the fully fledged Junior Computer

On board data busing provides for wider communication

Since Junior Computer Book 2 is now available, it is time we moved on to the next and final stage in the Junior Computer project. The addition of the forthcoming interface card will transform the Junior into the complete personal computer system. This card will from the essential link between the computer and the outside world as it includes additional RAM and EPROM memory and an extra I/O system. It also enables further memory cards and a cassette recorder to be connected to its buffered bus allowing the Junior Computer to communicate with the operator (and vice versa) in a far more sophisticated manner.

This article is the first in a series of three and propose to describe the theoretical and practical aspects of the hardware extension. These articles will form the basis of Books 3 and 4, to be published later this year, and may therefore be lacking in fine detail (due to lack of space) but the intention is to give our readers a good idea of what is in store.

Now that the interface card is to be added to the Junior, the computer will be 'fully grown', that is, it will have all the hardware it requires. Its 'brain', the software, may however be further expanded. It is time for the computer to 'talk like an adult', or rather, operate in a higher level programming language and it would of course be interesting for the operator to learn to work with such a language. Elsewhere in this issue machine language and additional system software (including two software 'aids') are described in detail, but the language involved is still only machine language.

The most obvious choice would be BASIC, an 8K version on cassette, for example, combined with 16K of RAM. Naturally, this would have to be based on an existing software system such as that developed by Apple or Kim and this rather expensive option is only worth considering for the Junior Computer, provided there are enough people interested in the idea. Think about it and let us know!

In any case, Elektor is planning to provide a 16K dynamic RAM card within the near future. This would constitute an economical alternative to two RAM/EPROM cards which both include 8K of RAM.

First things first, what extensions are required to bring the Junior Computer to adulthood?

To bus or not to bus . . .

Computer systems especially designed for hobbyists are available in two types: bus systems or single board systems. The former employs several cards (printed circuit boards) to incorporate the components. The cards are all the same size (the eurocard format is 100 x 160 mm, for instance) and are individually linked by means of a 'bus'. This is a printed circuit network in which equally positioned points (such as connector pins) are interconnected. Since the system is universally compatible, the sky is the limit as far as the *theoretical* expansion possibilities are concerned, the final result being a computer with multiple facilities.

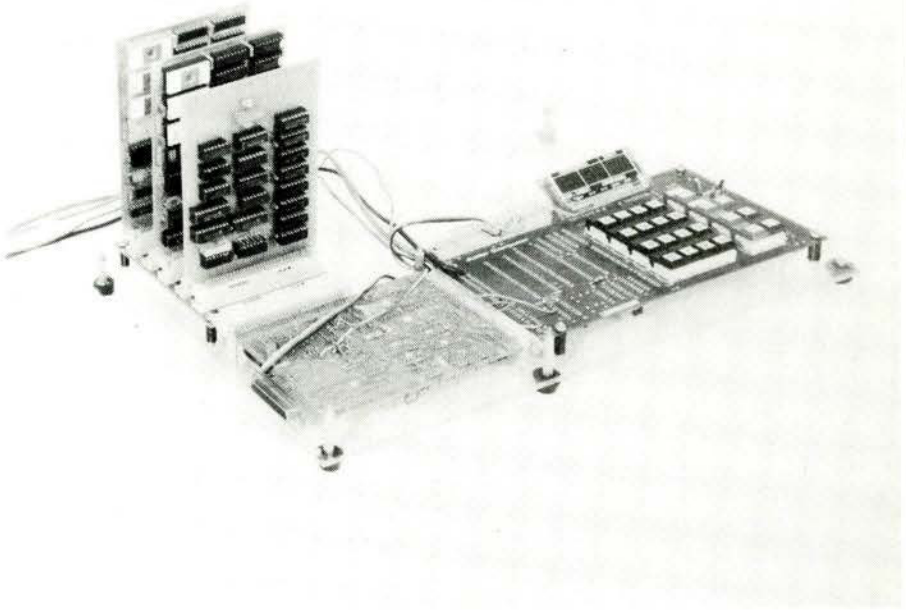
A single board system, on the other hand, incorporates everything on a single card. It all depends on what the 'everything' must comprise to suit the user's purposes (and therefore on the size of the card). Putting 'everything on a single card' is a bit of a gamble: you either win or you lose. It means the user has to estimate his/her needs very carefully from the start and this often proves to be quite a handicap, as it is difficult to plan ahead when the possibilities are largely unknown.

Single board computers are mainly used to teach operators and as part of relatively straightforward process control systems (where the computer is switched on permanently). In its standard version the Junior Computer is in fact a single board system, being designed to teach beginners. The standard version is already being used by many for all sorts of applications already, ranging from A/D conversion to process control in the manufacture of semiconductors. Thus, the interface is a purely optional addition and the expansion connector may be left untouched.

Nevertheless the expansion connector is there and for two very good reasons: it allows a cassette recorder to be connected as a backup memory (to boost a RAMshackle memory), thereby permitting full use of the total 64K memory capacity and second, it enables peripheral equipment in the way of a video terminal, etc. to be connected to the computer. This provides additional I/O and improves communication between the computer and the outside world.

Once the expansion connector is employed, the Junior Computer is no longer a single board system. This does not mean its possibilities as a bus system will be infinite. On the contrary, the computer is limited by its own hardware.

In any case, the expanded version bears no physical resemblance to the average basic system. All the extensions mentioned above are included on the interface card, which has the same size as the main board, so that they can be 'sandwiched' together. An existing SC/MP bus card may be linked to the interface card to house several other memory cards.
In spite of the bus card addition, therefore, the Junior Computer is not a bus system, but rather a 'double-decker' sandwich! However, never mind what category the computer could fall into, let's look at the system itself.



The interface card

The word 'interface' means 'link'. Here the interface card provides the Junior Computer with a vital link with the outside world in the form of various communication channels: additional I/O, a cassette interface, an RS-232-interface and an internal connection with the buffered bus board.
All the electronics involved may be found in figures 1 and 2. As can be seen, it is more elaborate than the main board, even though it is equally compact in size. Each component will now be discussed separately.

