Contents

Volume 25
No.12

& Features Projects

Regulars

ETI ELECTRONICS TODAY INTERNATIONAL News...

New Virtual Instruments

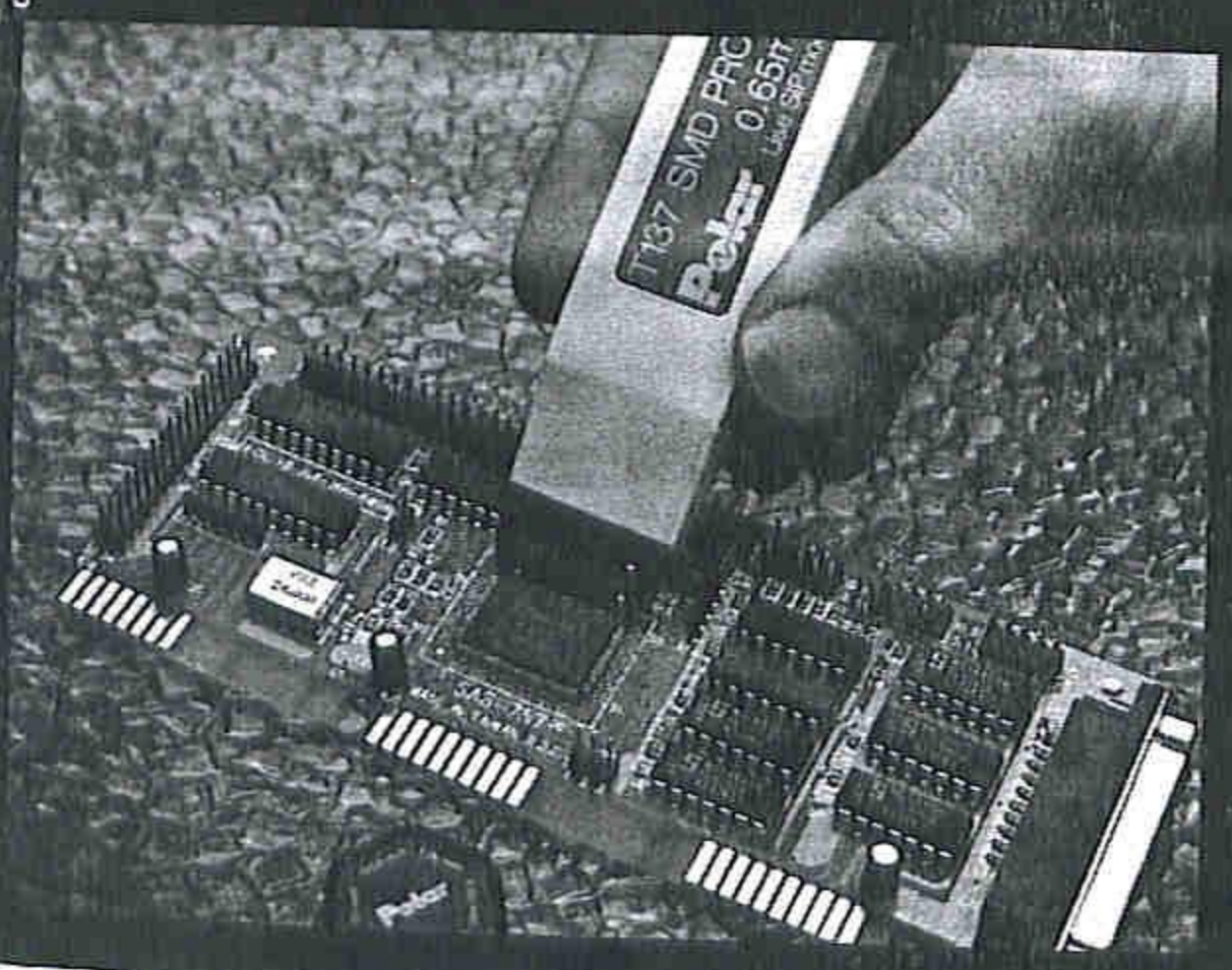
Cambridgeshire-based virtual instrument specialists Pico Technology have added three new products to their range of PC-based instruments. All the products in this range offer desktop instrument performance, coupled with the enormous benefits of PC connection.

Slightly smaller than a cigarette packet, the SLA-16 (£219) is probably the smallest and most economical PC-based logic analyser on the market today. It offers 16 channel operation with an 8K trace buffer. Flexible internal and external clocking modes are supported (up to 50MHz). The supplied software provides state listings and waveform displays, while two cursors can be used for time and frequency measurement. The SLA-16 is supplied with all necessary software, cables and probes; all it needs is a standard PC AT with a spare serial port.

Another new product is the ADC-100 (£199). With this plugged into a PC and the supplied PicoScope software, a PC can be used as a dual channel digital storage scope, spectrum analyser, frequency meter and voltmeter. For long-term data logging and chart recorder emulation PicoLog software is an optional extra. The ADC-100 offers 100kHz sampling and 12-bit resolution. Each channel can be set to any of seven input voltage ranges (+-20V to +-0.2V). The ADC-100 is powered from the PC's parallel port.

The last of the trio of new products is the ADC-22 (£199) which is a 22 channel data logging unit. Its input specifications are the same as Pico's best-selling ADC-11 data logger and it too plugs into the parallel port of a PC. All three of these new virtual instrumentation products are available directly from Pico Technology, 149-151 St Neots Road, Hardwick, Cambs. CB3 7QJ. Tel: 0954 211716.

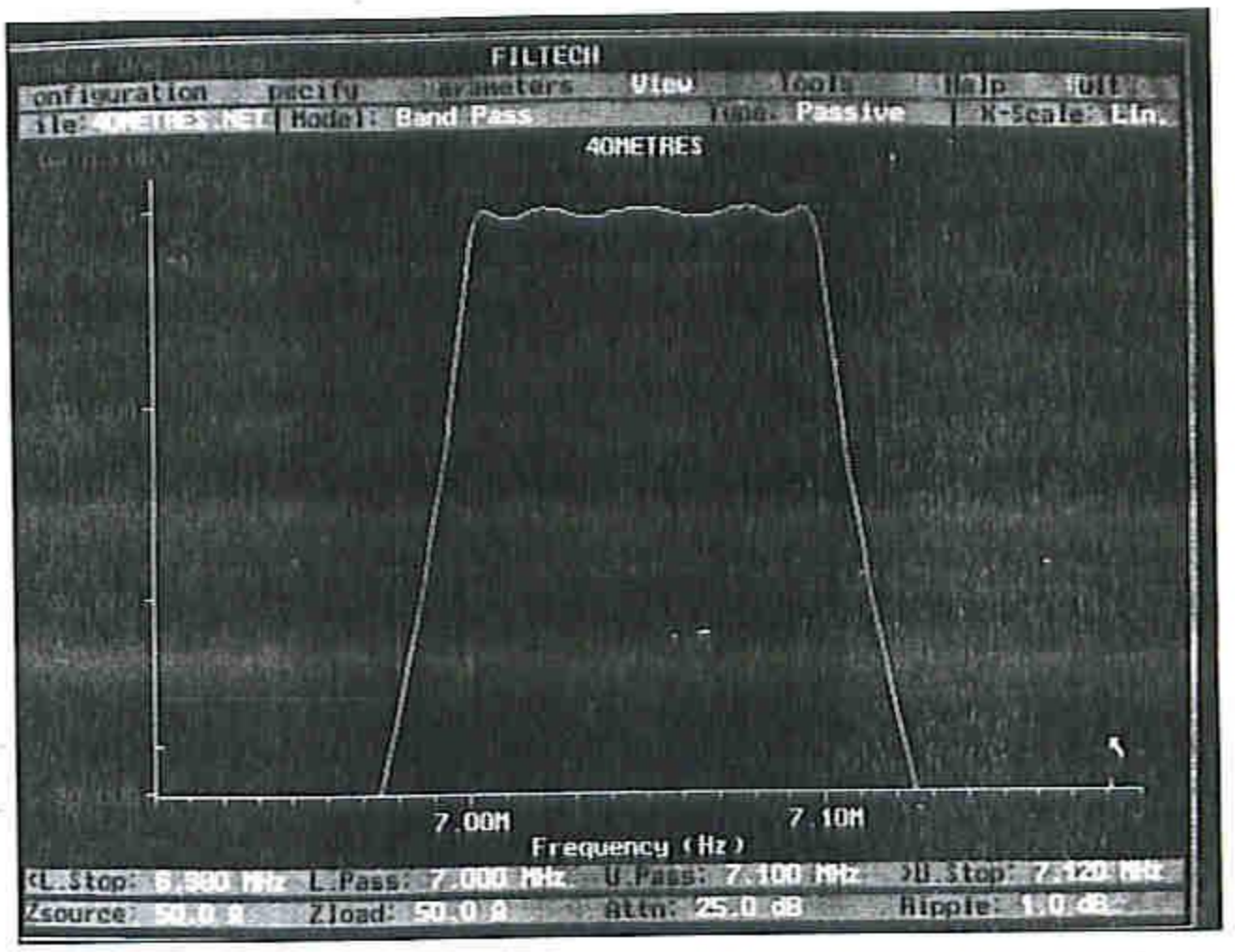
High density SMD probes



Polar Instruments has developed two new high density probes for state-of-the-art, surface-mount electronics components, providing an efficient test interface for ICs with fine geometry 0.65mm metric or 25thou Imperial lead pitches. Providing a row or 32 pins, the hand-held probes - known as T Series - interface with troubleshooting instruments equipped with the analogue signature analysis fault location technique. The probes provide a convenient and cost-effective interface for many of the latest VLSI peripherals and processor IC packages, and greatly accelerate the diagnostic tests needed for circuit investigation and repair.

