64K DRAM BOARD

Mucking around in memory? Seeking space? Look no further, here's a bounty (no connection with those distracting TV ads) of bits, rapacious in real-estate, for your 6502 or 6800 system to gorge itself on. Design and development by Bob Campbell.

Most microcomputer users find out fairly quickly that there is no such thing as too much memory. But even today with memory as cheap as it is, many systems are on sale with less, often considerably less than the 64K that most eight-bit microprocessors are capable of addressing.

The independent suppliers are usually very quick to provide units to fill this gap, but one system not well covered in this respect is the Tangerine Micron/Microtan 65. Until recently, there was only the TANRAM, but now there is the CMOS alternative. However, despite advantages in power consumption and battery back-up, the CMOS unit, like the TANRAM, is large and fairly expensive. More than one board is required to provide the maximum possible memory.

The approach here is to use the highest density dynamic RAM chips readily available and allow the user to access all of it except where it would clash with essential EPROM, I/O or CPU board RAM. This leads to an extremely flexible and cost effective system. Although specifically designed for the Microtan 65 computer together with either a disc system or TUG's Eprom Storage Card (the MOS Disc concept) the design retains enough flexibility to accommodate almost any desired configuration of computer and operating system, the only prerequisite is a 6502 or 6800 CPU.

Design

The board uses the latest 64K by 1 bit dynamic RAM chips, TMS 4164-15. These are decoded into 64 1K blocks, with all but four of the blocks used in its standard configuration. Making almost 6¼% of the RAM effectively redundant may at first sight seem a little extravagant, however even allowing for this the cost per K is less than £1.00. If one adds the other savings on hardware, sockets, power supply requirements board space etc., the 64K chip route stands out above all the other alternatives.

The heart of the system is the 74LS608 **memory cycle controller (MCC).** This chip generates all the signals the RAM requires to perform the two types of cycles necessary for proper operation. The MCC generates these signals from the CPU's clocks $\phi 1$ and $\phi 2$ together with the decoded signal RE, RAM enable. It is important not to confuse this signal with the Tanbus signal RAME. The only signals used from the bus are the address and data lines together with R/$\overline{W}$, $\phi 1$ and $\phi 2$, and because of this and the use of a PROM address decoder, this board is very flexible in design and easily adapted to suit other systems.

Dynamic RAMs

The two great advantages of dynamic RAM are its extremely low power consumption and its packing density. This is achieved by the design of the actual memory element which is in fact a very small capacitor. The logic level stored being defined by the presence or absence of a charge on that capacitor. Because all capacitors have a finite leakage, the charge on the capacitors must be periodically topped up. This procedure is called *refreshing* and is accomplished by performing what is known as a $\overline{RAS}$ only refresh cycle.

This $\overline{RAS}$ only refresh cycle consists of first setting up an eight bit address at the input latches and strobing $\overline{RAS}$ low, while maintaining $\overline{CAS}$ high. The complete chip is refreshed when all 256 row addresses have been treated similarly. Data retention is assured if all these 256 cycles are completed at least once every 4 msec.

Apart from the necessity to refresh every 4 msec there is one other penalty to pay for the 16 pin packing density and that is the multiplexed address bus. Figure 1 shows the internal architecture of the 4164.

To address every memory element within the IC, 16 address bits must be applied. These are separated into the row address and the column address, each latched onto the multiplexed address bus

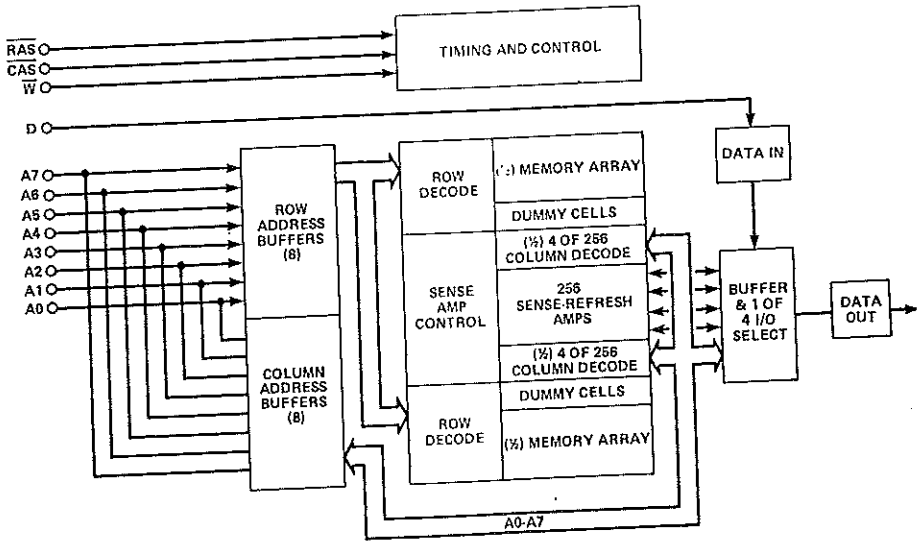


Fig.1 Internal architecture of the TMS4164 DRAMs used in the project.

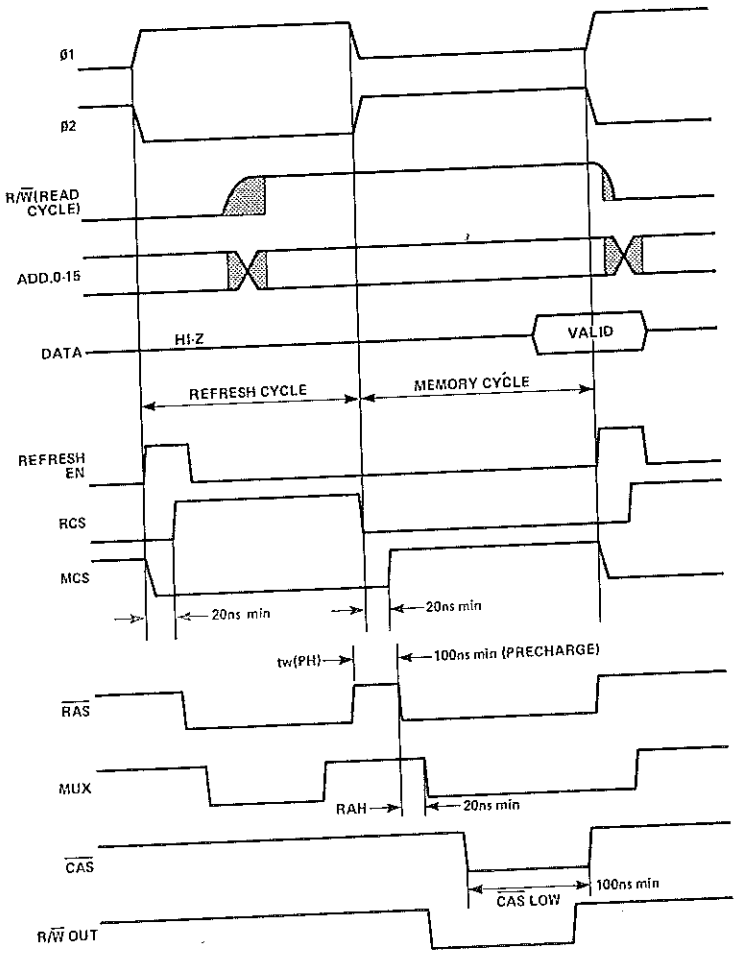


Fig. 2 Processor and memory cycle controller timing.

upon the appropriate signal $\overline{RAS}$ or $\overline{CAS}$.

In full, the memory cycle consists of five stages. Firstly the row address is set up on the eight address pins and $\overline{RAS}$ pulled low. Then the address multiplexers are switched placing the other eight bits, the column address, onto the address pins and $\overline{CAS}$ pulsed low. This last operation enables the chip and, depending upon status of the R/$\overline{W}$ line, enables the input or output buffers, thus completing a read or write cycle.

There are two other possible types of cycle, the *page mode read/write* and the *read modify write cycles*. However since neither of these apply to the 6502 or 6800 type of processor it is not necessary to consider them further here.

It is important to note that the 6502 operates in what is known as the **early write cycle** where the R/$\overline{W}$ line is set up long before $\overline{CAS}$ goes low. This enables the data in (D) and data out (Q) pins to be connected together and thus have a common data bus. Obviously the sequence and timing of the two cycles, refresh and memory, is extremely important. The $\overline{RAS}$ only refresh cycle is particularly significant for two reasons: firstly, it is necessary to perform it regularly (256 times every 4 msec), and secondly, it is effectively a dead cycle, when the CPU cannot access memory.

Refresh cycles can be carried out in either *burst mode* or *hidden transparent mode*. Burst refresh is a technique where all the memory elements are refreshed consecutively whilst the processor is held in a wait or halted state. This dead time is called the **refresh overhead**, which, more accurately, is defined as the ratio of the time taken to refresh all the memory elements and the maximum refresh interval. In well-designed systems with the 4 msec 64K rams this overhead can be as low as 2%. As the circuitry needed to maintain this type of refresh system is complex it is not commonly used outside the realms of very fast microcomputors, minis and mainframe systems.

The other technique, hidden refresh, is the more commonly used. This technique relies upon the fact that the CPU will always have a period within any instruction or machine cycle when it will not access the system bus, and one refresh cycle can be accomplished during this period. Thus after a maximum of 256 instruction cycles all the memory elements will have been serviced. This technique has the great advantage of a zero refresh overhead rate and is totally transparent to the CPU and thus the user.

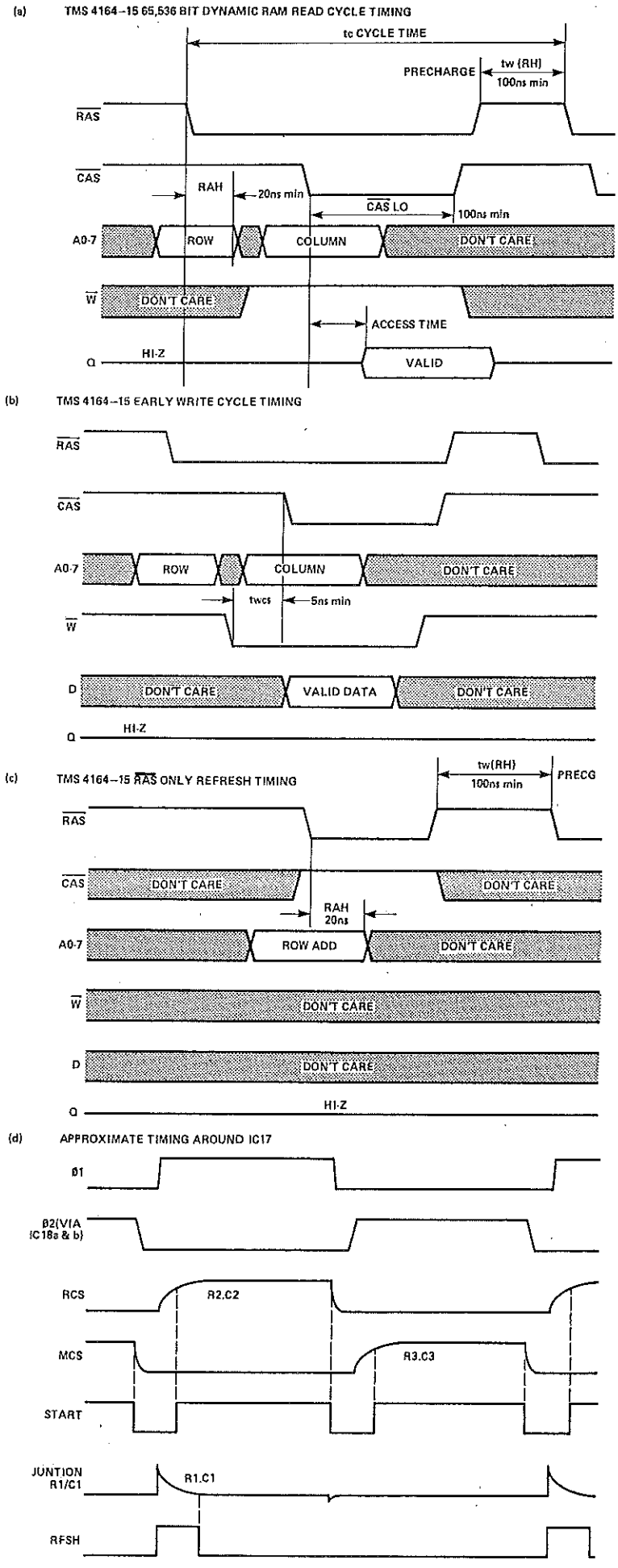
In this design the two cycles, refresh and memory, are sequenced by the main CPU clocks $\phi 1$ and $\phi 2$. While $\phi 1$ is high, the CPU sets up the R/$\overline{W}$ and address lines, the rising edge of $\phi 2$ signifying a valid memory address. This edge of $\phi 2$ is normally used to enable the address and data buffers. Thus while $\phi 1$ is high, the CPU is normally isolated from the system bus, and the refresh cycle can be accomplished during this period. In addition by using $\phi 1$ to clock the eight bit refresh row address counter all 256 row addresses can be refreshed sequentially. Figure 3 shows exactly the relationship and timing of these events.

