Photo 1: The complete setup of the IBM Selectric Keyboard Printer, typing under the control of a KIM-1 microcomputer with a 4 K memory expansion. The Selectric interface described in this article is housed in the equipment case in the center of this photo.

Interfacing the IBM

Dan Fylstra
Hamilton Hall C-23
Harvard Business School
Boston MA 02163

Photography by Carole Brock

One of the most desirable forms of computer output is high quality typewritten text suitable for preparing letters, reports and other documentation. A word processing system which speeds up the process of writing and revising text would be a very useful and feasible application for a small microprocessor based system, provided that a suitable hard copy output device can be found at a reasonable price.

An ideal output medium for such a word processing system would be an IBM Selectric office typewriter. Selectrics are moderately expensive when compared to ordinary typewriters ($630 to $830 depending on the options chosen), but they are ubiquitous in the office environment, produce very high quality typed output, and can be used to print in many different type styles simply by changing the ball shaped typing element. Special typeballs are available for printing mathematical symbols and for the APL character set (see "What is APL?", by Mark Arnold, November 1976 BYTE, page 20).

Unfortunately, the job of converting a Selectric office typewriter is made somewhat more difficult by the fact that (contrary to popular belief) the Selectric mechanism is entirely mechanical and not electronic in nature. The only use of electric power in an ordinary Selectric is for the motor which turns the drive shaft and various gears and cams. It is necessary to use solenoids to push levers and "bails" in the base of the mechanism to achieve printing under computer control. Similarly, contact switches must be installed in order to use the keyboard for computer input.

There is another alternative, however. A variety of computer terminals and other devices based on the Selectric mechanism are becoming available on the surplus market, often at a fraction of their original prices. These machines have their own built-in solenoids or other means for mechanical control, and present some sort of electrical or electronic interface to the outside world. The simplest, most commonly available, and of-

Selectric Keyboard Printer

(Teaching KIM to Type)

ten the cheapest of these are the Selectric Input/Output Keyboard Printers, Models 73, 731, 735 and others. They were manufactured by IBM, typically for use as IO devices in other companies' computer systems. As these systems have become obsolete, the Selectric Keyboard Printers have found their way into surplus channels.

As a business school student and experienced user of computers, I have always wanted to build a word processing system around my own home computer. Hence I seized a chance to acquire a Model 73 Keyboard Printer for $450 from the Computer Warehouse Store in Boston. (These units were sold out in a few weeks; I have heard of prices ranging from $250 to $1500 through other channels, but as interest in the units increases, their typical prices are bound to rise.) Armed with a couple of old IBM manuals provided by the Computer Warehouse Store, I set out to accomplish what I expected would be a straightforward interfacing process.

This article is a report of my experience, and a detailed description of the interface which I built. Briefly, the interfacing process, while simple in principle, was not at all straightforward in practice. But it was successful, even for such a mechanically inept person and relative novice in electronics as me. For about $50 in parts (including such extravagances as a pretty cabinet and a $20 IBM connector to plug into the Selectric's peculiar 50 pin receptacle), and lots of labor, I produced the unit shown in photo 1. It's only an interface to the Selectric printer, since I'm content to use my existing ASCII keyboard for input. It has its limitations, but it works.

This, of course, is hardly the last word on Selectric Keyboard Printer conversion. As a BYTE reader, I would be delighted to see information on more comprehensive interface designs, as well as actual experiences with several of the units currently on the market. Since most of them are sold on an "as is" basis, these machines can bring

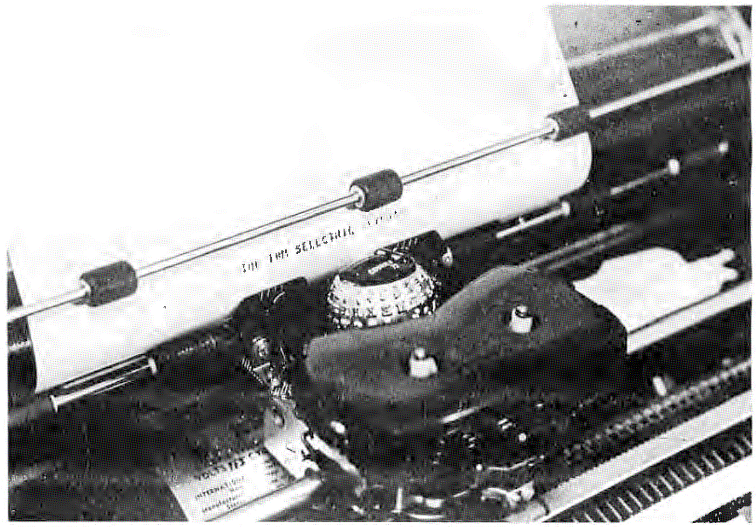


Photo 2: A closeup of the IBM Selectric ball mechanism on its moving carriage within the printer. The Courier 72 ball shown here is one of many balls available with the "Correspondence" coding arrangement.

you a lot of grief (read on). So it's wise to report on problems, and how you overcame them.

The Selectric Mechanism

To appreciate what the interface must do (and what can go wrong), it is first necessary to understand basically how the Selectric mechanism works. The typing element is a golfball sized hollow sphere embossed with up to 88 characters, arranged in four horizontal rings of 22 characters each. Photo 2 illustrates the ball in its rest position in the mechanism. All the lower case or unshifted characters appear on the "western hemisphere," the side normally closest to the paper. The upper case characters are in corresponding positions on the back side or "eastern hemisphere." Pressing the shift key causes the whole typeball to rotate 180°, thereby allowing the upper case characters to be printed. Hence, the actual typing operation can select any of 44 characters, four half rings of 11 characters each, with five to the left and five to the right of the center or "home" position on each ring. A particular character is selected by causing the typeball to tilt up or down and rotate right or left; then the ball jumps forward to strike the ribbon and paper. These movements account for the peculiar "dancing" motion seen when the Selectric is typing continuously. The typeball is mounted on a carriage which moves across the page, as opposed to traditional pre-IBM typewriters where the paper carriage moves and the typing mechanism remains stationary.

The actual tilting and rotation of the typeball is accomplished by an incredibly complicated system of latches, pulleys and levers which are driven by six moving "bails," or rods in the base of the machine. Although we need not understand the detailed mechanical linkages, we should appreciate the roles played by these six

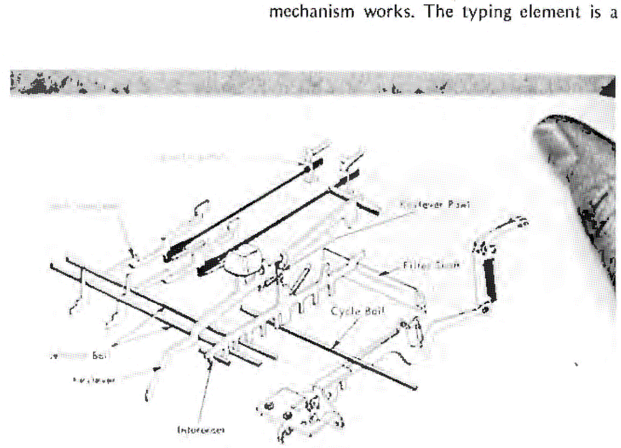


Photo 3: This diagram, from the IBM manual number 241-5159-3, shows how the various bails of the mechanism are connected in a typical case. The IBM manuals for the typewriter output unit are valuable reference materials and can be obtained by calling your local IBM office.

moving bails. Two of the bails, referred to as T1 and T2, are moved or not moved in one of the four possible combinations to provide the proper degree of tilt necessary to select one of the four rings. Three more bails, called R1, R2 and R2A, are moved or not moved in various combinations to provide 1, 2, 3, 4 or 5 increments of counterclockwise rotation, normally to select one of the five characters to the right of center on the given ring (as seen from above). Finally, when the bail named R5 is moved, the typeball rotates 90° clockwise so that the counterclockwise movement provided by R1, R2 and R2A can select one of the five characters to the left of center on the ring. (When none of the rotate bails is involved the center position on each ring is selected.)

