sinclair ZX80 REVIEW

By MIKE ABBOTT

PRODUCED by Science of Cambridge, the ZX80 is undoubtedly excellent value for money, whether purchased as a kit or ready-built. The Z80 based system incorporates a UHF modulator so that it can plug directly into your television aerial socket and go! All you need is a 9V 600mA unregulated supply. All leads are supplied, including that for a domestic tape recorder, to allow use of its built-in cassette interface for saving programs. Memory expansion is available up to 16K bytes, but the basic machine has 1K byte of RAM which Sinclair claim is used so efficiently that it's equivalent to 4K bytes on a "conventional" computer. We did not check this, but Sinclair also claim that the ZX80 is faster than all other personal computers, and so a comparison was made between a Commodore PET, our UK101 and the ZX80, each working through a simple benchmark test chosen to be independent of their different dialects of BASIC. The winning order was: ZX80, UK101 and PET.

FIRST IMPRESSIONS

The case comes in two halves, which fasten together with plastic rivets to form a pleasant unit in appearance, measuring 175 × 220 × 35 mm. These lightweight pressings are about as robust as plastic egg cartons, however, and it is immediately apparent that the p.c.b. gives strength to the case, rather than the reverse. Although there is a regulator with heatsink, the ventilation slots seen in the photograph turn out to be printed black stripes in reality. The case is probably not strong enough to take real slots! Affordability is the name of the game, and so mere adequacy should not be criticised, particularly as the all important p.c.b. is of *excellent* quality and design, in the ZX80.

The touch sensitive keyboard is fabricated integrally on the p.c.b. and looks smart enough. Membrane switches are used, and these sensitive and easy to clean keypads seem likely to enjoy increased popularity in the future.

CONSTRUCTION

Assembly was self-explanatory, with little risk to the i.c.s because the soldering stage involved mainly sockets and discrete components. A separate leaflet giving assembly instructions is supplied, and this is fairly thorough. It gives, in addition to the assembly procedure, convenient component identifications and colour codes alongside each item in the components list. With 47nF capacitors stamped as 473Z etc., the beginner will appreciate this.

At this stage I would like to reveal the fact that my ZX80 did not work initially due to a construction error on my part. In the belief that my kit was two diodes short, I had made up the deficit with surplus ones, only to find that the short-fall was due to my having put two diodes in what were intended to be vacant positions. If time is not on your side, mistakes like this are all too easily made, and so I proffer this warning. If at the end of the day your ZX80 fails to operate, check not only that components are in the right place, but that there is nothing where there shouldn't be! It costs £10 to put this microcomputer through Sinclair's debugging system—PE did not receive a bill, but you will!

UP AND RUNNING

Deviations from the norm are manifold in the ZX80, and this has made the machine fun to review but difficult to know where to begin.

A manifest curiosity relates to the keyboard, and what Sinclair describe as "key word" entry at a single touch. This refers to such words as INPUT, GOTO, RUN, LIST etc, being printed on the screen by one key press, and is claimed to save you up to 40 per cent typing time, although it must depend upon whether you are developing, or merely running a program. Many "key words" do not begin with the letter of the key to which they are assigned, which can lead to a conflict with one's own reflexes to begin with, that is if you are used to pecking out words like GOSUB in the old fashioned way.

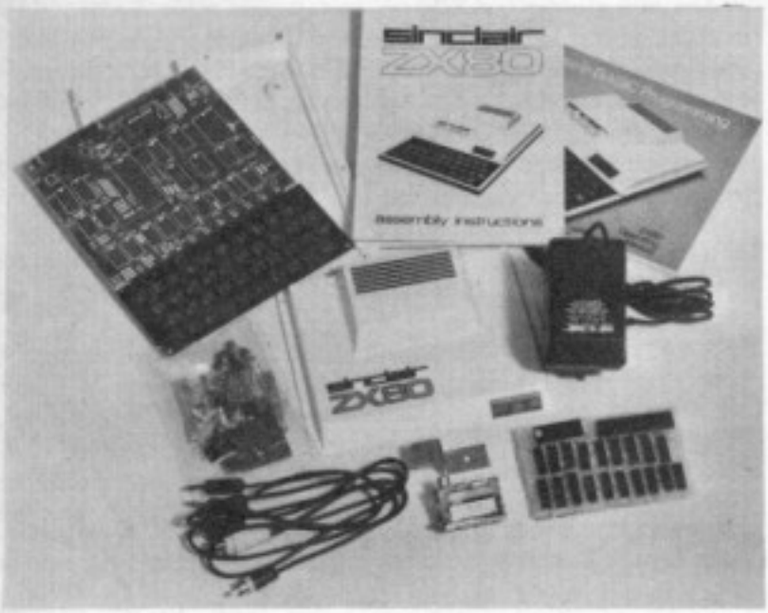
There is a kind of "anticipation logic" associated with the keyboard too! If you take, for example, the U Key; this has three functions. With Shift, it prints the dollar sign, and without Shift it prints U. It can also print IF, but just how can a two-condition switch provide three functions? Well, the key assignments are cleverly arranged so that the machine can assume certain needs. In the case of the U Key, there is no legal way in ZX80 syntax, that a U can follow a line number, and so the "anticipation logic" assumes you want IF . . . and that's what you get! All other key words are juggled thus.