The buffers: softening electronic blows

All the connections belonging to the INPUT CONNECTOR are shown to the left of figure 1. This connector together with five links to the port connector (see figure 2) allow for data transfer to and from the standard Junior Computer. With the exception of lines EX and K1 . . . 6, which serve the interface card only, all the lines lead to the OUTPUT CONNECTOR on the right-hand side of figure 1. This connects the interface to the bus board which in turn permits one or more memory cards to be connected.
The well-known A address lines are marked AB here and the D data lines are shown as DB; the B stands for 'buffered'. Why is buffering necessary? Well, for two reasons. First, it prevents an overload condition brought about by too many connections. Second, it enables data transfer along the data lines between the standard Junior Computer and the interface and memory cards to be controlled.
N1 . . . N16 act as address buffers. Since addressing always takes place by way of the 6502 on the main board, the triangles are pointing in the direction as indicated: left for inputs, right for outputs. N17 . . . N32 act as data buffers and here the even number buffer inputs point to the right and the outputs to the left, whereas the odd buffer inputs point to the left and the outputs to the right.
Three situations are feasible:

1) Ports N17 . . . N32 with odd numbers are active and the even ones are inactive when both $\overline{\text{WRITE}}$ and READ are logic 0. Data may then flow from the standard board to the extensions, if data is to be written either in the *interface* memory, the extension memory (*bus* memory) or in the I/O.
2) When $\overline{\text{WRITE}}$ and READ are both logic 1, ports with odd numbers between N17 . . . N32 are inactive (tri-state, that is to say, high impedance at both the inputs and the outputs) and the even ones are active. Data may then flow from the extension boards to the main board, if data is to be read either from the interface memory, the bus memory or from the I/O.
3) All ports between N17 and N32 are inactive when $\overline{\text{WRITE}}$ is logic 1 and READ is logic 0. No data will be transferred. This happens whenever data is being read or written inside the standard Junior memory or in the I/O. The $\overline{\text{WRITE}}$ and READ signals are generated by the PROM IC17. More about this later.

N.B. A fourth theoretical possibility is that $\overline{\text{WRITE}}$ = logic 0 and READ = logic 1, all ports in the N17 . . . N32 range being active at the same time. This should never happen.

Additional I/O

The VIA IC1, type 6522, merits individual attention. As a matter of fact, a whole chapter will be devoted to the subject in the second half of Book 3. The VIA (Versatile Interface Adapter) is remarkable in that it offers more facilities than the standard I/O nucleus, the 6532 PIA. As shown in figure 1, the VIA 'CONNECTOR' incorporates the relevant connections with the outside world. (It is put in inverted commas, as it is not a real connector). The 6522 is controlled by address lines AB0 . . . AB3 and by various signals produced by the control bus.
The eight data lines and the IRQ output signal (interrupt after a timer time out) will be familiar from Book 1.
Like the 6532, the 6522 features two chip select signals. Its $\overline{\text{CS2}}$ is connected to the output signal K6 of the address decoder IC6 on the Junior Computer's main board. The CS1 signal is linked to the output of N35 which is controlled by K6 and AB9. For the VIA to be activated $\overline{\text{CS2}}$ and CS1 will have to be 0 and 1 respectively. N35 will now act as a NOR gate, its output being logic 1 when both inputs AB9 and K6 are logic zero (in the case of the 6532, A9 has to be logic 1). Since A8 (AB8) is not connected to either the 6522 or the 6532, the VIA 6522 can be addressed as follows;
1800 = 1900 . . . 18FF = 19FF
(AB8=X:AB9=0:K6=1)
and the PIA 6532;
1A00 = 1B00 . . . 1AFF = 1BFF
(A8=X;A9=1;K6=1)
Since double addressing is excluded, as A8=AB8=X, 256 addresses are available for both the PIA and for the VIA. In the former case 19 different memory locations are available for the PIA in addition to 128 bytes of RAM. As can be seen in figure 1, address lines AB4 . . . AB7 are not connected. Thus, there are only 16 different memory locations for the VIA.