To print a particular character, then, we need to know its position on the typeball (which can vary from ball to ball), as well as what combination of bail movements – T1, T2, R1, R2, R2A and R5 – will take us to that position. Figure 1 presents the "coordinates" of each character in terms of the six bail movements for the two most common character arrangements, the ones used on the "BCD" and "Correspondence" encoded typeballs.

The Keyboard and Print Magnets

In an ordinary Selectric typewriter, the keys are mechanically linked to the various bails, as shown in photo 3. Striking a key depresses an "interposer" bar with a particular combination of fingers which arrest the motion of some of the bails. The interposer also moves a "cycle bail" which releases the drive shaft and allows it to turn 180°. On the drive shaft are a number of cams which control the series of movements necessary to print a character, as selected by the tilt and rotate bails. At the end of the cycle everything is back to normal, waiting for another key to be struck.

In a Selectric Keyboard Printer, the tilt and rotate bails are also mechanically linked to six electromagnets. The magnets pull down armatures which otherwise would arrest the motion of the bails. To print a character, some combination of the six magnets must be energized, the particular tilt and rotate "code" for that character as found in figure 1. In addition, something

TYPEHEAD LAYOUT

T – #s = Tilt latches active
R – #s = Rotate latches active

	Upper Case											Lower Case											Tilt	
	–5	–4	–3	–2	–1	0	+1	+2	+3	+4	+5	–5	–4	–3	–2	–1	0	+1	+2	+3	+4	+5	T2	T2
BCD TYPEBALL CODES	■	■	■	■	@	>	■	■	■	:	√	1	3	5	7	8	0	2	4	6	9	'	0	0
	■	■	■	‡	%	\	■	■	■	⌒	#	/	T	V	X	Y	b̸	S	U	W	Z	,	1	0
	■	■	■	|	✱	;	■	■	■	]	△	J	L	N	P	Q	—	K	M	O	R	S	0	1
	■	■	■	?	¤	>	■	■	■	[	‡	A	C	E	G	H	&	B	D	F	I	.	1	1
ROTATES R5	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0		
R2A	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	1	1		
R2	0	0	1	1	1	0	0	1	1	1	1	0	0	1	1	1	0	0	1	1	1	1		
R1	0	1	0	1	0	0	1	0	1	0	1	0	1	0	1	0	0	1	0	1	0	1		
CORRESPONDENCE TYPEBALL CODES	[	'	&	*	$	Z	@	%	¢	)	(	]	3	7	8	4	z	2	5	6	0	9	0	0
	X	U	D	C	L	T	N	E	K	H	B	x	u	d	c	l	t	n	e	k	h	b	1	0
	M	V	R	A	O	°	.	"	I	S	W	m	v	r	a	o	j	.	'	i	s	w	0	1
	G	F	:	,	?	J	+	P	Q	Y	_	g	f	;	,	/	j	=	p	q	y	—	1	1

HOME (upper case 0 column) HOME (lower case 0 column)

COLOR KEY

(orange) = Home Position of Ball
(yellow) = Tilt Rotate Command Bits

Figure 1: The graphic fonts of the BCD and Correspondence typewriter codes. The location of each character on the Selectric ball is described by a unique combination of case, tilt and rotate commands. The upper case versus lower case choice is made by a mechanical latch set up before printing, so the chart is broken down into two main sections for each code. The home position of the typeball is flagged in each case by a color shading. The binary command information for each matrix position is given by the rows in the center labeled "ROTATE" and the columns at the right labeled "TILT." Thus, to form the Correspondence code for the letter S, the tilt command bits are 01, and the rotate command bits are 0110 for a tilt rotate command code of 010110 to be used in the format described in figure 7.

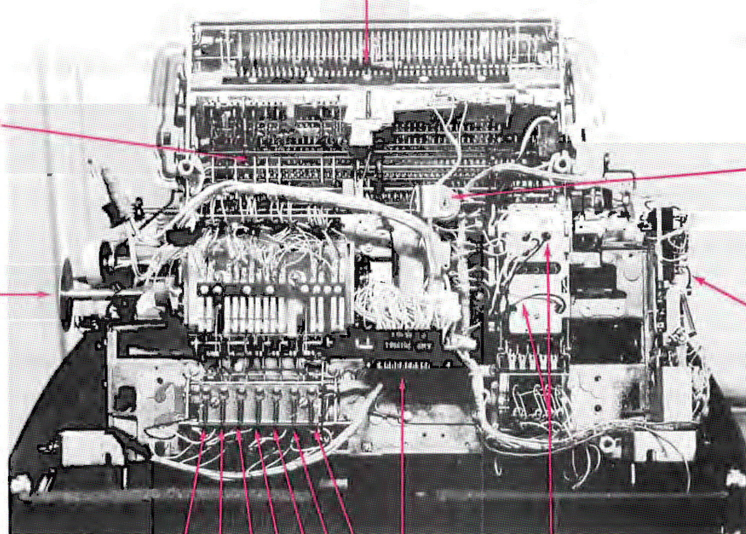


Photo 4: A detail of the underside of the Selectric Keyboard Printer with housings removed. The overlay shows several of the key points such as the location of various magnets, the switch contacts and interconnection receptacle.

must actuate the "cycle bail" to start the printing process. Hence a trip mechanism is provided which moves the cycle bail whenever any of the armatures is pulled down. However, there is one character on each hemisphere which should be printed when none of the magnets is energized, for the code 000000. Hence the trip mechanism is connected to a seventh magnet, called "check," which provides an odd parity function for the other six magnets. It is energized whenever necessary to ensure that the total number of magnets energized is odd. Thus the check magnet is energized on the code 000000, and this serves to actuate the cycle bail. (I didn't realize this when building my interface, so I can't print those two characters yet. Don't make the same mistake!)

Besides the print magnets, there are a number of other magnets and armatures inside the Keyboard Printer which control special functions such as space, backspace, tab, carriage return, index (ie: advance paper without returning), ribbon shift, and upper and lower case shift. Many of these magnets can be seen in photo 4, which exposes the underside of the machine and outlines the positions of many components. The upper and lower case shift magnets are latching, and hence they lock the machine into the new case until the opposite magnet is energized. Note that the operator cannot shift the machine back into lower case when the upper case magnet is latched! By Murphy's Law this is bound to happen whenever you are testing the interface, but it can be remedied by fooling around with the shift cam at the end of the drive shaft.

