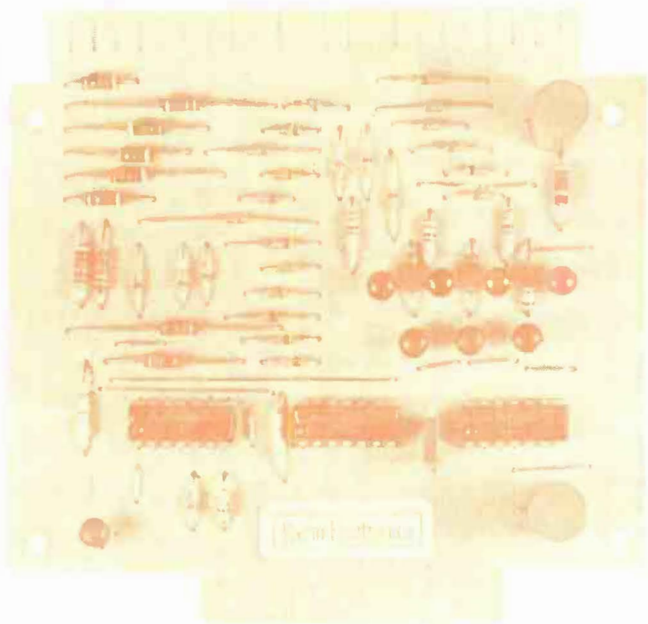


build an ASCII keyboard encoder

Here is what you need to couple the keyboard you built in February to a computer, teletype, or teaching machine.

by DON LANCASTER

THE LOW-COST KEYBOARD, LIKE THE majority of other typewriter style keyboards, provides only a single "make" contact for each key depressed (see **Radio-Electronics**, February 1973). Computer terminals, teaching machines, etc., cannot directly use a single-contact operation, and a device called an *encoder* must be placed between the keyboard and the computer. The encoder converts the single contact closure into a seven- or eight-bit IC logic compatible parallel code, usually following the ASCII encoding scheme, and allowing for *shift* and *control* key operations. After parallel encoding, there may follow a *parity* generator for error detection, and a 100-word-per-minute *parallel* to *serial* converter that allows the signals to be sent down a single wire or phone line.

The keyboard encoder described here costs only a tiny fraction of commercial equivalents. It uses three "dollar" integrated circuits and a small handful of surplus computer diodes. While this encoder was designed as a companion to the low-cost keyboard, it may be used with *any* keyboard, provided the make contacts are less than 2000 ohms when ON and provided that the keys *do not* have a common ground terminal. The encoder generates all the codes shown in Table I. This includes all the capital letters, all the often used punctuation, all numerals, and all of the transparent or control functions. Often used control functions such as DELETE, SPACE, LINE FEED, ESCAPE (ALT MODE), CARRIAGE RETURN, etc. are brought out to separate keys. The output is RTL, TTL, DTL and MOS compatible, and a single 10-volt, 25-mA power source is needed. If an ASCII code is not desirable, the same encoder may be used, through suitable rewiring, to generate EBDIC, SELECTRIC, BAUDOT or MORSE codes. Parity and the 100-wpm (words-per-minute) serial converter are easily added to the basic encoder.

What is the ASCII code?

Many years ago, the American Standards Association decided to adopt a standard code that computers could use to talk to each other, to their input/output devices, and to allow standardized connections between different brands of computer machinery. The resultant industry wide code is called ASCII, short for *A*merican *S*tandard *C*ode for *I*nformation *I*nterchange. This code is a sequence of six, seven, or eight *bits* (ones or zeros). It may be sent either in *serial* (bit by bit, least significant bit first) form, or in *parallel* (all bits at once, on 6, 7, or 8 lines) form. Usually, parallel words are used *inside* machines, while serial words are used *between* machines. Serial words are obviously slower, but they take far less wire and interconnections.