For more information about these probes contact Polar Instruments Ltd of Guernsey on: 0481 53081.

Filter Designing Made Easy



The filter circuit is one of the basic building blocks in analog circuit design. Indeed almost every piece of electronic equipment, of whatever sort, contains at least one filter circuit and their use is becoming more common as EMC legislation progressively comes into force.

For many electronics engineers, particularly those trained in digital electronics, the design of active and passive filters is a tedious and error-prone process, which invariably involves many repetitive and tedious calculations or the looking up of dozens of normalised co-efficients in tables.

Thankfully, such days are now past. A new design program, FILTECH, from circuit simulation software specialists Number One Systems, will do all these calculations automatically, thus making filter design a relatively simple process.

The software will also simulate the behaviour of the filter circuit and display a graphic plot of the calculated frequency response superimposed on the specified filter limits, thereby giving the user immediate confirmation of the accuracy of the design.

Both active and passive filters up to sixth order with frequency ranges extending from fractions of a Hertz to GigaHertz can be designed and simulated using FILTECH. There is a no-nonsense approach to filter specification, Passband and Stopband frequency limits, Ripple and Attenuation levels and terminating impedances are all the parameters that the design program needs to complete a design. The user has full override capability on both design and simulation.

The FILTECH program costs £145, and is available from Number One Systems Ltd and includes all documentation plus example designs and a full discussion of filter circuits and characteristics. They can be contacted at: Harding Way, Somersham Road, St Ives, Huntingdon, Cambs. PE17 4WR. Tel: 0480 461778

New range of circuit protectors

Raychem have just launched a new version of their re-settable surface-mount Polyswitch circuit protection devices for electronics applications including computer interface ports (e.g., SCSI, Ethernet, keyboard and mouse ports) and computer peripherals (e.g., disk drives). This new range offers higher current carrying capability and lower resistance than earlier versions.

The Polyswitch SMD products have hold current ratings that range from .30A to 2.0A, maximum voltage ratings range from 15 to 60V and resistance ratings as low as 100 milliohms. These units can be used as replacements for fuses but, unlike fuses, they do not need to be replaced after they have blown, they will reset once the fault current is removed. They are made from solid state conductive material which is fabricated to form a positive temperature coefficient resistor that has a trimmed resistance under normal current flow conditions. If an overcurrent occurs, then it switches to a very high resistance state, thereby interrupting current flow and preventing damage to the circuitry.

For more information on these devices contact Raychem of Swindon on 0793 628777.

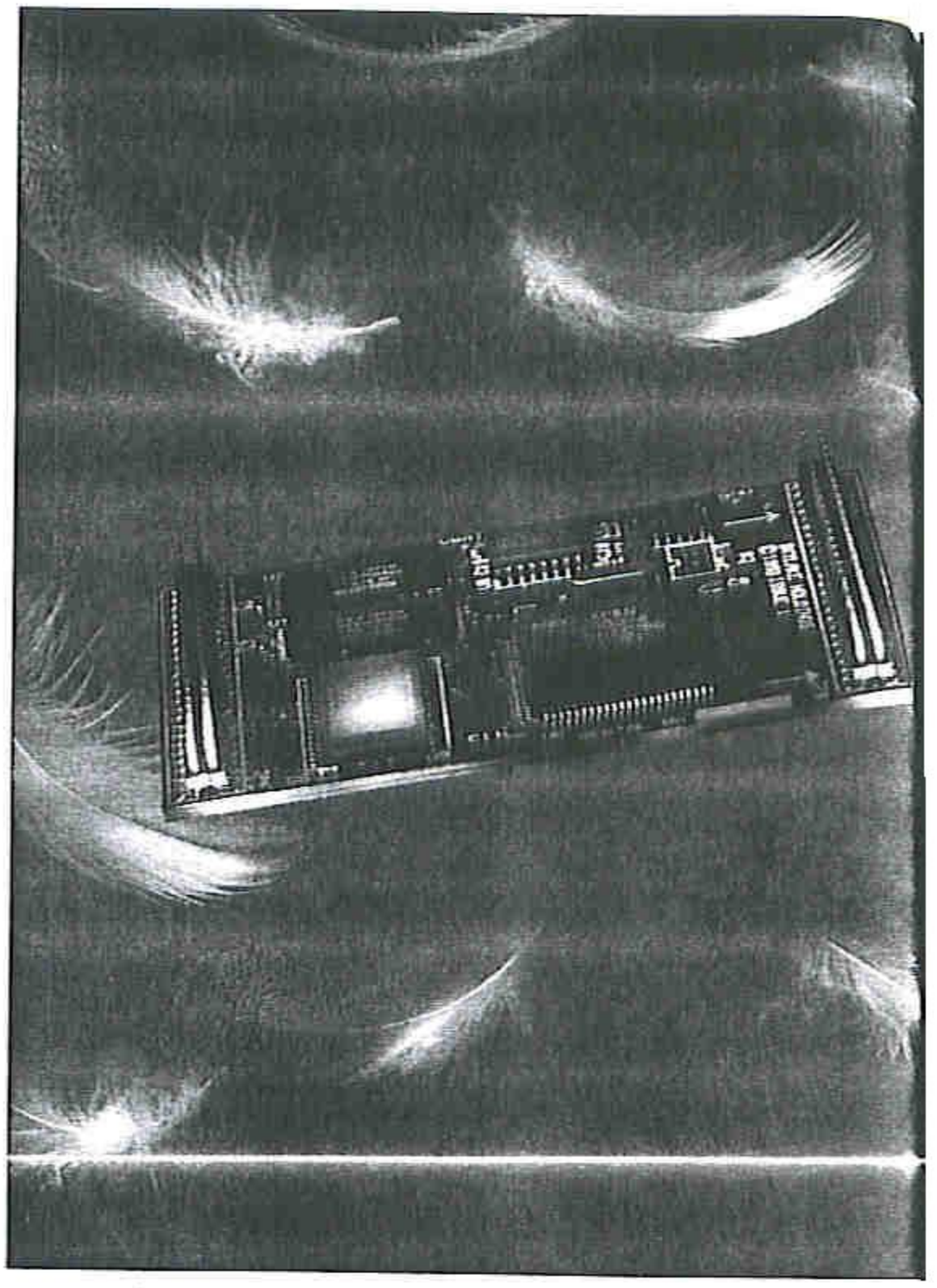
Very small embedded PC module

Skylake Talix of Southampton have developed what is almost certainly one of the smallest and least expensive PC-compatible computers for embedded applications that is on the market today. Known as Talix uV25, the computer board measures just 52 x 80mm, similar to a credit card, and weighs a mere 30g. Priced at £75, it sets a new price/performance standard for embedded control and data acquisition applications.

Developed to meet the ever-growing requirement for compact and economic PC-compatible processing power, Talix uV25 contains a V25 processor running at 8 or 10MHz, plus up to 1MByte of memory in dual half megabyte RAM and ROM/Flash-EPROM arrays. Also onboard are three parallel I/O ports offering up to 32 I/O lines - two serial channels, a real-time clock, interrupt controller, and interface/expansion connectors with terminals for back-up battery. Power consumption is 50mW typical at 5V.

The talix uV25 may be operated as a PC-compatible running DOS applications software. In an agreement between Skylake Talix and Great Western Instruments, General Software's powerful multi-tasking embedded DOS has been ported onto the module. Software development for PC-compatible target systems is simplified by a suite of utilities provided, including a debugger and a silicon disk utility for booting PC operation system and applications software from non-volatile memory. Alternatively, the processor can be used in its native 8086-compatible mode, and the module can optionally be supplied with Paradigm's TDREM monitor for the V25 CPU.