No electric power is provided for any of these magnets inside the Keyboard Printer, but the coil connections are brought out to the 50 pin receptacle at the back of the machine. The magnets are rated for 43 to 53 VDC at 125 to 300 mA, applied for at least 10 ms in order to pull down the armatures and cause the desired action.

Switch Contacts

The other major addition to the basic

Continued on page 133

Continued from page 52

Selectric mechanism found in the Keyboard Printer is a set of switch contacts which are closed by movement of the tilt rotate bails, and by movement of the cams in various stages of the printing cycle. These contacts can also be seen in photo 4. Again, no electric power is applied to these contacts inside the Selectric, but six of them, called C1 to C6, are wired together thru certain pins in the receptable at the back of the machine (more on this later). For printed output, these contacts can be tested to determine when the printing cycle is complete. For keyboard input, there is another set of contacts which must be tested at the proper instant in order to capture the code for the key just depressed. Other contacts are provided which make it possible to determine whether the machine is currently locked in upper or lower case, whether the end of line margin stop has been reached, and so on. According to the documentation, the contacts are rated for 40 mA at 10 V (minimum) to 300 mA at 48 V (maximum).

BCD and Correspondence Machines

At this point, I should clear up the mystery surrounding the differences between the so-called "BCD" and "Correspondence" versions of the Selectric Keyboard Printer. There are differences in three areas:

1. The arrangement of characters on the typeball that is used.
2. The arrangement of the fingers on the interposers connected to particular keys.
3. The code obtained for keyboard input at the 50 pin receptacle when a key is pressed.

The Correspondence version is the simpler of the two. All of the office typewriters are built this way, and nearly all the typeballs available from IBM use the Correspondence arrangement of characters. In a Correspondence encoded Keyboard Printer, the tilt and rotate bail contacts are wired directly to the 50 pin receptacle, and so the code obtained when a key is pressed is the actual tilt rotate code. Note that the tilt rotate code is the same for, say, an upper case A and a lower case a, so the current state of the shift contacts must be checked whenever a character is read.

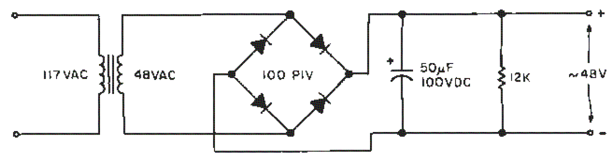
Many Selectric Keyboard Printers were built for use in equipment which employed a 6 bit byte and the old BCD (binary-coded decimal) character code, and so IBM developed the "BCD" version of the Selectric. In this machine, the tilt and rotate contacts (there are several sets of contacts for each bail) are wired through a maze of diodes and shift contact connections to yield a unique 6 bit code for all of the essential characters in the BCD set. Hence the code which reaches the 50 pin receptacle can be read directly into a 6 bit byte, and the shift contacts themselves need not be tested. Of course, a 6 bit byte can represent only 64 different characters, and after allowing for the digits and various special characters, there was room for only the upper case alphabetics. In fact, because of the limitations of wiring through diodes and switch contacts, only 48 distinct codes are actually produced. Even so, in order to accomplish this wiring feat, it was necessary to move some of the essential characters to convenient spots on the typeball, and hence the interposers with certain finger combinations also had to be moved around in order to preserve the usual layout of the keyboard. This is why the characters are all mixed up when you type manually on a BCD machine with a Correspondence typeball. Indeed, just to make everything fit together, IBM puts only the upper case characters on most of the typeballs intended for use with the BCD machine. (An exception is the Model 963 typeball which is used in many timesharing terminals.) But, in fact, the mechanism is still capable of tilting and rotating to any character position.

What does all this mean for the computer hobbyist? If you are using the Selectric as a printer only, it makes no difference whether you have a BCD or a Correspondence machine, since in either case you have direct access to the tilt and rotate magnets. By energizing the proper combinations of the seven magnets, you can use both BCD and Correspondence typeballs with either machine. (My Selectric is a BCD machine and I regularly use it with a Correspondence encoded Courier 72 typeball.)

If you want to use the Selectric keyboard for computer input (and you want upper and lower case), or if you want to use the machine off line with a variety of Correspondence encoded typeballs, you are considerably better off with the Correspondence version of the Keyboard Printer. But, since most of the units available through surplus channels (at least at reasonable prices) are BCD machines, you may have to settle for one of these. With some mechanical and electronic skill (and lots of courage), you could convert a BCD machine into a Correspondence version by:

1. rearranging the interposers to match the Correspondence typeball arrangement.

Figure 2: A very simple power source for the unregulated DC used to power the solenoids of the Selectric Keyboard Printer.



2. tearing out all the wiring for BCD code generation and replacing it with direct connections from the bail switch contacts to the 50 pin receptacle.

So much for the theory of operation of the Selectric mechanism. Now let's get on to the design of an interface unit which will let us control the Selectric printer using standard TTL level signals from a computer output port. Mindful always of our potential exposure to Murphy's Law, we will keep this interface as simpleminded as possible. Readers with more sophistication in electronics may use this approach as a jumping-off point (so to speak) for their own designs.

Interface Design

To control the operation of the Selectric printer we must provide three types of functions:

1. Signal conversion of TTL levels to magnet currents.
2. Code conversion of ASCII codes to tilt rotate code.
3. Control and timing to type successive characters, wait for carriage return, etc.

It seemed to me that the most appropriate division of labor was to provide the first function in hardware, and the second one in software. Signal conversion requires an external power source, while code conversion requires some flexibility to accommodate different typeballs. For the third function, I have experimented with both open loop control (realized entirely in software) and closed loop control (which uses a hardware feedback signal); both approaches will be discussed briefly here.

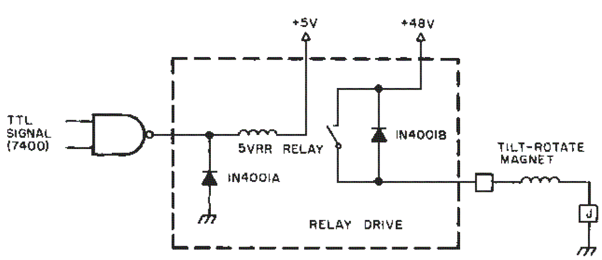


Figure 3: Switching of the solenoid actuator magnets in the Selectric Keyboard Printer is accomplished by this basic circuit. A reed relay which is within the drive capabilities of TTL is driven from a TTL logic gate, with protection against back EMF provided by the diode A. The reed relay, in turn, drives the magnet in the printer from the 48 V (nominal) supply of figure 2. Diode B provides back EMF protection for the relay contacts to prevent arcing which would shorten the life of the relay. The dotted line outlines the detailed circuit repeated many times in figure 4.

Signal Conversion

For signal conversion, we simply need a power source for the Selectric magnets and a means of switching the power on and off using TTL level signals. For the power source, we need a maximum of about 1 A of DC (for seven simultaneously energized magnets at 125 mA per magnet) in the range of 43 to 53 V. The source need not be regulated nor even filtered. (See "Watts Inside a Power Supply," by Gary Liming, January 1977 BYTE, page 42, for a further discussion.) Figure 2 is a circuit diagram for the power supply which I built around a $4 surplus transformer. The only really essential element is the full wave rectifier. The capacitor was included simply to jack up the voltage of the particular transformer I was using to the point where it would energize the magnets.