PROM Program Design

The memory map of the RAM board is controlled directly by the TBP24S10 PROM, which acts as a complex address decoder. Before programming the PROM, the desired memory map must be established. The minimum requirement for most systems will be the system monitor, the I/O area and unless there is a serial VDU as the screen, some screen memory. Some systems use a relocatable area of memory for the screen RAM, the video controller accessing the system bus directly. If the target system is of this type then no provision should be made for the screen RAM in the PROM program. Remember the overriding factor when designing the memory map is that there must not be two components within the system which have the same address. Taking the standard configuration of the Microtan as our worked example, the minimum memory map is as shown in Fig 4.

Once you've determined the memory map(s) required, the upper six address lines should be written out bit fashion (bit by bit . . .?). Each bit corresponds to a PROM address bit; however because of the PCB board layout, the one-to-one cor-

Fig. 3 Memory timing for various operations and approximate timing round the MCC.



respondance is not in numerical order.

In addition, by using the two extra PROM address lines A7 and A8, there is the facility to have up to four programs and therefore four memory maps resident on the board at one time, selectable by means of the DIL switch SW1. Using the two tables 1 and 2 it is possible to calculate all the PROM addresses which are required to be 'blown'.

Remember that PROMs are not erasable, once a memory location is altered from the "all 1's" condition, it cannot be reversed. There is however an escape route if a mistake is made during programming. The program is created by blowing only the operative bits within the data word from a 1 to a 0. In this design, only one of the four bits available is used (bit 4). If an error is made during programming, then it is possible to use an alternative bit by breaking the PCB track at pin 9 IC15, installing a link to either pin 10, 11 or 12 (bits 1-3 inc.) and reprogramming the PROM using the appropriate data word. (Alternatively, this would make it possible to hold a total of 16 memory maps in the PROM).

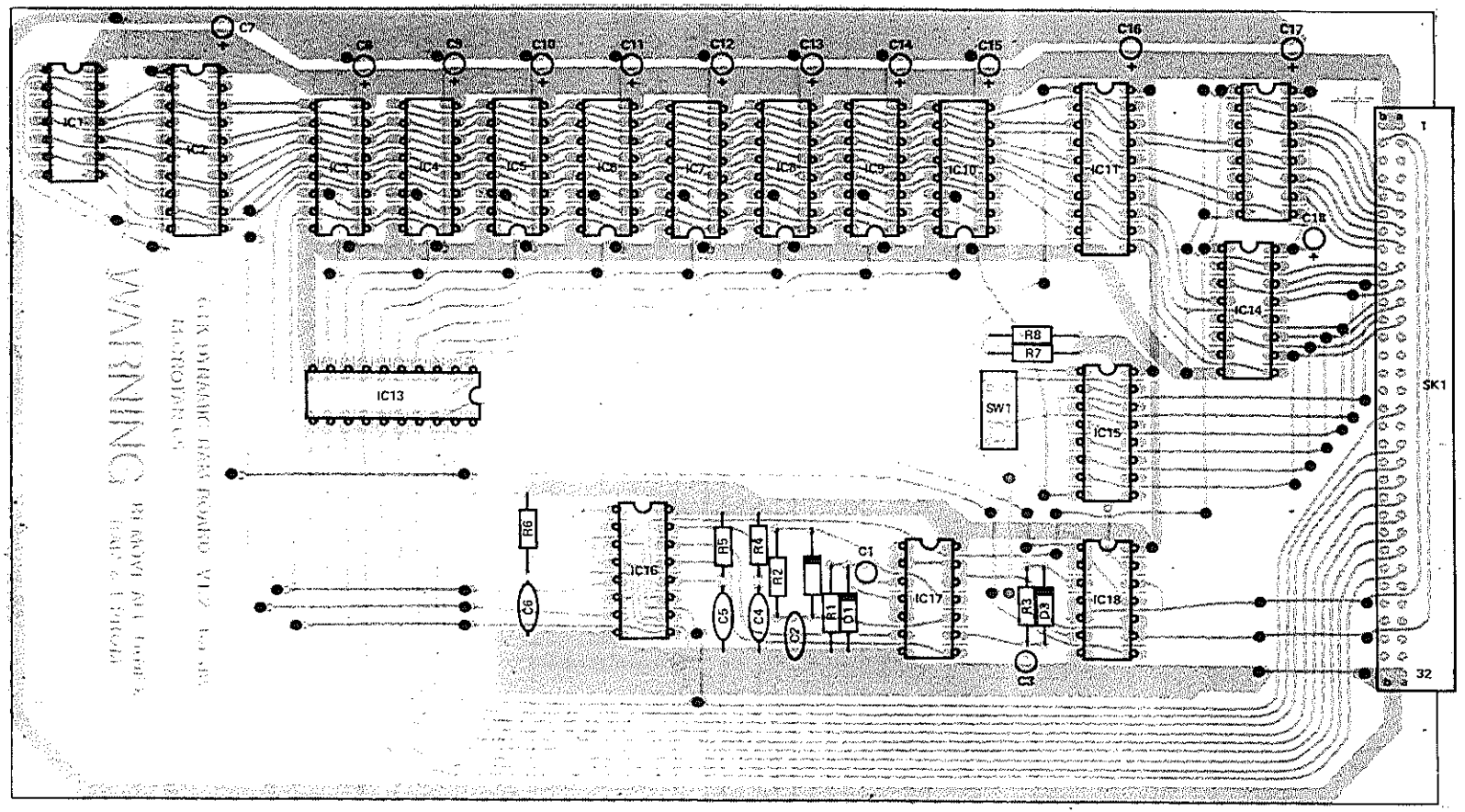
It is beyond the scope of this article to describe the methods for actually programming the PROM, suffice it to say that the amount of programming by the nature of it's use, is small, so it would be feasible to use the switchbox type of programmer.

Construction and Setting Up

The construction of the board is very straightforward, particularly if the PCB design presented here is followed exactly: there are, after all, only 18 ICs. The PCB is a double-sided design but to keep costs down it doesn't use plated-through holes. To make the necessary interconnections, track pins or short lengths of wire must be soldered between the two in the positions marked on the overlay diagram with a black dot. These pins must be soldered in first,

AREA	HEX ADD.	SIZE
A) TANBUG	FFFF F800	2K
RAM	F7FF C000	14K
B) I/O	BFFF BC00	1K
RAM	BBFF 0400	46K
C) CPU BOARD RAM	03FF 0000	1K

Fig. 4 Minimum memory map for the Microtan.



PARTS LIST

RESISTORS		C3	68p ceramic plate 2% or better, or silvered mica 1%	**SEMICONDUCTORS**	
R1, 2, 3	1k0 hystab metal film 1%	C4	220p ceramic plate 2% or better, or silvered mica 1%	IC1	74LS393
R4, 6	4k3 hystab metal film 1%	C5	68p ceramic plate 2% or better, or silvered mica 1%	IC2, 11	74LS244
R5	1k1 hystab metal film 1%	C6	120p ceramic plate 2% or better, or silver mica 1%	IC3-10	TMS4164-15p
R7, 8	1k0 carbon ¼W 5%	C7, C17	10u tant or low leakage solid aluminium electrolitic	IC12, 13	74LS157
CAPACITORS		C8-C16, C18	1u tantalum	IC14	74LS245
C1	100p ceramic plate 2% or better, or silvered mica 1%			IC15	TBP24S10
C2	150p ceramic plate 2% or better, or silvered mica 1%			IC16	74LS608
				IC17	74LS32
				IC18	74LS00
				D1, 2, 3	1N4148

MISCELANEOUS

DIN 41612 64 way double-sided connector; DIL 2 pole on/off switch; DIL sockets: 3 off 20 pin, 12 off 16 pin, 3 off 14 pin; PCB.

BUYLINES

One or two not so easy to obtain items here. The TBP24S10 PROM and 74LS608 memory cycle controller chip were tracked down to Farnell Electronic Components Ltd, Canal Road, Leeds LS12 2TU. At £2.42 and £6.44 respectively plus 55p p&p plus VAT these shouldn't break the bank. The 1% metal film resistors are available from Rapid, Cricklewood, Watford and many others. The specified memories and other TTL devices are advertised by Midwich Computer Company Ltd. And, in case you hadn't guessed, the PCB will be available through our own service (see page 77).

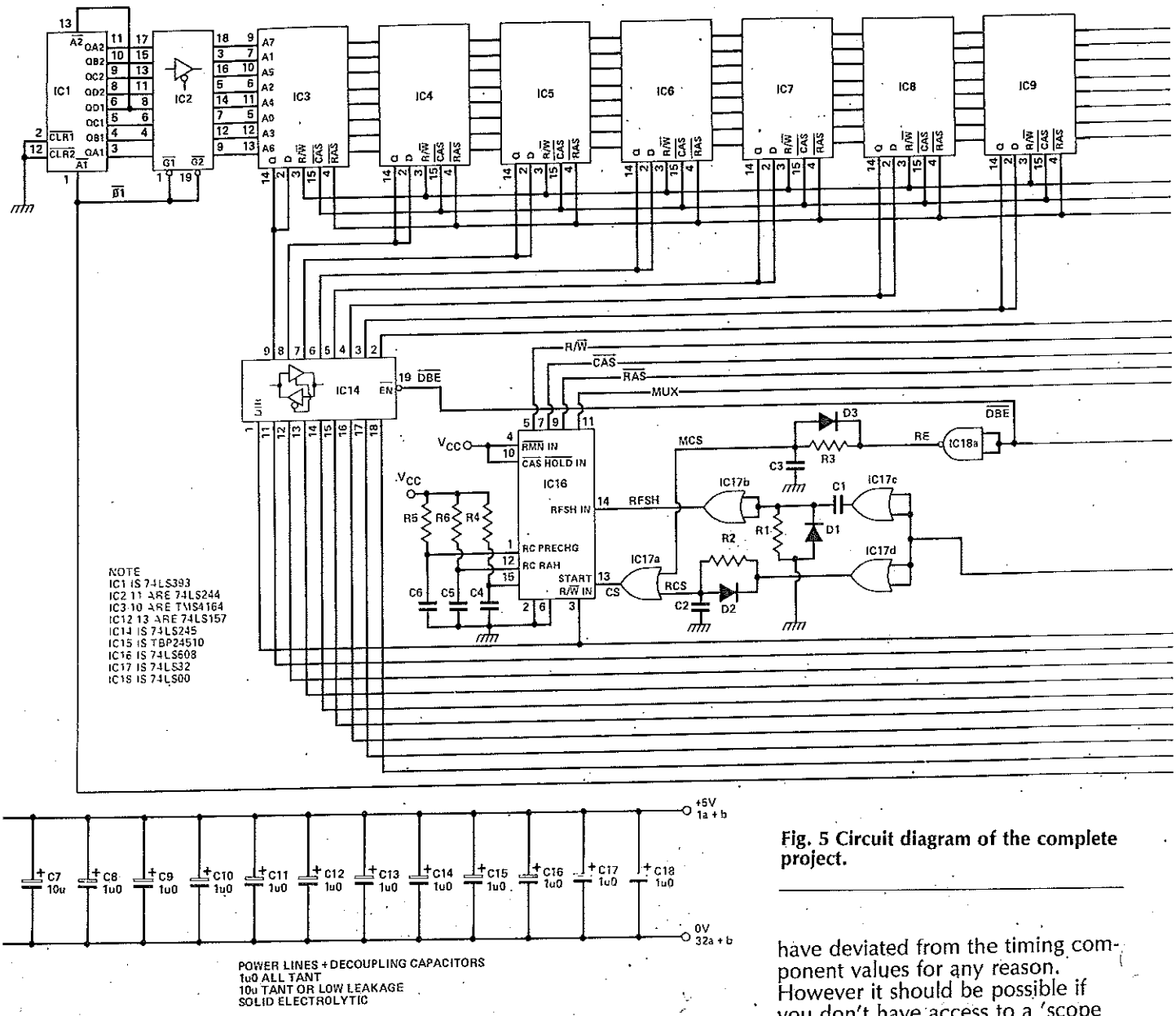


Fig. 5 Circuit diagram of the complete project.

prior to any other components, as there are some beneath the DIL sockets; I advise checking the continuity of each one thoroughly, as mistakes are difficult to rectify later. The remainder of the soldered components can be assembled in almost any order, but I've found that it pays to be systematic and to follow a list, checking off each component as it is soldered in.

All the usual checks should be carried out before the ICs are inserted into their sockets. Particular attention should be given to avoiding solder bridges in the daisy-chained RAM area of the board.