The basic code consists of *seven* bits. If we look at all possible combinations of seven ones and zeros from 000-0000, 000-0001 . . . through to . . . 111-1111, we'd find a total of 128 different sequences. Each of these may be used to represent something distinct. 64 of these code sequences are used for alphanumeric capital letters, numbers, a blank, and punctuation. 32 more sequences are used for *transparent* or *control* commands that never appear on a screen or in print. These commands tell the machinery on the other end what to do—things like returning carriages, clearing, line feeds, bell ringing, and other control functions. A final 32-code sequences are reserved for lower case alphabet and some little used punctuation. This last group is very seldom used as most computer communications can be handled with only capital letters, numerals, control commands, and common punctuation.

The complete code appears in Table II. It is arranged in a *matrix* form to make it compact and easy to read. For instance, the transparent command "Carriage Return", or "CR" has a code of 000-1101, starting with bit 7 on the left and going to bit 1 (the least significant) on the right. A numeric "6" has the code 011-0110. Note the right half of this code is the same as a binary or a binary-coded-decimal six. A capital L has the code 100-1100, while the lower case L is a 110-1100.

There are several ways to use the code, depending on how much you want the code to do. If we are *only* interested in upper-case alphabet, numerics, and punctuation, we can use the middle of the code and get by with a six-bit code, sometimes called ASCII-6. This is useful in character generators and displays that do not need transparent commands or lower case alphabets. Many MOS integrated circuits are now available that convert the six-bit subset into a recognizable bunch of dots on a TV screen or a line printer; these are called ASCII Character Generators.

Or, we can use all seven bits, either with or without the lower case stuff, picking up both alphanumerics and control commands. This is often called the ASCII-7 code. Finally, if we

TABLE I OUTPUT CODES

The output codes below are shown in HEXADECIMAL notation to conserve space. Thus "3D" is an ASCII 011-1101, or output $a_1 = 1$, $a_2 = 0$, $a_3 = 1$, etc. . . . The "Key Depressed" output does *NOT* appear for the SHIFT or CONTROL buttons depressed separately. All other keys, whether or not they are used with SHIFT or CONTROL, produce a *Key Depressed* output.

KEY	NORMAL CODE	SHIFTED CODE	CONTROL CODE
@	40	40	00 (null)
A	41	41	01 (soh)
B	42	42	02 (stx)
C	43	43	03 (etx)
D	44	44	04 (eot)
E	45	45	05 (eng)
F	46	46	06 (ack)
G	47	47	07 (bell)
H	48	48	08 (bs)
I	49	49	09 (ht)
J	4A	4A	0A (Lf)
K	4B	4B	0B (vt)
L	4C	4C	0C (FF)
M	4D	4D	0D (cr)
N	4E	4E	0E (so)
O	4F	4F	0F (si)
P	50	50	10 (dle)
Q	51	51	11 (DC1)
R	52	52	12 (DC2)
S	53	53	13 (DC3)
T	54	54	14 (DC4)
U	55	55	15 (NAK)
V	56	56	16 (SYN)
W	57	57	17 (ETB)
X	58	58	18 (CAN)
Y	59	59	19 (EM)
Z	5A	5A	1A (SUB)
0	30	20 (space)	10 (dle)
1	31	21 (!)	11 (DC1)
2	32	22 (")	12 (DC2)
3	33	23 (#)	13 (DC3)
4	34	24 ($)	14 (DC4)
5	35	25 (%)	15 (NAK)
6	36	26 (&)	16 (SYN)
7	37	27 (')	17 (ETB)
8	38	28 (()	18 (CAN)
9	39	29 ())	19 (EM)
:	3A	2A (*)	1A (SUB)
;	3B	2B (+)	1B (ESC)
<	2C	3C (,)	0C (ff)
-	2D	3D (-)	0D (cr)
>	2E	3E (.)	0E (so)
?	2F	3F (/)	0F (si)
↑	5E	5E	1E (rs)
—	5F	5F	1F (us)
SPACE	20	20	00 (null)
LINEFEED	0A	0A	0A (Lf)
C. RETURN	0D	0D	0D (cr)
ESCAPE (ALT)	1B	1B	1B (esc)
DELETE	7F	7F	1F (si)

like, we can add an eighth bit and use it for error detecting. It is called the *parity* bit. In an *even parity* system, the parity bit makes the total number of *ones* in the word *even*. If there are 1, 3, 5, or 7 ones in the word before the parity bit is added, the parity bit is a one. If there are 0, 2, 4, or 6 ones, the parity bit is a zero. This way, there is always an *even* number of ones sent. At the receiving end, parity is once again tested. If an odd number of ones shows up, a mistake has been made, and the receiver can substitute a "?" or ask for the information over again. We could also use an odd parity system just as well, provided both ends are playing the game with the same rules. This is called the ASCII-8 code.