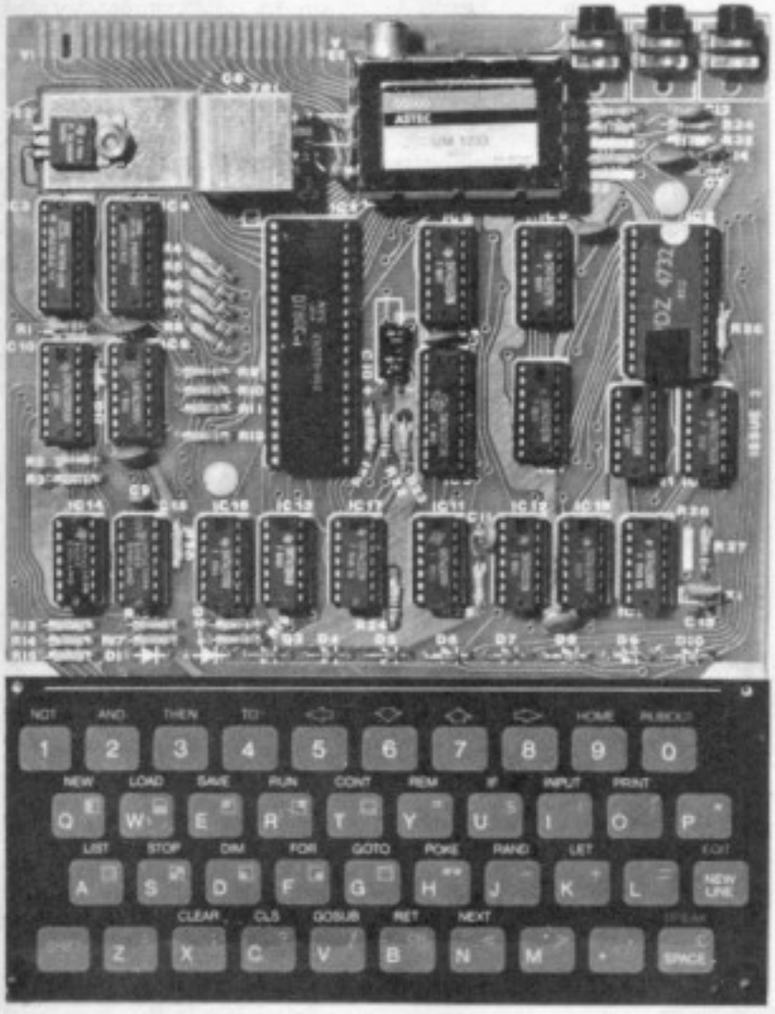
Some of the inflexibilities of the ZX80 may stem from the aforementioned. Only one instruction per line is allowed, and LET is not optional. If it was, you might start a line with U = X + 1, which would confuse the "anticipation logic".

THE DISPLAY

Large clear characters appear on a rock steady display in reverse field video (black characters on white background). A full screen contains 32 columns by 24 rows of characters, and amazingly enough, the VDU operating firmware does not facilitate scrolling. When a full screen is reached, if your program does not execute a *Clear Screen* instruction (CLS) before undertaking further printing, it will "kick out" and throw up the *Screen Full* error message.