To switch power on and off, I used a set of reed relays (optoisolators or power transistors could be used instead). These particular reed relays have a coil resistance of 290 ohms, so they can be driven by an ordinary TTL gate (17 mA at 4.8 V, or 10 TTL loads). They are available from Digi-Key Corporation, POB 677, Thief River Falls MN 56701, for $1.70 each (part number 5VRR). I used a total of 12 relays, six for the print magnets (since I forgot about the "check" magnet) and six for the most important control functions (space, backspace, tab, carriage return, and upper and lower case shift).

The reed relays were each connected to a computer output port and a Selectric magnet through the circuit diagram shown in figure 3. Here the 1N4001 diodes protect the TTL gate and the reed switch from voltage transients in the two coils. Since I needed a standard TTL buffer to provide enough current for each reed relay, and since I wanted to economize on my use of output ports, I used a seventh control line to switch between the six print magnets and the six control function magnets. The resulting circuit diagram is shown in figure 4. The lettered squares which terminate the reed switch contact lines refer to pin designations on the Selectric's 50 pin receptacle (see below). Photo 5 shows the physical layout of the components of figure 4 in the interface which I built. Most of the wiring is Vector Slit n' Wrapped on the other side of the square piece of Vectorboard.

This construction layout is not recommended! Allow yourself much more room for repairing, replacing or adding components (like a seventh pair of reed relays!). A length of scrapped telephone cable makes a good connection between the interface and the Selectric itself. Also shown in photo 5 is a 50 pin connector which plugs into the

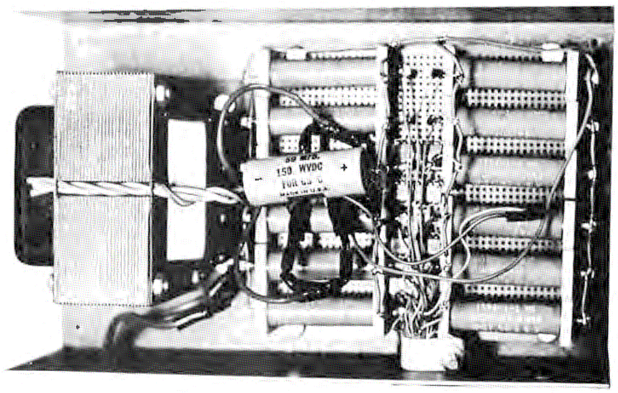


Photo 5: Physical layout of the components of the interface box which houses the circuit described in this article.

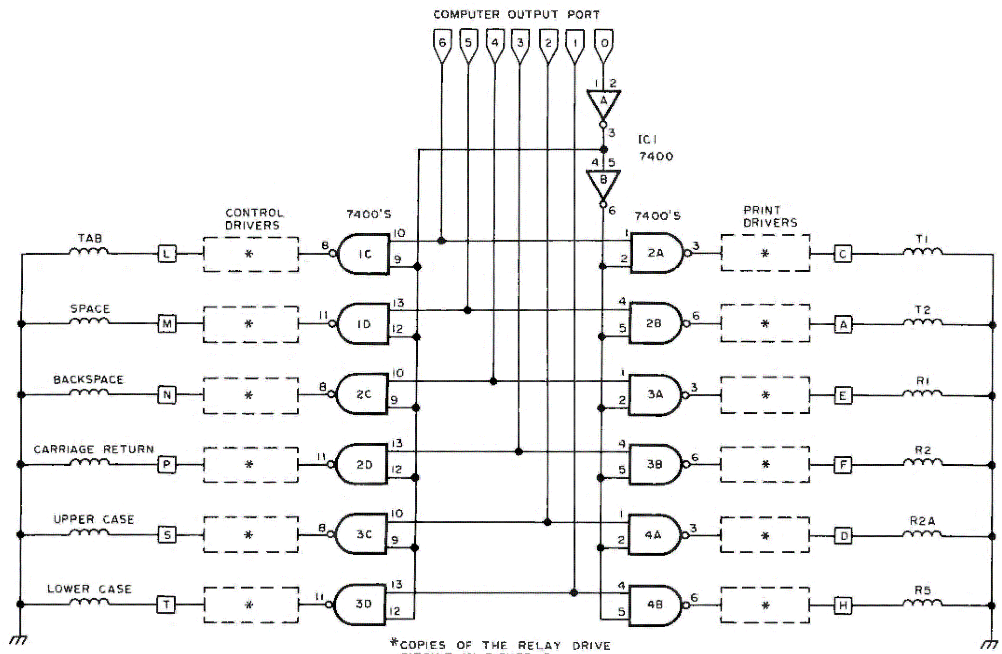


Figure 4: The complete interface schematic. The 7400 NAND gate logic is used to select either the drivers for the miscellaneous control functions, or the drivers for the print commands. The basic drive circuit of figure 3 is repeated once for each magnet in the printer.

Pin		Function
A	←	T2
B	←	Check
C	←	T1
D	←	R2A
E	←	R1
F	←	R2
H	←	R5
J	←	Magnet Common
K	←	Keyboard Lock
L	←	Tab
M	←	Space
N	←	Backspace
P	←	Carriage Return
R	←	Index
S	←	Upper Case Shift
T	←	Lower Case Shift
U	←	Red Ribbon Shift
V	←	Black Ribbon Shift
W	→	C1 N/C
X	→	Contact Common
a	→	Feedback N/C
b	→	Feedback N/O
e	→	End of Line N/C
f	→	End of Line N/O
n	→	C1 N/O
r,s,t,u,v,w	→	BCD Bit Lines

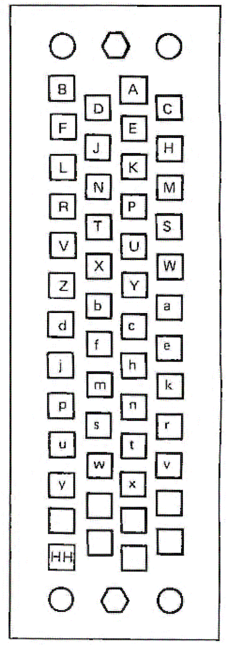


Figure 5: The Selectric Keyboard Printer receptacle pin identifications. This receptacle can be purchased as a spare part through an IBM office. The arrows in this table indicate direction of the signal: A left arrow indicates drive to the printer (typically a magnet) from a source in the interface; a right arrow indicates a sensor contact in the printer.

receptacle at the back of the Selectric, which I obtained from my local IBM branch office for $20 (IBM part number 1167134). The more important pin designations on this connector are shown in figure 5.

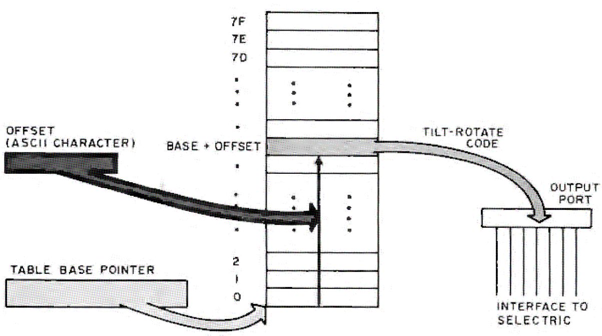


Figure 6: Table structure for the conversion of ASCII to Selectric coding. The table base pointer identifies the start of the table. There should be one table for each different ball coding scheme employed. The ASCII character value is added to the base address giving an address in the table. At this address is found the code which is sent to the output port. The logic of sending the code to the output port is given in detail by figure 8.

