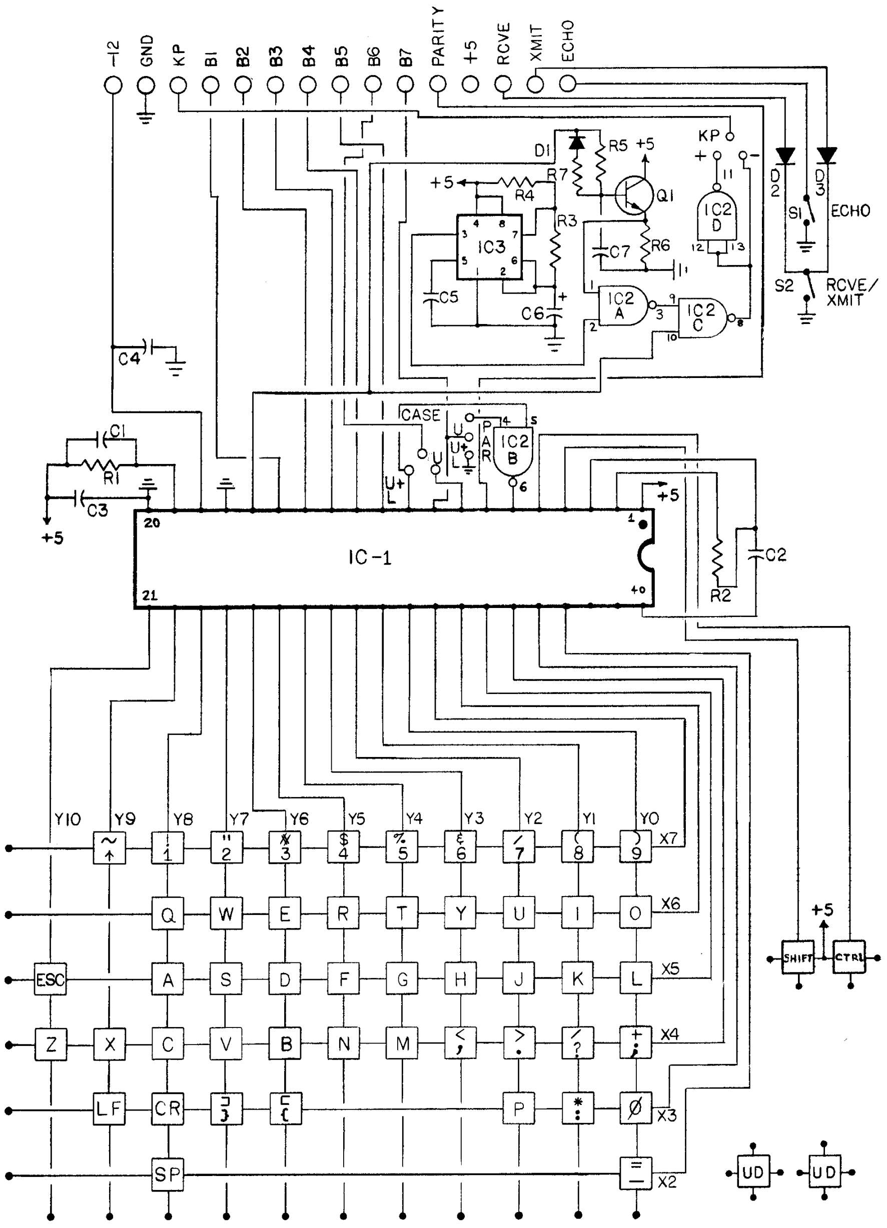
BIT NUMBERS								0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_7 ↓	b_6 ↓	b_5 ↓	b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	COLUMN → ROW↓	0	1	2	3	4	5	6	7
			0	0	0	0	0	NUL	DLE	SP	0	@	P	\	p
			0	0	0	1	1	SOH	DC1	'	1	A	Q	a	q
			0	0	1	0	2	STX	DC2	"	2	B	R	b	r
			0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
			0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
			0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
			0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
			0	1	1	1	7	BEL	ETB		7	G	W	g	w
			1	0	0	0	8	BS	CAN	(	8	H	X	h	x
			1	0	0	1	9	HT	EM	)	9	I	Y	i	y
			1	0	1	0	10	LF	SUB	*		J	Z	j	z
			1	0	1	1	11	VT	ESC	+		K	[	k	{
			1	1	0	0	12	FF	FS	,	<	L	\	l	(
			1	1	0	1	13	CR	GS	-	=	M	]	m	}
			1	1	1	0	14	SO	RS		>	N	∧	n	~
			1	1	1	1	15	SI	US	/	?	O	-	o	DEL