Seven bits are rarely sent between machines. An eighth wire or bit space is usually added, so that if parity is added later, it doesn't take a bunch of rework. Similarly, most paper tape punches and magnetic tape is in an eight-track code; if parity isn't used, the eighth bit is usually all ones.

You may wonder what all those transparent commands stand for. Actually, a lot of them are only used on very big and very complicated machines, and thus aren't really very common. The ones you'll probably use are few in number. For instance, **CR** is a carriage return that starts a new line on a typewriter. **LF** is the line feed, used to skip a line. **BEL** rings a bell or signals an operator. **BS** is backspace. It can only be used in some systems, for many CRT terminals and teletypes cannot back up. The direct control commands are labeled *DC1*, *DC2*, *DC3*, and *DC4*. These are usually yours to do anything you want with, such as turning on and off equipment, remote signalling, etc. **NUL** is a do-nothing command that everything sits in while not

ASCII ENCODER PARTS LIST

R1—390 ohms, ¼ watt (sets operating force—see text)
R2, R3, R4—100,000 ohms, ¼ watt
R5, R7, R10, R16—4700 ohms, ¼ watt
R6, R8, R9, R11, R12, R13, R14—9100 ohms, ¼ watt 5%
R15—470 ohms, ¼ watt
R17, R18, R19—100 ohms, ¼ watt
R20—2200 ohms, ¼ watt
R21, R22, R23—10,000 ohms, ¼ watt
C1—0.1-μF disc ceramic capacitor
C2, C3—100-μF 10V electrolytic
D1 thru D26—1N914 or similar silicon computer diode
D27—1N4736 or similar 6.8-V Zener diode
IC1, IC2, IC3—MC 9818P mwrtl hex inverter
Q1, Q5, Q6, Q7—2N5139 pnp transistor
Q2, Q3, Q4, Q8—2N5129 npn transistor
MISC: PC board, jumpers, sleeving, solder, hardware

in use. Finally, **ESC** is called Escape or Alternate mode. This lets you break out of the ASCII code if you ever need a longer sequence or something else special. The technique is called *code extension*. One common use is on a timesharing terminal, where you can switch back and forth between BASIC, FORTRAN, and EXECUTIVE MODE languages using the **ESC** or exactly equivalent **ALT** key.

DEL is a 111-1111 code that's used to delete a previous command or fill a paper tape full of holes.

Construction steps

The schematic and parts list is shown in Fig. 1. The keys are arranged electrically to form an "8 x 8" array, and pnp transistors translate the positive end of the array back down to logic levels. This technique requires far fewer diodes than a direct encoding does. The *control* and *shift* keys suitably alter the code of only those keys they're supposed to change.