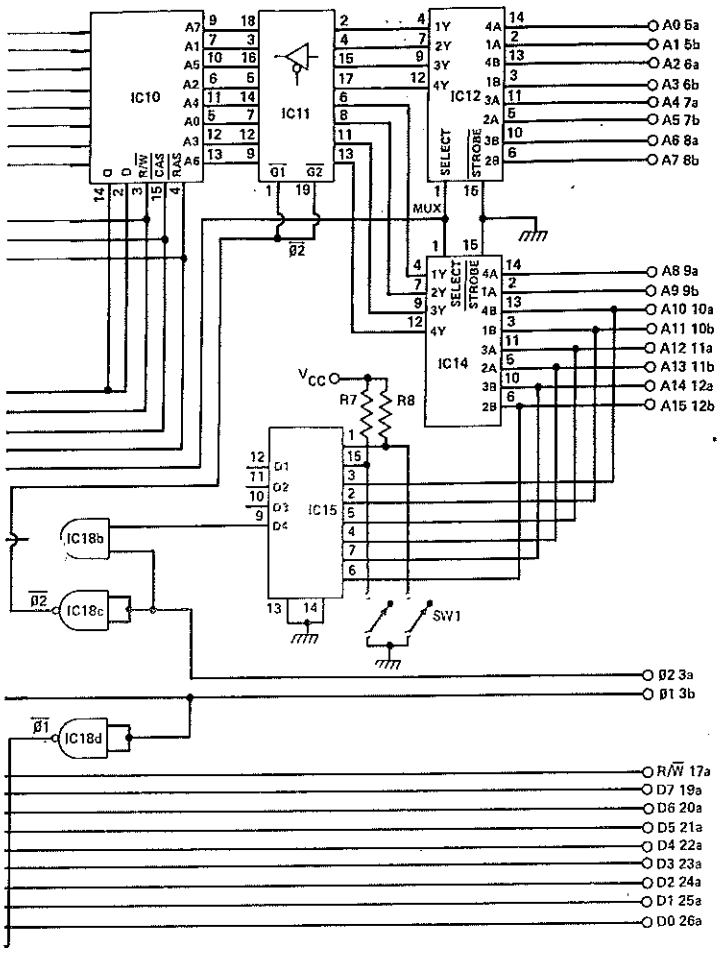
It is useful to insert the chips in three stages and perform some functional checks on the system at each stage. The first of these stages is to insert the PROM and all the TTL, with the exception of the 74LS608 (IC16) and the 74LS245 data bus buffer (IC14). Now powering up the board on the bus can be performed with all the Tanex RAM and EPROM still resident without the risk of any memory conflict occurring. This procedure will allow you to check the following items with the system running.

A dual beam oscilloscope is really desirable particularly if you have deviated from the timing component values for any reason. However it should be possible if you don't have access to a 'scope to use a good logic probe to check that all the apropriate signals are present.

The most relevant signals to check first are $\phi 1$, $\phi 2$ and their complements $\overline{\phi 1}$, $\overline{\phi 2}$. RE and $\overline{DBE}$ should be active only when a valid address within your programmed memory map is accessed. Next check that the two address buffers, IC2 and IC11, are switching correct-

SYSTEM ADDRESS						HEX ADD.	COMMENTS
A15	A14	A13	A12	A11	A10		
1	1	1	1	1	1	FFFF	TANBUG
1	1	1	1	1	0	F800	
1	0	1	1	1	1	BFFF	I/O
1	0	1	1	1	1	BC00	
0	0	0	0	0	0	03FF	CPU BOARD RAM
0	0	0	0	0	0	0000	

Table 1 Revised system memory map.



HOW IT WORKS

As so much detail has been given in the general section, this 'How It Works' is going to be fairly brief. During $\phi 1$ high the main bus buffers IC11 and IC13 are disabled, removing the RAM from the system bus. The refresh row address counter IC1 is connected directly to the RAM ICs (IC3-10) via the enabled buffer IC2. The rising edge of $\phi 1$ is first buffered by two OR gates and then, via the pulse generator network D1, C1, R1, IC17, it applies a pulse to the REFRESH ENABLE pin (14) of the memory cycle controller IC16. The same rising edge is delayed by D2, C2, R2, IC17, before reaching the CYCLE START pin 13 of IC16. This delay is necessary to satisfy the refresh hold time of the memory cycle controller, and must be maintained at 20 ns minimum. The MCC then responds by pulsing $\overline{RAS}$ low for a period of time determined by the RC network at pin 12, the row address hold time. The rising edge of $\overline{RAS}$ is the end of the refresh cycle.

The memory cycle starts with the rising edge of $\phi 2$ (falling edge of $\phi 1$) at which point the address bus buffer is enabled directly by $\phi 2$ and assuming the address is within the memory map, the PROM output D4 is already high. This output combined with $\phi 2$ produces via IC18 two signals $\overline{DBE}$ and RE.

$\overline{DBE}$ enables the data buffer IC13; RE delayed via D3, C3, R3, IC17 is fed to the CYCLE START input of IC16 the memory cycle controller. This last event causes the MCC to start the actual memory access cycle. The $\overline{RAS}$ output (pin 7) goes low then, after the programmed RAH time, the $R/\overline{W}$ line is allowed to pass through and the MUX output then goes low switching over the address multiplexers IC12 and IC14 to the column address. $\overline{CAS}$ then goes low for a period of time $\overline{CAS}$ LO. All three outputs $\overline{RAS}$, $\overline{CAS}$, MUX then go high. This point should coincide with the falling edge of $\phi 2$ when the data from or to the RAM is latched by either the CPU or the memory depending on the status of the $R/\overline{W}$ line.

The next refresh cycle then occurs on the rising edge of $\phi 1$ and so the system carries on until the power is removed.

Table 2 Programming sheet for the PROM.

64K DYNAMIC RAM BOARD PROM PROGRAMMING SHEET . . . OF . . . 4									
SYSTEM ADDRESS	SW1	SW2	11	10	13	14	15	12	HEX PROM ADD.
PROM ADDRESS	8	7	6	5	4	3	2	1	BLOW TO 07
	0	0	0	0	0	0	0	0	00
	0	0	1	1	1	1	1	1	3F
	0	0	1	0	1	1	1	1	2F
	0	0	1	1	1	0	1	1	3B

ly, ie exactly 180° out of phase with each other, and that the refresh address counter IC1 is functioning correctly as an eight-bit counter.

The final check at this stage is to measure the pulse delay and shaper circuits formed by the diode/resistor networks and IC17. The three signals $\overline{RAS}$ cycle start (RCS), memory cycle start (MCS) and refresh (RFSH) should all correspond to the timing diagram in Figure 3. Any deviation should be adjusted by altering the value of the capacitor and/or resistor within the relevant RC network. However if the stated tolerances of the components are adhered to there should be no problems.

Having completed all the checks and adjustments so far the next stage is to insert the 74LS608 memory cycle controller, which should produce the necessary signals $\overline{RAS}$, $\overline{CAS}$, MUX and $R/\overline{W}$. These four main signals should be checked against the timing diagrams in Figures 2 and 3. The important factors are the relationships between cycle start, CS, and $\overline{RAS}$, MUX, $\overline{CAS}$ sequence and the $\overline{RAS}$ refresh cycle. The row address hold time RAH, CAS low and the precharge time are the major controlling times and are all programmable via the three RC networks on the 74LS608. Under standard conditions with the 750KHz Microtan system clock these times have quite a large latitude. However with faster clock rates the times become proportionally more critical. All these times can be calculated from the memory data sheets.

One fault which may occur at this point has the symptoms $\overline{RAS}$ permanently low, $\overline{CAS}$, MUX and $R/\overline{W}$ permanently high. If this situation exists try shorting **very briefly** pin 12 to ground. If the controller then starts to function correctly then the 74LS608 is at fault. I understand from Texas that on a number of the older batches of chips there is a fault with the power-on-reset circuit, newer batches, I am assured, are all O.K.

Having checked that all the relevant signals are present at the RAM chip sockets, the RAM chips themselves can now be inserted. **Power down first.** These are very static sensitive so take all the usual precautions, they are also **upside down** in relation to the other ICs on the board. Be warned that if they

are inserted with pin 1 to the upper edge of the board they will be irrevocably damaged, and at £4 each a mistake could be very expensive. Finally insert the data buffer IC13.

With construction and testing completed there is still one task to finish before the board is inserted back into the rack and powered up. **Remove all Tanex RAM and EPROM**, and all other memory map conflicts, for example the hires graphics board, failure to do this will probably destroy ALL the memory components in the system.

After powering up the board in the now "minimised" system, unless you've chosen to create a memory map option which retains the Tanex EPROM, your system will be running in Tanbug or TUG bug. The quickest way to check the RAM from here is to boot up Basic and XBUG from disc or ESC and let it do the check. 47103 BYTES FREE should appear as the message header. Note some difficulties may be experienced because the F7F7 error jump will not exist immediately. This will show up only if an error occurs during the boot up procedure e.g. miss keying; simply RESET and start again to recover. Assuming this initial check appears to be OK then a more comprehensive memory test routine should be performed; the one published in the November 1981 issue of Computing Today is most suitable. However it should be noted that these types of test do not pick out the periodic bit drop out and only extensive usage in BASIC or similar will show up this problem.

Other Systems

The board relies only upon signals derived directly from the CPU i.e. $\phi 1$, $\phi 2$, R/$\overline{W}$ and the address and data buses. Since all these signals will be present in any 6502-6800 system, conversion is relatively simple. The only component that needs to be altered in any way is the PROM which does all the decoding. The essential considerations are those concerning the design of the memory map and, in particular, possible address conflicts. Remember no two components, be they RAM or I/O should have the same address! A suggestion for those with a Microtan but no discs or ESC is to leave the XBUG EPROM resident (F000-F7FF) and use the tape routines instead.

Those who design their own PCB should take care to heed the memory manufacturer's recommendations on decoupling and PCB layout around those chips. Particular attention should be given to the ground and power supply lines, which effectively surround each chip; the arrangement of interlocked fingers on the typical breadboard is definitely out.

Similarly the decoupling of the TTL chips should be comprehensive enough to avoid too much power supply noise, a major culprit of periodic bit drop out. Lastly, the 74LS608 MCC gets hot, but since the lead-frame is directly coupled with both the substrate and the ground pin, a large area of copper around pin 8 should alleviate the problem and improve reliability.

With regards to systems employing faster clock rates than 1MHz, as long as the $\overline{RAH}$, PRECHARGE $\overline{CAS}$ low times and the refresh hold time for the MCC are satisfied (calculating them from the manufacturer's data sheets), no significant problems should occur. **ETI**

EXPERIMENTERS' 64K DRAM CARD

Gnosis: knowledge of spiritual mysteries (Concise Oxford Dictionary); Gnos-ex: expandable memory system (ETI Dictionary). Phil Walker tries again for the obscure pun of the year award.

The ETI GNOS-EX is the expandable, flexible dynamic memory system for the keen experimenter. Using the 4416 16K X 4 dynamic memory devices, the system can be populated and configured for from 16K to 64K blocks with the capability of deleting or including memory in 1K blocks.

Last time we published a DRAM card for the 6502, we used the 74LS608 memory controller. Since then, we've discovered there are problems with this device (don't worry if you're trying to get that board going — we're working on a fix!). So this system was designed to do without any very special control devices and rely, so far as possible, on absolutely standard ICs which will be (we hope!) readily available for some time to come and cheap. In the final design the most unusual devices are the 4416 memories and the PROM. This latter device is not actually absolutely essential for the operation of the project and could be replaced by suitable logic.

The layout of the PCB is intended to be such that it will plug into a Microtan system bus, although at the time of writing this it has not been tested. The original development work was carried out on the author's Ohio Superboard, somewhat modified with the processor running at 1.25MHz.

The Circuit

The basic ideas behind this project are much the same as any other which makes use of dynamic memories. There are two distinct phases of operation; the first, and most important as far as the user is concerned, is the reading or writing data, ie actually using the memory. The second is the periodical refreshing of the stored data to make sure that it is remembered correctly. Ideally, the refresh operation should not be apparant to the user, and so it must take place when the processor is not using the memory.

In the case of the 6502 microprocessor, for which this project is designed, the processor is concerned with accessing the memory for only half the time. The remaining time can be used for refreshing the memory with no effect on the processor. With the 6502 running at 1MHz there is about 500ns in which the processor will read or write data as necessary, followed by 500ns or so in which the processor is doing internal operations and not interested in the outside world; this is the time we use to do a refresh operation.

The 4416 specification requires that the whole memory be refreshed at least every 4ms. To do this, 256 different addresses must be put on the address lines and the RAS input pulsed low for a certain time for each one. All this must be done within the 4ms allowed. In this design it will be done every 256μs with a 1MHz processor clock.

The circuit consists of several elements. First, there is an address multiplexer which takes the 16 address lines from the processor and switches them to the eight address lines of the memories during the processor access cycle. Only 14 of the address lines are used, eight are latched into the memory ICs by the RAS signal and six by the CAS signal.