Available codes and characters for the UPPER PLUS LOWER CASE mode

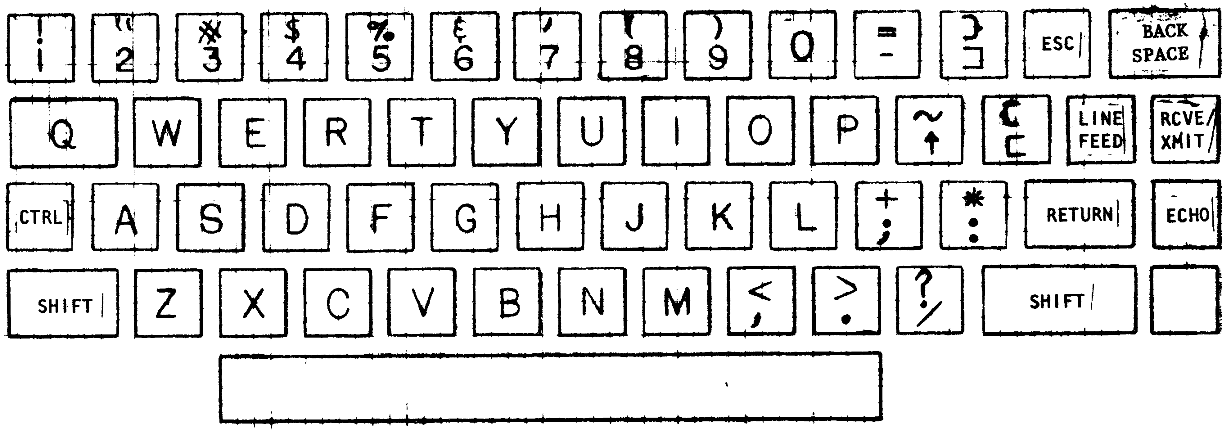
AY-5-2376 Keyboard Encoder

CONNECT → ↓ TO OBTAIN ↘		X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7
Y_0	NORMAL	NUL	DLE	-	Ø	;	l	o	9
	SHIFT	NUL	DLE	=	NUL	+	L	O	)
	CONTROL	NUL	DLE	NUL	NUL	NUL	FF	SI	NUL
Y_1	NORMAL	SOH	K	FS		/	k	i	8
	SHIFT	SOH	[	FS	*	?	K	I	(
	CONTROL	SOH	VT	FS	NUL	NUL	VT	HT	NUL
Y_2	NORMAL	STX	L	GS	p		j	u	7
	SHIFT	STX	\	GS	P	>	J	U	'
	CONTROL	STX	FF	GS	DLE	NUL	LF	NAK	NUL
Y_3	NORMAL	ETX	N	RS	_	,	h	y	6
	SHIFT	ETX	^	RS	DEL	<	H	Y	8
	CONTROL	ETX	SO	RS	US	NUL	BS	EM	NUL
Y_4	NORMAL	EOT	M	US	@	m	g	t	5
	SHIFT	EOT	]	US	`	M	G	T	%
	CONTROL	EOT	CR	US	NUL	CR	BEL	DC$_4$	NUL
Y_5	NORMAL	ENQ	NAK	<	BS	n	f	r	4
	SHIFT	ENQ	NAK	<	BS	N	F	R	$
	CONTROL	ENQ	NAK	NUL	BS	SO	ACK	DC$_2$	NUL
Y_6	NORMAL	ACK	SYN	>	[	b	d	e	3
	SHIFT	ACK	SYN	>	{	B	D	E	#
	CONTROL	ACK	SYN	NUL	ESC	STX	EOT	ENQ	NUL
Y_7	NORMAL	BEL	ETB	.	]	v	s	w	2
	SHIFT	BEL	ETB	,	}	V	S	W	"
	CONTROL	BEL	ETB	NUL	GS	SYN	DC$_3$	ETB	NUL
Y_8	NORMAL	DC1	CAN	SP	CR	c	a	q	1
	SHIFT	DC1	CAN	SP	CR	C	A	Q	!
	CONTROL	DC1	CAN	SP	CR	ETX	SOH	DC1	NUL
Y_9	NORMAL	P	EM	.	LF	x	FF	HT	^
	SHIFT	@	EM	.	LF	X	FF	HT	~
	CONTROL	DLE	EM	NUL	LF	CAN	FF	HT	RS
Y_{10}	NORMAL	0	SUB	_	DEL	z	ESC	VT	\
	SHIFT	_	SUB	_	DEL	Z	ESC	VT	
	CONTROL	SI	SUB	US	DEL	SUB	ESC	VT	FS

Code Assignment Chart - AY-5-2376 Keyboard Encoder



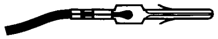
Schematic - KBD - 5 Keyboard and Encoder



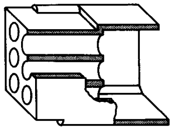
Key Configuration - KBD - 5 Keyboard

Connector Reference Sheet

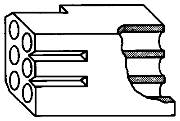
In order to avoid confusion in distinguishing between the various connectors supplied with our many kits, we are including this connector reference sheet with the kit instruction set. We have had a great many customers interchange the male and female connector shells when assembling their kits so we have clearly illustrated each connector along with its proper name and gender on this reference sheet. All are shown actual size.



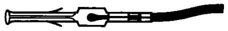
Male Pin



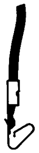
Molex Female Shell Connector



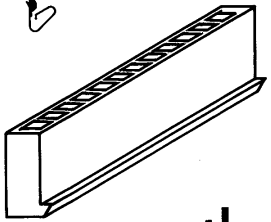
Molex Male Shell Connector



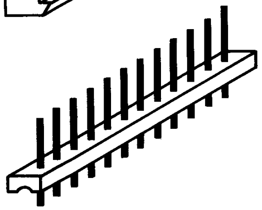
Female Pin



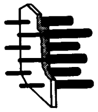
Pin for Harness Connector



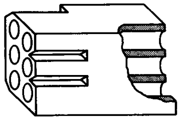
Harness Connector



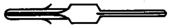
Straight Pin Edge Connector



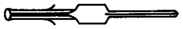
Molex Wafercon®



Molex Male Shell Connector



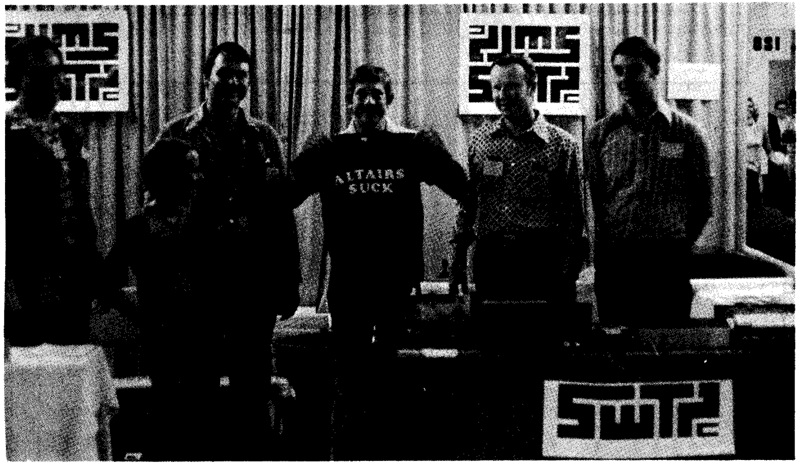
Male Solder Tail Pin PC Type



Female Solder Tail Pin PC Type

SWTPC 6800 COMPUTER NEWSLETTER Issue No. 2. October 1976

Southwest Technical Products Corporation
219 W. Rhapsody
San Antonio, Texas 78216



SWTPC WELL REPRESENTED AT PCC '76

Southwest Technical Products had a busy double booth at the Personal Computing Convention on August 28th and 29th in Atlantic City, New Jersey. We had two complete systems up and running and were amazed at the number of people who came by to ask questions and buy their copy of 4K Basic (C). Microcomputer Systems Inc., one of our Florida dealers, helped us out by providing us with some knowledgeable personnel and stock for those customers wanting to take their purchases with them

The convention was obviously an outstanding success. Our only complaint was that the convention was held in Atlantic City. It seemed that most all of the hotels/motels in the area left a lot to be desired. We always had to travel in groups(safety in number algorithm). The lack of air conditioning in the convention exhibit area was of no help either. The equipment worked but the operators didn't. Maybe us Texans have been spoiled by all of this good living.