Code Conversion

Assuming that the ASCII code is used for characters inside the computer, the process of code conversion is basically just a simple table lookup: The 7 bit ASCII code is used as an index into a 128 byte table to obtain the 6 bit tilt rotate code. Since the tilt rotate code for a given character may vary depending on the typeball that is used, it should be possible to switch between several 128 byte tables. This is easily done by indexing from a pointer to the base of the table as shown in figure 6.

The main complication in code conversion is the handling of upper and lower case. At any given time the Selectric Keyboard Printer is locked into one case or the other. If the machine is locked in upper case and the next character to be printed is an upper case A, we need only send out the appropriate tilt rotate code. But if the next character is a lower case a, we must energize the lower case shift magnet, wait for the machine to shift into lower case, and then send out the tilt rotate code. This is easily accomplished by using a seventh bit in the table entry byte for each ASCII character to indicate whether it is to be printed in upper or in lower case.

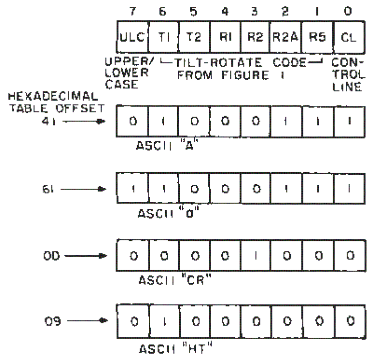


Figure 7: The coding scheme for each conversion table entry is given by the general box at the top of this diagram. Bit 7 tells the software whether the mechanism should be in the upper or lower case mode. (The need to shift explicitly in a Selectric is reminiscent of the shift requirements of Baudot Teletypes.) The tilt rotate code contained in bits 6 thru 2 is derived from figure 1 for each character in the table. (For other ball arrangements, a version of figure 1 would need to be generated.) The low order bit of the word is used to indicate to the logic of figure 4 whether a control command (0) or print command (1) is being sent.

The last problem in code conversion is the handling of control functions such as carriage return, tab, backspace, etc. Fortunately, the ASCII character set assigns unique 7 bit codes for functions such as these. For example, the ASCII carriage return character (hexadecimal code 0D) can be used for carriage return, and the ASCII horizontal tab (hexadecimal code 09) can be used for the tab function. Since in my interface a special control line determines whether the six output ports affect the print magnets or the control function magnets, I can use the eighth bit in each table entry byte to set the control line appropriately. The table entries for the printable characters have this bit set to 1, with six bits providing the tilt rotate code; the entries for the control characters have this bit set to 0, with the bit corresponding to the given control function magnet set to 1 and the other five bits set to 0. This encoding is illustrated in figure 7.

Once we have this encoding of the information needed for code conversion, the actual program logic to accomplish the conversion is straightforward. A flowchart of the logic is presented in figure 8, and an

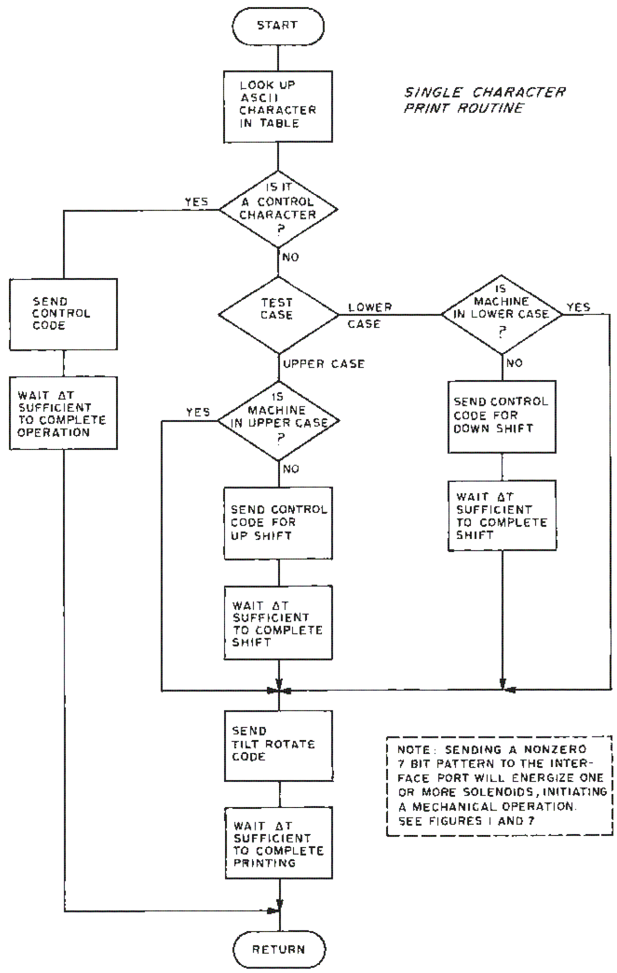


Figure 8: A flowchart giving the logic of a simple open loop driver program which takes a given ASCII character, looks up its table entry, and then takes appropriate printer actions. As an open loop program, each time delay in this chart (the ΔTs) is picked to reflect the worst case response time for the action involved. This makes the Selectric type successfully, but does not optimize operation for the maximum speed, since as everyone knows, the worst case is often not identical with the typical value of a parameter.

CHARACTER OUTPUT ROUTINE FOR SELECTRIC KEYBOARD PRINTER

```
OUTCH   TAY                 ASCII character to index register
        LDA   (TABPT), Y    get code byte from table
        LSR   A             test low order bit
        BCC   CTL           0 means control character
        ROL   A             test high order bit
        BMI   LOWER         1 means lower case character
        LDX   #4            code for upper case shift
        LDY   CASE          check current case
        BEQ   OK            0 means upper case
        INC   CASE          indicate shift to upper case
        JMP   SHIFT         go initiate shift operation
LOWER   LDX   #2            code for lower case shift
        LDY   CASE          check current case
        BNE   OK            -1 means lower case
        DEC   CASE          indicate shift to lower case
SHIFT   STX   PORT          send shift code to port
        JSR   ENERG         for 10 milliseconds
        LDY   #60           delay for 60 milliseconds
        JSR   WAIT          until shift operation is done
OK      STA   PORT          send tilt rotate to port
        JSR   ENERG         for 10 milliseconds
        LDY   #50           delay for 50 milliseconds
        JSR   WAIT          until print operation is done
        RTS                 return to calling program
CTL     ROL   A             restore control code
        STA   PORT          send to output port
        JSR   ENERG         for 10 milliseconds
        LDY   #120          delay for 120 milliseconds
        JSR   WAIT          until control operation is done
        RTS                 return to calling program
ENERG   LDY   #10           set up for 10 millisecond delay
        JSR   WAIT          loop for that long
        LDY   #0            send 0s to output port
        STY   PORT          to turn off magnet current
        RTS                 return to caller
WAIT    LDX   #200          number times thru inner loop
LOOP    DEX                 decrement inner loop count
        BNE   LOOP          loop until count is 0
        DEY                 decrement outer loop count
        BNE   WAIT          loop until count is 0
        RTS                 return to caller
```

Listing 1: 6502 assembly language source code of a program which implements the logic of the flowchart in figure 8. This program is a subroutine which will drive the Selectric Keyboard Interface in an open loop mode and is run on a KIM-1 system.

equivalent assembly language program for the MOS Technology 6502 used in my system is shown in listing 1. In this simple version of the program, delay loops are used for timing purposes, and sufficient time is allowed either to print a character or to complete the worst case control function (carriage return across the entire length of the page). Of course, this version of the program will operate the Selectric at far less than its maximum rated speed, and will monopolize the processor's time while waiting for completion of each operation. In order to improve on this, we turn next to the subject of control and timing.