Second, and allied to the above, there is the refresh address counter and buffer. The eight-bit counter is incremented at the end of each refresh cycle and provides the 256 addresses necessary for the complete operation. The tri-state bus buffer puts the output from the counter on the memory address pins starting mid-way through the previous processor

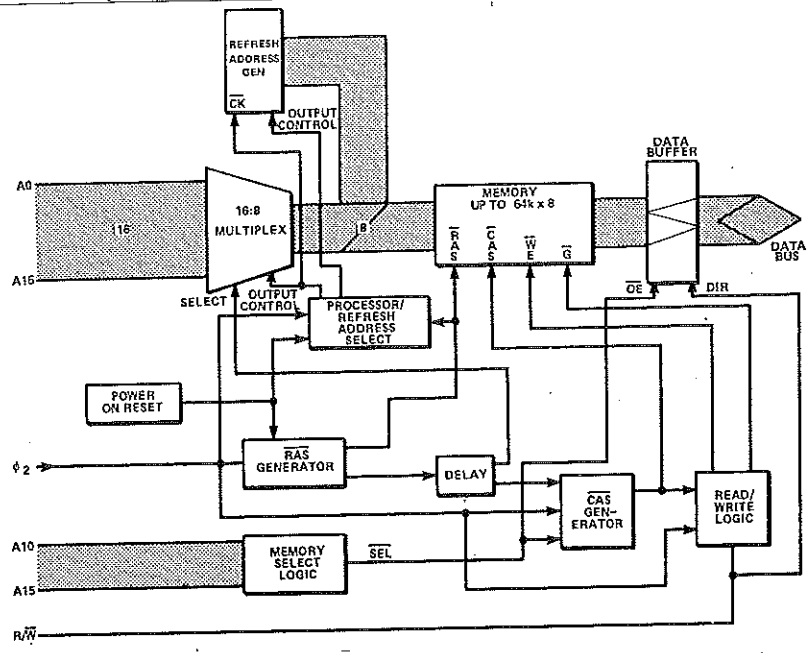


Fig. 1 Block diagram of the card

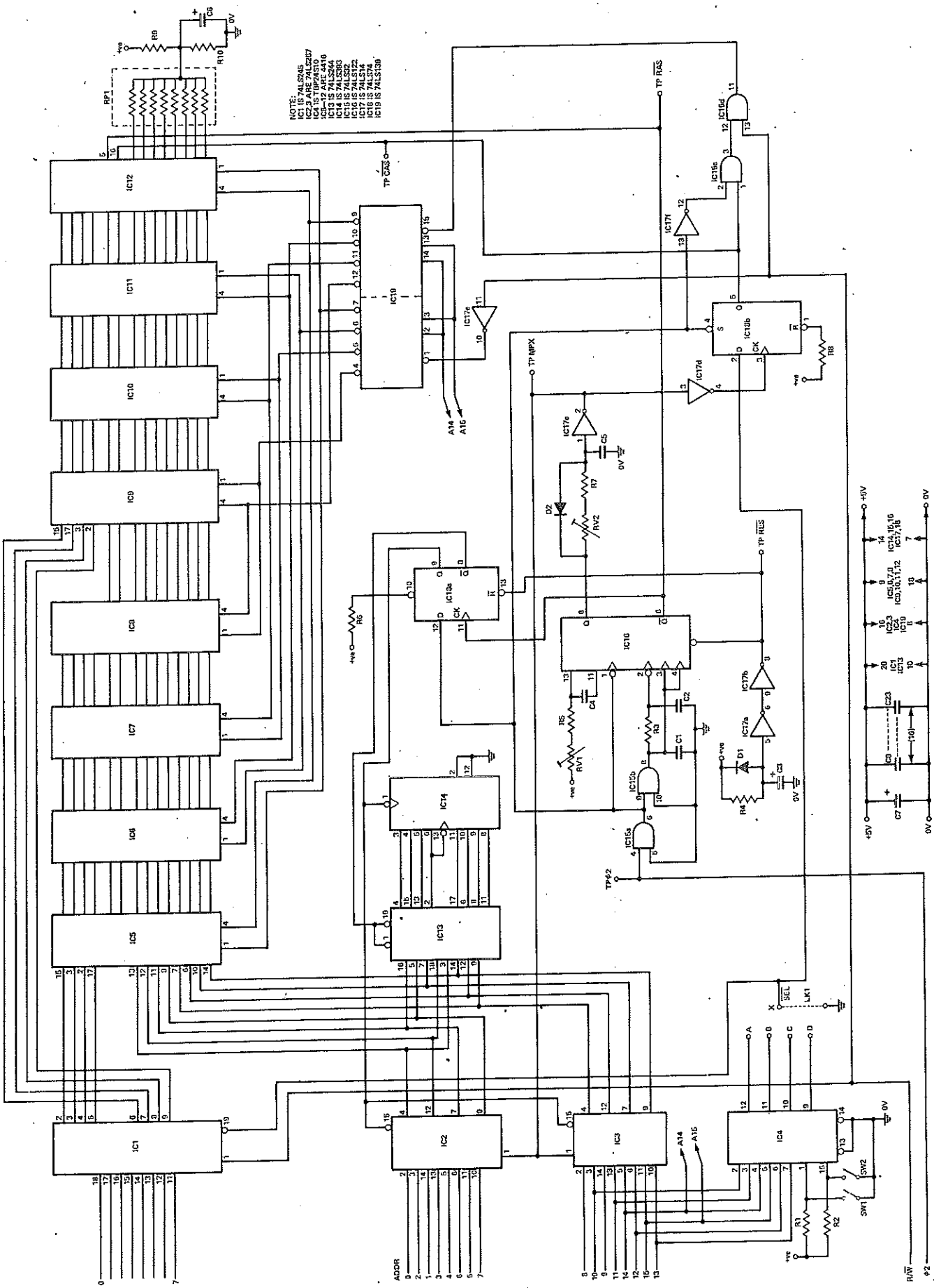


Fig. 2 Main circuit diagram

HOW IT WORKS

In this section we shall dispose of the simpler functions first. IC1 is an eight bit bi-directional data buffer. Its direction of transmission is determined by the state of the R/W line from the processor. It is enabled by the SEL signal from the address decoding PROM when the processor requires access.

The address decoder PROM is about the easiest way of providing full decoding of the address space into 1K byte blocks. As supplied by the manufacturers the TBP24S10 has all its memory cells at a logic high level. In this project this corresponds to the board not being selected. In order that the memory will respond to any block of addresses, the corresponding locations in the PROM must be blown to the low state. Since the PROM has four bits per location, three more memory maps can be blown into the device to cater for future modifications to your system and link LK1 moved to use the appropriate bit. Also, since only six of the eight address lines are used for the decode, SW1 and 2 are provided so that you can have up to four address maps for each link position.

If the blowing of a PROM is difficult for you then a 74LS151 one of eight selector IC can be used to simulate some of its operation.

In order to ensure that the circuit starts up correctly when power is applied, R4 and C3 together with IC17a and b form a power-on-reset circuit which gives a logic low for 25ms or so after the power is first applied. D1 across R4 ensures that C3 is discharged rapidly when power is removed and also protects the input circuitry of IC17a.

IC2 and 3 are quadruple two-to-one multiplexers which are used to switch the 16 processor address lines onto the eight memory address lines at the right time. 'LS257s are used here instead of the more usual 'LS157s because they have outputs which can be made high impedence by the state of a single input pin. By using this facility the processor address bus is applied to the memory at the start of the processor access cycle. At the end of the processor cycle the outputs of IC2 and 3 are turned off and IC13 outputs are turned on, allowing the state of the refresh counter IC14 to be applied to the memory address inputs in readiness for a refresh cycle.

IC13 is an eight-bit bus buffer while IC14 is a dual four-bit binary counter connected to provide the eight bit refresh address. To ensure that the outputs of IC14 are steady while the refresh cycle is in progress, IC14 is clocked as the outputs of IC13 are disabled thus the transitions of IC14 will be over long before the next refresh address is gated onto the memory inputs.

The next part of the circuit is where the main work is done. The main timing signal φ_2 from the processor goes to IC15a where it is buffered; from IC15a output it passes through IC15b where it is delayed a little and through R3 and C2 where it is delayed a little more. When φ_2 goes low the falling edge at IC16 pin 1 causes IC16 to generate a pulse at its output. IC16 is a monostable whose period is set by RV1, R5 and C4. Note that due to the delays in IC15b and R3/C2, pins 2, 3 and 4 will still be at a high level when this occurs. This satisfies the trigger conditions for the device in this case.

When, on the other hand, φ_2 goes high, pin 1 will go high first followed by pins 3 and 4 and then pin 2. The combination of pin 2 low and pins 3 and 4 going high is another valid triggering combination for IC16 and it will trigger again. This arrangement gives a pulse at both the rising and falling edges of the φ_2 input signal.

The low-going output from the Q output of IC16 is used as the RAS or row address strobe signal for the memories. Its falling edge causes the first eight bits of the address to be latched internally.

The high-going signal from the Q output of IC16 is somewhat delayed by RV2, R7 and C5 and inverted by IC17c. This signal causes the address multiplexers IC2 and 3 to apply the other eight address bits to the memory inputs ready for the next operation. The output from IC17c is inverted and, incidentally, slightly delayed by IC17d before going to the clock input of IC18b.

Normally the Q output of IC18b is set to the high state every time φ_2 is low, but if φ_2 is high and the clock input goes high, the state of the SEL signal is transferred to the Q output. If the memory board is selected, SEL will be low and IC18b Q output will go low, otherwise it will remain high.

The output from IC18b is used as the CAS or column address strobe signal for the RAMs. Its falling edge causes six more addresses to be latched into the memories, making the total up to the 14 necessary to access one of the 16384 locations.

A little while later, the monostable IC16 finishes its pulse. The delay network RV2, R7 and C5 is reset quickly via D2 ready for the next RAS cycle. As the Q output or RAS signal goes high it also clocks IC18a. This transfers the state of the φ_2 signal to its outputs which in turn control the outputs of the refresh buffer and processor multiplexers.

If φ_2 is high the next RAS cycle will be a refresh and IC13 will be enabled. If φ_2 is low the next RAS cycle will be a processor access if required and the outputs of IC2 and 3 will be enabled.

Note that the power-on-reset circuit ensures that the control logic starts up with the RAS generator disabled and the multiplexers enabled. It must be arranged that the software allows at least eight RAS only cycles of the memory before it is accessed. This would normally occur while the processor registers are being initialised.

The remaining logic is concerned with reading and writing the memories. The design of the project does not aim for minimum power consumption so all the memories are accessed during each cycle. However, to read data from one pair of them, its G input must be low. This signal is derived by simply inverting the R/W signal from the processor and passing its through IC19a. This is half of a one-of-four demultiplexer and its output driven is determined by the states of the two most significant address lines. The selected output then enables one of the four pairs of memories and allows it to output its data.

To write data into the memories, the WE input must be taken low. To accomplish this, the R/W signal is gated with the inverted φ_2 signal from IC17f and the CAS signal. This is done to ensure that it cannot occur at the wrong time. The outputs from IC15c and d then enable IC19b which routes it to the WE inputs of the pair of memories determined by the states of A14 and A15 of the address bus. This arrangement also delays the write command a little and allows a little extra time for the data to arrive from the processor.

The last components on the PCB worth a mention are the resistor pack and R9, 10 and C6. These are provided to terminate the address lines and suppress reflections of the signals travelling back along them. On this size of board, they may well not be necessary.

The final point to note is that the top address lines A15 and A14 are multiplexed onto the memory address inputs during the CAS cycle, but the memory ICs ignore them. This may prove useful if 64K by 4 bit devices become available (with eight multiplexed address pins) in the near future.