At each and every keystroke there is an extremely irritating flicker on the screen, particularly when typing in NEW, which causes a complete eruption. There is no key-repeat function, and when repetitive "dabbing" at any of the membrane switches is necessary; Rubout, for example, concentrating one's eyes on the restless display becomes difficult.





There are ten special graphics characters accessible direct from the keyboard, each available in reverse video by printing the appropriate CHR$, thus providing twenty fairly standard graphics symbols of the kind found on larger machines. Producing bar graphs is quite feasible.

The video display is not memory mapped. In fact, with a total of only 1K of RAM there is no way the VDU could be memory mapped, since 768 locations would be snatched away for screen refresh alone. The processor presumably returns to PRINT statements on an interrupt basis, which would explain why the screen goes blank during any processing—such as FOR/NEXT loops—and this places animated *real-time* graphics outside the realms of possibility. Many disadvantages may result from this.

EDITING

Microcomputers running an interpreted BASIC generally have a small vocabulary of error messages, such that if a program with an error is run, when the mistake is encountered, the machine will stop and display the appropriate message. This might read:

SYNTAX ERROR IN LINE 150
or UNDEFINED STATEMENT ERROR IN LINE 70 etc.

Many machines have numbered error codes which amount to the same thing, and although no computer can comment on the logic of your program, it *can* tell you where you have entered invalid instructions—*after* you have tried running the program! This is where the ZX80 differs, at least in the case of syntax, because it will not accept a line in which you have a syntax error, at the *programming* stage.

The cursor plays an important roll in this function because it can enlighten you the moment you make a mistake, or merely remind you that something needs correcting sooner or later. Let me give you an example.

Supposing you wish to type the following instruction:

10 PRINT "SOFTWARE RULES OKAY"

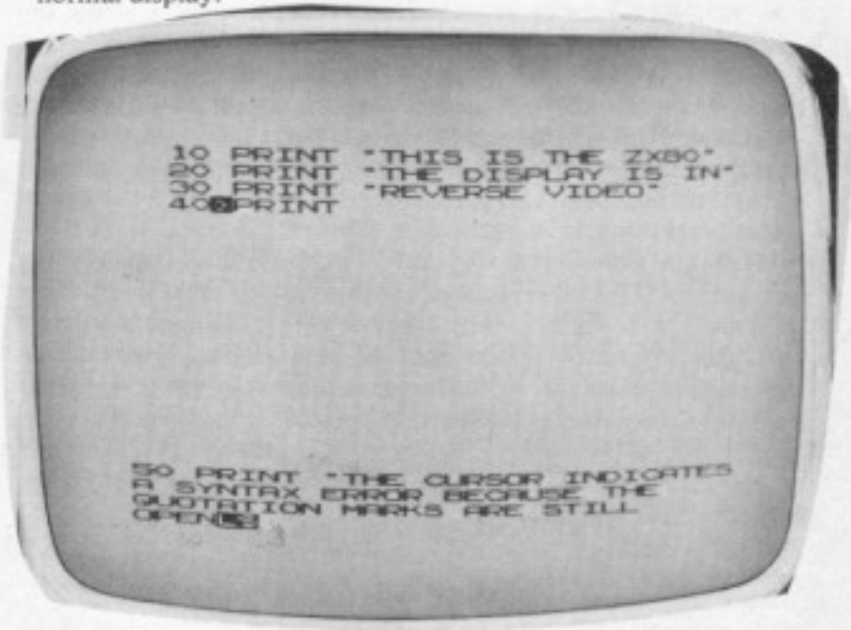
The normal cursor awaits at the bottom of the screen:

K

You type in:

10 PRINT L (three key presses)

The cursor suddenly turns into a reverse-field L. Note that the cursor symbol is described as reverse-field *relative* to the ZX80's normal display.



The L is telling you that the computer is not expecting any further single-touch key words. This relates to what I described earlier as "anticipation" logic, so that, using the former example, the U Key would actually print U now, and not IF. In other words, the cursor reflects the computers expectations.

To continue, opening quotes are now needed.

10 PRINT "LS

Now you find yourself with a two-character cursor clumsily lopping along. The S stands for Syntax error, and while it is present, the line will not be accepted. If you were to hit New Line (Return) now, nothing would happen, and this is because you have opened quotes and not yet closed them.

To finish, typing the remainder gives

10 PRINT "SOFTWARE RULES OKAY" L

Pressing New Line now sends line 10 to the top of the screen to join any previously written lines . . . A "*greater than*" symbol always points to the last line entered.