Interface memory

The interface card has room for up to 5K supplementary memory which is

1

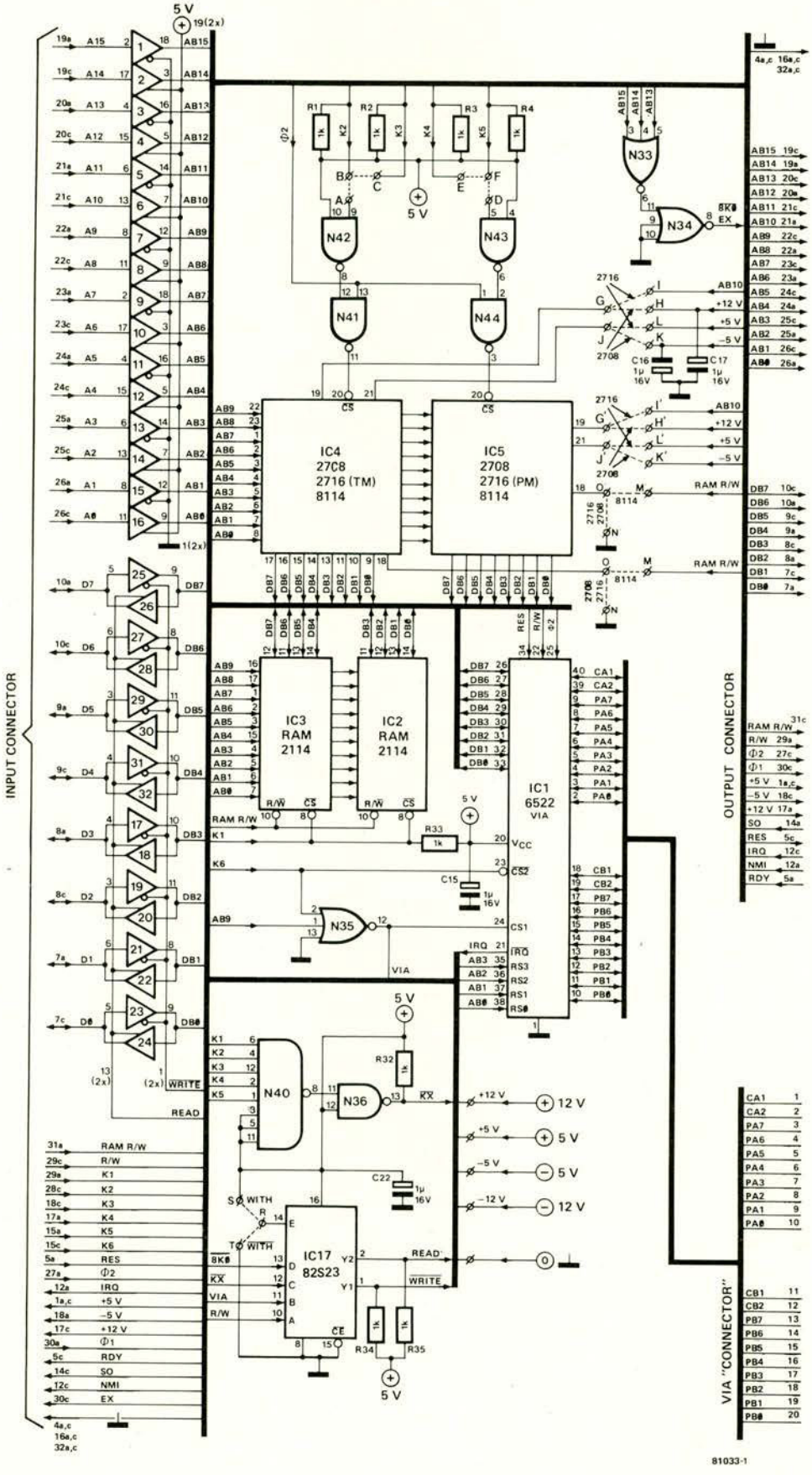


Figure 1. The interface card. It provides more memory, additional I/O, buffering including data buffer control and complete address decoding.

elected by K1 . . . K5. This is enough o allow the main board's decoded ddresses (8K) to be fully utilised including 1K for the PIA plus VIA).

n the first place, additional RAM is ow available. Even without using other xtension facilities, the operator can lready dispose of 1K of RAM on the nterface card by way of IC2 and IC3 1024 data nibbles per IC). The RAM is elected when K1 = $\overline{CS}$ = logic Ø. Thus, ts address range covers:

Ø4ØØ . . . Ø7FF.

n other words, four pages (Ø4 . . . Ø7) are available linking up nicely with the standard RAM pages ØØ . . . Ø3, so that engthy user programs can now be stored 'in one piece'.

Both IC4 and IC5 may be 1K RAM 8114), 1K EPROM (2708) or 2K EPROM (2716) types. One or two signals K2 . . . K5 select each IC with the aid of the clock signal Φ2 (ports N41 . . . N44). The latter is necessary to time the reading and (if appropriate) the writing process.

Addressing is as follows:

K2 = logic 0 → addresses Ø8ØØ . . . ØBFF
K3 = logic 0 → addresses ØCØØ . . . ØFFF
K4 = logic 0 → addresses 1ØØØ . . . 13FF
K5 = logic 0 → addresses 14ØØ . . . 17FF

For 1K of RAM (8114) or 1K of EPROM (2708) each IC provides various selection possibilities. For 2K of EPROM (2716) two K signals are combined: K2 & K3 for IC4 (2716) (address range Ø8ØØ . . . ØFFF) and K4 & IC5 for IC5 (2716) (address range 1ØØØ . . . 17FF). Solder pins A . . . F are supplied to select these addresses. They are interconnected to suit the type of IC8. The same is true of pins G . . . O and G' . . . O', which are linked according to the value of the supply voltages and also according to whether an additional address line AB10 or the RAM-R/W signal is required or not.

Full address decoding

In the standard Junior Computer only 8K out of a total of 64K is decoded. Address lines A13 . . A15 are not used, as a result of which pages ØX, where X = Ø . . . F, are also accessed on pages 2X, 4X, 6X, 8X, AX, CX and EX, and pages 1X, where X = Ø . . . F, are also accessed on pages 3X, 5X, 7X, 9X, BX, DX and FX. In this particular situation pin D belonging to the address decoder IC6 on the Junior's main board is grounded via a wire link.

If the memory is to be extended by more than 5K, we will have to set up an address decoding system that enables us to access any address within the 2ØØØ . . . 2FFF range.

This is achieved as follows. First pin D of IC6 in the standard Junior Computer is linked to pin EX — *move the wire bridge to EX!* Pin EX=pin D is connected to the output of inverter N34 which is controlled via the NOR gate N33 and this in turn is connected with AB13 . . . AB15. As soon as one of the three address lines mentioned becomes logic 1, EX and therefore D of IC6 become logic 1 too. As a result, none of the output signals KØ . . . K7 of IC6 will be low. If all three address lines AB13 . . . AB15 are logic zero, pin D of IC6 will also be low (grounded) and one of the KØ . . . K7 signals will become logic zero. This means the I/O and memory on both the main board and the interface board can only be accessed in pages ØX and 1X, where X=Ø . . . F; pages 2X onwards can be accessed via the bus card. Supplementary address decoding is required on either the RAM/EPROM card or on the 16K dynamic RAM card to select one or more memory areas in the relevant address range 2ØØØ . . . FFFF.

The EX output signal is also sent, under the denomination $\overline{8KØ}$, to an address input of the PROM (IC17). This contains 32 bytes in all, but only two bits in each byte, Y1 (=$\overline{WRITE}$) and Y2 (=READ), are used. The two bits control data buffers N17 . . . N32. The 32 bytes can be accessed via the five address lines E . . . A, which will be described later on.