For further details contact Skylake Talix on: 0703 666403

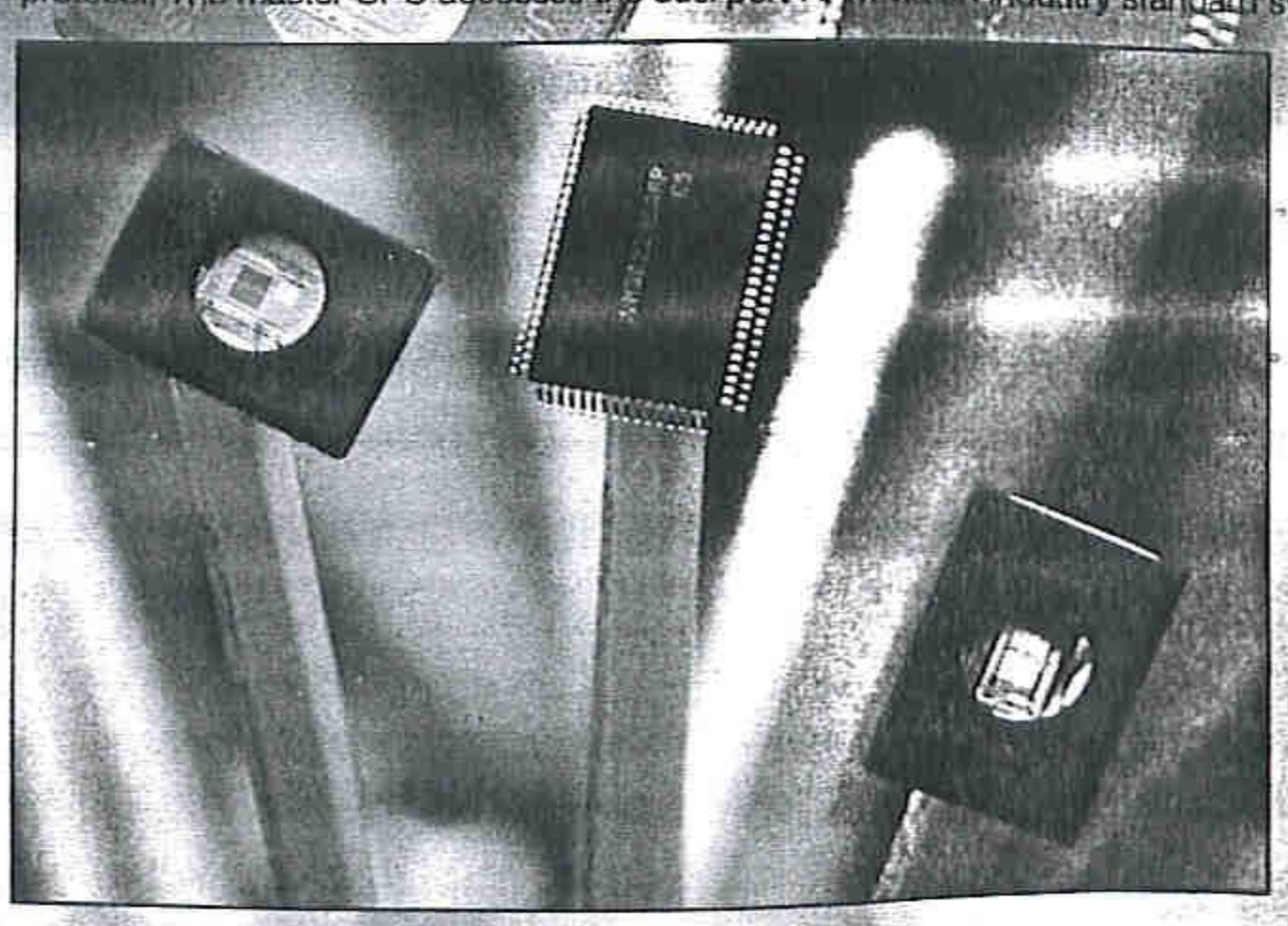


Intelligent protocol controllers

Mitsubishi Electric have introduced the M3888x Group of intelligent protocol controllers for multiprocessor systems with an internal control bus. The new devices join the company's wide existing range of 8bit microcontrollers.

The first device in this Group is the M38881 which features 240 bytes of RAM and 8Kbytes of ROM, plus 3UARTS including independent baud rate generators and CRC, an independent bus structure and dual port RAM.

The new Mitsubishi microcontroller operates as a slave controller to provide high-speed system communications. Featuring program memory ROM, work RAM and dual port RAM, the slave can be customised to any communications protocol. The master CPU accesses the dusl port RAM via an industry standard system bus and the slave accesses the dual port RAM via a local bus to control data transmission.



The 3888x microcontrollers feature full semaphore handling and provide all system control for the dual port memory. They cater for all possible data collisions. If a second device, for example, sends data to the same address, the M3888x's automatic arbitration resolves any conflict between local and system buses.

The intelligent protocol controller is capable of handling different system events and uses semaphores to flag each event as it occurs.

For further details on this range of devices contact Mitsubishi in Hatfield on 07[illegible]876100.

24-bit colour in a low-cost package

A new 24-bit RAMDAC from AT&T Microelectronics allows designers of personal computer systems to provide high resolution graphics facilities at a price comparable to standard VGA. Featuring a 16-bit pixel port and maximum video rate of 135MHz, the ATT20C498 displays 24-bit colour at resolutions of 800x600, and up to 64K colours at resolutions of 1280x1024.

Packaged in a low cost 44-pin PLCC, the ATT20C498 is backward compatible with the 8-bit VGA standard. An internal multiplexer allows the 16-bit pixel port to load two 8-bit pixels simultaneously; this means that designers can use slower, less expensive frame buffer memory, and run clock signals at half the video rate, reducing EMI. An on-chip clock doubler simplifies design even further. Switching between the ten graphics modes of the chip can be implemented "on the fly" or on a pixel by pixel basis, allowing the device to display several windowed applications supporting different graphics capabilities. Each application can be run at its highest resolution, using the correct colours, regardless of the primary system display mode.

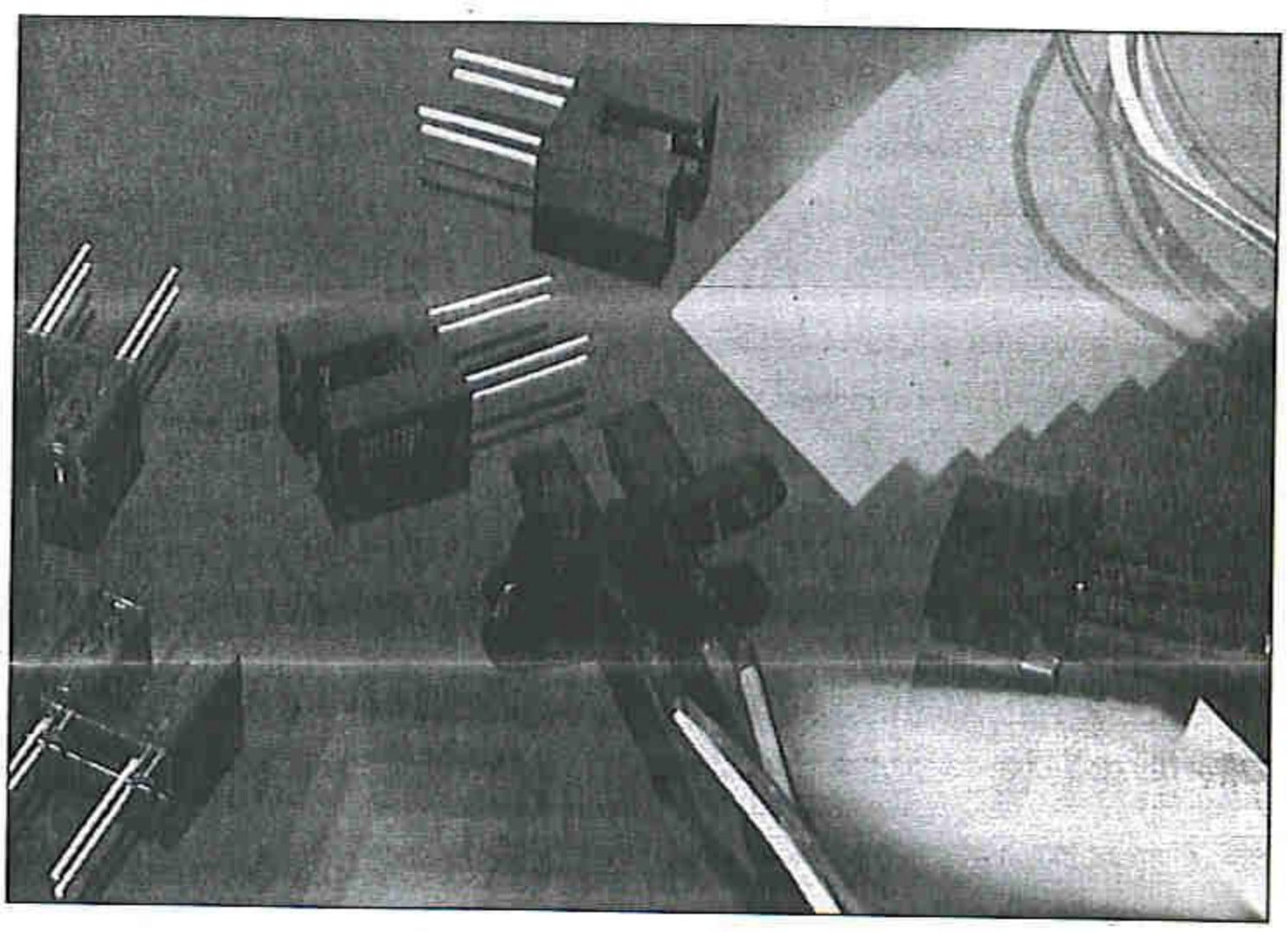
For further information contact AT&T Microelectronics, on: 0732 742999

High resolution slotted optical switches

Quality Technologies' new range of fully enclosed, slotted optical switches has a unique housing design which incorporates ambient light protection and has a smooth external surface to prevent any build up of dust or dirt in the optical path.