A printed circuit board is used for

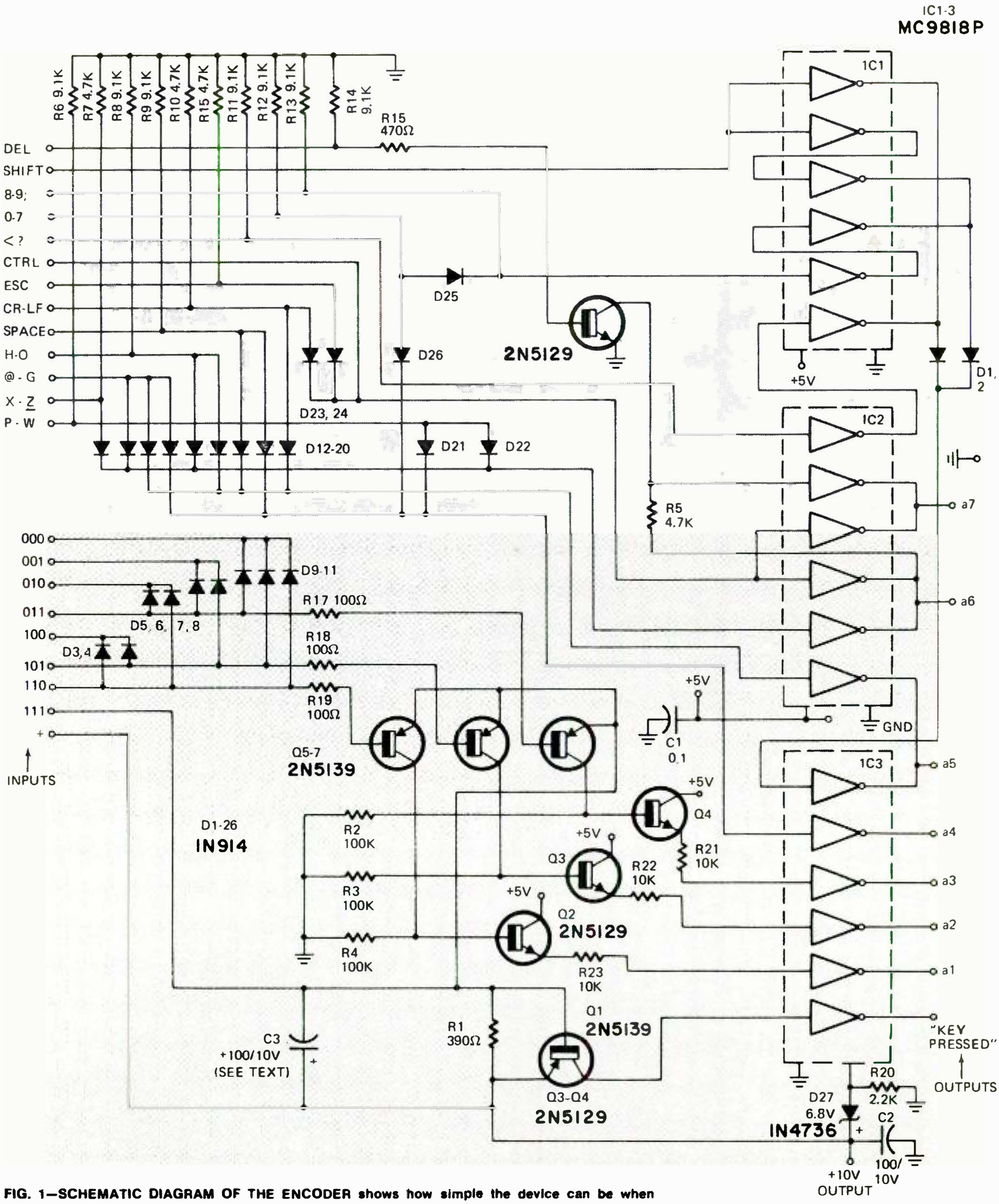


FIG. 1—SCHEMATIC DIAGRAM OF THE ENCODER shows how simple the device can be when three inexpensive IC's are used.

assembly. It's available commercially (see Fig. 1 parts list) or you can make your own using the pattern in Fig. 2 and following the drilling and assembly guides of Figs. 3 and 4. Use a small iron and fine solder for assembly, and be careful to observe the code notch and dot on the IC's and the polarity bands on the diodes and capacitors.

Resistor R1 determines the operating force required on the keyboard. It is chosen to be low enough in value that each key's output code is set up and correct at ⅓ to ¼ the pressure required to get the "key pressed" output command. This way the code is set up and stable before it is sent. Capacitor C3 further delays the "go" command to insure reliable operation. Be sure you use only the *leading edge* of the "key pressed" command, for it lasts longer than the rest of the code does. Should a second key be pressed before the first one is released, it will not be sent, giving a form of "2 key rollover" protection.