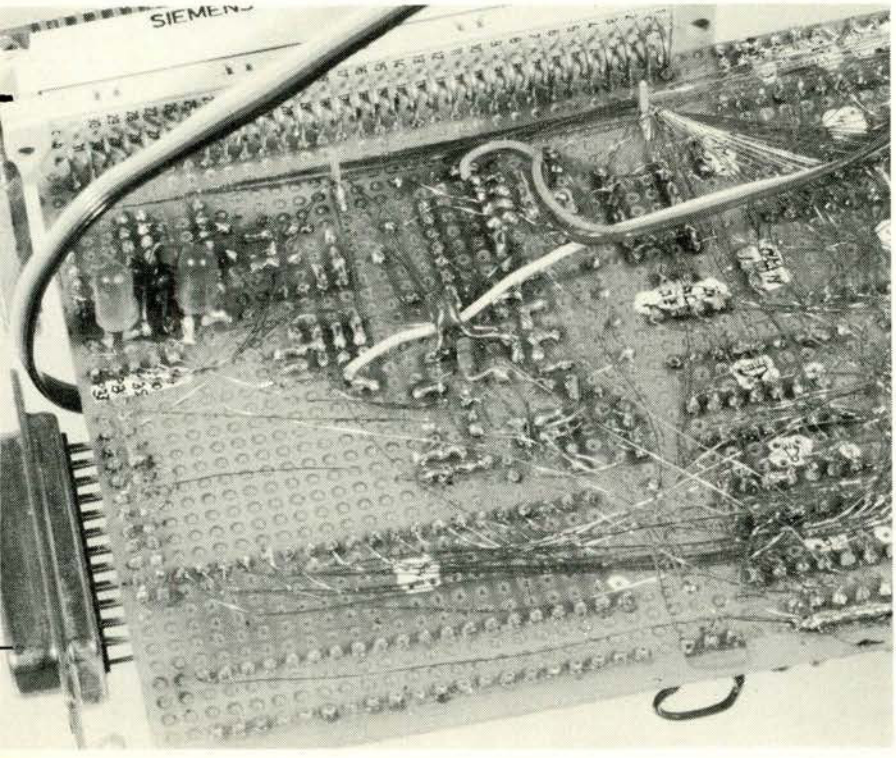
Seeing as the PROM is so complicated to access, why have one at all? Why not simply make do with the R/$\overline{W}$ signal in the manner provided on the Junior's main board? To find out, let's see what happens when data is read from the RAM, EPROM or I/O stationed on the main board. If the R/$\overline{W}$ signal is directly connected to READ and $\overline{WRITE}$ signals on the extension card, the eight data buffers with inputs facing the interface card will be enabled. These inputs will in fact not be connected to anything at all — in other words they will be left hanging in mid-air — as the interface card is not being addressed! Thus, the inputs will be in a random logic state and, as their outputs are activated, the information that appears on the data lines is bound to be incorrect. The solution is therefore to control the data bus buffers with the address system of each circuit. In the example given above (reading data into one of the ICs on the main board) the CPU 6502 does not require anything beyond the buffers and so these must be disabled. It should be noted, however, that such problems do not arise when data is being *written* on the main board.

The PROM is included for another reason as well. The NMI, RES and IRQ vectors can now (once the address decoding is complete) be stored at their true addresses FFFA . . . FFFF, instead of 1FFA . . . 1FFF (in the standard EPROM). This will only work, of course, if page FF is really stored in EPROM memory. Since page FF is included on a RAM/EPROM card here that is connected to the computer by means of a bus board, we will refer to it as bus board memory fron now on.

N.B.: As soon as a bus board is used tc connect a memory card, page FF must be included on EPROM with the correct vectors, so that 'double crossing' addressing (where FF=1F, etc.) is no longer possible.

Now to get back to addressing the PROM IC17. The five address signals with which the $\overline{WRITE}$ and READ signals are obtained are as follows:

A. The R/W or R/$\overline{W}$ signal (the dash above the W merely means: 'enabled when logic zero' and helps avoid confusion).
B. The 'VIA' signal. This is obtained from N35's output and is none other than the CS1 signal inherent to the



VIA IC1. Inside the 1K zone decoded by K6=logic zero, the PIA is situated above the data buffers and the VIA below them.

C. The $\overline{KX}$ signal. This is derived from N36 which is enabled by the output of N40. This is logic zero when one of the K1 . . . K5 lines is logic zero, in other words, when the interface memory is being accessed.

D. The $\overline{8K0}$ signal. This is another name for the EX signal which was mentioned earlier and will be logic zero when the first 8K (of memory and I/O on both the main and interface board) is being addressed and will be logic 1 when bus board memory is being addressed.

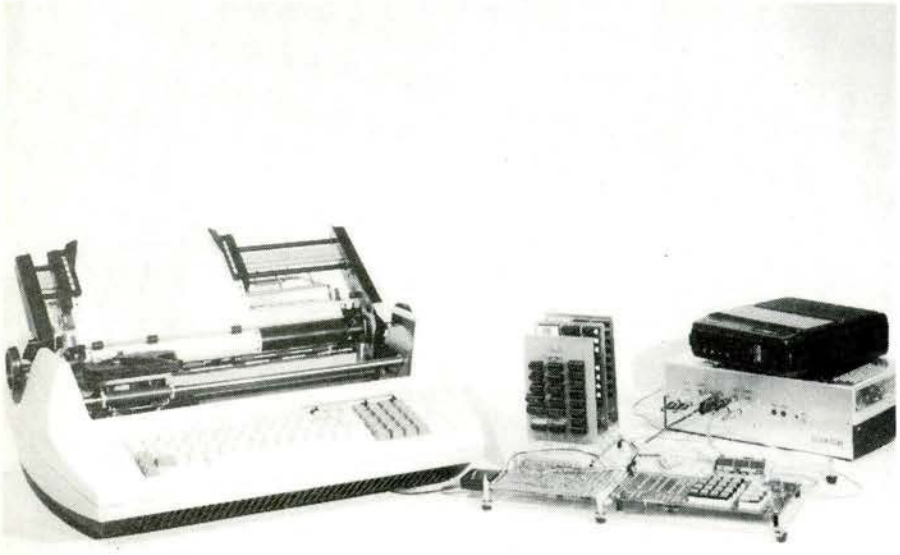
E. Pin E of IC17 is either linked up with +5 V (wire link RS) or grounded (wire link RT). Pin E *must* be logic zero when no bus board memory is connected (indicated as $\overline{WITH}$, meaning WITHOUT) and must be logic 1, when it is in fact connected (WITH, meaning $\overline{WITHOUT}$). The wire link is RS or RT is necessary for the simple reason that it enables the user to decide whether the bus board memory is to be connected or not.