Control and Timing

Now that we have a working Selectric interface, we can turn our attention to two major improvements: driving the Selectric at maximum rated speed, and minimizing use of the processor's time for Selectric control.

To drive the Selectric at full speed we can adopt an approach of "open loop" control or "closed loop" control. Open loop control involves keeping track of the carriage position, margin, tab stops and similar information in software (changing the margin and tab stop information via software interpreted commands), and calculating the delay time necessary for each operation. Closed loop control involves testing the Keyboard Printer's switch contacts to determine when each operation has been completed. The worst case delay approach used in the program of listing 1 is a simplified version of open loop control. For full speed operation, the closed loop approach is much simpler and more reliable; so let's consider it here.

Nearly every mechanical operation opens or closes some set of switch contacts inside the Selectric. Sets of contacts are wired to the 50 pin receptacle in a variety of ways to reflect operations such as printing, tabbing, backspacing, etc. We will not consider all the possible methods of achieving feedback control using these contacts, but will outline one particularly simple approach, which remains to be tested in my own system. The pin labeled a on the receptacle is wired through a set of normally closed contacts, and the pin b through corresponding normally open contacts, associated with the set of common contacts connected to pin X. Figure 9 shows how these contacts may be debounced to yield a clean TTL level signal (ignoring the nominal voltage ratings for the contacts). Here we use the last half of the 7400 package left over from figure 4. During any printing or control function operation, pin a will go from ground to +5 V and back to ground again, while pin b does the reverse. Hence the feedback line will go from logic 1 to 0 to 1. By sensing this change in software through a loop testing the feedback input port after energizing the magnets, we can closely control the operation. When the line goes to logic 0, we can turn off current to the magnets, and when it returns to logic 1, we are ready to start the next operation.

The second problem we face in control and timing is how to minimize use of the processor's time for Selectric control. Here, of course, is where the interrupt system comes into play. If we are using the circuit outlined in figure 9 for closed loop control, we can tie the feedback line to a processor interrupt rather than to a data input port. If we are relying instead on open loop control, we can use a programmable interval timer which is capable of causing an interrupt as an alternative to delay loops. The software to handle interrupts from the Selectric is slightly complicated by the need to shift between upper and lower case prior to typing the next character, but this can be handled by initiating the shift operation and then arranging to retry the character printing operation on the next interrupt, at which time the Selectric will be locked into the proper case.

Actual Experience

Hopefully this article has given the reader all the information he or she needs to build a Selectric Keyboard Printer interface similar to, or better than mine. Lest you are unduly emboldened by the foregoing discussion, however, consider what can go wrong.

I carefully tested the interface in stages, by using an ohmmeter to verify that bit patterns sent to my computer output port closed the proper combinations of reed switches, and by testing the power supply on some of the Selectric's magnet coil connections. Nevertheless, when I first tested the entire setup, I thought I saw a blue flash around one of the reed relays when I tried to pulse the R2 magnet. Nothing seemed to happen when I tried again, except that the R2 magnet wasn't being energized. Then, listening carefully, I heard a telltale simmering sound that sent me leaping for the electric outlet. The R2 reed relay had stuck closed, and on further examination I found that most of the arc suppressing diodes inside the Selectric had been destroyed. After painstakingly replacing the R2 reed relay and installing the diodes visible in photo 5, I tried again. This time I found out why the reed relay, like its replacement, was sticking closed! The R2 magnet in the unit I purchased had been burned out and was a short circuit. No wonder the unit was a surplus item.

Not willing to give up, I managed to remove the coil from the R2 magnet core, and replace it with the coil from the unused (by me!) check magnet. After this feat, I found that when I typed manually on the keyboard, only @s, 0s, and a few other characters could be printed! Only after hours of reading and experimentation did I discover that the adjustment of the plate holding the magnet armatures in place (which I had removed to change the coils) was critical, and could be set only by considerable trial and error.

These are the kinds of things that can go wrong. You cannot be too careful in playing with these machines! Readers certainly should investigate the possibility of an IBM maintenance contract on at least the mechanical portion of the Keyboard Printer, which need not be too expensive.

And, to conclude, although I probably never would have undertaken this project had I known at the outset what it would ultimately entail, it certainly is satisfying to have that Selectric typing away under the control of my home computer. To anyone else who is ready to undertake such a project, I hope that this article has helped, and I wish you the best of luck.■

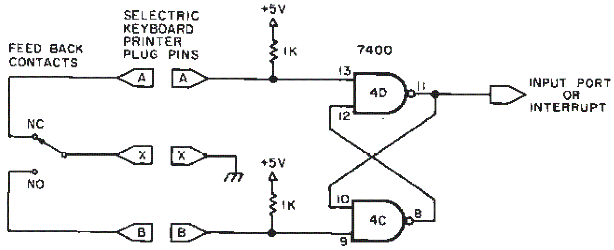


Figure 9: A circuit for debouncing the feedback information generated by contacts in the printer which are mechanically linked to the action. Using the feedback pulse to drive an input port or interrupt line can result in operation at the maximum possible speed since the timing is now on an "each case" basis rather than "worst case."

BIBLIOGRAPHY

"IBM Selectric Input-Output Writer: An Exciting Advance in the Field of Input-Output Media," Form # 543-0033-1. This manual is absolutely essential since it gives circuit diagrams, timing charts, and end views of the magnets and switch contacts.

"IBM Selectric I/O Keyboard Printer: Customer Engineering Manual of Instruction," Form #241-5159-3. This or a similar manual is very valuable for understanding the mechanical functioning of the Keyboard Printer.