Designated the QVA and QVB series, the switches comprise an infra-red emitting LED, facing an NPN photo-transistor, across a 3.18mm gap. The phototransistor detects the presence of infra-red light from the LED, until it is blocked by a mechanical object. Moulded internal apertures of 1.27mm or 0.25mm in the optical path make these switches ideal for high resolution applications, such as encoders and precise position sensing.

For further details contact Quality Technologies of Aylesbury, on: 0296



Event diary

1-3 November	Windows Exhibition, Olympia, London. Tel: 01256 381 456
5-6 November	Eighth North Wales Radio and Electronics Show, Abergoonway Conference Centre & The New Theatre, Llandudno. Tel: 01745 591704
13 Nov	Midland Amateur Radio Society rally at Stoddard Road Leisure Centre, Slade Road, Erdington, Birmingham. Doors open at 10AM, admission £1, for further details ring 021 422 9787 or 021 443 1189(evenings only).
14 November	Operation Raleigh by John Leyton G4AAL, Stratford upon Avon and District Radio Society. Tel: 01789 740073
28 November	Microphones by Jack Cluley G4YIG, Stratford upon Avon and District Radio Society. Tel: 01789 740073
1-4 December	Christmas Computer Shopper Show, Grand Hall Olympia, London. Tel: 0181 742 2828
6 December	Open Forum - Questions and answers on anything connected with amateur radio, Sudbury and District Radio Amateurs. Tel: 0787 313212
6-8 December	CD-ROM 94, Olympia 2, London. Tel: 01865730275
12 December	Open house, Stratford upon Avon and District Radio Society. Tel: 01789 740073

If you are organising an event which you would like to have included in this section please send full details to: ETI, Argus House, Boundary Way, Hemel Hempstead, Herts HP27ST Clearly mark your envelope Event Diary.

Electronics in space

Space, the final frontier, a frontier that would be unconquerable without the help of sophisticated electronics. Nick Hampshire investigates

It is probably true to say that without electronics, and in particular computers, mankind's conquest of space would never have happened.

The history of the conquest of space began during the latter part of World War II, with the development of the German V2 rocket. This was a revolutionary weapon that could be launched deep in continental Europe on a trajectory that took it into near space, and that on its return to Earth brought so much terror, death and devastation to large parts of the city of London.

The V2 was a liquid fuelled rocket designed by Wernher von Braun, the brilliant engineer who later went on to design some of the most famous US rockets, including the giant Saturn V which launched all the Apollo moon missions. Indeed the V2 was the basic design on which all subsequent liquid fuelled rocket designs were based.

Although a very sophisticated system at the time, the V2 was, by today's standards, very simple. It was designed to follow a preset trajectory; once set on that trajectory the rest was automatic - gravity and the Laws of Physics would do the job. This meant that all it needed was a simple electromechanical inertial guidance system that would ensure that the rocket stayed on the desired trajectory during the flight.

Electromechanical systems were used because they were better able to stand up to the vibration and acceleration force experienced during a rocket launch. At that time the only electronics available were valve based, and valves are not the most physically robust devices.

The development of the much more robust transistor in the early 1950s, allowed designers to replace the earlier electromechanical systems with purely electronic systems. Rocket guidance and control systems consequently became much more sophisticated, and included telemetry links. Telemetry was very important because, up to the early 1960s, a great many of the missile guidance computers in use were ground based. This was because they were too large and required too much power to be fitted into the actual rocket.

The greater sophistication of flight control systems, whether ground or rocket based, allowed the flight trajectory to be controlled more accurately, something which was of increasing importance given the much longer range of the bigger rockets that were being produced.

At the height of the Cold War, the accuracy of long range ballistic missiles was of paramount importance, as the military wanted to be able to deliver a nuclear weapon to a precise target on the other side of the world. Control systems therefore needed to be even more complex, and more reliable.

The engineers given this task quickly realised that the electronic components and construction techniques available at the time were not well suited to the requirements of extreme reliability, or of packing large amounts of circuitry into small spaces. The result was the Tinkertoy project which sought to produce ultra-miniaturised, highly robust circuit modules which could then be assembled into the control system.

It was the developments generated by the Tinkertoy project in the late 1950s which effectively led to the creation of the first integrated circuit in the early 1960s.

By this time, ballistic missiles had developed to a stage where things could be put into orbit around the Earth; the days of Sputnik, Yuri Gagarin, and Gemini. Satellites and spacecraft called for even more sophisticated and miniaturised electronics that could fit into the very small spaces that were available. The first satellite launched in 1957, the Russian Sputnik 1, only weighed 75Kg. And, for the first time, electronics which could stand up to the very harsh environmental conditions found in space, and electronics which could run on the very minimal amount of power in these early spacecraft. were available. All these factors helped to further accelerate the development of the integrated circuit.

President Kennedy's acceleration of the 'space race' by declaring that the first man on the moon would be an American, put the full power of the US government budget behind the development of all the systems necessary to achieve this goal - development which was being funded to the tune of over 4% of US GDP. It is thanks to this enormous expenditure that we have most of the technologies that we take for granted today.

The space program gave an enormous boost to the US computer industry thanks to NASA's requirement that subcontractors were fully computerised. It also boosted the development and use of numerically controlled machine tools and a host of new production techniques.

But electronics was the area which saw some of the biggest advances. When Kennedy made his famous declaration, integrated circuits were confined to a few simple gate circuits but, by the end of the Apollo program, there were the first pocket calculators, electronic watches, memory chips and, just a couple of years later, microprocessors. All products where the development process had been directly, or indirectly, accelerated by the US space program.

Today the impact of space technology on electronics is less pronounced, but still enormously important. No longer is the acceleration provided by geopolitical factors. Instead it is

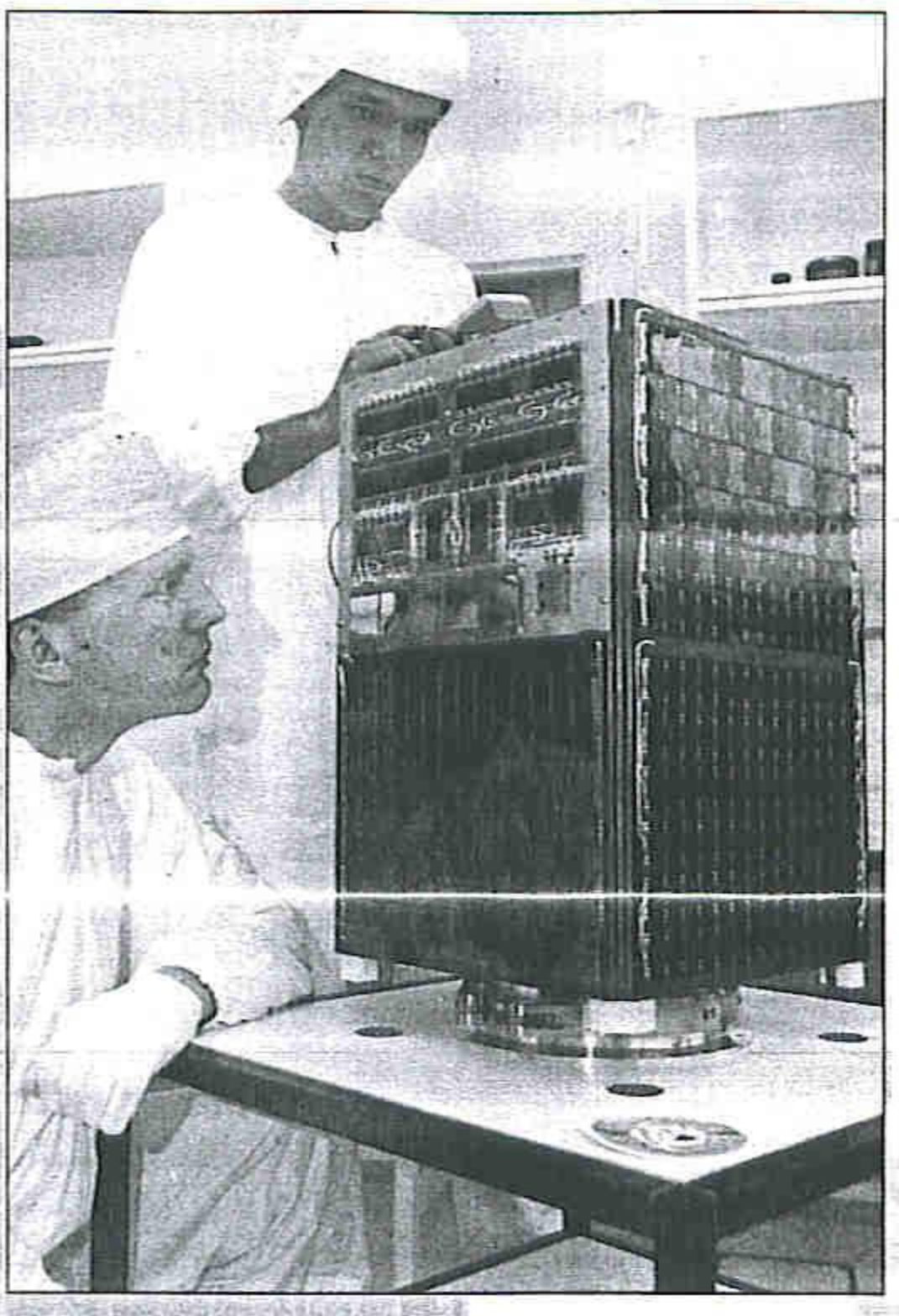
provided by hard commercial factors. Satellite communications systems have become commonplace. We now take it for granted that an overseas phone call could be relayed via a satellite.