Pictured above from left to right are Bill Thames, Rep. of Microcomputer Systems; Ted Uiterwyk, Software Debugger; Robert Uiterwyk, Author of Microbasic and 4K Basic(C); Gary Kay, SWTPC Engineer; Dan Meyer, President of SWTPC; and Joe Deres, SWTPC Engineer. The photo was taken by Jim Stratigos of the Atlanta Area Microcomputer Hobbyist Club. Jim spent most of his time at the convention helping us out at the booth. Also helping were Bill Blomgren, Rep. of Microcomputer Systems; Forrest Hurst, Mgr. of Microcomputer Systems; Warren Startup, Representative of Microcomputer Systems; and last but not least Steven Uiterwyk, Software Debugger.

SWTPC 6800-SOFTWARE

WARNING – It has been determined that reading this ad may be hazardous to your health, if you own another type computer system. We will not be responsible for ulcers, heartburn, or other complications if you persist in reading this material.

4 K BASIC© — 8 K BASIC©

- * Full floating point math
- * 1.0E-99 to 9.99999999E+99 number range
- * User programs may be saved and loaded
- * Direct mode provided for most statements
- * Will run most programs in 8K bytes of memory (4K Version)
  or 12K bytes of memory (8K Version)
- * USER function provided to call machine language programs
- * String variables and trig functions—8K BASIC only

COMMANDS

LIST
RUN
NEW
SAVE
LOAD
PATCH

STATEMENTS

REM
DIM
DATA
READ
RESTORE
LET*
FOR

END
GOTO*
ON. . .GOTO*
ON. . .GOSUB*
IF. . .THEN*
INPUT
PRINT*
NEXT

STOP
GOSUB*
PATCH*
RETURN
† DES
† PEEK
† POKE

FUNCTIONS

ABS
INT
RND
SGN
CHR
USER
TAB

† VAL
† EXT$
† LEN$
† LEFT$
† MID$
† RIGHT$

† SIN
† COS
† TAN
† EXP
† LOG
† SQR

** Direct mode statements*
† 8K Version only

MATH OPERATORS

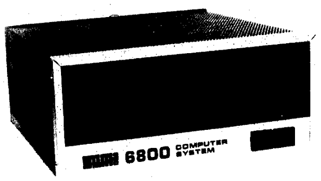
-	(unary) Negate
*	Multiplication
/	Division
+	Addition
–	Subtraction
† ↑	Exponent

RELATIONAL OPERATORS

=	Equal
< >	Not Equal
<	Less Than
>	Greater Than
< =	Less Than or Equal
>=	Greater Than or Equal

Your BANKAMERICARD welcome here

We honor master charge THE INTERBANK CARD



You guys are out of your minds, but who am I to complain. Send –

□ **4K BASIC CASSETTE $4.95** □ **MP-68 Computer**
□ **8K BASIC CASSETTE $9.95** **Kit $395.00**

NAME ______________________

ADDRESS ______________________

CITY ____________ **STATE** ______ **ZIP** ______

Southwest Technical Products Corp.
Box 32040, San Antonio, Texas 78284

SWTPC 4K BASIC VERSION 2.0 (C)

We are now distributing 4K Basic Version 2.0 (C), for those of you who bought versions 1.0 (C) or 1.1 (C). The differences and improvements are noted below. At the prices we are charging we cannot afford to continuously update tapes for those customers who bought an outdated version. So if you want version 2.0, send us another $4.95 and we will send you a new cassette tape and manual.

Features

1) An APPEND command has been added which functions as a LOAD command, but does not erase the previous source file.

2) On an INPUT command, failure to enter multiple variables when required is now properly handled. Also, entering an erroneous value such as a null or string is responded to by a reply of "RE-ENTER" instead of a forced exit from the Basic program.

3) Using Control X for line cancel now gives the reply "DEL" for deleted.

4) You may now use either CHR or CHR$ to call the character function. The CHR$ allows you to enter hex values rather than decimal values which are used by the CHR function.

5) FOR-NEXT loops will now allow a premature exit from a nested group without issuing an error 18. It should also be noted that 4K Basic Versions 1.1 and 2.0 (C) allow negative steps. This was not noted in the manual. Example: FOR I=10 TO-1 STEP -1

6) The INT function has been corrected so that values between 0 and -1.0 now correctly return-1.

7) The overflow checking routine (error 14) did not catch an overflow if it wrapped around memory. Example: DIM C4(255,255). This has been corrected.

8) Addition, subtraction and comparisons did not work correctly if one element was zero, and the other less than 1.E-10. This has been corrected.

9) Entry of an incorrectly formatted LIST command (Example: LIST 40,10) will now merely list from line 40 to the end of the source file.

10) Occasionally Basic 1.1 (C) gave an error 13 when it should have been an error 6. This has been corrected.

11) One of the fixes incorporporated into Basic 1.1 (C) removed the truncation of some mathematical results to nine digits. This has been restored.

12) The system will no longer insert a null line. A null line is an entry with a line number but no statement following it.

13) 0.001 - 0.01 erronously gave a positive answer. This has been fixed.

14) An underflow of 1.E-100 or 1.E-101 was not recognized and the value was not set to zero, thus a funny print value was returned. This has been corrected.

15) A multiplication or equivalent division of a small number (Example: 1E-70 * 1E-70) such that the result was less than 1E-127 gave a result of 9.99999999E99 rather than zero. This has been fixed.

The following corrections were made between Basic 1.0 (C) and 1.1 (C):

1) A divide 90 by 9.XXXX gave a result of zero. This was fixed.

2) Repeated multiplications of zero times a number occasionally created an error. This has been corrected.

3) SGN(0) was originally defined to be 1 whereas it should have been 0. This was changed in version 1.1 and you are asked to change your manuals accordingly.