A 6800 Selectric IO Printer Program

Listing 1: The listing of the Selectric printer interface routine for a 6800 system driving the IO version of the standard office typewriter. This listing is extracted from two assemblies done using the Southwest Technical Products Corporation's version of the M6800 self assembler. The first part of the listing is the actual code, and the second part is a table of Selectric correspondence codes which is referenced using ASCII codes as an index into the table which is computed at CONV1.

```
SWTPC M-6800 ASSEMBLER
ENTER PASS : 1P,1S,2P,2L,2T

                          NAM    SELECTRIC
                   *SELECTRIC DRIVER PROGRAM FOR SWTPC 6800 ASSEMBLER
                          OPT    S
                          OPT    O
                          OPT    L
0100                      ORG    $0100
0100 7E 17CD              JMP    START
0212                      ORG    $0212
0212 BD 17F4              JSR    START1   CALL OUTPUT(NEW)
08F7                      ORG    $08F7
08F7 18FF                 FDB    $18FF    MAKE ROOM FOR PATCH
093D                      ORG    $093D
093D 1900                 FDB    $1900
13D6                      ORG    $13D6
13D6 1900                 FDB    $1900
17CD                      ORG    $17CD
17CD C6 FF   START        LDA B  #$FF     INITIALIZE PIA
17CF F7 8000              STA B  PIAOUT
17D2 7F 8002              CLR    PIACHK
17D5 C6 04                LDA B  #$04
17D7 F7 8001              STA B  PIAOUT+1
17DA F7 8003              STA B  PIACHK+1
17DD C6 81                LDA B  #$81     START ALWAYS IN LOWER CASE
17DF F7 8000              STA B  PIAOUT
17E2 F7 18CA              STA B  CASE
17E5 FE 181A              LDX    COUNT1   SETUP TIMER FOR SHIFT CYCLE
17E8 FF 18CB              STX    COUNTR
17EB BD 18C3              JSR    TIMER
17EE 7F 8000              CLR    PIAOUT
17F1 7E 0300              JMP    $300     GOTO MAIN PRGM
17F4 84 7F   START1       AND A  #$7F     RESET PARITY
17F6 FF 18CD              STX    SAVEX    SAVE XREG FOR MAIN PRGM
17F9 81 10                CMP A  #$10     TRAP HOME-UP
17FB 26 02                BNE    CR
17FD 20 04                BRA    CR1      PRINT IT AS CR,LF
17FF 81 0D   CR           CMP A  #$0D     TRAP CR
1801 26 0B                BNE    SP
1803 86 84   CR1          LDA A  #$84
1805 FE 181C              LDX    COUNT2   SETUP TIMER FOR CR,LF
1808 FF 18CB              STX    COUNTR
180B 7E 18BA              JMP    EX1
180E 81 20   SP           CMP A  #$20     TRAP SPACE
1810 26 05                BNE    CONV0
1812 86 88                LDA A  #$88
1814 7E 18B4              JMP    EX2      GO PRINT BUT DO NOT RESET MSB
1817 7E 1880 CONV0        JMP    CONVRT
181A 2000    COUNT1       FDB    $2000
181C 1000    COUNT2       FDB    $1000
181E 0400    COUNT3       FDB    $0400
1880                      ORG    $1880
1880 81 20   CONVRT       CMP A  #$20     IS IT A PRINTING CHARACTER?
1882 22 02                BHI    CONV1    YES
1884 20 39                BRA    EXIT     NO
1886 B7 18D0 CONV1        STA A  TABLEP+1 CONVERT CODE
```

The following letter and listing 1 were received from an Italian reader of BYTE, Fulvio Guzzon of Rome. Fulvio purchased the same print mechanism (IBM Model 735 IO typewriter) which is described by Dan Fylstra in his article in this issue. We're treating Fulvio's letter as a short article, since its technical content is far above that of the usual letter. The listings photographically reproduced here were typed on pin feed paper using his printer mechanism. The text of his letter was submitted using a text editor with the Selectric IO mechanism as its output.

I understand there is some interest among your readers in using a Selectric typewriter for hard copy. As you can see I have funneled an editor program (SWTPC) and an assembler program (SWTPC, too) through a Selectric typewriter. *[The original of this note was typed on the Selectric.]* I bought the machine on the surplus market in Boston and it had some problems: It was stuck in upper case by a bolt screwed on the right side of the frame, it had some unrecoverable backlash in the head rotate mechanism, and many feedback and interlock contacts were missing or badly damaged. I had the machine serviced here in Rome (Italy) and at last, with a new carriage, a new motor (here we have 220 V 50 Hz power), and a new set of shift magnets, the printer was ready. I decided to use it only as a printer in order to reduce the hardware and software effort to a minimum.

On the underside of the machine there are seven printing magnets. In table 1 I have paired them with the bits from 0 to 6. Seven transistors provide for the interface between the PIA and the printer.

There are seven more magnets for the machine commands: space, backspace, tab, carriage return, index (line feed), upper case, and lower case: so seven more transistors are required. Seven output lines from the PIA in slot 0 are switched between the two sets of magnets by digital logic. The various feedback and interlock contacts were wired in series and filtered for bounce by a condenser and a software loop. The conversion table shown in the assembly listing provides for the characters used on the so called "Correspondence" balls. As I later found out, there are minor variations between the balls of this series.

The MSB in the table is set when the character to be printed is on the upper case half of the ball. (The upper or lower case of ASCII code bears no relation to the upper or lower side of Selectric golf balls). The MSB of the output byte to the printer

Listing 1, continued:

```
1889 FE 18CF        LDX    TABLEP
188C A6 00          LDA A  0,X
188E 26 02          BNE    CASECK   IS IT AVAILABLE SOMEWHERE ON
                   * THE BALL?
1890 20 2D          BRA    EXIT     NO,RETURN
1892 2A 04   CASECK BPL    CASELW   MSB CLEAR?
1894 C6 C0          LDA B  #$C0     NO,CHECK IF PRINTER IS IN UC
1896 20 02          BRA    SKIP
1898 C6 81   CASELW LDA B  #$81     YES,CHECK IF PRINTER IS IN LC
189A F1 18CA SKIP   CMP B  CASE     NEW CHAR. SAME HALFBALL
                   *AS THE PREVIOUS ONE?
189D 27 13          BEQ    PRINT1   YES GO AND PRINT IT
189F F7 8000        STA B  PIAOUT   NO, ROTATE BALL 180 DEGREES
18A2 F7 18CA        STA B  CASE     AND RECORD IT
18A5 FE 181A        LDX    COUNT1   SETUP TIMER FOR SHIFT CYCLE
18A8 FF 18CB        STX    COUNTR
18AB 8D 16          BSR    TIMER
18AD 7F 8000        CLR    PIAOUT
18B0 8D 11          BSR    TIMER
18B2 84 7F   PRINT1 AND A  #$7F     RESET CASE BIT AND
18B4 FE 181E EX2    LDX    COUNT3   SETUP TIMER FOR PRINT CYCLE
18B7 FF 18CB        STX    COUNTR
18BA B7 8000 EX1    STA A  PIAOUT   NOW PRINT
18BD 8D 12          BSR    WAIT1
18BF FE 18CD EXIT   LDX    SAVEX    RESTORE X REG
18C2 39             RTS             GO AND FETCH NEXT CHARACTER
18C3 FE 18CB TIMER  LDX    COUNTR
18C6 09      LOOP   DEX
18C7 26 FD          BNE    LOOP
18C9 39             RTS
     8000    PIAOUT EQU    $8000
     8002    PIACHK EQU    $8002
18CA 00      CASE   FCB
18CB 0000    COUNTR FDB
18CD 0000    SAVEX  FDB
18CF 1800    TABLEP FDB    $1800
18D1 8D F0   WAIT1  BSR    TIMER
18D3 C6 01          LDA B  #1       SETUP MASK
18D5 F5 8002        BIT B  PIACHK   PRINT CYCLE STARTED?
18D8 27 F7          BEQ    WAIT1    NO
18DA 7F 8000        CLR    PIAOUT   YES ON IT'S WAY
18DD 8D E4   WAIT2  BSR    TIMER
18DF F5 8002        BIT B  PIACHK   READY FOR A NEW ONE?
18E2 26 F9          BNE    WAIT2    NO
18E4 39             RTS             YES!
18E8                ORG    $18E8    SHIFT START OF SYMBOL TABLE
                   *TO NEXT PAGE
18E8 C1             FCB    $C1
18E9 20             FCC    5,
18EA 20
18EB 20
18EC 20
18ED 20
18EE FFFF           FDB    $FFFF
18F0 C2             FCB    $C2
18F1 20             FCC    5,
18F2 20
18F3 20
18F4 20
18F5 20
18F6 FFFF           FDB    $FFFF
```