Millions of us watch TV programmes broadcast from a satellite. We watch news programs sent live via satellite from remote corners of the world, or watch the weather forecast with its satellite-generated images of terrestrial weather-systems. We use information systems which rely on global networks of computers often linked via satellite.

Then ,of course, we depend on satellites to give us information about other planets in our universe. We also use them to monitor the health and state of our planet, to track down pollution, to monitor crop yields, to locate new sources of natural resources. And, as we saw in ETI a few months back, we can use satellite systems to accurately determine the position of a plane, a boat, or even a lorry-load of goods travelling across Europe.

Without the myriad satellites which now circle the Earth, much of what we do today would be impossible. Satellites would in turn be impossible without sophisticated and highly reliable electronics - electronic systems which can work 24 hours a day, 365 days a year for ten years or more without a single fault despite operating in one of the harshest environments known and being subjected to enormous mechanical stress

In the rest of this article we look at some of the problems facing electronic systems that are designed for use in space.



Types of spacecraft

However, before taking a closer look at the actual electronics, we need to look at the different types of space mission; each has different needs and different design criteria. We can divide space missions into three broad categories. Firstly, there are the manned missions, such as the regular Shuttle flights and the Mir space station. Secondly, there are the unmanned orbital satellites; these have a wide range of different functions; from weather and communications satellites, through the Hubble Space Telescope, to navigation satellites and military reconnaissance satellites. Lastly, there are the unmanned Interplanetary probes, These include some of the most sophisticated and complex spacecraft, with future missions scheduled to carry robot explorers.

For manned missions safety of the crew is of paramount importance. Consequently, all the critical electronics are designed to be completely fail-safe. However, the presence of a human crew means that non-critical areas can, in the event of failure, probably be repaired and replaced by the crew or over-ridden by direct human control. In addition, flights are of short duration, and much of the electronics will be within the pressurised crew compartment and thus will not be exposed to the very harsh environment of space. The Shuttle has five general purpose computers, four of which can be configured as a triple-redundancy fail-safe system for highly mission-critical operations such as flight control during re-entry.

In unmanned systems, the main criteria is reliability over a very long period, ten years or more is not unusual, with full exposure to the environment of space. This means that they have to be capable of being remotely configured, maintained, and repaired. Despite the use of the Shuttle to repair the Hubble Space Telescope, there is, in general, little possibility of having someone physically repair a faulty satellite, and absolutely no chance of intercepting and repairing an interplanetary probe.

This need for very long term reliability in a very hostile environment means that such systems have to incorporate a great deal of redundant circuitry. They have to be testable and reconfigurable from the ground, so that a faulty part of the circuitry can be switched out and replaced with a functioning circuit. For interplanetary probes, some of this testing and reconfiguration must be done automatically by the spacecraft itself, since signals between the spacecraft and Earth can take too long.

The rating of spacecraft electronics

I have already said that space is an extremely hostile environment. We all know that space consists of a near vacuum and this fact by itself is of little concern to the designers of spacecraft electronics. Indeed, it can be beneficial since having no atmosphere means that there is no oxygen and therefore no problems with oxidation.

However, operating electronic circuitry in a vacuum brings other sorts of problems, the biggest of which concerns the dissipation of heat. In operation, all electronic circuitry generates heat. Normally this is dissipated by a combination of convection with the aid of air currents, and conduction through the housing PCB etc plus a certain amount of heat radiation.

Convection is the primary means of cooling electronic circuitry, hence the inclusion of a fan in many pieces of equipment. But in the absence of any air, it is impossible to cool any circuitry in this manner. Conduction only moves unwanted heat from one place to another (specialist high efficiency conduction devices known as heat pipes are frequently used in spacecraft for this purpose). This leaves just one method of getting rid of unwanted heat; radiating it out into space.

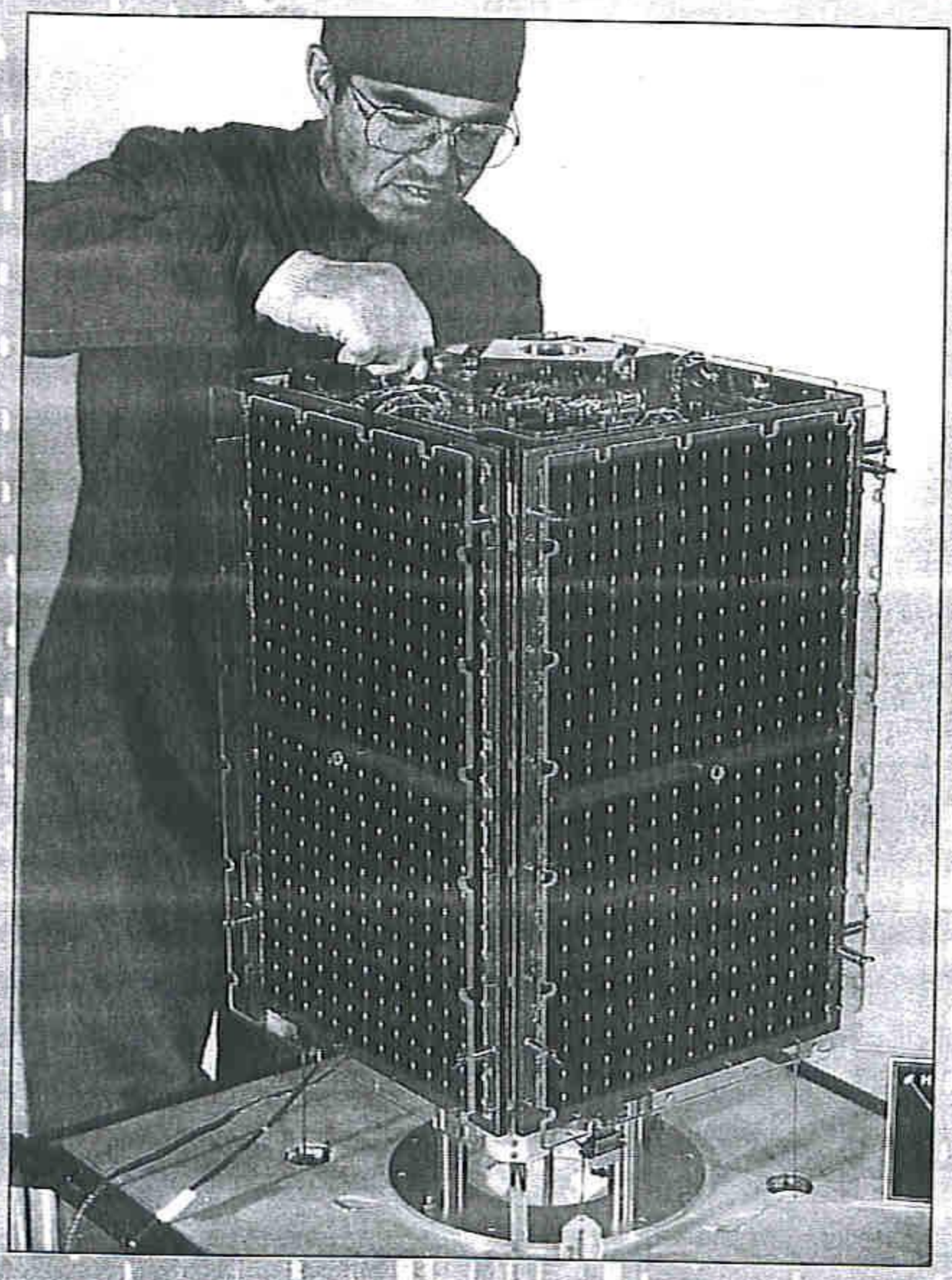
Reducing the heat generated by circuitry is therefore a very important component in the design of space-based electronic systems. This is a requirement which also coincides with the

need for circuitry which uses very little power (power is usually provided by an array of solar cells feeding a battery, battery power being used when the satellite is in the Earth's shadow).