The PC board mounts on short spacers directly below the keyboard, and connects to the keyboard with a double connector, a flat cable, or direct jumpers. An Amphenol 143-012-01 connector may be used as an output. A 12-volt supply may be used by going to an 8-volt Zener diode. Operation at 5 or 6 volts may be obtained by lowering all the resistors, but the required keyforce will be

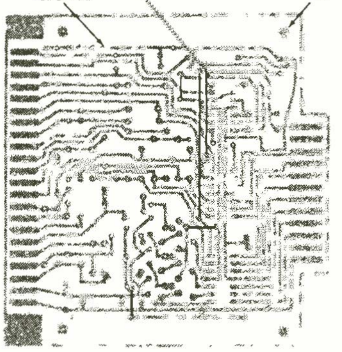


FIG. 3 (left)—DRILL GUIDE for the PC board. Solid-wire jumpers are on the component side.

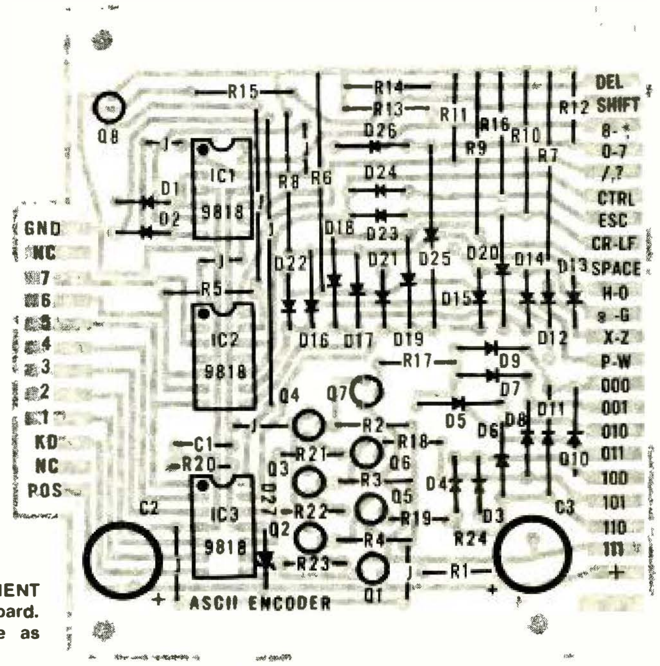


FIG. 4 (right)—COMPONENT SIDE of the encoder board. Jumpers are shown here as well as in Fig. 3 at left.

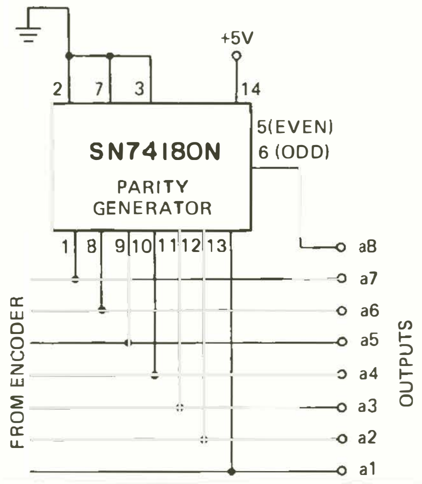


FIG. 5—PARITY GENERATOR for the eighth bit (a8) needs only one IC for odd or even parity.

TABLE II COMPLETE ASCII CODE

To read the matrix, choose your character. Then read the three high order bits off the top and the four low order bits off the left side. For instance, a small alphabet "f" is a binary 110-0110, otherwise known as a hexadecimal 66. Each control function has a specific meaning. For instance "LF" stands for a line feed, "CR" is a carriage return, "BEL" is a bell to attract an operator's attention, "ESC" is an escape for complicated control instructions, etc.

b_7					0	0	0	0	1	1	1	1
b_6					0	0	1	1	0	0	1	1
b_5					0	1	0	1	0	1	0	1
				Column								
BITS b_4	b_3	b_2	b_1	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	a	LF	SUB	*	:	J	Z	j	z
1	0	1	1	b	VT	ESC	+	;	K	[	k	}
1	1	0	0	c	FF	FS	,	<	L	\	l	|
1	1	0	1	d	CR	GS	-	=	M	]	m	}
1	1	1	0	e	SO	RS	.	>	N	↑	n	-
1	1	1	1	f	SI	US	/	?	O	—	o	DEL