Table 1 gives a survey of the contents of the PROM IC17. Bits Y3 . . . Y8 are logic zero and are not used. In principle, 32 different situations are possible leading to one of the three 'legal' READ & $\overline{WRITE}$ combinations is selected. In practice, however, only eight situations remain if the separation into read and write is disregarded:

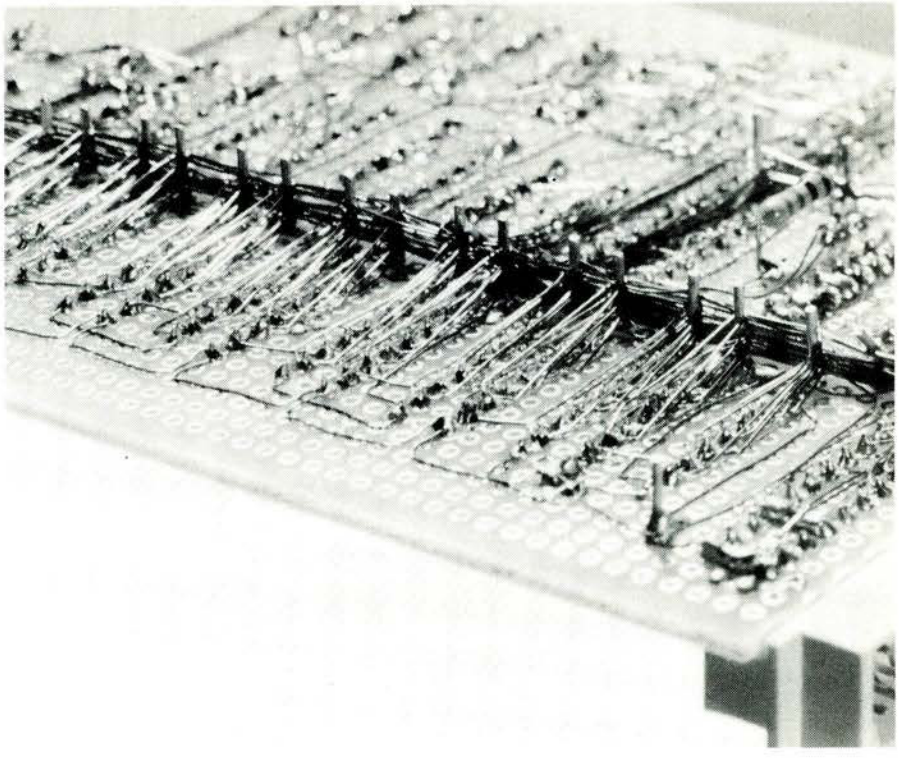
1. Writing or reading memory on the interface card (IC2 . . . IC5). Eight data buffers are enabled to indicate either read or write in PROM addresses 00 and 01.
2. Reading EPROM and either reading or writing RAM or PIA on the main board. The data buffers are all disabled (PROM addresses 04 and 05).
3. Reading or writing into the VIA. Since this involves the interface card eight data buffers will be enabled to indicate either process (PROM addresses 06 and 07).
4. Writing or reading bus board memory (PROM addresses 0C and 0D). Since once again E is low ($\overline{WITH}$=WITHOUT), no bus board memory is connected and so the data buffers *must* all be disabled. The NMI, RES and IRQ vectors are automatically defined by the standard EPROM of the Junior Computer.
5. See point 1 (PROM addresses 10 and 11).
6. See point 2 (PROM addresses 14 and 15).
7. See point 3 (PROM addresses 16 and 17).
8. This partly corresponds to point 4 (PROM addresses 1C and 1D). As in cases 5 . . . 7, E is logic 1 ($\overline{WITH}$=$\overline{WITHOUT}$) which means the bus board memory must be connected. Now however the data buffers will have to be enabled to allow data transfer to take place in one direction or another. The

tabel 1

PROM-address (hex)	E = WITH of $\overline{WITH}$	D = $\overline{8K0}$	C = $\overline{KX}$	B = VIA	A = R/W	Y8	Y7	Y6	Y5	Y4	Y3	Y2 = READ	Y1 = $\overline{WRITE}$	PROM-data (hex)
00	0	0	0	0	0	0	0	0	0	0	0	0	0	00
01	0	0	0	0	1	0	0	0	0	0	0	1	1	03
02	0	0	0	1	0	0	0	0	0	0	0	0	0	00
03	0	0	0	1	1	0	0	0	0	0	0	0	0	00
04	0	0	1	0	0	0	0	0	0	0	0	0	1	01
05	0	0	1	0	1	0	0	0	0	0	0	0	1	01
06	0	0	1	1	0	0	0	0	0	0	0	0	0	00
07	0	0	1	1	1	0	0	0	0	0	0	1	1	03
08	0	1	0	0	0	0	0	0	0	0	0	0	0	00
09	0	1	0	0	1	0	0	0	0	0	0	0	0	00
0A	0	1	0	1	0	0	0	0	0	0	0	0	0	00
0B	0	1	0	1	1	0	0	0	0	0	0	0	0	00
0C	0	1	1	0	0	0	0	0	0	0	0	0	1	01
0D	0	1	1	0	1	0	0	0	0	0	0	0	1	01
0E	0	1	1	1	0	0	0	0	0	0	0	0	0	00
0F	0	1	1	1	1	0	0	0	0	0	0	0	0	00
10	1	0	0	0	0	0	0	0	0	0	0	0	0	00
11	1	0	0	0	1	0	0	0	0	0	0	1	1	03
12	1	0	0	1	0	0	0	0	0	0	0	0	0	00
13	1	0	0	1	1	0	0	0	0	0	0	0	0	00
14	1	0	1	0	0	0	0	0	0	0	0	0	1	01
15	1	0	1	0	1	0	0	0	0	0	0	0	1	01
16	1	0	1	1	0	0	0	0	0	0	0	0	0	00
17	1	0	1	1	1	0	0	0	0	0	0	1	1	03
18	1	1	0	0	0	0	0	0	0	0	0	0	0	00
19	1	1	0	0	1	0	0	0	0	0	0	0	0	00
1A	1	1	0	1	0	0	0	0	0	0	0	0	0	00
1B	1	1	0	1	1	0	0	0	0	0	0	0	0	00
1C	1	1	1	0	0	0	0	0	0	0	0	0	0	00
1D	1	1	1	0	1	0	0	0	0	0	0	1	1	03
1E	1	1	1	1	0	0	0	0	0	0	0	0	0	00
1F	1	1	1	1	1	0	0	0	0	0	0	0	0	00



three vectors are now sought in page FF. This must be connected to the EPROM which specifies the three vectors at addresses FFFA . . . FFFF.
It can be concluded that half of the 32 bytes inside the PROM IC17 are truly necessary. The other 16 logic states are irrelevant, since such combinations as D=$\overline{8K\emptyset}$, C=$\overline{KX}$ and B=VIA just do not arise. In the other 16 cases however Y1 and Y2 are always low. This indicates that the data buffers are prepared for a write operation and so perfectly harmless.
That covers figure 1, now for figure 2..