The movement of a satellite in and out of the Earth's shadow is the cause of yet more heat-related problems. The lack of any atmosphere in space means that a spacecraft is exposed to the full radiation from the sun. Much of this solar radiation can be reflected away from the electronics, but it still means that temperatures can rise to +55C when the spacecraft is in the sun and then plummet to -30C or lower when it is in the Earth's shadow. The electronic systems on a spacecraft must therefore be capable of operating successfully over this very wide temperature range.

This repeated oscillation between being very hot and very cold can lead to enormous thermal stresses within components due to differential expansion and contraction rates. These stresses can easily lead to failures of joints, and thus failure of the circuit. Overcoming these problems requires very special construction techniques which reduce the expansion differentials to a minimum.

Thermal stress is just one source of mechanical failure in spacecraft electronics. There are also the enormous stresses caused during launch. Components need to be designed to withstand forces up to 30 gravities. In addition, there are high levels of vibration associated with the launch, all of which can easily lead to component or joint failure.



Solar radiation is not the only source of radiation encountered in space; there are other forms of radiation which are potentially far more damaging to electronic circuitry. There are high levels of electromagnetic disturbance present in space, capable of causing induced currents that will destroy ordinary circuitry. In addition, electromagnetic disturbances can, if the system is not very carefully designed with electromagnetic shielding, arise from other satellite systems due to the fact that all the circuitry is in very close proximity.

In space there are also large numbers of high energy particles (cosmic rays) which can, at best, cause a serious glitch in circuit operation and, at worst, literally blast holes in integrated circuits. To a degree, semiconductors will, if switched off, heal themselves of much of the damage caused by a high energy particle strike. But this is often impossible in mission critical applications.

One way of limiting the potential damage is to shield the circuitry using aluminium plates. However, this approach often has limited success since one high energy particle hitting a plate can, if its energy is high enough, produce more than one high energy particle coming out the other side of the plate. This is therefore a potential cause of problems since the effects of radiation damage are cumulative; the more high energy particles which hit an IC the sooner it will fail.

The standard unit of radiation is the rad, and a standard commercial IC will, in general, fail after it has been exposed to several thousand rads of total radiation. However, susceptibility to damage depends upon the fabrication technique employed in making the IC. Thus, amongst commercial chips, old-fashioned TTL fabrication is probably the least prone to damage. Somewhat more prone to damage are PMOS, and CMOS fabricated circuits, whilst NMOS is probably the most susceptible.

For applications where radiation damage is a problem, circuits can be produced in versions which are designed to be radiation hardened. Making such chips involves special fabrication techniques, usually based upon CMOS, and rigorous testing procedures, and in consequence are enormously expensive. But whereas a commercial chip will fail after cumulative exposure to a few thousand rads, special radiation hardened chips will stand up to a cumulative exposure of several million rads, thereby ensuring a long life in a high radiation environment such as space.

The combined effects of the harsh space environment mean that one cannot build a satellite with the same types of components and construction techniques which one would employ to build, say, a personal computer. If the system is to work reliably for a reasonable period then special care has to be taken in selection of components and the way that they are assembled.

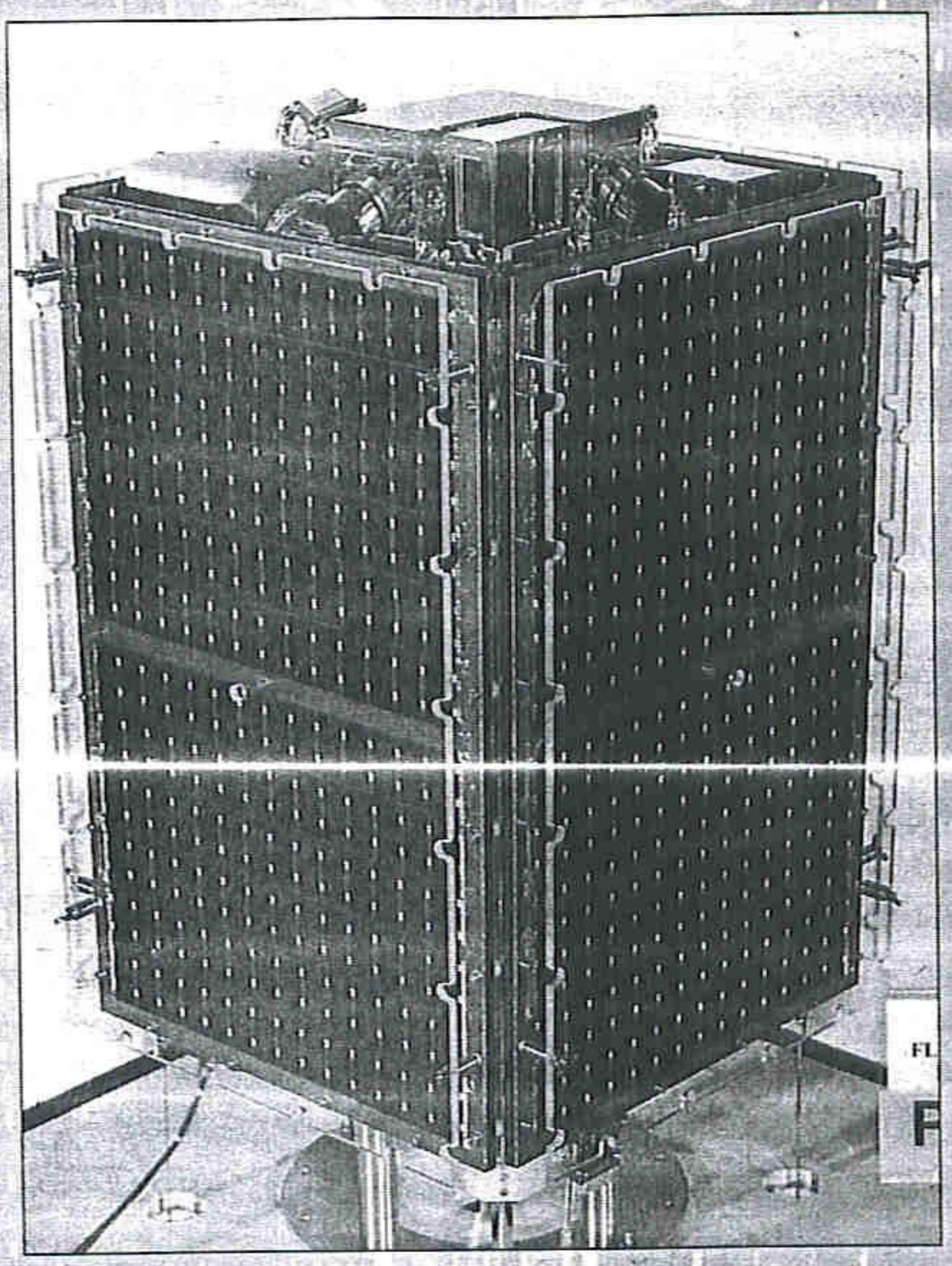
Types of spacecraft electronics.

Broadly speaking, spacecraft and satellite electronics can be broken down into four broad categories. These are the power supply system, radio transmitter/receiver equipment, the main control system and the main payload, which could range from specialist scientific experiments to highly sensitive cameras, plus all the associated data input equipment.

The power supply systems usually consist of a large array of solar cells, organised either as wings that can be tilted towards the sun or actually wrapped around the outer body of the spacecraft. On manned missions such as the space shuttle, power is usually provided by fuel cells which produce electricity as a result of the catalytic oxidation of hydrogen or alcohol. The continuous power generated by a spacecraft is rarely more than a few kilowatts, and there are battery storage systems to allow for fluctuations in power generated, such as when the satellite moves into Earth shadow, and fluctuations in power requirements.

Without a power source, a satellite will cease to function as soon as its batteries are drained. This happened just a few weeks ago to the US weather satellite NOAA13, which functioned for just 13 days after launch. Analysis of the system showed that it failed because primary power was not supplied thanks to a 30mm screw shorting out the circuit - $77million worth of satellite is now dead thanks to a very simple design failure.