PATCH FOR 4K BASIC (C)

If you have been having problems with the 4K Basic (c) SAVE and LOAD commands not working properly with the AC-30 cassette interface try the following patches: Version 1.0 and 1.1 - make $029D_{16}$ a $1E_{16}$. Version 2.0 - make $02AF_{16}$ a $1E_{16}$.

series with the wire going to jack J10 terminal A. Install the resistor right at jack J10.

( ) Solder a 0.1 mfd capacitor across jack J10 terminals A and B.

( ) Insert a 2.2 ohm 1/2 watt resistor in series with the wire going to jack J11 terminal A. Install the resistor right at jack J11.

( ) Solder a 0.1 mfd capacitor across jack J11 terminals A and B.

If you have easy access to the REMOTE jack terminals on the inside of the tape recorder, attach a 200 PIV 1 amp diode (1N4001) across the two terminals. Be sure you install the diode so it is oriented with the banded end

power supply, so the benefits of having both inside the same enclosure may be realized.

The case is made of 1/4" PLEXIGLASS and is 15" wide X 17" deep X 6" tall. It comes in two colors: clear and smoked. The smoked color is a dark bronze and makes it difficult to see the components inside the case. There are holes provided for two DB-25 (RS-232) connectors as well as several holes for ventilation. There is a large cutout for the SWTPC KBD-5 keyboard with PLEXIGLASS standoffs for mounting it. The case even has moulded feet and polished edges. The cost is $65.00 ppd. in the US. (specify color when ordering)

Order from: Microcomputer Systems,Inc.
144 S. Dale Mabry
Tampa, Florida 33609
(813) 879-4301 879-4225

SWTPC 8K BASIC

SWTPC 8K Basic (C) is now in the field testing stage and we expect to start delivering tapes and manuals soon. The price will be $9.95 for the "Kansas City" cassette tape and manual combination. 8K Basic (C) has the features of 4K basic (C) plus string handling capabilities with string arrays, trigonometric functions, exponentials and all sorts of other neat things. It is fundamentally full ANSI Basic with some additions and some small limitations due to the fact that we are using a microcomputer rather than an IBM 370.

MOD FOR MICROBASIC

If you are still using the SWTPC Microbasic package you might want to change memory location Ø36C from a Ø8 to a 1B. This will insure more random numbers when using the random number generator. If you have been having problems with your Microbasic not working properly and the memory diagnostics all check good, then you probably transposed some B's with some 8's or vice versa when you loaded Microbasic. The version of Microbasic printed in the first newsletter is accurate and does work!

ANIMALS FOR THE SWTPC 6800

SWTPC is now offering the ANIMALS game for the SWTPC 6800 thanks to the efforts of Doug Domke. The game of "Animals" was originally written in BASIC at DEC by Nathan Teichholz and is presented in "101 Computer Games" as an example of artificial intelligence.

This version of the game unlike the original which uses disk files, operates completely within the RAM of a SWTPC 6800 Computer System Although the program itself only requires 650 bytes of code, it theoretically has the ability to use up any amount of RAM as it"learns". In practice, however, even a 2K system is large enough to make the game very enjoyable.

The program itself is actually a guessing game. The program attempts to guess wnat animal the user is thinking of. If it fails, it will request the name of the animal, followed by a request for the user to supply a question which, when answered to the affirmative, would distinguish the correct answer from that guessed (incorrectly) by the program. This information then becomes a part of the program.

In response to the questions "Are you thinking of an animal?", the user may respond in one of four ways:

Y(ES) - the game continues
N(O) - the program exits to MIKBUG
L(IST) - the program lists its vocabulary of animals
C(LEAR) - the program clears everything it has learned and reverts to its initialized (dumb) state.

SWTPC CASSETTE TAPE PROGRAM LIBRARY

The following programs are available from SWTPC on AC-30 (Kansas City) formatted audio cassette tape:

MP-EC Editor/Assembler package with manual. Requires 8K of memory to run. $14.95 ppd. in US.

BAS4C 4K Basic Version 2.0 (C) with manual. Requires at least 6K of memory - 8K preferred. $4.95 ppd in US.

GAMIC Tic-Tac-Toe and Blackjack as listed in the notebook and newsletter. Require 6K of memory to run. $4.95 ppd in US.

BAS8C 8K Basic (Available shortly) with manual. Requires at least 8K of memory-12K preferred. $9.95 ppd in US.

ANIMC Animals program. A children's learning game. Requires 2K of memory. $4.95 ppd. in US.

SWTPC BASIC AVAILABLE ON PAPER TAPE

SWTPC has decided to offer 4K and 8K Basic (C) on paper tape:

BAS4P 4K Basic Version 2.0 (C) with manual. Requires at least 6K of memory - 8K preferred. $10.00 ppd. in US.

BAS8P 8K Basic with manual. Requires at least 8K of memory - 12K preferred. $20.00 ppd. in US. (Available Nov. 15, 1976.)

MOD FOR THE AC-30 CASSETTE INTERFACE

If you have been having trouble with your AC-30 not loading programs properly to the computer or while operating with the terminal ld)

MODIFICATION TO BASIC 2.0 (C) TO DRIVE

THE SWTPC PR-40 ALPHANUMERIC LINE PRINTER

By: W. C. Thames
Microcomputer Systems, Inc.

When the following code is added to Basic Version 2.0 (C), the PR-40 printer will be operated directly from the Basic interpreter. Basic will then print data on the terminal first, followed by the same data on the PR-40 printer. Changing the data in memory location 1219$_{16}$ from a 7E to a 39 will eliminate the terminal's printout entirely. If you wish to disable just the PR-40's printout, simply turn off the PR-40. Do not turn off the printer while the computer is outputting data. Doing so will interrupt the normal handshake routine and will cause the computer to lock up. This modification is only usable on the SWTPC PR-40 Alphanumeric Printer and in most cases will not drive other printers.