Listing 1, continued:

```
00108 18F8 D8          FCB   $D8
00109 18F9 20          FCC   5,
      18FA 20
      18FB 20
      18FC 20
      18FD 20
00110 18FE FFFF        FDB   $FFFF
00111                  END

START  17CD
START1 17F4
CR     17FF
CR1    1803
SP     180E
CONVO  1817
COUNT1 181A
COUNT2 181C
COUNT3 181E
CONVRT 1880
CONV1  1886
CASECK 1892
CASELW 1898
SKIP   189A
PRINT1 18B2
EX2    18B4
EX1    18BA
EXIT   18BF
TIMER  18C3
LOOP   18C6
PIAOUT 8000
PIACHK 8002
CASE   18CA
COUNTR 18CB
SAVEX  18CD
TABLEP 18CF
WAIT1  18D1
WAIT2  18DD

TOTAL ERRORS 00000

SWTPC M-6800 ASSEMBLER
ENTER PASS : 1P,1S,2P,2L,2T

00001                  NAM   TABLE
00002                  OPT   L
00003                  OPT   S
00004 1821             ORG   $1821
00005 1821 FE          FCB   $FE
00006 1822 D4          FCB   $D4
00007 1823 BE          FCB   $BE
00008 1824 CE          FCB   $CE
00009 1825 D6          FCB   $D6
00010 1826 D8          FCB   $D8
00011 1827 54          FCB   $54
00012 1828 86          FCB   $86
00013 1829 C6          FCB   $C6
00014 182A 9E          FCB   $9E
00015 182B B0          FCB   $B0
00016 182C 18          FCB   $18
00017 182D 01          FCB   $01
00018 182E 34          FCB   $34
00019 182F 48          FCB   $48
00020 1830 46          FCB   $46
00021 1831 7E          FCB   $7E
00022 1832 36          FCB   $36
00023 1833 3E          FCB   $3E
00024 1834 4E          FCB   $4E
00025 1835 56          FCB   $56
00026 1836 16          FCB   $16
00027 1837 5E          FCB   $5E
00028 1838 1E          FCB   $1E
00029 1839 06          FCB   $06
00030 183A D8          FCB   $D8
00031 183B 58          FCB   $58
00032 183C 00          FCB   $00
00033 183D 30          FCB   $30
00034 183E 00          FCB   $00
00035 183F C8          FCB   $C8
00036 1840 B6          FCB   $B6
00037 1841 9C          FCB   $9C
00038 1842 82          FCB   $82
00039 1843 9A          FCB   $9A
00040 1844 DA          FCB   $DA
00041 1845 D2          FCB   $D2
00042 1846 B8          FCB   $B8
00043 1847 F8          FCB   $F8
00044 1848 C2          FCB   $C2
00045 1849 94          FCB   $94
00046 184A F0          FCB   $F0
00047 184B 92          FCB   $92
00048 184C CA          FCB   $CA
00049 184D FC          FCB   $FC
00050 184E D2          FCB   $B2
00051 184F CC          FCB   $CC
00052 1850 D0          FCB   $D0
00053 1851 90          FCB   $90
00054 1852 DC          FCB   $DC
00055 1853 C4          FCB   $C4
00056 1854 F2          FCB   $F2
00057 1855 BA          FCB   $BA
00058 1856 BC          FCB   $BC
00059 1857 84          FCB   $84
00060 1858 FA          FCB   $FA
00061 1859 C0          FCB   $C0
00062 185A F6          FCB   $F6
00063 185B 00          FCB   $00
00064 185C 00          FCB   $00
00065 185D 00          FCB   $00
00066 185E 00          FCB   $00
00067 185F 81          FCB   $81
00068 1860 00          FCB   $00
00069 1861 1C          FCB   $1C
00070 1862 02          FCB   $02
00071 1863 1A          FCB   $1A
00072 1864 5A          FCB   $5A
00073 1865 52          FCB   $52
00074 1866 38          FCB   $38
00075 1867 78          FCB   $78
00076 1868 42          FCB   $42
00077 1869 14          FCB   $14
00078 186A 70          FCB   $70
00079 186B 12          FCB   $12
00080 186C 4A          FCB   $4A
00081 186D 7C          FCB   $7C
00082 186E 32          FCB   $32
00083 186F 4C          FCB   $4C
00084 1870 50          FCB   $50
```

interface is set to select a machine command. Only one input line of the PIA is used to sample the status of the printer READY or BUSY.

Since the shift feedback and interlock contacts were missing, a timing loop provides for the timing here; however, for the carriage return it has been necessary to build an interlock contact to lock out the printing function till the completion of a carriage return which takes a variable time.

A commented assembler listing of the program driving the printer was written for a 6800 and assembled with output to my Selectric (see listing 1). It can be loaded after the original SWTPC tape has been read in. A refinement which could be added is to provide for motor on or off via software as the printer can be powered up only after the program is running. This is because the power up reset of the computer leaves the PIA LINES all programmed as inputs, ie: open circuited and this simultaneously turns on all the machine magnets.

Another refinement could be to sense via an unused input line if the motor is on or off and steer the output to a TV terminal when the printer is off. To probe into undocumented programs like the SWTPC assembler or editor, I used a little program which searches the memory for a particular string of bytes and prints out the address of the first byte when and if found. I think it can save lots of time.

Fulvio Guzzon
c/o L Alessio
Via Anassagora 63
Casalpalocco 00124
Rome ITALY■


```
00085 1871 10          FCB   $10
00086 1872 5C          FCB   $5C
00087 1873 44          FCB   $44
00088 1874 72          FCB   $72
00089 1875 3A          FCB   $3A
00090 1876 3C          FCB   $3C
00091 1877 04          FCB   $04
00092 1878 7A          FCB   $7A
00093 1879 40          FCB   $40
00094 187A 76          FCB   $76
00095                  END

TOTAL ERRORS 00000
```

Table 1: Assignment of bits.

BIT 6	ROTATE+1	when energized removes the ROTATE+1 latch
BIT 5	ROTATE+2	when energized removes the ROTATE+2 latch
BIT 4	ROTATE+2A	when energized removes the ROTATE+2 supplementary latch
BIT 3	ROTATE-5	when energized activates the ROTATE-5 latch
BIT 2	TILT 1	when energized removes the TILT 1 latch
BIT 1	TILT 2	when energized removes the TILT 2 latch
BIT 0	CHECK	this one unlatches the print clutch (and so does every one of the previous six)