All spacecraft, with the possible exception of some spy satellites, have radio communications capability with Earth based control and communications systems. In the case of communications satellites which are used to transmit TV programs to dish equipped viewers, or relay phone conversations and computer data between ground stations in different parts of the globe, the communications equipment is the primary payload. In other applications, such as weather satellites, the communications system is simply a means of controlling the operation of the satellite and transferring data back to Earth.



The communications hardware is a crucially important component of any spacecraft since without communications back to Earth the satellite is effectively dead, an event which has happened recently with several interplanetary probes including an important mission to Mars.

Computer systems

The rest of the electronics on a satellite consist primarily of computers and various data acquisition devices such as cameras. Computers and their associated data systems are now a crucial component of any spacecraft design, although it is surprising to discover that the first spacecraft to actually incorporate an on-board general purpose computer was the Orbiting Astronomical Observatory launched in 1972.

Today, all spacecraft have on-board computers, and it is thanks to them that satellites have the reliability, flexibility, and autonomy of function which is required for today's applications.

We can divide the functions of the software that runs on a spacecraft's computer (today it is more likely to be an array of computers) into six broad sections. These are: attitude determination, command storage, executive, housekeeping, telemetry format control, and instrument sequencing and control.

The attitude determination function is very important since it is used to ensure that the spacecraft always maintains the same attitude in space so that communications antennae, cameras and solar arrays will always be pointing in the right direction. It is, after all, no good having a weather satellite where the camera might sometimes be pointing at the Earth and sometimes at the Moon.

The data required to ensure that the correct attitude is maintained is derived from a number of specialist sensors. Theses include star and sun trackers, earth sensors and gyroscopes and of course, for orbital satellites GPS (see ETI August issue), all of which provide data for the attitude determination program. This program uses the data from all these various sources to compute the actual attitude of the spacecraft with respect to some predetermined datum. The calculated attitude is then compared with the expected attitude and corrections are made if necessary with the aid of small thrusters or reaction wheels.

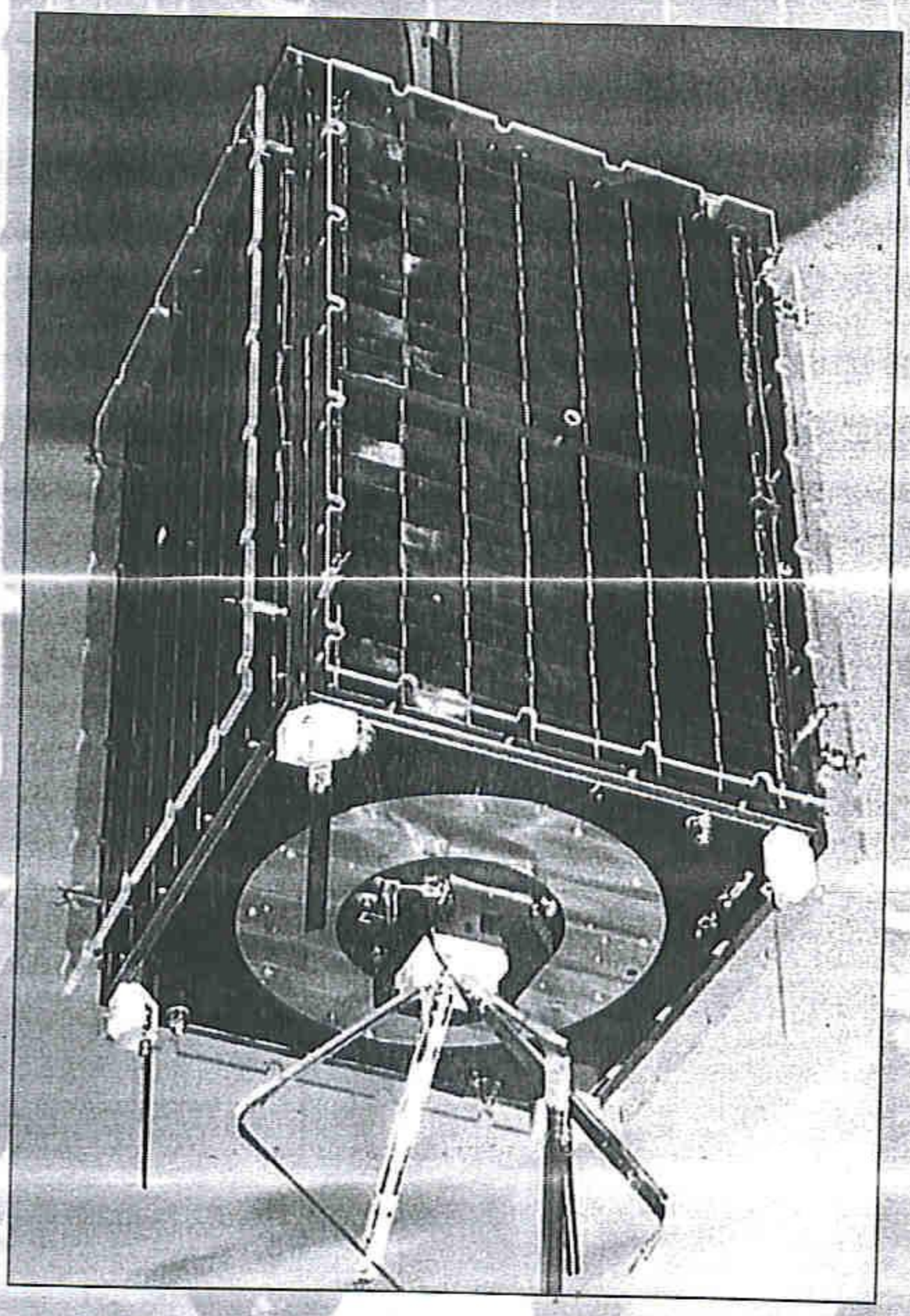
The command storage function is used to store sequences of commands uploaded from Earth based control. These types of commands are time-tagged for execution at specific times and are often necessary because the spacecraft may be out of direct communication for short periods as it orbits over parts of the Earth which do not have communications links. It should be noted that only semiconductor memory can be used on a spacecraft. Disk drives will operate in space but have the unfortunate side-effect that disk rotation will impart a momentum to the spacecraft that will, over time, alter its attitude.

The executive function is simply a means of co-ordinating spacecraft functions using a real-time clock. The executive actually comprises a real-time, multi-tasking, interrupt-driven software system that is capable of sequencing all tasks as well

as handling all necessary inputs and outputs.

One of the regular tasks initiated by the executive is the housekeeping function. This is probably one of the most important functions since with the aid of an array of specialist sensors it monitors the state of the spacecraft. It will look at the power in the batteries, the temperature at key positions, run diagnostic tests on the circuitry and the software. In general, it will check that all the spacecraft's systems are behaving in the proper manner. If they are not, then the fault data will be relayed to ground control and special software routines will be called into play which will allow the system to be reconfigured to overcome the fault.

The key to this procedure is the telemetry format control which performs the actual sampling and format of the spacecraft's telemetry points. These are the measurements which are constantly being relayed to ground control. This code will then perform any necessary reconfiguration in response to both failure and system degradation (note that spacecraft are designed to degrade gradually rather than fail catastrophically).



All the software functions mentioned so far have dealt with the operation of the actual spacecraft. It is only in the sixth and last of the program section that we come upon the functions which control the actual operation of the spacecraft, its instrumentation, communications equipment etc. This is the instrument sequencing and control function and it is used to either monitor and control the instruments directly or to communicate with the instrument's own computer systems. This function is also designed to handle data coming from the instrumentation, perhaps pre-processing it to remove noise, and distortion, perhaps compressing it to remove unnecessary data and also storing it so that it can be downloaded during the periods when there is a direct link with an Earth station

As can be seen from the above, a spacecraft's computer systems can be called upon to perform an enormous amount of computation. The programmes themselves are written in a wide variety of different languages and run on an equally wide range of different processors. As far as languages are concerned Pascal and Ada are common favourites, and, as for processors, they range from the 80286 to the T8000 Transputer. NASA have standardised their computer systems and use the NASSC or NASA Standard Spacecraft Computer. They have also standardised both electronics and software modules for various spacecraft control functions (see Box 1 for an insight into an actual computer system).

The spacecraft's computer software is designed in such a way that it can easily be changed, debugged and reconfigured by a ground based operator using the communications links. Using a ground based support computer, a programmer can reconfigure the software and then reload the spacecraft's memory using the comms links. In this way, it is possible to not only maintain a system but actually improve it.

Into the future

The electronics payload is becoming an ever more important component of the average spacecraft. Most now contain several computer systems networked together and, because each microprocessor based computer is smaller, faster and uses less power than its predecessors of even five years ago, it is enabling designers to build much smaller and cheaper satellites.

A lot of effort is now being spent on using some of this increased amount of on-board computing power to make the system more intelligent. By adding intelligence into the software it is hoped that satellites will become more autonomous and able to repair and maintain themselves automatically. This has the obvious advantage that it does away with the some of the very expensive need for human monitoring of satellite telemetry data.