The patch has been written to output all printer data to a MP-L I/O board located at I/O card position #7 of a SWTPC 6800 Computer System. If you wish to relocate this interface to another card position simply change the address in memory locations 12Ø4$_{16}$ and 121D$_{16}$ to the address of the desired MP-L parallel interface board.

```
1200 37     121A E1     1234 F5     0101 12
1201 DF     121B D1     1235 8D     0102 1C
1202 1E     121C CE     1236 F3     0273 12
1203 CE     121D 80     1237 86     0274 2A
1204 80     121E 1C     1238 20     0276 12
1205 1C     121F C6     1239 20     0277 33
1206 A7     1220 FF     123A 0E     027B 12
1207 00     1221 E7     123B 44     027C 00
1208 C6     1222 00     123C 44     02AF 1E
1209 36     1223 C6     123D 44     07F9 12
120A E7     1224 3E     123E 44     07FA 4C
120B 01     1225 E7     123F 84
120C C6     1226 01     1240 0F
120D 3E     1227 7E     1241 8B
120E E7     1228 08     1242 30
120F 01     1229 0E     1243 81
1210 6D     122A A6     1244 39
1211 01     122B 00     1245 23
1212 2A     122C 8D     1246 02
1213 FC     122D 0D     1247 8B
1214 E6     122E A6     1248 07
1215 00     122F 00     1249 7E
1216 DE     1230 08     124A 12
1217 1E     1231 20     124B 00
1218 33     1232 0C
1219 7E     1233 8D
```

D.W. Ekstrand
P.O. Box 1260E
Southgate, Calif. 90280
(213) 566-1677

This man has been making attractive cases for the terminal systemfor some time now. The cases are allodyned aluminum with welded seams. Configurations are offered for several keyboards so write the man for complete information and pricing.

SWTPC 6800 PROTOTYPING BOARDS

If you have been looking for prototyping boards for the SWTPC 6800 Computer System, Personal Computing Company is now selling prototyping cards for both the large and small board positions. The PC boards are single sided plated boards and will accept 14, 16, 24, and 40 pin IC sockets. Both boards are provided with holes for SWTPC 6800 compatible connectors on both the top and bottom edges, however the molex connectors are not supplied. Provisions have been made on each card for an on board 7805 regulator also not supplied. The large board sells for $19.95 while the small goes for $9.95. See their advertisement with photographs on page 112 in the October 1976 issue of 73 magazine.

Personal Computing Company
3321 Towerwood Drive, Suite 107
Dallas, Texas 75234

NEWS FROM MIDWEST SCIENTIFIC

Midwest Scienfific tells us they have two Basic packages, a disassembler, and mini-assembler that are SWTPC 6800 compatible. They are also advertising a PROM board and a floppy disk with software Check with them on prices and delivery. See their add on pages 58 and 59 of the October 1976 issue of Byte magazine.

Midwest Scientific Instruments Inc.
220 West Cedar
Olathe, Kansas 66061

EPROM BOARDS COMING FOR THE SWTPC 6800

We have been notified by two independent firms that they are working on EPROM boards plug compatible with SWTPC 6800 Computer System. Neither manufacturer has a sellable product at this time, however, both should be available by the time of our next newsletter. The PROM boards will use either 2704 or 2708 EPROMS. 1702's are too slow for the SWTPC 6800's memory cycle time.

PR-40 CHARACTER ADDITION

According to David M. Alexander, the underline character (5F$_{16}$) can easily be added to the SWTPC PR-40 Printer's Character font by the addition of two short wires which are tack soldered to the top or bottom of thePR-40 main circuit board. A spare open collector buffer is used to "wire-or" the pin 15 and 16 outputs of the character generator, IC2, into the input circuit of solenoid driver #7. The modification should be made as follows:

1) connect a wire from IC2 pin 15 to IC7 pin 11.

2) connect a wire from IC7 pin 10 to IC7 pin 6.

MORSE CODE PROGRAM FOR THE SWTPC 6800

If you are interested in a morse code program for the SWTPC 6800 Computer System, you might want to check out Wayne Sewell's program which appeared on page 42 of the October 1976 issue of Byte Magazine. Although the program was written to run on any 6800 system, it was written specifically for the SWTPC 6800.

FASTER PROGRAM LOADS FROM CASSETTE

SWTPC will be supplying the longer program cassette and paper tapes such as Basic (C) in binary form. By doing this program loading time will be decreased by a factor from two or three to one, depending upon the length of the program. At the beginning of each tape is an ASCII formatted binary load routine. When you start loading your program tape, the computer actually loads in the binary loader program first. The G necessary to start this program is actually recorded on the tape so the program initiates itself. Following the binary loader program is the actual program to be loaded. The program is stored on the tape in binary so its length is about one third that of the same program stored in ASCII. The memory locations used by the binary loader program have been chosen so as not to conflict with the memory locations of the program to be loaded into memory.

RELATIVE ADDRESS CALCULATOR PROGRAM

By: Russel Yost

This Relative Address Calculator Program may be used to calculate relative addresses for branch instructions. This is especially useful when calculating long branches where you are more likely to make an error if you do it by hand. This program lets the computer do the work for you.

To use the program, type in the machine code listing on the next page. The entire program fits inside the scratchpad RAM used by Mikbug®. Be sure to save the program on tape if you have a tape unit connected to your computer. After loading the program type a G for "Go to User Program". The computer will home the cursor and erase the screen on those systems using the CT-1024 Terminal System with the CT-CA option. It will then print out a BA which stands for "branch address". To this you should respond with the address of the branch instruction and not the address following it. The program will then output a T which stands for "TO". Now you type the destination address of the branch instruction. The program outputs a = followed by the relative address. If branching forward, the outputted address will be 00XX and you must be sure not to have XX greater than 7F. If branching backwards, the outputted address will be FFYY and you must be sure to have YY greater than 7F. Only the last two digits of the outputted address are used for the relative address.

If any non-hex character is input at either address, the program jumps to Mikbug® and outputs a *. Upon entering Mikbug®, typing a G will restart the Relative Address Calculator Program. After calculating each relative address, the program prepares itself for new data. When using the CT-1024 Terminal System, the program will home and erase the terminal's screen after each calculation.

Mikbug® is a registered trademark of Motorola, Inc.