100-WPM ADAPTOR PARTS LIST

R1—1000 ohms, ¼ watt
R2—2200 ohms, ¼ watt
R3—10 ohms, ¼ watt
R4—10,000 ohms, ¼ watt
R5—potentiometer, 1000 ohms, linear
C1—0.1-μF 10V disc ceramic
C2, C3—100-μF 6V electrolytic
C4—4-μF 10V tantalum, 10%
D1—1N4002 silicon power diode or equal
IC1—MC4024 TTL multivibrator
IC2—SN74165 TTL 8-bit shift register, PISO
IC3—SN7474 TTL dual-type-D flip-flop
Q1—2N1613 npn silicon transistor
MISC: PC board, jumpers, sleeving, mounting adaptor hardware.

greater and less uniform from key to key. TTL (*T*ransistor *T*ransistor *L*ogic) fanout is 1 standard load. RTL (*R*esistor *T*ransistor *L*ogic) fanout is one medium-power gate.

The unit is tested by noting the proper codes in Table I. It's particularly important to watch all the bits at once with lamp drivers, IC testers, or something similar during initial checkout to be sure the code is up and stable before the keypressed command is sent for each and every key.

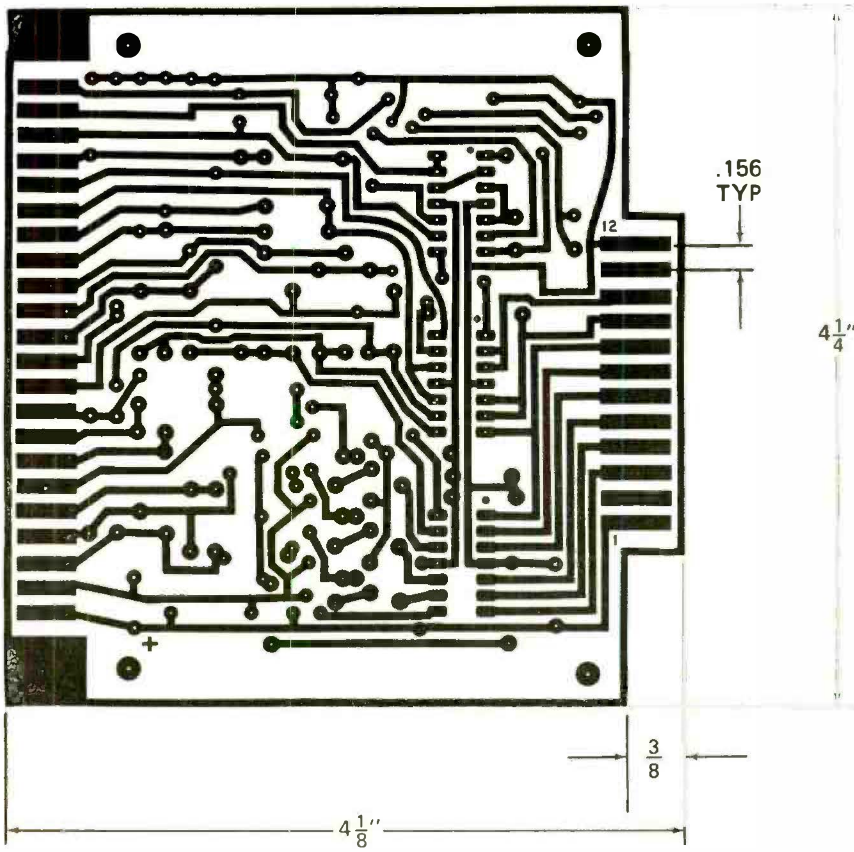


FIG. 2—FOIL PATTERN FOR THE ENCODER PC BOARD. This board mounts just below the keyboard and connects to it through flat cable, PC connectors or direct wired jumpers.