This leads automatically to discussion of the Editing firmware. Two arrows, obtained by Shift 6 and 7, allow the > symbol to be moved up and down the listing to select the line to be edited. After pressing the Edit key, two further arrows, Shift 5 and 8, allow the cursor to be moved *along* the Edit line, which by now will have appeared at the bottom of the screen. Insertion, alteration and deletion of characters in the usual way is then possible.

You may be wondering how lengthy programs are listed on a machine with no scroll capability. In short, there *is* a scroll which can be executed incrementally on a listing by moving the > symbol downwards using the "downward arrow" key. When the > symbol (which, incidentally, is called the *Current Line* Cursor) hits the bottom line, it scrolls the listing from then on.

Hyphenated LIST commands are not valid for listing blocks of program, but, according to the manual, if you type LIST, followed by a line number, the machine will list your program from that line down to the bottom of the screen. My machine seemed to have a mind of its own about that! I always got the line I wanted, but embedded in a block of program listing of the machine's own choice. If I asked for LIST from a line number already displayed further down, I got the same program block back with the Current Line Cursor readjusted to that line. Fair enough I suppose!

There is in the manual an explanation of the computer's logic for listing specific lines. This seemed little help to me when seeking out certain blocks of lines.

ZX80 BASIC

The ZX80 can only perform "integer" arithmetic; that is to say, whole numbers. Calculations on low value figures can therefore produce pretty meaningless results unless you have implemented a software solution—which is easy enough although heavy on memory! This means that the machine will not accept a decimal point in a listing (the dreaded S appears). It is interesting to note that the S came up even when I tried to reply to an INPUT with a number containing a decimal point. That cursor is ever vigilant! Any number which contains a decimal point as a result of an algorithm will be truncated, even at an intermediate stage. An example, for readers who are not familiar with the limitations of integer arithmetic:

```
10  LET A = 5/2
20  LET A = A*2
30  PRINT A
    gives the result 4
```

Enough criticism! There is one versatile aspect of ZX80 BASIC which impressed me, and that is GOTO n, where n can be a *variable*. This, I think, does more than compensate for the absence of ON n GOTO, found on other machines.

I decided not to waste editorial space repeating facts about the ZX80 which can be found in Sinclair's ubiquitous two-page advertisements, but to confine this review primarily to subjective judgements. For this reason there are no lengthy lists of BASIC verbs and functions here. There are indeed no mathematical functions, although the manual shows how square roots can be resolved by software.

There are a couple of novel string functions, and these are:

TL$ *(string)* which returns the string contained within the brackets minus the first character.

CODE *(string)* which returns the code number of the first character in the string. *This is not the ASCII code*!!!

The non standard ZX80 character code table is given in the manual, and it is interesting to note that complete key words such as CONTINUE, RANDOMISE, CLEAR etc., can all be thrown up on the screen by a single PRINT CHR$ *(code number)* statement.

A pseudo-random number generator allows the following functions:

RND(X) which provides a random interger in the range 1 to X.

RANDOMISE which initialises the random sequence to a number, using the number of frames supplied to the television since it was switched on. This eliminates the likelihood of a random number based program following the same trend each time the machine is switched on.

RANDOMISE n which sets the start of the random number sequence to n. This would be useful if you wanted a random number based program to deliberately do what RANDOMISE avoids, i.e. repeat a pattern or trend each time after switch-on, or execution of this function.

The ZX80 *advertisement* refers to a "Timer under program control". This refers to the PEEKing and POKEing of the TV frame counter, and not standard BASIC real-time.

Undoubtedly software and expansions for the ZX80 will be forthcoming, but for the benefit of those who know absolutely nothing about computers, yet have read the Sinclair *advertisement*, in its minimal form you would at present be hard pressed to run a power station or even play chess with the ZX80.

Another claim which might mislead the novice is: *"Lines of unlimited length"*. They may indeed be! But this can only be in the form of extended PRINT statements or prolonged Boolean expressions, such as:

IF A = 1 OR A = 5 AND K = 6 . . . etc.

This may be very handy indeed, but the hidden truth is that only *one statement per line* is legal, thereby relegating the colon to its literary function only.