4K AND 8K BASIC (c) EXTERNAL SUBROUTINE CALLS

Basic Version 1.0 and 1.1

```
0260 7E E0BF OUT2H  JMP    $E0BF    OUT2H IN MIKBUG


0263 7E E0C8 OUT4HS JMP    $E0C8    OUT4HS IN MIKBUG


0266 8D 09   OUTCH  BSR    BREAK
0268 7E E1D1        JMP    $E1D1

026B BD E1AC INCH   JSR    $E1AC    INEEE IN MIKBUG
026E 36             PSH A
026F 20 08          BRA    BREAKO

0271 36      BREAK  PSH A
0272 B6 8004        LDA A  $8004
0275 2B 09          BMI    BREAK1
0277 8D F2          BSR    INCH
0279 81 03   BREAKO CMP A  #$03
027B 26 03          BNE    BREAK1
027D 7E 0812        JMP    READY
0280 32      BREAK1 PUL A
0281 39             RTS
```

Basic Version 2.0

```
0272 7E E0BF OUT2H  JMP    $E0BF    OUT2H IN MIKBUG


0275 7E E0C8 OUT4HS JMP    $E0C8    OUT4HS IN MIKBUG


0278 8D 09   OUTCH  BSR    BREAK
027A 7E E1D1        JMP    $E1D1    OUTEEE IN MIKBUG

027D BD E1AC INCH   JSR    $E1AC    INEEE IN MIKBUG
0280 36             PSH A
0281 20 08          BRA    BREAK0

0283 36      BREAK  PSH A
0284 B6 8004        LDA A  $8004
0287 2B 09          BMI    BREAK1
0289 8D F2          BSR    INCH
028B 81 03   BREAK0 CMP A  #$03
028D 26 03          BNE    BREAK1
028F 7E 0815        JMP    READY
0292 32      BREAK1 PUL A
0293 39             RTS
```

BINARY LOAD AND PUNCH

To easily decrease the amount of time it takes to load a long tape (Cassette or paper) a BINARY formatting technique can be used instead of the conventional ASCII format used by the punch and load routines in MIKBUG. The two following programs,BILOAD and BIPNCH are two such programs necessary for punching your own binary data tapes. The punch routine is designed to automatically punch a program that is in several sections or a program and its program counter. Load your program and the BIPNCH into your system and set up the following locations in the MIKBUG RAM:

AØ14	- number of sections to be dumped (Ø2 if you have a one piece program and program counter)
AØ15	- Starting address of program dump (1st block)
AØ16	
AØ17	- End address of program dump (1st block)
AØ18	
AØ19	- Starting address of second block (or pgm. ctr.)
AØ1A	
AØ1B	- End address of second block
AØ1C	
etc.	

(Above similar to setting up AØØ2-etc. for MIKBUG punch)

AØ28 AØ29	- The hex value of the program counter for your program. The data in these locations is transferred to AØ48 and AØ49 automatically and punched in binary.

Executing the BIPNCH program at 1EØ4 will punch your program onto either cassetteor paper tape. Note - Be sure to have your READ switch on a cassette tape loader such as the AC-30 in the off position during a binary punch.

Since you are dumping in binary rather than ASCII, do not expect to see the usual S11 format as during a MIKBUG punch. All you will see will be random characters.

Using the BILOAD program is quite straightforward. Simply use it the same way you use the "L" command in MIKBUG except that you are executing a loader program at 17Ø3 instead of typing an L. The loader will give you a register dump when loading is complete.

If, you desire, the loader program can be put on the beginning of each binary tape to save you time in loading.
Use the following procedure to make such a tape.