An optional parity generator for the eighth bit is shown in Fig. 5 and may be used for even or odd parity. A 100 word per minute teletype adaptor is shown in Fig. 6. The 100-wpm adapter consists of an oscillator whose period must be exactly 9.09 ms. Upon the Key Pressed command, an ASCII code, a START bit and a SYNCHRONIZING bit are loaded into a parallel-load shift register. After loading is completed, the oscillator marches out the code bits in proper sequence to be teletype and computer compatible. The circuit may accept a second character anytime after the 110-ms transmission time. The output consists of a transistor that normally shorts the teletype line. It *breaks* the line anytime a "1" is to be transmitted. The proper polarity must be observed on the output, and the 20 or 30 mA loop current source is located elsewhere. **R-E**

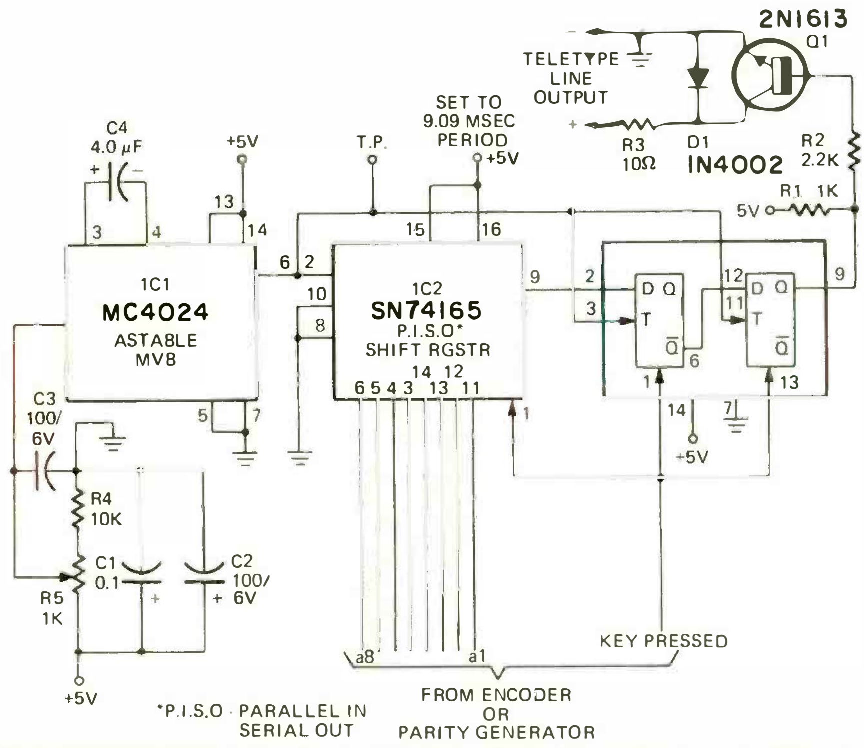


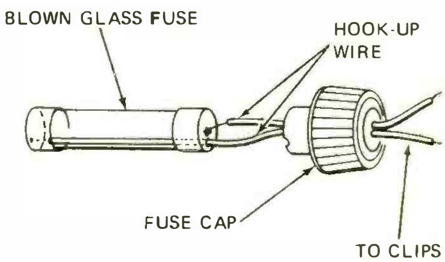
FIG. 6—TELETYPE FEED is through a special adapter circuit. This one handles up to 100 words per minute. A precision oscillator controls the storage and output from the shift register.

CIRCUIT BREAKER SUBSTITUTION BOX

A substitution box with circuit breakers selected by a switch is one of the handiest gadgets on my service bench. With breakers of different ratings, I'm ready to check radios, amplifiers and TV's with blown fuses and questionable circuit breakers.

Generally, I can clip onto the fuse or circuit breaker if the chassis has been pulled. I've rigged up a handy adapter that lets me jump fuses without pulling the chassis when the fuse holder is a post type on a panel or chassis skirt. The drawing shows its construction.

Drill a ⅛ inch hole through the



center of a spare fuse-holder cap. Drill small holes slightly off center in the ends of a blown cartridge fuse. Drill a second hole, just large enough to pass a piece of thin insulated hook-up wire, in the center of one end of the fuse. Strip about ⅛ inch of insulation off one end of a piece of hook-up wire and pass it through the center of the fuse so the short exposed wire goes through the hole in the far end. Solder. Solder a second piece of hook-up wire to the other end and then thread both leads through the fuse-post cap and then add clips for connecting to the breaker substitution box.—*Arthur M. Padmore* **R-E**