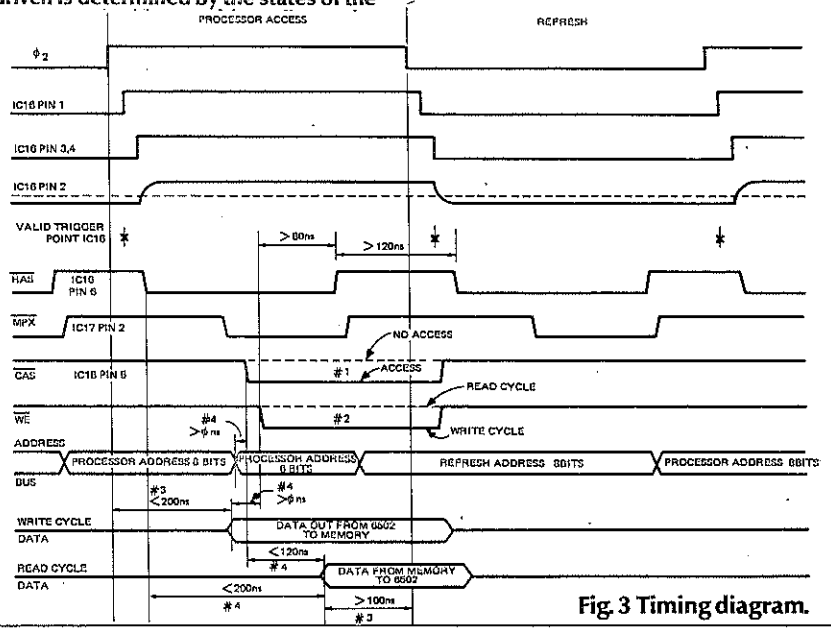
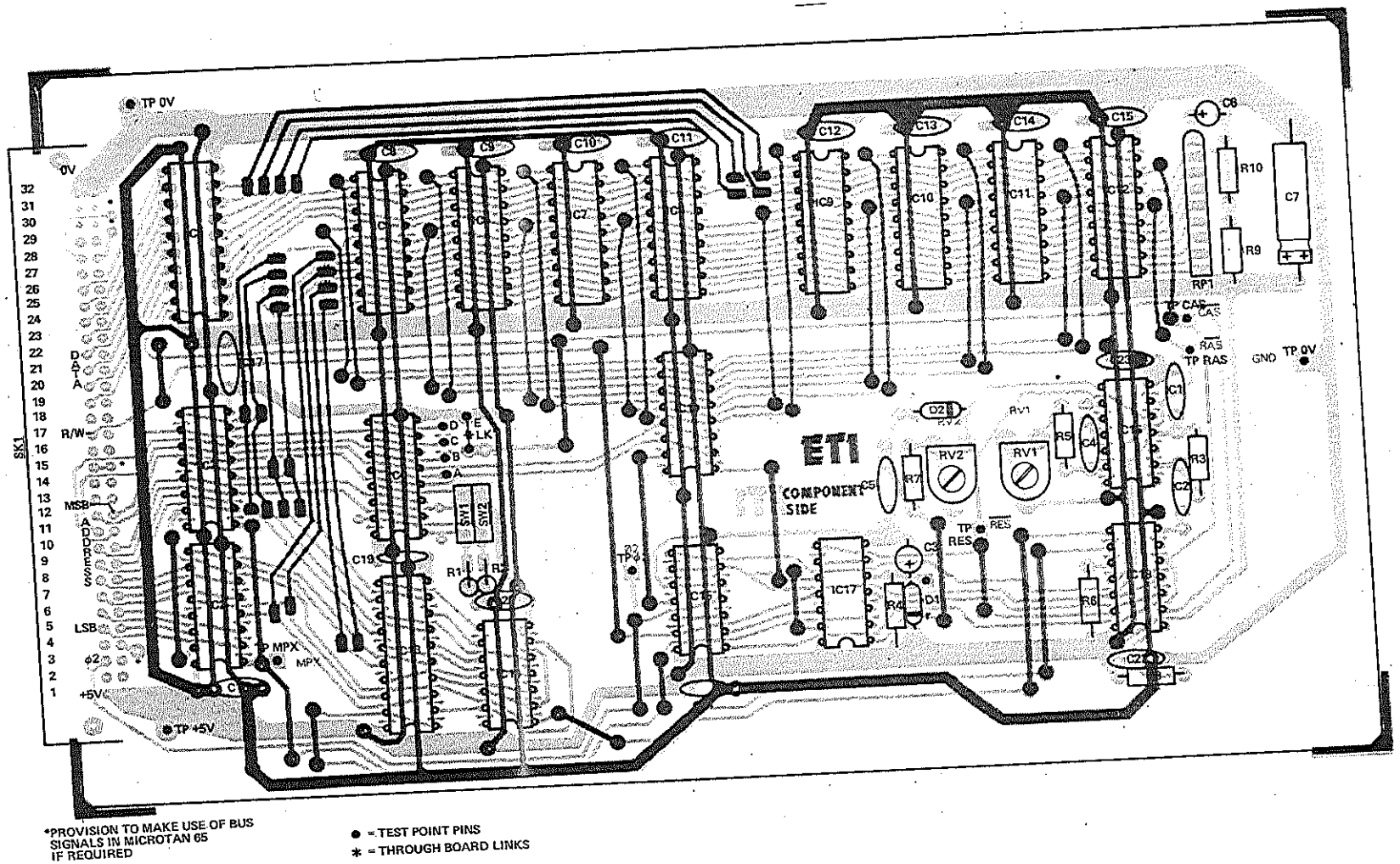


Fig. 3 Timing diagram.



*PROVISION TO MAKE USE OF BUS SIGNALS IN MICROTAN 65 IF REQUIRED

● = TEST POINT PINS
✱ = THROUGH BOARD LINKS

Fig. 4 Overlay for the PCB

PARTS LIST

RESISTORS (¼ watt 5% carbon film)

R1, 2, 6, 8	2k2
R3, 7	470R
R4	10k
R5	5k6
R9, 10	330R
RP1	SIL resistor pack 8 x 1k0
RV1	10k min. preset (horiz.)
RV2	4k7 min. preset (horiz.)

CAPACITORS

C1, 2, 5	100p 1 or 2% silver-mica
C3	10μ tantalum bead
C4	22p 1 or 2% silver-mica
C6	100μ tantalum bead
C7	100μ aluminium electrolytic axial
C8 to 15 inc.	47n (8 off) ceramic disc
C16 to 23 inc.	100n (8 off) ceramic disc

SEMICONDUCTORS

IC1	74LS245
IC2, 3	74LS257
IC4	TBP24S10 (if required — see text)
IC5, 6, 7, 8, 9, 10, 11, 12	4416 (150ns preferred)
IC13	74LS244
IC14	74LS393
IC15	74LS32
IC16	74LS122
IC17	74LS14
IC18	74LS74
IC19	74LS139
D1	1N4148
D2	OA91

MISCELLANEOUS

SW1, 2	2 section DIL switch 2x single pole
SK1	64 way a-b DIN 41612 card connector

2 x 20 pin, 8 x 18 pin, 4 x 16 pin and 5 x 14 pin DIL sockets; tinned 20swg wire for links; PCB, wire, etc.

cycle until mid-way through the refresh cycle. During this time, the outputs from the multiplexer ICs are made high impedence to avoid conflict.

The next section, and possibly the most important as far as this design is concerned, is the $\overline{RAS}$ generator. This is basically a monostable but with extra logic to make it trigger from both the rising and falling edges of the φ_2 signal. The φ_2 signal is the main timing output from the 6502 processor. The dual triggering capability enables the circuit to generate the row address strobe ($\overline{RAS}$) signal for both refresh and processor access with one device and one adjustment.

Following on from the $\overline{RAS}$ generator, there are two functional blocks, one of which determines the memory address source for the next cycle from the current state of the φ_2 signal. The other provides a short delay in which the first eight bits of the processor address are held steady before switching over to the next six bits in readiness for the $\overline{CAS}$ signal.

The delay circuit also triggers the column address strobe ($\overline{CAS}$) generator which, if all its input conditions are correct, will provide a low output signal to the end of the cycle. Note that this signal will only occur if φ_2 is high and the select logic output is low. The state of the select logic is sampled after a delay from the $\overline{RAS}$ signal falling edge; if it is low then the $\overline{CAS}$ output may go low. If, however, φ_2 is low then $\overline{CAS}$ will stay high. This occurs during a refresh cycle. If, during a processor cycle, the $\overline{CAS}$ output does go low it will remain low until the end of the cycle when φ_2 going low will force it high again. This allows data to remain available at the output of the memories until the end of the processor cycle without the use of a separate latch.

The last major section of the circuit consists of the read/write select logic. This also performs the function of selecting which pair of memory chips is accessed. Each memory device has a write enable ($\overline{WE}$) input and an output control ($\overline{G}$). The latter has the effect of allowing data to be output from the device only when this line is low. The former causes data to be written into the memory matrix when it is taken low provided that the $\overline{RAS}$ and $\overline{CAS}$ signals have been properly set up. The $\overline{WE}$ signal is allowed to be generated only if a $\overline{CAS}$ signal is present, and so will occur only in a processor cycle when the select logic output is low and the R/$\overline{W}$ is also low.

In order to keep the loading on the processor data bus low and to avoid handling problems due to static on the memory data pin, a data buffer is provided. Its direction of transmission is determined by the state of the R/$\overline{W}$ line and it is enabled when required by the SEL signal from the select logic.

The select logic on the board is intended to be a TBP24S10 256 x4 bit PROM. Only the six MSBs of the address bus are connected to the PROM and thus only 64 of

Programming The PROM

As already stated, the PROM address lines are not used in order, the connections are as follows:

Address Line	PROM Pin	PROM Pin Designation
A15	5	A0
A14	4	A3
A13	7	A2
A12	6	A1
A11	3	A4
A10	2	A2

When you come to do the programming, program one location at a time. The manufacturers do not recommend manual programming of this device, but you may get away with it. The recommended programming procedure is as follows:

1. Address the word to be programmed, apply 5 volts to V_{CC}.
2. Verify the status of a bit location by checking the output level.
3. Decrease V_{CC} to 0 volts.
4. For bit locations that do not require programming skip steps 5 through 11.
5. Increase V_{CC} to $V_{CC(pr)}$ with a minimum current capability of 250 milliamperes.
6. Apply $V_{S(pr)}$ to all the S, E or G inputs. $I_I \leqslant 25$ milliamperes. Active-high enables may be left high.
7. Connect all outputs, except the one to be programmed, to V_{IL}. Only one bit is to be programmed at a time.
8. Apply the output programming pulse for 20 microseconds. Minimum current capability of the programming supply should be 250 milliamperes.
9. After terminating the output pulse, disconnect all outputs from V_{IL} conditions.
10. Reduce the voltage at S, E, or G inputs to V_{IL}.
11. Decrease V_{CC} to 0 volts.
12. Return to step 4 until all outputs in the word have been programmed.
13. Repeat steps 2 through 11 for each word in memory.
14. Verify programming of every word after all words have been programmed using V_{CC} values of 4.5 and 5.5 volts. Note that registered PROMs must be clocked to verify the output condition.

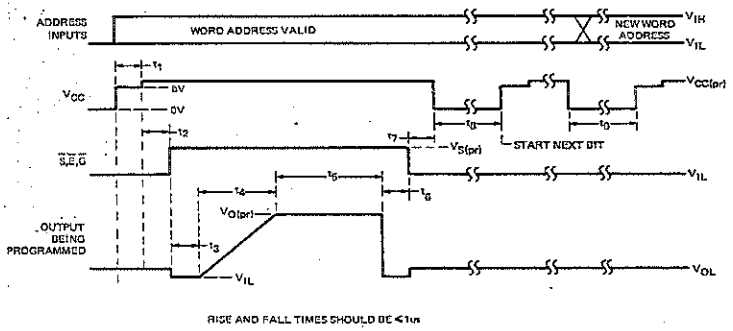


Fig. 5 PROM programming timing

$V_{CC(pr)} = 6 \pm 0.25V$
$V_{S(pr)} = 9.75 - 11V$
$V_{IL} = 0$ to $0.5V$

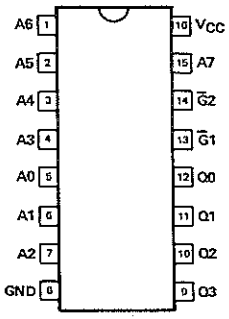


Fig. 6 PROM pin-out

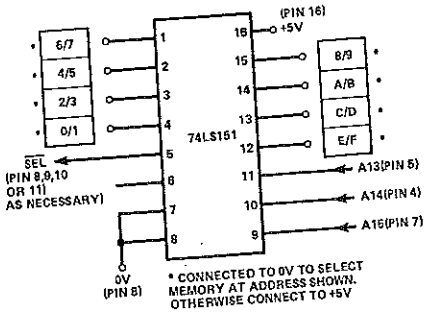


Fig. 7 Use of 74LS151

the locations are available. SW1 and SW2 can be used to gain access to three others sets of 64 locations. Also, only one of the four output bits of the PROM is used — selected by LK1 — so a total of 16 different memory maps can be held by each PROM. Note that unlike a previous design for a memory board using this device, the output of the PROM must be programmed LOW to enable the appropriate part of the memory map. Note also that the address lines are not used in order.

If desired it should be easy to wire one or two chips to a 16 pin DIL plug for use in place of the PROM.

Construction

This stage of the project is not difficult but just seems exceedingly tedious. Step one is to check that all the components will fit their holes. Note that the DIN 41612 connector usually needs 1 mm holes for its leads as does C7. All the other components, except RV1 and 2 which need 1.2 to 1.5 mm holes, will fit into 0.8 mm holes.

Step two is to take all the components off the PCB (you didn't solder them on — did you?) and make all the through-board links. The easiest way we know of doing this cheaply is to take a length of 22 swg tinned copper wire, stretch it a little to make it straight and stiff, squeeze the very end with pliers to flatten it out so that it will not fall through the holes in the PCB and then cut off about ¼ inch (6 mm). Repeat this process until you have enough pieces to go through all the link holes.

Support the PCB clear of the table top with the component side uppermost. Working from one end of the board, put about a dozen of the links in the proper holes and solder them in place. Turn the board over and put it flat on the table with a piece of kitchen tissue for protection, and solder all the links on this side as well. Clip off all excess wire and repeat until all the links are made.

Step three is to fit all the IC sockets. Note that IC5 to 12 are the opposite way round to the others. Then fit the DIN 41612 edge connector and the other passive components except R9. Make sure that the diodes and electrolytic capacitors are the right way round.

At this stage it is advisable to check that there is not short circuit on the power supply lines. If this test is OK then R9 can be fitted. Check also that 0 V and +5 V supplies are connected to each IC socket. Examine the PCB tracks carefully, especially around the edge-connector socket, for breaks or solder bridges, as these will be very difficult and possibly expensive to find later.

Step four is to insert IC15, 16, 17 and 18. Apply power to the board and check that it does not draw more than 100 mA or so. Now connect a 1 MHz TTL compatible square wave signal to the φ_2 test point. With an oscilloscope (or otherwise, as equipment allows) monitor the $\overline{RAS}$ test point and adjust RV1 such that the high time is about 150 ns. If this cannot be done, check your PCB again and verify the component values of RV1, R5 and C4. Also check that there are two pulses per 1 μs — check C1, 2 and R3 if not. $\overline{CAS}$ testpoint should be continuously high.