The cassette interface

Most of the components shown in figure 2 refer to the cassette interface. This includes everything that is needed in the way of hardware (the software involved is dealt with elsewhere in this issue) to transfer data to and from a cassette recorder. When data is being read it is the tape that is emitting (playing back) data and when it is being written into the cassette it is recorded on tape.
The data is transferred to and from the 6502 μP by way of port line PB7 belonging to the port connector. When data is being written during the DUMP/ DUMPT subroutine of the TAPE MANAGEMENT system program, PB7 functions as an output and so do PB5 and PB6. Post PB5 will then be low and PB6 high. As a result, the input of N38 with pin number 8 will also be low and so its output will be high (N38 has an open collector output which is high ohmed whenever it is logic 1). Since PB5 is logic Ø the PNP darlington T3 is connected up via R15 and so the *red* LED D5 (OUTPUT ON) lights and relay Re2 is activated. The two contacts of output J4 are interconnected. If these are connected in series with the motor of the recorder OUTPUT (remote control), the recorder can be switched on by means of software, provided that the recorder is ready to record.
Since PB6 is logic 1 during a write operation, T2 will not conduct, the green LED D4 will not light and relay Re1 will not be activated. The output of N39 leads to P2 via R20 and C14 and P2 presets the maximum recording level. J2 is the data output.

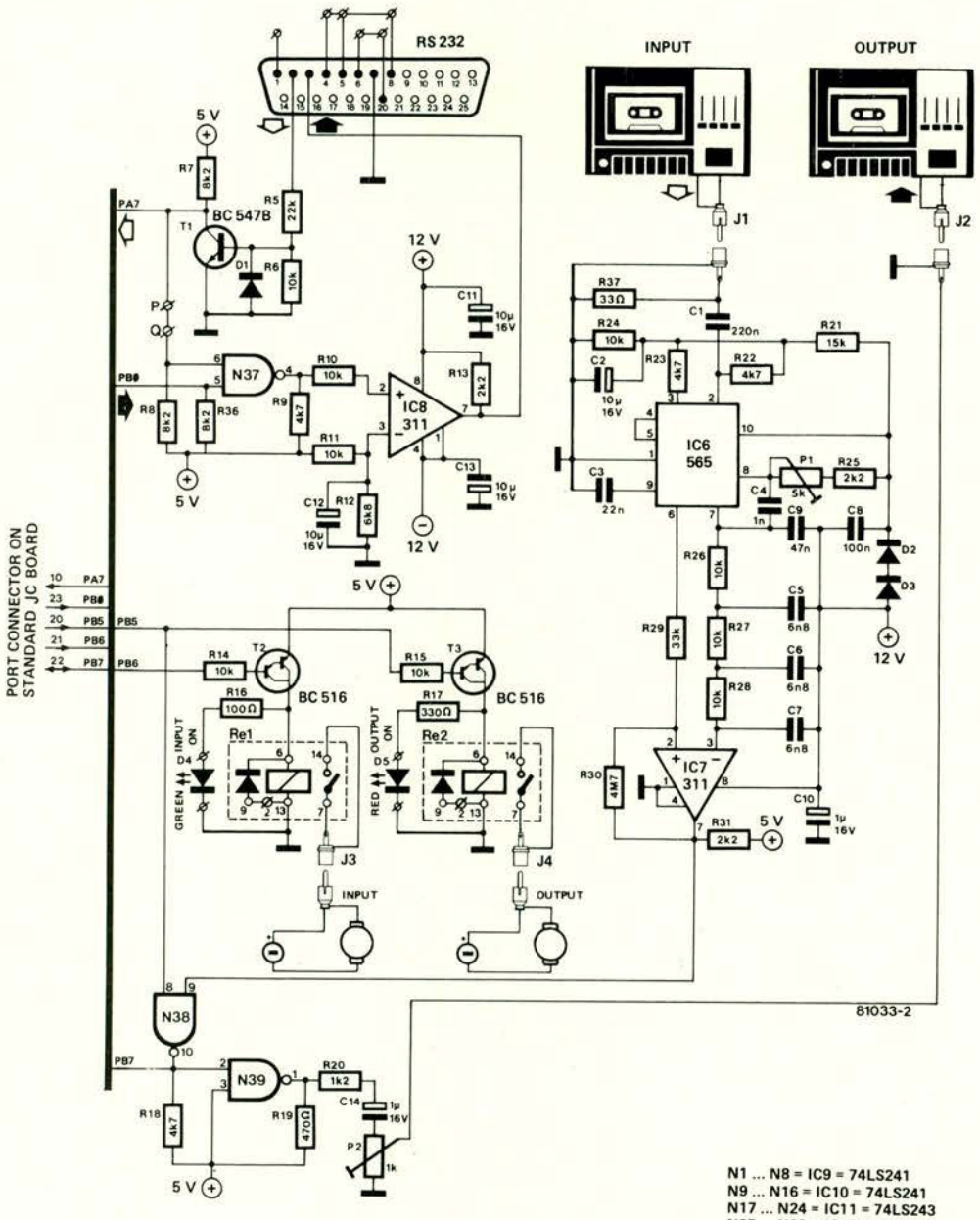
When data is read on tape during the subroutine RDTAPE in TAPE MANAGEMENT, PB7 acts as an input and PB5 and PB6 will again be outputs. Port PB5 is logic 1 now and PB6 logic Ø. As a result, N38 can pass on the output signal of IC7 to PB7 after inverting it. Port 39 remains connected through, so that signals also reach the data output J2, but without doing any harm.
Now that PB5 is logic 1, the red LED D5 will not light. PB6 will now be low causing T2 to conduct, the green LED D4 (INPUT ON) to light, relay Re1 to be activated and the contacts of J3 (remote control for the playback recorder INPUT) to be disconnected.
There is no absolute need to use separate recorders for storing and fetching data. If one is used for both purposes, remote control will no longer be necessary and the relays will then seem superfluous. However it is best to mount them *both* on the board just in case!
In addition, the T2 and T3 controls provide useful visual information about what is actually going on. The green LED lights when data is being read and the red LED is lit when data is being written.