RELATIVE ADDRESS CALCULATOR (May be freely copied. No rights with

```
Hex.   Hex. Instr.  Labels   Op.    Operand          Comments
Addr.   Code                 Nmen.
A014 | 8E A0 47     BEGIN    LDS#   NEWSP            Saves BEGIN in
                                                     A048,49
A017 | CE A0 6F              LDX#   MSETUP           Clears screen & types BA .
A01A | 8D 4F                 BSR    PDATSR              See subroutine below.
A01C | BD E0 47     NEXT     JSR    MIGBUG^R BADDR   Gets 4 hex from term &
                                                     stores in X reg.
A01F | FF A0 02              STX    BRA              Stores brnch addr in BRA
A022 | CE A0 77              LDX#   MT               Outputs " T "
A025 | 8D 44                 BSR    PDATSR
A027 | BD E0 47              JSR    MIKBUG^R BADDR
A02A | FF A0 04              STX    DEST             Stores dest'n addr. in DES
A02D | CE A0 00              LDX#   RAMSTART         Prepare for indexed
A030 | 0C                    CLC                     Addr. mode.
A031 | A6 04                 LDAA  X  DEST_H
A033 | E6 05                 LDAB X   DEST_L
A035 | 20 13                 BRA    CONTN
-----------------------------------------------
A048 | A0 14        PCNTR                            Load  BEGIN in MIKBUG^R
                                                     Prog. Cntr Stack Locn.
A04A | C0 02        CONTN    SUBB# 02                Subtract 0002 from
A04C | 82 00                 SBCA# 00                Destination Addr.
A04E | 0C                    CLC
A04F | E0 03                 SUBB X  BRA_L           Subtract Br Addr from
A051 | A2 02                 SBCA X  BRA_H           (Destn. - 2)
A053 | A7 69                 STAA X  REL_H           Store result at REL
A055 | E7 6A                 STAB X  REL_L
A057 | CE A0 7B              LDX#     M=             Outputs " = "
A05A | 8D 0F                 BSR      PDATSR
A05C | CE A0 69              LDX#     REL
A05F | BD E0 C8              JSR      MIKBUG^R OUT4HS    Outputs 4 hex's + Sp
A062 | CE A0 71              LDX#     MBA            Outputs cr, lf, "BA "
A065 | 8D 04                 BSR      PDATSR
A067 | 20 B3                 BRA      NEXT
-----------------------------------------------
A069 | XX XX        REL      EQU
-----------------------------------------------
A06B | BD E0 7E     PDATSR   JSR      MIKBUG^R PDATA1    Outputs string term'
A06E | 39                    RTS                         by 04_16
-----------------------------------------------
A06F | 10 16        MSETUP
A071 | 0D 0A 42 41  20 04  MBA
A077 | 20 54 20 04   MT
A07B | 20 3D 20 04   M=
174C B7 1700          STA A  CKSM
174F 32               PUL A
1750 39               RTS
1751 86 11     LØAD   LDA A  #$11
1753 BD E1D1          JSR    ØUTEEE
1756 86 3C            LDA A  #$3C
1758 B7 80 07         STA A $8007
175B 39               RTS
175C 37        INCHP  PSH B
175D BD E1A5          JSR    SAV
1760 A6 00     IN1    LDA A  0,X
1762 2B FC            BMI    IN1
1764 6F 02            CLR    2,X
1766 BD E1F3          JSR    DE
1769 BD E1EF          JSR    DEL
176C C6 04            LDA B  #4
176E E7 02            STA B  2,X
1770 58               ASL B
1771 BD E1EF IN3      JSR    DEL
1774 0D               SEC
1775 69 00            RØL    0,X
1777 46               RØR A
1778 5A               DEC B
1779 26 F6            BNE    IN3
177B BD E1EF          JSR    DEL
177E 7E E1E3          JMP    IØUT2
                      END


      BIPNCH

1DF7 A7 00     STRT   STA A  0,X
1DF9 B6 A029          LDA A  PCT+1
1DFC A7 01            STA A  1,X
1DFE 20 0F            BRA    BIPNCH
1E00 01        CNTR   FCB    0.1
1E01 0001      TW     RMB    1
1E03 0002      TEMP   RMB    2
1E04 8E A047          LDS    #$A047
1E07 CE A048          LDX    #$A048
1E0A B6 A028          LDA A  PCT
1E0D 20 E8            BRA    STRT
1E0F BD 1E93 BIPNCH JSR    PNØN
1E12 BD 1E99          JSR    PNLDR
1E15 FE A015  BEG     LDX BEGA
1E18 FF 1E01          STX    TW
1E1B F6 A018 PUND10 LDA B  ENDA+1
1E1E F0 1E02          SUB B  TW+1
1E21 B6 A017 BEG1     LDA A  ENDA
1E24 B2 1E01          SBC A  TW
1E27 27 02            BEQ    PUND25
1E29 C6 FF            LDA B  #$FF
1E2B 86 58     PUND25 LDA A  #'X
1E2D BD E1D1          JSR    ØUTEEE
1E30 86 31            LDA A  #'1
1E32 BD E1D1          JSR    ØUTEEE
1E35 37               PSH B
1E36 5F               CLR B
1E37 30               TSX
1E38 8D 69            BSR    PUN
```

```
1E3A 32               PUL A
1E3B 4C               INC A
1E3C B7 1E03          STA A  TEMP
1E3F CE 1E01          LDX    #TW
1E42 8D 5F            BSR    PUN
1E44 8D 5D            BSR    PUN
1E46 FE 1E01          LDX    TW
1E49 8D 58     PUND30 BSR    PUN
1E4B 7A 1E03          DEC    TEMP
1E4E 26 F9            BNE    PUND30
1E50 FF 1E01          STX    TW
1E53 53               CØM B
1E54 37               PSH B
1E55 30               TSX
1E56 8D 4B            BSR    PUN
1E58 33               PUL B
1E59 FE 1E01          LDX    TW
1E5C 09               DEX
1E5D BC A017 CP1      CPX    ENDA
1E60 26 B9            BNE    PUND10
1E62 B6 1E00          LDA A  CNTR
1E65 B1 A014          CMP A  MAX
1E68 27 1C            BEQ    EXIT
1E6A 7C 1E00          INC    CNTR
1E6D B6 1E17          LDA A  BEG+2
1E70 8B 04            ADD A  #4
1E72 B7 1E17          STA A  BEG+2
1E75 B6 1E23          LDA A  BEG1+2
1E78 8B 04            ADD A  #4
1E7A B7 1E23          STA A  BEG1+2
1E7D B7 1E5F          STA A  CP1+2
1E80 4C               INC A
1E81 B7 1E1D          STA A  PUND10+2
1E84 20 89            BRA    BIPNCH
1E86 86 58     EXIT   LDA A  #'X
1E88 BD E1D1          JSR    ØUTEEE
1E8B 86 39            LDA A  #'9
1E8D BD E1D1          JSR    ØUTEEE
1E90 7E E0E3          JMP    CØNTRL
1E93 86 12     PNØN   LDA A  #$12
1E95 BD E1D1          JSR    ØUTEEE
1E98 39               RTS
1E99 C6 05     PNLDR  LDA B  #5
1E9B 4F        PNULL  CLR A
1E9C BD E1D1          JSR ØUTEEE
1E9F 5A               DEC B
1EA0 26 F9            BNE    PNULL
1EA2 39               RTS
1EA3 A6 00     PUN    LDA A  0,X
1EA5 BD E1D1          JSR    ØUTEEE
1EA8 EB 00            ADD B  0,X
1EAA A6 00            LDA A  0,X
1EAC 08               INX
1EAD 81 14            CMP A  #$14
1EAF 27 E2            BEQ    PNØN
1EB1 81 94            CMP A  #$94
1EB3 27 DE            BEQ    PNØN
1EB5 39               RTS
```

1) Load in BILOAD and BIPNCH into memory. Load in the program to be dumped. Set AØ48 and AØ49 to 17Ø3. Set AØØ2 - AØØ5 to 17ØØ 177F. Set up locations AØ14, AØ28, etc. as described earlier.

2) Execute the MIKBUG P command. Set AØØ2 - AØØ5 to AØ48 and AØ49 and execute P.

3) Switch to <u>local</u> and put an S9 G on your tape. Be sure to leave a second or two dead time on both sides of the G.

4) In the remote mode, change AØ48 and AØ49 to 1EØ4. Type G and the program will be punched in binary (Be sure to have locations AØ14, etc. set up correctly as described earlier).

The tape made this way will have the following stored on it:

BINLD FORMATTED IN ASCII	BINLD PGM CTR (ASCII)	S9	G	USER PROGRAM FORMATTED IN BINARY

IMPORTANT NOTE:

Some terminals (such as the SWTPC CT-1Ø24) will treat a 94_{16} the same as a 14_{16} (Punch off). The BIPNCH is set up to correct for this. If your terminal does not see a 94_{16} as a 14_{16} you must change locations IEBI-IEB4 to NOP'S (Ø1).

If the program you with to dump or load occupies the same area of memory as either BIPNCH or BILOAD, you will need to re-assemble them to move them to other areas of memory. Be careful because the programs use the EXTENDED addressing mode in several places.