If you have got this far successfully, remove power from the PCB and link X to E. Reapply power and check RAS signal again. Now check that the $\overline{CAS}$ testpoint has a low pulse while the φ_2 signal is high. Adjust RV2 if necessary to see this. If this signal does not appear check RV2, R7, D2 and C5 and the signals at IC17c and d.

If all is correct, adjust RV2 such that the $\overline{CAS}$ signal goes low about 100 ns after the $\overline{RAS}$ signal goes low. This should set the main timing to about the right area for normal operation.

Switch off the power again and insert IC2, 3, 13, 14 and 19. Switch on again and check that all eight outputs of IC14 are counting. Check that the outputs of IC18a are switching on the rising edge of the $\overline{RAS}$ signal. Check also that IC19 pin 4 is permanently low and all other outputs from IC19 are high. Now connect the R/$\overline{W}$ input to the board to 0 V and check that all outputs from IC19 except pin 9 are permanently high. Pin 9 should be pulsing low with approximately the same signal as that on the $\overline{CAS}$ testpoint. Pulling A14 or A15 inputs low should alter the pin numbers but not the signal.

If you have got this far successfully there is only one more thing to do before inserting the memory devices. This is to check the power-on-reset circuit. Incidentally if this does not work correctly it could have given you problems earlier. Temporarily short-circuit C3 and monitor the output of IC17b. This should be low. Remove the short from C3 and check that the output of IC17b stays low for at least 200 ms (probably nearer 500 ms). During this time the $\overline{RAS}$ and $\overline{CAS}$ signals will be high. Note that φ_2 signal should be present as early as possible to ensure that the $\overline{CAS}$ signal is forced high, although the φ_2 line being low will also accomplish this.

The last thing to do now is to insert the 4416 memory devices and IC1. The memory ICs are accessed in pairs, so if you are not using the full complement you must insert IC8/9, 7/10, 6/11 and 5/12 in pairs. This is also the order in which they appear in the memory space. Remove the X-E link and insert a TBP24S10 suitably programmed into the IC4 socket and link X to A, B, C or D as appropriate. Alternatively, plug a 16-pin header into IC4 socket with, for example, a 74LS151 connected up to select the memory in 8K blocks. However you do it, the $\overline{SEL}$ signal at X must be low to read or write to or from the board.

A feature of this design is that the $\overline{SEL}$ signal needs to be low only a short time before the $\overline{CAS}$ signal is generated in order to activate the memory control but must be held until the end of the φ_2 cycle for a read operation or the end of the $\overline{RAS}$ signal for a write cycle in order for valid data to be read or written by a 6502 processor. This should not be a problem for any normal address decoder logic. **ETI**

BUYLINES

Everything you need is readily available. Technomatic, Watford, Cricklewood and others suppy all of the semiconductors (but note that the TBP24S10 is usually listed simply as a 24S10) and the PCB is available from our PCB Service.

DRAM BOARD UPDATE

Ahh! Doesn't it take you back to the balmy days of September 1983 when ETI first published its 64K DRAM board? The sheer technical excellence of the design, the excitement as you completed your very own memory card, the horror when you found it didn't work. Never mind, says Phil Walker, memories are e-made like this....

In September 1983 we ran a design for a 64K DRAM board to connect to the Microtan 65 system or indeed any 6502 processor system. Although the original worked satisfactorily we received a number of letters from readers who could not get theirs to work. At first it looked as though there was a faulty batch of the 74LS608 controller chip around which the design was based but after a while it became apparent that it was "a problem device".

To overcome the shortcomings of the original design and to simultaneously incorporate some new features, last month's Experimenter's DRAM Card was designed. This removed the need for special ICs by using standard components. The only unusual items were the 4416 16K x 4 bit dynamic RAMs which were used because they allowed the PCB to be only partially populated when a significant proportion of the address space was not required.

Once this board was working on the author's 6502 system — which runs at 1.25MHz, a little faster than the tangerine — it seemed reasonable that we should go back and do something for all those people who had built the 1983 project. To this end we have designed a small PCB which contains all the necessary control logic to replace both the 74LS608 and two other devices on the original board.

This PCB is mounted 'piggy-back' fashion on the PCB once all the original control ICs and timing components have been removed.

All the original features relating to address space allocation are retained and the same PROM can be used. One thing which may be of interest to non-Tangerine users is that there is no longer any need for the $\varphi 1$ signal to be provided as all timing is taken from the edges of the $\varphi 2$ signal.

The Circuit

This is identical in most respects to the control logic in last month's project. One significant difference is that the incoming select signal from the PROM is high to enable rather than low. However, since

HOW IT WORKS

This is very much the same as for last month's Experimenter's DRAM Card project but we shall go through it briefly in relation to the original design.

All timing is performed relative to the rising and falling edges of the $\varphi 2$ signal from the 6502 processor. $\varphi 2$ is buffered by IC22a and by means of the delays in IC22b, C21 and 22. IC23 is triggered on both edges of 02 to produce the RAS pulse. The width of this pulse is controlled by RV21 in conjunction with C23 and R23.

A delayed version of the RAS signal is used as the MUX signal to operate the row/column address multiplexers. Shortly after this IC24a is clocked, and if $\varphi 2$ is currently high and the inverted SEL line is low, the CAS output will go low and stay low until $\varphi 2$ goes low at the end of the processor access period. This keeps the data at the outputs available for longer when a read cycle is required. If $\varphi 2$ was low then the CAS output will stay high and a refresh operation will occur.

If all other conditions for a processor access cycle have been satisfied, to write data to memory the R/W line will be low. This condition is gated with the state of the inverted $\varphi 2$ signal in IC22c and with the MUX signal in IC22d. This ensures that the WE low condition is only asserted at the requisite time.

At the end of each RAS pulse IC24b is clocked to sample the state of the $\varphi 2$ signal. The outputs of IC24b are used to enable and disable the address buffers and allow the processor or refresh addresses onto the memory chip address pins at the appropriate times. Doing it this way rather than with the $\varphi 1$ and $\varphi 2$ clock signals allows more settling time for the buffers and the refresh address counter before the RAS strobe occurs to start the next operation.

The last piece of circuitry to describe is the power-on-reset. This is produced by IC1b and c together with C25 and R21. D21 allows C25 to discharge rapidly when the power is turned off. By means of this circuit, IC23 and 24b are held in a defined state for a short while after power is applied. This allows the internal circuitry of the memories to become operational. This operation is required when power is applied, not when the processor is reset. Note also that eight RAS only cycles should by performed after the power-on-reset before the memories are fully operational. However this will usually be taken up by the system initialisation routines reading from ROM.

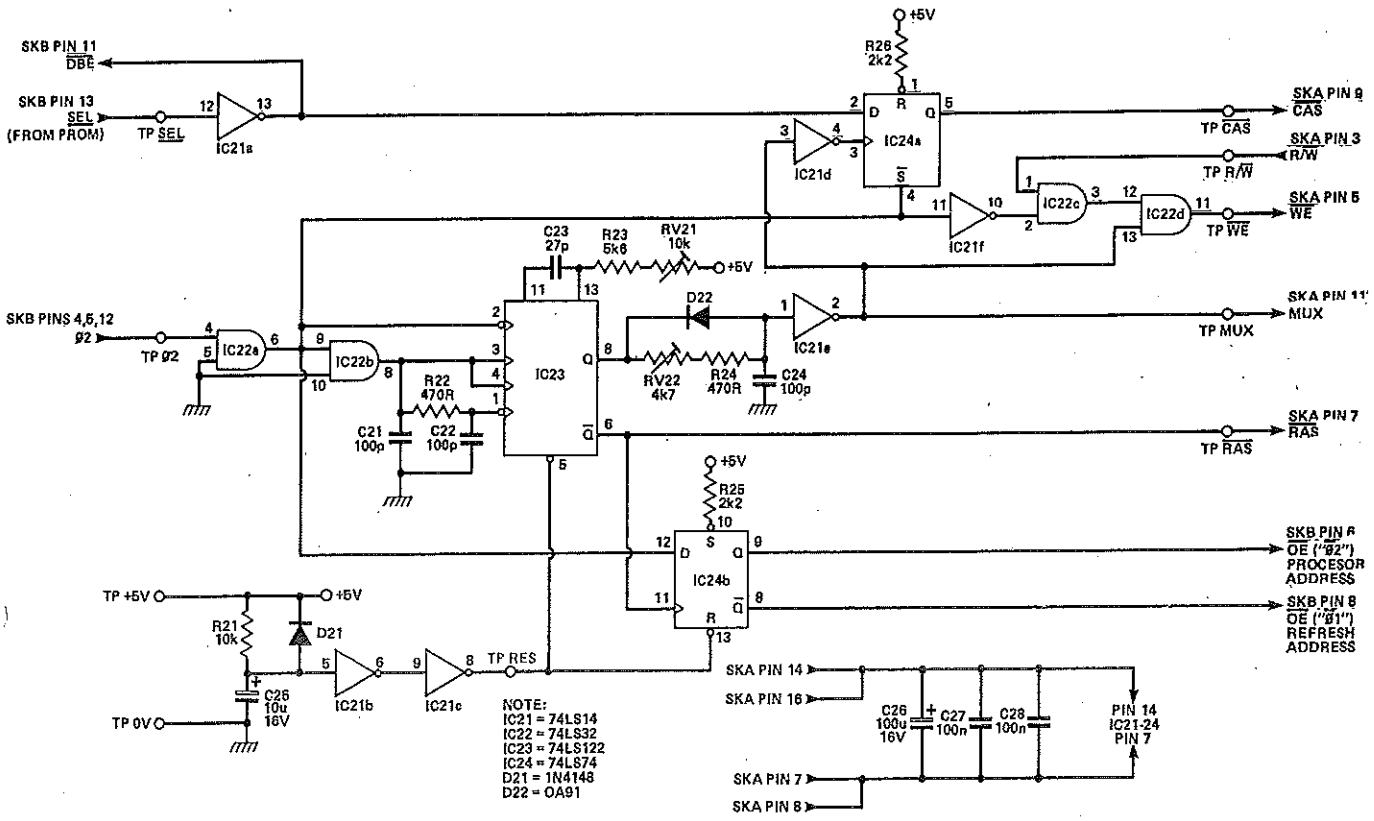


Fig. 1 Circuit diagram of the new control logic.

there is no G pin on the 4164 devices, the inverter which previously provided this signal can now be used to invert the SEL line and enable the data buffer at the appropriate time.

The other difference is in the $\overline{WE}$ circuitry. The lack of the G pin on the 4164 means that, if we want to connect the data - in and data - out pins of the memory chips to the same data bus as in the original design, the write cycle must be the so-called 'early write' detailed in the data sheet. This requires that the $\overline{WE}$ signal go low before the $\overline{CAS}$ signal goes low. If this is not done the data outputs of the memory chips will become active and may try to drive the data bus into a state opposite to that of the bus buffer.

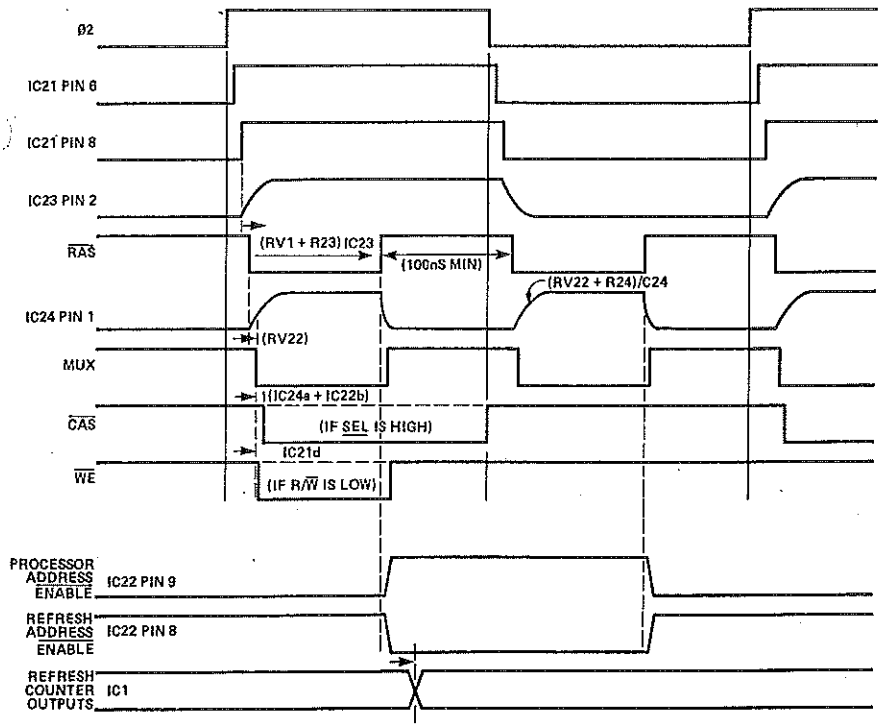


Fig. 2 Timing diagram for the control logic showing how all of the signals are derived from $\varphi 2$.

Construction

Construction of the PCB itself is quite straightforward. Remember that there are four wire links to be inserted as this is a single sided board. If height is likely to be a problem then solder the ICs directly into the PCB but otherwise use sockets. The other components are simple to install but the usual care should be taken to get polarities correct.