The section in figure 2 between the data input J1 and the input of N38 (pin number 9) is next on the list. This is the circuit around IC6 and IC7 which will undoubtedly look very familiar to KIM operators. It is in fact similar to part of the KIM hardware, albeit improved, because we didn't feel like reinventing the wheel.
Before discussing IC6, IC7 and co. it is useful to know that the data is recorded on tape in the form of a series of audible signals. These are rectangular in form, a 3600 Hz signal being followed by a 2400 Hz one, and so forth. A frequency of 2400 Hz corresponds to a low level bit (logic Ø) while 3600 Hz corresponds to a high level bit (logic 1). The circuit around IC6 and IC7 makes sure the output of IC7 is high when there is a frequency of 3600 Hz at the input J1 and that it is low when the frequency at the input is 2400 Hz. The software of RDTAPE distills either a zero or a null from each cycle, depending on the 3600 Hz time in relation to the concluding 2400 Hz time. Together with the associated components the circuit around IC6 constitutes a PLL (Phase Locked Loop). To explain this in great detail (with complex formulas) would fill the pages of Elektor for a whole year! We hope readers won't mind if instead we describe its operation very briefly.

Where the interface card is involved, the PLL may be regarded as a *frequency follower*. This is because it is similar to an emitter follower, where the output voltage imitates the input voltage, an internal oscillator here adapts its frequency to that of the input signal within a certain frequency range and above a minimum level of the input signal. The internal oscillator produces a frequency that is proportional to a control voltage (VCO). Without the input signal the frequency is about 3000 Hz, so exactly half-way between the 2400 Hz. If the frequency at the input is 3600 Hz, the frequency will rise by 500 Hz; at 2400 Hz this will drop by 600 Hz. The control voltage has to be increased for the frequency to drop and lowered for it to rise. Thus, the level of the control voltage is directly related to the frequency of the input signal and after comparing it to a fixed voltage it allows a logic distinction to be made between the two frequencies: locig 1 for one frequency and logic Ø for the other. This is in accordance with the FSK (Frequency Shift Keying) principle.

The PLL is supplied with a +12 V voltage by means of D2 + D3 and C8 which are connected in parallel for transient suppression. The 'clean' 12 V (about 11 V due to the diodes) is also used to preset the input DC voltages (pins 2 and 3 of IC6) by way of R21 . . . R24. In the KIM a 5 V supply is used and any interferance will pass

2



N1 ... N8 = IC9 = 74LS241
N9 ... N16 = IC10 = 74LS241
N17 ... N24 = IC11 = 74LS243
N25 ... N32 = IC12 = 74LS243
N33 ... N35 = IC13 = 74LS27
N36 ... N39 = IC14 = 74LS01
N40 = IC15 = 74LS30
N41 ... N44 = IC16 = 74LS00
Re1,Re2 = DIL reed relay 380 Ω

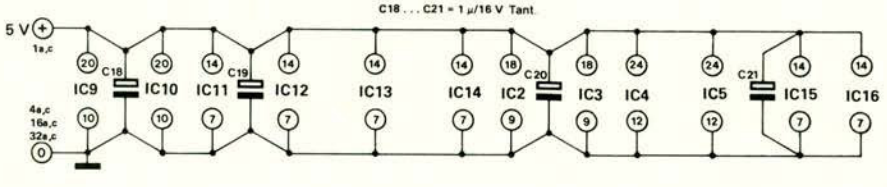
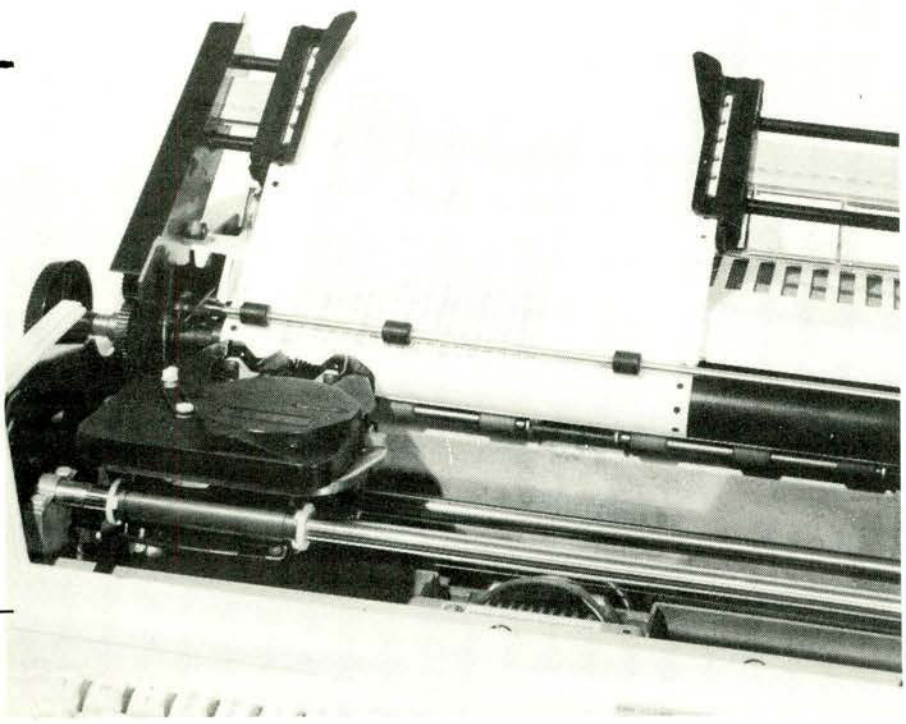


Figure 2. The circuit diagram of the interface card. Part of the circuit adapts the computer to RS 232 standards and part of it establishes communication links between the computer and two cassette recorders.

directly into the (in theory balanced) inputs. Another difference with the KIM is that the input signal is not attenuated by a factor of 10 until it reaches pin 2. All these improvements make the equipment so easy to work with that you'd have to be very careless to make a mistake when inputting data. Look out for 'drop outs' on the tape, dirty or maladjusted recording heads, etc. We happened to discover once – after a perfect data entry – that the ground connection between the computer and the recorder was missing!
Only pin 2 of IC6 (a kind of differential amplifier input) is used as an asymmetrical control. The system is connected to J1 via C1 which has a much lower value than the KIM in order to filter out the maximum of frequencies below 2400 Hz. Resistor R37 is needed in case the loudspeaker output (or headphone output) of a cassette recorder is employed.
In the absence of an input signal the VCO frequency is set by C3, R25 and P1. It is very important to preset P1 correctly, as this largely determines the quality and reliability of the data reading. Various methods will be stipulated for the correct adjustment of P1 in a later article.
Pin 7 of IC6 forms the output of the PLL. This supplies the control voltage, which was considered previously, to make sure the VCO frequency is proportional to the input signal. This pin is connected to +12 V by way of C9. Together with a 3K6 resistor inside the IC the capacitor constitutes the loop filter. This enables the PLL to react well to changes in frequency at the output without getting 'over emotional' (overshoot). By way of the ladder filter consisting of R26 . . . R28 and C5 . . . C7 the output of the PLL is connected to the negative input of the comparator IC7. Its positive input is connected to a fixed direct voltage produced by IC6 (at pin 6) via R29.