The BILOAD and BIPNCH are very similar to the ones SWTPC will use when formatting long cassette tapes in binary.

```
     BILØAD

1700 0001     CKSM   RMB     1
1701 0002     TW     RMB     2
1703 8E A047         LDS     #$A047
1706 8D 49    BILØAD BSR     LØAD
1708 8D 3C    ØVER   BSR     INPUT
170A 81 58           CMP A   #'X
170C 26 FA           BNE     ØVER
170E 8D 36           BSR     INPUT
1710 81 31           CMP A   #'1
1712 27 07           BEQ     READ
1714 81 39           CMP A   #'9
1716 26 F0           BNE     ØVER
1718 7E E115         JMP     DMPREG
171B 7F 1700  READ   CLR     CKSM
171E 8D 26           BSR     INPUT
1720 16              TAB
1721 5C              INC B
```

```
1722 8D 22           BSR     INPUT
1724 B7 1701         STA A   TW
1727 8D 1D           BSR     INPUT
1729 B7 1702         STA A   TW+1
172C FE 1701         LDX     TW
172F 8D 15    STØRE  BSR     INPUT
1731 A7 00           STA A 0,X
1733 01              NØP
1734 A1 00           CMP A   0,X
1736 26 0B           BNE     ØUT
1738 08              INX
1739 5A              DEC B
173A 26 F3           BNE     STØRE
173C 8D 08           BSR     INPUT
173E 7C 1700         INC     CKSM
1741 27 C5           BEQ     ØVER
1743 7E E040  ØUT    JMP     LØAD19
1746 8D 14    INPUT  BSR     INCHP
1748 36              PSH A
1749 BB 1700         ADD A   CKSM
```

PAPTAP   HIGH SPEED PAPER TAPE LØADER PRØGRAM
         DEVELØPED BY DR. CHARLES ADAMS
         TEXAS A&M UNIVERSITY

```
1F00 86 2E                LDA A  #$2E
1F02 B7 800B              STA A  PIA2
1F05 B7 800A              STA A  PIA1
1F08 8D 28     ØV         BSR    SUB1
1F0A 81 53                CMP A  #$53
1F0C 26 FA                BNE    ØV
1F0E 8D 22                BSR    SUB1
1F10 81 31                CMP A  #$31
1F12 26 F4                BNE    ØV
1F14 7F 1F43              CLR    CLR1
1F17 8D 39                BSR    SUB2
1F19 80 02                SUB A  #2
1F1B B7 1F42              STA A  TMP3
1F1E 8D 24                BSR    SUB3
1F20 8D 30     BR2        BSR    SUB2
1F22 7A 1F42              DEC    TMP3
1F25 27 05                BEQ    BR1
1F27 A7 00                STA A  0,X
1F29 08                   INX
1F2A 20 F4                BRA    BR2
1F2C 7C 1F43   BR1        INC    CLR1
1F2F 27 D7                BEQ    ØV
1F31 3F                   SWI
1F32 B6 800B   SUB1       LDA A  PIA2
1F35 2A FB                BPL    SUB1
1F37 B6 800A              LDA A  PIA1
1F3A 84 7F                AND A  #$7F
1F3C B7 800A              STA A  PIA1
1F3F 39                   RTS
1F40 00        TMP1       FCB    0
1F41 00        TMP2       FCB    0
1F42 00        TMP3       FCB    0
1F43 00        CLR1       FCB    0
1F44 8D 0C     SUB3       BSR    SUB2
1F46 B7 1F40              STA A  TMP1
1F49 8D 07                BSR    SUB2
1F4B B7 1F41              STA A  TMP2
1F4E FE 1F40              LDX    TMP1
1F51 39                   RTS
1F52 8D 10     SUB2       BSR    SUB4
1F54 48                   ASL A
1F55 48                   ASL A
1F56 48                   ASL A
1F57 48                   ASL A
1F58 16                   TAB
1F59 8D 09                BSR SUB4
1F5B 1B                   ABA
1F5C 16                   TAB
1F5D FB 1F43              ADD B CLR1
1F60 F7 1F43              STA B CLR1
1F63 39                   RTS
1F64 8D CC     SUB4       BSR SUB1
1F66 80 30                SUB A #$30
1F68 81 09                CMP A #$09
1F6A 2F 02                BLE RT
1F6C 80 07                SUB A #7
1F6E 39                   RTS
```

QUICKLØAD

```
0000 CE 1F00              LDX #1F00
0003 86 2E                LDA A #$2E
0005 B7 800B              STA A $800B
0008 B7 800A   HERE       STA A $800A
000B B6 800B   LØØP       LDA A $800B
000E 2A FB                BPL LØØP
0010 B6 800A              LDA A $800A
0013 A7 00                STA A 0,X
0015 08                   INX
0016 7E 0008              JMP HERE
```

HIGH SPEED PAPER TAPE READER SOFTWARE

Recently an ad has appeared in several magazines for an optical high speed paper tape reader by Oliver Audio Engineering, 7330 Laurel Canyon Blvd., North Hollywood, Calif. 91605, (213) 765-8080. Some SWTPC 6800 compatible software has been developed for this unit which we are passing along here.

The following programs assume that the required parallel interface is in the #2 card position in the 6800. The program PAPTAP is the loader that takes the parallel data in from the reader through the PIA and stores it in memory. This program is rather long and it is not convenient to load it in by hand each time. If you currently have some type of save/load device (AC-30 cassette, teletype, etc.) you will have no problems in loading the loader program. If you have no AC-30 or teletype but can steal a few minutes time on someone else's paper tape punch, punch out the program QUICKLOAD using a binary format (no header characters, checksum, address pointers, etc. - just send the data to the punch using OUTEEE). Be sure to use only black paper tape. Do not use the BIPNCH program in this newsletter.

If you are using an AC-30, etc. simply load in the PAPTAP program and set the program counter to IFØØ. Place the paper tape in the reader and type G. Pull the paper tape through the reader and your program will be loaded. If a software interrupt is encountered an error was seen and the program should be reloaded.

If you are using the binary formatted tape,type in the QUICKLOAD program and insert the binary loader tape in the reader. Start execution at ØØØØ and pull the tape through the reader. This loads the PAPTAP program from locations IFØØ-IF6E. The loading of your program can now be accomplished as described earlier.

Instructions come with the tape reader concerning assembly and use. If you have an questions concerning the mechanics, price, availability, etc. of the loader, please contact Oliver Audio, not SWTPC.

Note: You may have to re-write the loader programs to move them to a convienent area of memory for your computer. Also, the reader has a jumper that must be installed on it - Jumper A to ACK, not $\overline{\text{ACK}}$.