Assembly onto the main board should be postponed until the add-on board has been tested. You will need access to a signal generator giving a 1MHz TTL compatible square wave signal and an oscilloscope with which to see the results.

Connect a suitable +5V supply to the board and check that the current drawn is not more than

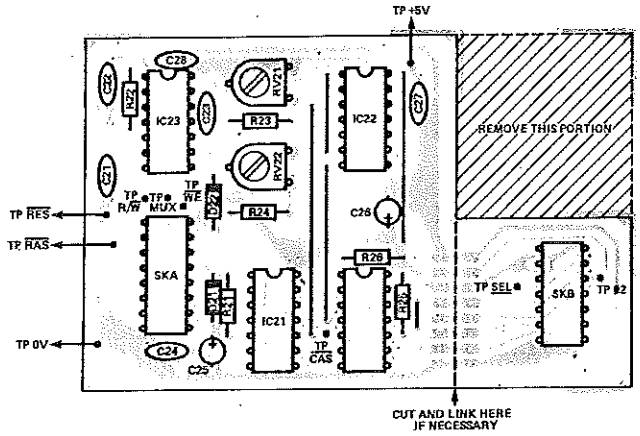


Fig. 3 Component overlay of the PCB.

PARTS LIST

RESISTORS (¼ watt 5% carbon film)

R21	10k
R22, 24	470R
R23	5k6
R25, 26	2k2
RV21	10k miniature horizontal preset
RV22	4k7 miniature horizontal preset

CAPACITORS

C21,22,24	100p silver mica or polyestyrene
C23	27p silver mica or polystyrene
C25	10u 16V tantalum bead
C26	100u 6V tantalum bead
C27, 28	100n ceramic disc

SEMICONDUCTORS

IC21	74LS14
IC22	74LS32
IC23	74LS122
IC24	74LS74
D21	1N4148
D22	OA91

MISCELLANEOUS
PCB; 4 off 14 pin DIL Sockets if required; thin sheet of insulating material.

100mA or so. Connect the SEL testpoint to 0V and a 1MHz square wave TTL compatible signal to the $\varphi 2$ test point and monitor the signals on the RAS, CAS and WE test points. Only RAS should show any activity at this stage and its low time should be set to about 300ns by means of RV21 (one half to three-quarters clockwise rotation). Both the CAS and WE signals should be high. A worthwhile check at this stage is to see that SKB pins 6 and 9 are switching at the same rate as the $\varphi 2$ signal but phase shifted from it. Also check that the MUX signal at SKA pin 11 is similar to the RAS signal but slightly delayed from it (RV22 should be set to approximately one quarter of its clockwise rotation). Note that the RAS and MUX signals are at twice the $\varphi 2$ frequency.

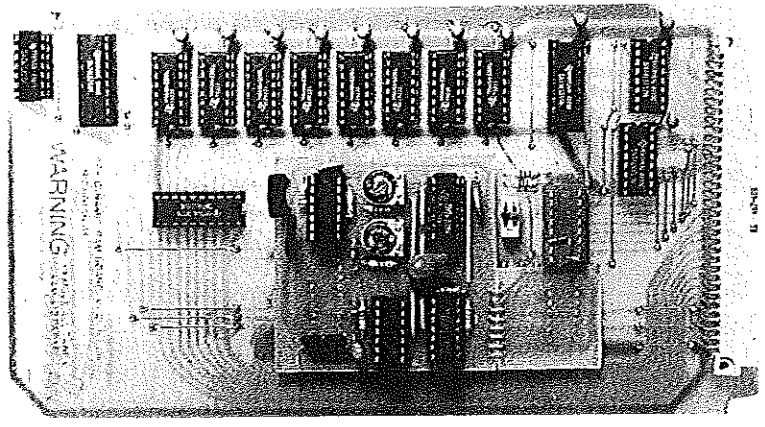
So far we have checked that IC21e, 22a & b, 23 and 24b are working. Now set the SEL test point to +5V, or just open circuit the link you previously inserted, and check that the CAS test point shows low pulses but only while $\varphi 2$ is high. The RAS to CAS delay time can be adjusted by means of RV22 if required but will only be critical if you have a slow processor.

There are only two other sections to check. Monitor the WE output and check that, with the R/W input open circuit or logic high, this output is also permanently high. Applying a logic low or 0V signal to the input should cause the WE output to produce a series of low pulses, at the same rate as $\varphi 2$ but with the same width as the MUX low signal. The WE pulses should only occur when $\varphi 2$ is high.

The last thing to check is the power-on-reset. If possible, monitor the RES test point and the voltage across C25. Temporarily short circuit C25 and check that the test point stays high for a few milliseconds after the short is removed.

By now you should have a fully tested board and will be ready to connect it to the original DRAM. First disconnect all the wires you used in the testing phase and connect 22 SWG tinned copper wires to the SKA and B positions. Cut these off flush with the top side of the board but leave them about 25mm (1 inch) long on the wiring side.

Next remove IC16, 17 and 18 from the original PCB together with their sockets and R1 to 6, C1 to 6 and D1 to 3. If you have not corrected the original PCBs you should now connect IC1 pins 2 and 12 to 0V, IC3 pin 9 to IC4 pin 9 and the tracks going to EC2a and 2b to EC3a and 3b respectively instead.



In order to fit the PCB into the Microtan rack we found it necessary to remove C7 and trim the through-board link near IC1 pin 14.

Having done all necessary corrections, the time has come to put the two PCBs together. First remove all of the ICs from both boards and put some thin insulating sheet under the new PCB to prevent shorts to tracks on the top of the old board. Now carefully feed the wires from SKA through the holes vacated by IC16 and the wires from SKB through those left by IC18. Make sure that none of the wires are misplaced or strained and carefully ease the two PCBs as close as possible to each other. Solder the wires on the underside of the old PCB and clip off excess length.

Insert all ICs (except 16, 17 and 18) and the assembly should be ready for use. If necessary, some adjustment may be made to RV21 and RV22 but this will usually not be necessary if the other stages have been carried out.

ETI

BUYLINES

Absolutely everything here is readily available from any number of different suppliers and the PCB is available from our PCB service.

DESIGNING MEMORY

Just what is involved in obtaining a working design? Phil Walker uses the example of the 64K DRAM card and the DRAM fix (ETI Dec '84 and Jan '85 respectively) as an example to show how it's done.

The design of electronic projects can, like most human undertakings, be long or short term, easy or difficult, with all shades of grey in between. In both the hobby world and professional life, it is unusual to go straight from original concept to final hardware with no deviations or modifications along the way. So in the long winter evenings as you sit by the fire working out the details of your next piece of electronic wizardry, don't be in too much of a hurry to commit pen to cheque-book or solder to iron.

The first step is to design what you want: write it down in as much detail as possible. Now try to come up with as many different ways of getting there as possible. Choose two or three of these to consider in detail, and examine their good and bad points.

At this stage you will need information on the various devices and components you may want to use. The catalogues of the larger distributors can be very helpful, but obviously, they do not have the detail that you will find in manufacturers' data books. (One point though, those readers who have access to RS Components will probably already have found that they do produce some very useful data sheets for the semiconductors they distribute.)

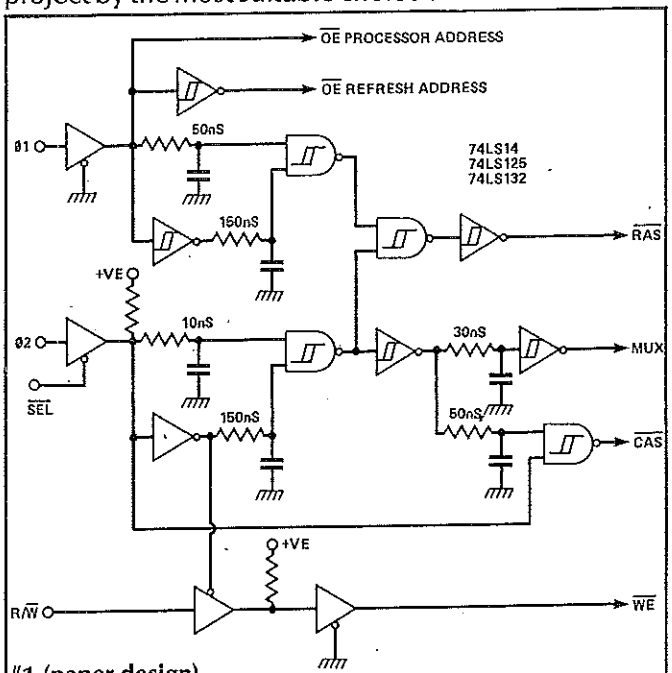
When you have all the information, choose a design approach that satisfies your requirements easily, in other words, an approach which does not require you to operate close to the limits of the devices. In particular, be careful that power dissipation, gain and bandwidth at the lower end of the specified (on the device data) range will not lead to trouble in analogue circuits; on digital circuits, look out for propagation delays and over-critical timing, at either the fast or slow limits (or a combination of both).

After this, you should be in a position to draw circuit diagrams and even outline physical dimensions to your project if they are important. Make sure that, at least on paper, your design will do what you want every time and has no hidden modes of operation, especially at power-on and power-off. Examine what you have and check that it is what you want. Look back to your original requirements and see if it fulfils them (it probably won't).

Now is the time to build something. If it is a large project, then it is wise to build the essential part or parts of the circuitry and test them on some sort of breadboard system. Make sure that these bits do what your theory predicts. If not, **find out why!** If you don't do this you will have really wasted your time. You will learn more from understanding why thing's don't work (first time) than from a series of first-time successes. Naturally it is very frustrating to have your brain-child fail miserably at the first hurdle, but with perseverence your success rate will improve or the complexity of project will increase. With any luck both will occur.

Having chased all the gremlins out of the wirework so far it is time to take the plunge and make the whole thing. Two things are likely to trip you up now, apart from the sillies such as wrong wiring or PCB error. The first is that your original idea had a flaw which has been overlooked or was not apparent until the whole thing came together. The second is that some external equipment or system does not interact with it as planned. In either case some careful thought will be needed to find the appropriate action. This part is often the most frustrating and expensive and leads to red faces, lost tempers and much burning of midnight oil in industry.

A good designer is one who by experience, imagination or both can avoid most of the likely problems in a project by the most suitable choice of method and com-



#1 (paper design)

For:
- Timing derived from processor clocks.
- All timing periods controllable.
- Only three active packages.

Against:
- Timing affected by temperature.
- Timing for RAS determined by differences between networks.
- Too many passive components.
- Long time delay φ_1 and φ_2 to RAS.
- Address buffer controls too near RAS timing.
- Uses φ_1 and φ_2 signals.

ponents. Often he or she will use well-tried circuit elements and configurations as building blocks for the new design. New concepts or components will be tested both in theory and practice before being relied upon. In this way it is made likely that problems arising at the prototype stage will be trivial for the most part (wiring error, etc.) or relatively easy to isolate. The unfortunate thing is that most designers operate under a "wanted yesterday" regime and cannot take all the time needed for such deliberation. This is true possibly to an extreme degree amongst hobbyists.

The alternative to designing projects yourself is, of course, to let someone else do it. In this case you must either pay someone to design what you want (or think you want) or get it from a book or magazine (ETI of course!). In this latter case you are restricted to what is published but must still make the effort to understand fully the circuitry you are making. If you do not understand what you have made, you are going to be in big trouble if anything goes wrong. You may get a little help by writing in but publishers usually have neither the time nor money to employ clairvoyant faultfinders.

To illustrate the design process I have included a section on the recent GNOS-EX memory expansion card. Not that this is necessarily a shining example of perfection but it serves to illustrate some of the foregoing comments.

GNOS-EX Development

This started in earnest about six months ago. For some time we had been aware that all was not well with our original 64K DRAM card for the Microtan system, especially in the control logic area. Since this was mostly invested in the 74LS608 memory control device there was not a great deal we could do about it.

In the meanwhile, we had published a DRAM card for Z80-based systems using the TM4500 memory controller. It would have been relatively simple to adapt this to 6502 use, but we did not think this the most useful or illuminating approach. Alternatively, we decided to design the project using the simplest standard TTL devices possible for the control section, along with the lesser known 4416 DRAMs. The 4416 is a 16K x 4 variant on the 64K x 1 devices used in the earlier designs. Although it's slightly more expensive than the latter, it offers the considerable advantage that the user need not populate the memory fully to obtain a workable system, as memory devices need be used in just pairs so 16, 32 or 48 Kbyte options are available, besides the full 64K.