The values of the components in the ladder filter depend on the speed at which the 3.6 kHz and 2.4 kHz frequencies succeed each other. This in turn is determined by the speed with which the bits are written on tape and then read from it. This is know as the baud rate, the number of bits transmitted or received per second, and is 800 (baud) for both the Junior cassette hardware and software.
As you will remember, increasing the VCO frequency to 3.6 kHz caused the output voltage at pin 7 to drop and decreasing it to 2.4 kHz caused the voltage level to rise. Thus, since the filtered output of IC6 at the inverting input of IC7 is either higher or lower than the DC level at its non inverting input, this can be expressed as 'high' (+5 V, via R31) when the input frequency is 3.6 kHz and 'low' (practically grounded) when the input frequency is 2.4 kHz. And that is exactly what is required. The output voltage of IC7 is inverted by way of N38 ('high' goes 'low' and vice versa) and is applied to PB7.
N.B. There is a limit to the speed at which the PLL can react to a change in input frequency. Consequently, the output of IC7 will not turn over in one go from logic '1' to '0' or vice versa, but will fluctuate between the two levels before eventually making a definite choice. This is called PLL jitter and resembles contact bounce when keys are depressed. Don't worry, subroutine RDTAPE's software knows how to handle such behaviour.

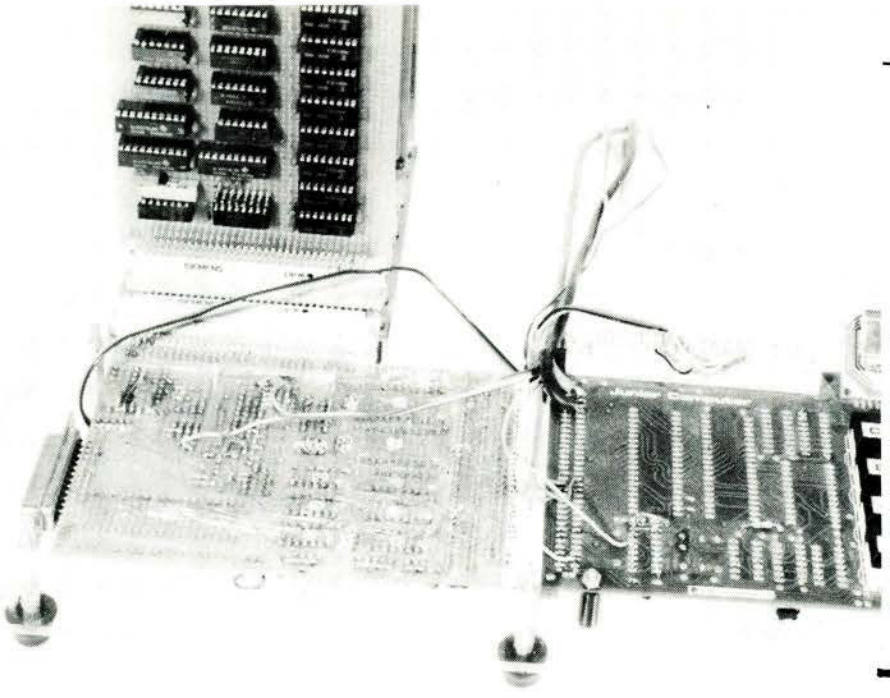


Connecting peripherals by means of the RS 232C

The small circuit at the top left-hand corner of figure 2 is surprisingly straightforward, considering the enormous possibilities that it has to offer. For it enables highly complex peripheral equipment to be connected to the Junior Computer. It consists of a data transmitter and a data receiver with port PA7 acting as an input and PBØ as an output. The receiver is situated around T1 and a few other components. T1 converts logic ones at the input (top of R5) into logic zeros and vice versa.
If we suppose that input 6 of N37 is connected to +5 V by way of R8 and forget about wire link P-Q for the moment, N37 will invert the input level received from PBØ and its output will control the comparator IC8 by way of R10. A comparison is made with the

voltage at the junction R11/R12/C12. If the output of N37 is logic 1, the output of IC8 will be about +12 V and −12 V when N37 is logic Ø. Again, it can be seen that the output signal of PBØ is inverted irrespective of whether the logic levels are adapted to any particular voltage (+ and −12 V) or not. The data input (via R5) and output (IC8) are connected to a standardised 25 pin D connector, the RS 232 connector. This number refers to the universal standard that has been established for data communications. Each byte is preceded by a start bit to indicate that it has started to be transmitted. The bytes are coded according to the ASCII code.

The RS 232 standard determines the two possible logic levels and their corresponding voltage values. Later on the D connector itself was included in these standard levels. A logic zero is represented by a voltage between +5 V and +15 V (RS 232C version) and a logic one by a voltage between −5 V and −15 V. In the Junior Computer this is about + and −12 V respectively. In other words, a low voltage corresponds to a 'high' logic level and a high voltage to a 'low' level. This is known as negative logic. By inverting the logic levels in the interface twice both during transmission and reception, however, the computer does not have to bother with such subtle distinctions. The D connector features a number of pins that are interconnected. These links may be changed to suit the peripheral equipment. The first suitable device that springs to mind is the Elekterminal, a video terminal with an ASCII keyboard which was originally designed for the SC/MP, but which is equally suitable here. The system program PRINTER MONITOR is based on using the ASCII keyboard as an input (through various key commands) and the actual Elekterminal or a suitable printer (*not* – at least not for the moment – the metal foil printer published several months ago in Elektor) as an output (display).



Now all the hardware involved in the extensions has been discussed, apart from the 'revised' Junior Computer power supply and a few other modifications which need to be made to the main board. These will be dealt with in the article on the constructional details (in next month's issue) and, of course, in Book 3 which should be available soon.

Table 1

The contents of the PROM IC7. This can be programmed by the reader with the aid of circuit no. 97 in the Summer Circuits' Issue '80 or purchased ready programmed from Technomatic Ltd. Various pin compatible PROMs are suitable, such as the 74S188, but these cannot be programmed using the above circuit. ◀