Some ZX80 statements are: ABS(n), REM and STR$(n). There is no TAB function, although a software implementation of this is given in the manual.

Other minor observations are that NEXT must be followed by the operator; NEXT Z for instance. INPUT statements may not contain messages or prompts between quotes. For example:

INPUT "NAME"; N$

is illegal. A prior print statement would be necessary for "NAME?"

INPUT statements can only handle single variables. For example:

INPUT X,Y,Z

is illegal. You would require three separate INPUT statements to do this.

With regard to speed, a quick check was carried out to verify Sinclair's claim to processing speed. Below is the program used, and the time taken by two other machines which happened to be available at the time.

		TIMES
10 FOR A = 1 TO 5000		
20 LET X = A*2		
25 LET X = X/2	ZX80	30 sec.
30 NEXT A	UK101	40 sec.
40 PRINT "FINISHED"	PET	50 sec.

It should be borne in mind that the ZX80 had the advantage of processing integer only arithmetic, but Clive Sinclair is quoted as saying that, with its clock speed of 3·25MHz, the ZX80 will *remain* faster when used with the *floating point ROM* which is now in the pipeline. This ROM, incidentally, will incorporate logarithm and trigonometry functions too.

HARDWARE

The unit measures 218 × 170 × 50mm and weighs 340g (12 oz.), and is based on the NEC Z80 μP.

All firmware is stored in one ROM, christened the "Super-ROM". This i.c. houses the *BASIC interpreter, operating system, monitor and character set.* A 46-way edge connector pad (23 + 23 pin) at the edge of the p.c.b. allows for expansion and interfacing. Memory expansion boards are available which can add up to 3K bytes each.

There is no PIO (Parallel Input/Ouput) device on-board. The polarized expansion connector carries the Address Bus and Data Bus, along with the Z80's "single wire" control lines.

Memory comprises two 2114L RAMs; the remaining i.c.s, excluding the regulator, being 74LS series TTL devices.

OPERATING MANUAL

The ZX80 Operating Manual is dubbed "A Course in BASIC Programming". It contains minimal inside information about the hardware and firmware and so would displease the "wires and machine code" man, but instead is an excellent introduction to BASIC, or at least, the ZX80's dialect of the language.

There is little one can say about this as a training manual because it is so good! It is well printed, clearly laid out, and very considerately written for the beginner, and it has some quaint chapter titles. A latter chapter called "4K BASIC FOR ZX–80" gives a summary of the "user's view" of the machine, and for the technically hungry, an elementary "computer's eye view" of how things are organised. There is also a glossary of terms and statements. The appendix gives a number of useful address locations in the ZX80's memory.

So, the ZX80, plus this manual, will make learning BASIC easy and pleasurable. Although there are many more instructions at your finger-tips with an extended BASIC, they are the ones which make programming *easier*, and so learning the language with Sinclair's package should nevertheless be effective and worthwhile.

CASSETTE INTERFACE

The cassette interface is Sinclair's own (300 baud), and conventional SAVE and LOAD commands are used to write and read data using a domestic cassette recorder. There is no VERIFY or *named program* firmware, and curiously for a machine of this kind, *variables* and *data* are SAVEd too; after LOAD you should start with GOTO 1 instead of RUN to get the benefit of this.

The manual implies that you may not record successfully with your cassette player running on mains power, and that you may need to switch to battery power. I did not have this trouble, but using a Sony TC–207, recording programs could not be achieved with the Earphone lead connected simultaneously with the Mic. lead. The earphone plug had to be removed. I later discovered an addendum slip to this effect.

The display, unrefined as ever, produced ugly patterns whilst cassette handling.

CONCLUSION

I see the ZX80 in the classroom, and in workshop control applications. Perhaps even hidden in the executive's top draw, to be pulled out at lunch times to resume training. For these situations, the machine is excellent, and eminently suited to teaching children the art of computer programming. It is of little use scientifically at present, with only integer capability and no mathematical functions, and this to some extent wastes the boasted processing speed of the machine.

At the time of writing it seemed appropriate to advise that a firm delivery date be secured before purchasing the ZX80 microcomputer.

A ZX80 Users Club has been formed, membership of which costs £6 p.a. (£10 overseas), which includes the newsletter and the Software Bank Index. For details, write to:

ZX80 Users Club, PO Box No. 159, Kingston upon Thames, Surrey KT2 5UQ.