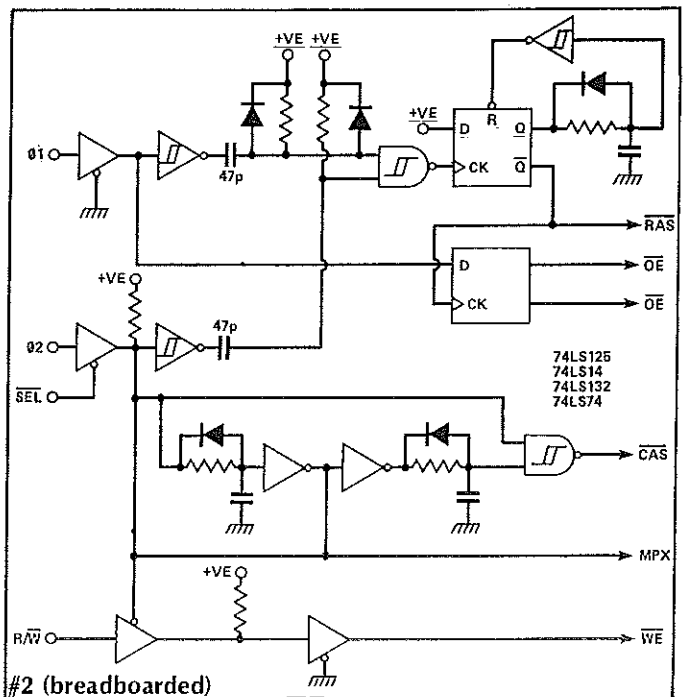
The requirements for this project can be summarised as:

1. To provide up to 64K of dynamic memory;
2. To operate from the signals from a 6502 microprocessor;
3. To use standard TTL devices, preferably SSI and MSI types.

Other factors were considered but were deemed subsidiary to these for this purpose.

Design Approaches

#1 The first ideas on the possible control logic design tended to follow the original circuitry. Both φ_1 and φ_2 were used to obtain the requisite timing signals. In the 6502 data sheet these two signals are supposed to be non-overlapping, ie one falls to zero before the other rises. Looking at these on an oscilloscope showed that if this was the case, and it certainly did not appear so, it was of little practical use as a little stray capacitance or variation in gate propagation delay would nullify any benefits it gave.



#2 (breadboarded)

For: Timing for $\overline{RAS}$ provided by edges of φ_1 and φ_2.
Common control of $\overline{RAS}$ in processor and refresh periods.
Address buffers switched at end of $\overline{RAS}$ cycle, not near φ_1 or φ_2.

Against: Timing dependent on temperature – made worse by Schmitt thresholds being near 0V rail.
No provision for start-up reset sequence.
Quite a lot of passive timing components.

Did not work due to differentiator capacitors being too small to trigger NAND Schmitt. Increasing capacitors did not help as they could not discharge sufficiently in the time available to be ready for the next cycle.

The first paper design used the φ_1 and φ_2 signals to generate the $\overline{RAS}$, $\overline{CAS}$ and MUX signals with the aid of R-C delay networks and Schmitt input gates and inverters. This in fact used only three IC packages – one less than the final design. However it was felt that the number of passive components was excessive and that the timing they would give would be rather critical and temperature dependent.

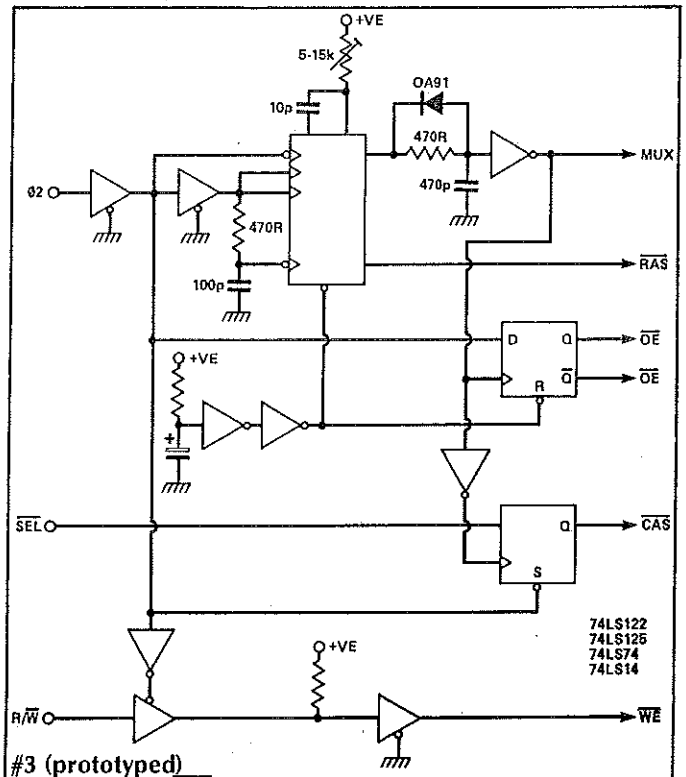
#2 The next serious attempt used a D-type latch connected as a monostable to generate the $\overline{RAS}$ pulse triggered by two differentiator networks from the φ_1 and φ_2 inputs. The other half of the latch package was used to control the processor and refresh address buffer outputs. This was clocked on the rising edge of the $\overline{RAS}$ signal and sampled the state of the φ_1 input to determine whether the next operation would be a refresh or processor operation. This was felt to have great merit as it ensured that the address presented to the memories was stable well before and after the active $\overline{RAS}$ falling edge. The idea persisted into the final design.

The MUX and $\overline{CAS}$ signals were still generated by R-C delays. In spite of this the circuit was felt to have some merit and was breadboarded. Unfortunately the differentiator networks did not function as planned and the monostable was not triggered reliably. The fault was found to be that the capacitors were too small to overcome the capacitance at the inputs to the gate, and increasing them did not help since then they were too large to recharge before the next cycle.

#3 A deliberate attempt was made in this design to eliminate as many of the R-C delays as possible. This led to the use of the 74LS122 monostable, as it is claimed to operate quite reliably at these time periods. Also with the multiplicity of trigger inputs, it was found possible to make the '122 trigger on both the rising and falling edges of the same input signal. This made it possible to dispense with the φ_1 signal and refer all timing to the φ_2 input. This also suited the designer quite nicely, as the φ_1 signal is not available on the system expansion bus connector.

With the critical timing taken care of by the monostable, the only other timing delays needed were between $\overline{RAS}$ and $\overline{CAS}$ and non-critical delay to make the monostable trigger on both edges. The former was very simply accomplished by an R-C network and, although vital, this was found to be stable enough for this project.

The data sheet on the 4416 states that the column addresses must be stable by the time the $\overline{CAS}$ signal goes low. In this design there is one extra device propagation delay in the $\overline{CAS}$ signal path more than in the address multiplexer path. In practice, this was found to give about 20ns separation in the right direction. In going from the D-type latch monostable circuit to the 74LS122 and retaining the buffer control part there was a spare D-type latch available. This was quickly put to use to sample the select signal at an appropriate time and provide a $\overline{CAS}$ signal which could remain low until the end of a processor access cycle. This configuration not only removed a lot of constraints on the select decoding logic but also keeps data available at the memory devices until the 6502 has captured it. It also makes sure that there are no short or erroneous $\overline{CAS}$ outputs which would lead to mis-operation of the memory chips.



#3 (prototyped)

For: $\overline{RAS}$ **timing set by one control and not temperature dependent.**
$\overline{RAS}$-$\overline{CAS}$ delay set by one network.
Only two main timing networks.
Address buffers switched half-way through cycle.
Once $\overline{CAS}$ triggered stays low to end of cycle – makes data remain on outputs.

Against: **Stopped working when processor warmed up**

Refer to data sheets for 6502 and 4416 to get probable reason that 6502 guarantees data out valid for 200ns after φ_2 goes high. 4416 requires data valid whenever the later of $\overline{WE}$ or $\overline{CAS}$ goes low for a write cycle.

The essential control logic as designed together with the various address multiplexers, refresh counter and 16K of memory were constructed on a small PCB. This was connected via a simple interface to the designer's Ohio Superboard. This is a rather old 6502 system, which has been much modified, and now runs at a clock speed of 1.25MHz. After sorting out all the usual bugs and making the necessary adjustments, the project seemed to work well . . . for 10 minutes. No reason for the latch-up was apparent, so freezer spray was applied to various parts to see if there was a thermal problem. Lo and behold, it was the 6502 processor! However, since this had been working satisfactorily for some years, and still did so when not connected to the new memory, it was felt that the fault must be a little more obscure.

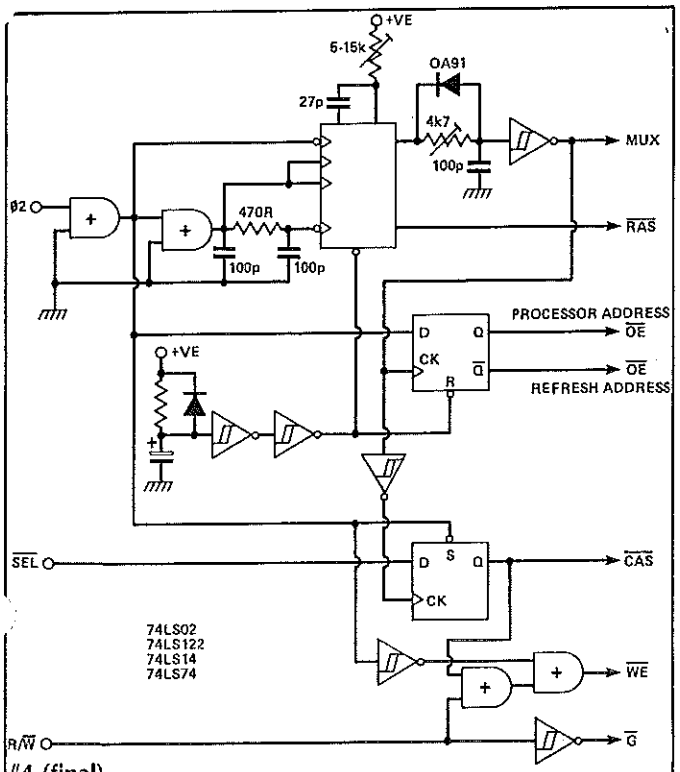
The next thing was to dig out all the data on the 6502 and the 4416. A combined timing diagram was drawn to illustrate the interaction between the two devices and the effect of the control logic. It had been noticed that although the data read from a location was not the same as that written to it each read operation was consistent with the last. This pointed to the possibility that there was a problem in the write operation. In fact the specification of the 6502 states that data in a write cycle is only guaranteed stable 200ns after the φ_2 rising edge. The 4416, however, requires that the data be stable whenever the later of $\overline{CAS}$ or $\overline{WE}$ goes low. In the design at this time it was the $\overline{CAS}$ which was critical. It was easy to adjust the delay to get round this problem, but this immediately raised another in the read cycle.

Here, the 6502 requires that data from the memories be stable 100ns before the φ_2 falling edge. The 4416 is guaranteed to supply this data 120ns after the $\overline{CAS}$ signal goes low. Being really pessimistic this would require a minimim φ_2 high time of 420ns, which looks a little adrift from the observed 360ns. To get around this, at least partly, the $\overline{WE}$ signal was gated with the $\overline{CAS}$ and φ_2 signals such that it occurred after the $\overline{CAS}$ signal. This meant that there was a little more leeway in the timing and the $\overline{CAS}$ signal could occur earlier. This was even better in the final design where there was an extra delay through the 'LS139 decoder.

#4 Having got the prototype working satisfactorily, the time arrived to lay out a PCB for the final design. This took about four days for the double-sided artwork and a further three days to have it photographically reduced and a board etched, drilled and populated. A new interface to the Superboard was constructed and the design tested. This brought to light one or two things like the $\overline{RAS}$ and $\overline{WE}$ being swapped on the memories and the desirability of more test points on essential signal paths. These items were corrected on the PCB artwork to yield the final PCB pattern.

Once the $\overline{RAS}$ and $\overline{WE}$ problem had been solved the project worked as required although it was not possible to check operation with the slower 4416-20 devices as only 4416-15 were available from the suppliers.

One extra feature had been added to the design between the #3 and #4 versions and that was the $\overline{G}$ signal derived by an inverter from the R/$\overline{W}$ line. This is necessary because during an active processor access cycle, all the memory devices will perform a read operation except for a pair selected to have data written if the processor wants a write cycle. The $\overline{G}$ signal is routed by



#4 (final)

For:
- All advantages of previous design.
- Uses quad NOR package instead of buffers – one less resistor.
- $\overline{WE}$ signal slightly after $\overline{CAS}$ instead of before it.
- $\overline{G}$ signal available to disable memory data output in WRITE mode.
- Works on a 1.25 MHz 6502 system.
- All edge triggered from φ_2 – samples $\overline{SEL}$ signal, CAS cannot be aborted once started.

Against:
- Requires adjustment to be made for best operation.
- Relies on short time delays around inputs of monostables.
- Memories must not be accessed for eight cycles after power-up.
- As it stands, is not usable for 4164 devices with data in and data out pins commoned.

the 'LS139 to the pair of memories to be accessed only during a read operation. At all other times the $\overline{G}$ pins of the memories will be high thus preventing their data bus drivers fighting for control of the bus. This does not affect the internal operation of the memories.

As a bonus and a service to readers who had built the 1983 DRAM design the control logic was re-configured to fit a small PCB such that it could be used to replace the control logic of the earlier design. Some small changes were necessary because of the select logic polarity difference and the use of 4164 memory devices. This was built onto the original project and tested under the same conditions and worked perfectly, as far as could be ascertained.

This concludes the description of how this project was designed. Although it cannot cover all the thought processes, prejudices, scribblings on odd bits of paper and strange flights of fancy which make up the design process, it serves as an illustration of how one starts with one set of ideas and progresses to the final solution discarding most or all of the original concepts on the way. This is probably quite common in designing electronic or other hardware (and software?) and is probably more useful than doggedly trying to get the first idea to work.

ETI