Indeed spacecraft, in particular interplanetary probes, are becoming more 'robotic'. NASA is already planning a new wave of planetary exploration with the aid of small autonomous robots that can be landed on the surface of planets, the moons of planets, comets and asteroids. Preliminary experiments with this type of spacecraft robotic explorer are being carried out right now with the NASA-funded Dante project. Dante is a walking robot that has been designed to explore one of the most hostile environments on Earth, an active volcano in Antarctica.

The Dante robot has gone through several generations, with engineers gradually overcoming problems associated with its

operation. But within the next few years they will be solved and, by the end of the decade, we should see the first true autonomous robots exploring parts of our solar system. Yet another stage in the application of advanced electronics systems in space and a pioneering use of artificial intelligence and robotics.

Satellites also hold the key to future developments in communications technology. There are now several commercial proposals for using large numbers of small communications satellites in low Earth orbit (LEO) to provide world-wide communications using hand-held terminals. The ultimate in portable phone technology. These proposed networks include the Motorola inspired IRIDIUM project and the more recent Teledesic project proposed by Bill Gates (founder of Microsoft) and Craig McCaw (founder of McCaw Communications).

The IRIDIUM project calls for the use of some 66 satellites in LEO. but this is dwarfed by the $9billion Teledesic project to create an orbiting packet switched data network of 840 LEO satellites, 40 in each of 21 polar orbits that will take them some 700km above Earth. Using a 30GHz carrier uplink it will be possible for users of specially equipped computers and hand-held terminals to communicate with the satellites.

These communications will then be automatically routed around the satellite network and sent via a downlink to the receiver computer or terminal. The Teledesic system is being designed to handle voice, 16kbit/sec data, and because of the increasing use of multimedia systems 2GBits/sec video.

The infinite frontier

The satellite and space technology are now a vital part of our technology and information infrastructure. In the future they will become even more important. What started as a weapon of mass destruction has now become a force that is bringing the people of the world together.

The development of modern space technology and the development of advanced electronics has gone hand in hand. Each has propelled the growth of the other. In the near future, concepts like IRIDIUM and Teledesic will provide the driving force for new generations of consumer electronics products, new applications and new industries, whilst the work on robotic explorers will inevitably lead to the development of advanced mobile, intelligent, robotic systems for use on Earth.

Such developments are only the beginning. Space is the final frontier, but it is also an infinite frontier full of infinite possibilities for technology, and electronics in particular.

Microsatellites in Guildford.

Just outside Guilford in Surrey, a small pioneering British company is building and launching satellites. Since 1981 they have launched eleven, the twelfth is just being built. Not the great big highly expensive satellites that we are all familiar with, but small (50Kg), relatively cheap ($1-2million), very high technology satellites, the so-called microsatellites.

The company is Surrey Satellite Technology (SSTL) and they are an unknown British success story in an area of high technology which most people think is dominated by the Americans and Russians. Indeed, so successful have they been in designing a revolutionary class of small satellites that NASA have just ordered two slightly larger satellites to be designed and built by SSTL at a cost of $110million each.

This NASA order is an acknowledgement of the fact that this small company just 34 people, is way ahead of NASA, or anyone else, in this particular area of technology. The company was founded by its technical director Professor Martin Sweeting in 1979 as part of the University of Surrey and has now become an University owned company, the income from which is used to support an academic Centre for Satellite Engineering Research at Surrey.

Besides building satellites SSTL also provides ground stations and telemetry support for the satellites owners. It also provides training and technology transfer for its customers as well as supporting an extensive R&D program. Total R&D expenditure now stands at over $1million per annum and climbing.

The researchers at SSTL are currently working on development of a hybrid rocket motor for in-orbit propulsion, a special three-axis attitude control system, adaptive modulation/coding schemes for low earth (LEO) communications, VHF/UHF/SHF communications systems, and autonomous on-board attitude determination and orbit navigation systems. In addition many of the satellites contain technology which is being proved and tested for other companies, in particular electronics companies.

Two of their recent launches are the HEALTHSAT II, and PoSAT. The HEALTHSAT has been designed to be part of the HealthNet global communications system that is run by a non-profit making US-based organisation called SateLife and is designed to provide desperately needed low-cost communications links between medical institutions and health programmes in the developing world. Indeed, HEALTHSAT has played a vital part in helping the medical program in Rowanda. PoSAT is primarily an Earth resources satellite that has been developed by SSTL in conjunction with a consortium of Portuguese academia and industry.

HEALTHSAT II.

The HEALTHSAT II satellite is a 'state-of-the-art' design that is based upon SSTL's considerable 9 year experience in flying 'store and forward' communications satellites. It was launched as part of a 'piggyback' load on an European Space Agency Ariane (as a result of special deals the Ariane will put a microsatellite into orbit for about $1/2million, in space launch terms a phenomenal bargain) in September 1993.

The project was completed from concept to launch in just one year. SSTL being responsible for the complete design, construction, test and in-orbit commissioning, as well as organising the launching and insuring the satellite during launch. All this within a contract price of just £1million.

This satellite has been designed to support cost-effective store and forward communications using low-cost portable groundstations, basically a transceiver and a portable PC. It has the following characteristics:

Communications link:

* 3 uplink receivers at VHF
* 2 redundant downlink transmitters at UHF
* Omin-direction satellite antennas
* Switchable AFSK/FSK modulation
* 1200/9600 bps uplink, 9600/38,400 bps downlink

* Output transmitter power adaptive from 1W to 10W under computer control
* AX.25 packet communications protocol

Payload:

* Dual redundant 80C186 on board computers at 8MHz
* Highly integrated 80C188 back-up on-board computer at 10MHz
* 48MBytes SRAM on-board memory
* EDAC memory protection against radiation induced Single Event Upsets (SEUs)
* Advanced high efficiency(17.5%) GaAs solar cell array of 1344sq.cm, producing 30W

The large amount of memory on the satellite is needed to store messages coming in on the uplink so that they can be retransmitted on the downlink when the satellite is in range of a groundstation. The memory is organised as a RAM disk, and the controlling software is all written in C. The RAMdisk is protected from SEUs by special software coding and 'wash routines' plus the extensive use of error correction in the transmission protocols

PoSAT - 1

The PoSAT - 1 was launched on an Ariane rocket in September '93. It is primarily a research satellite designed to test a range of satellite systems, but its primary function is as an Earth imaging satellite. In its design, SSTL have drawn upon three previous research and two commercial missions that used similar technology, in particular KITSAT - 1.that was built and launched for the South Koreans in 1992.

The Earth Imaging System (EIS) on board the satellite consists of two charge coupled device (CCD) imagers, two lenses and a Transputer based on-board image processing and data compression system. There are two imaging/lens systems in order to provide a wide field of view with a 2Km resolution and a narrow field with 200m ground resolution.

The imagers are EEV(UK) 576x576 pixel area CCDs digitised to 256 levels of grey. The digitised data derived from these CCDs is stored in a special 2Mbyte SOS (silicon on sapphire) RAM. This RAM can be accessed by two T-800 transputers to allow the image data to be processed to enhanced quality and also to compress the data and thereby reduce storage and transmission requirements.

The EIS data, once it has been compressed and processed, is transferred via the satellite's local area network to an 8MHz 80C186 based on-board computer which then stores the images as files on its 16MByte RAMdisk for subsequent transmission to a ground base. There is sufficient space on the RAMdisk to store 60 images.

To use the EIS system, a ground base sends a command to the satellite's computer to initiate imaging over a particular area of the Earth's surface. In order to ensure that the proper area is imaged, the ground control team use a model of the satellite's orbit to determine at exactly what time the satellite will be over the desired location. Usually a sequence of such image time commands will be sent.

On-board the satellite, the computer runs a real-time, multi-tasking, operating system that is responsible for the automatic operation of the satellite's mission. One of these tasks is to periodically check a 'diary' of timed events, some of which might be the uploaded times at which images must be taken.

It should be noted that the SSTL EIS is different from that commonly used in Earth imaging satellites. It is different because it uses a CCD array rather than a mechanically scanned linear CCD, an arrangement which is usually chosen because it gives a higher resolution image. However, the SSTL approach has a number of major advantages, the biggest of which is the fact that it has no moving parts and is thus far more reliable. In addition, the absence of a motor means that it uses a lot less power and can be much more compact, both important factors in spacecraft design. The CCD array approach also has the advantage that exposure times can be very short, just a few milliseconds, a factor which means that spacecraft stability is less of a problem.

Besides the EIS, PoSAT payloads include a star sensor which is based upon the same technology as the EIS but is optimised for imaging the faint light from stars for use as part of the spacecraft's attitude determination system. The star-field image is analysed by the same transputer based system that is used by the EIS and the measurement data returned to the on-board 80C186 computer.

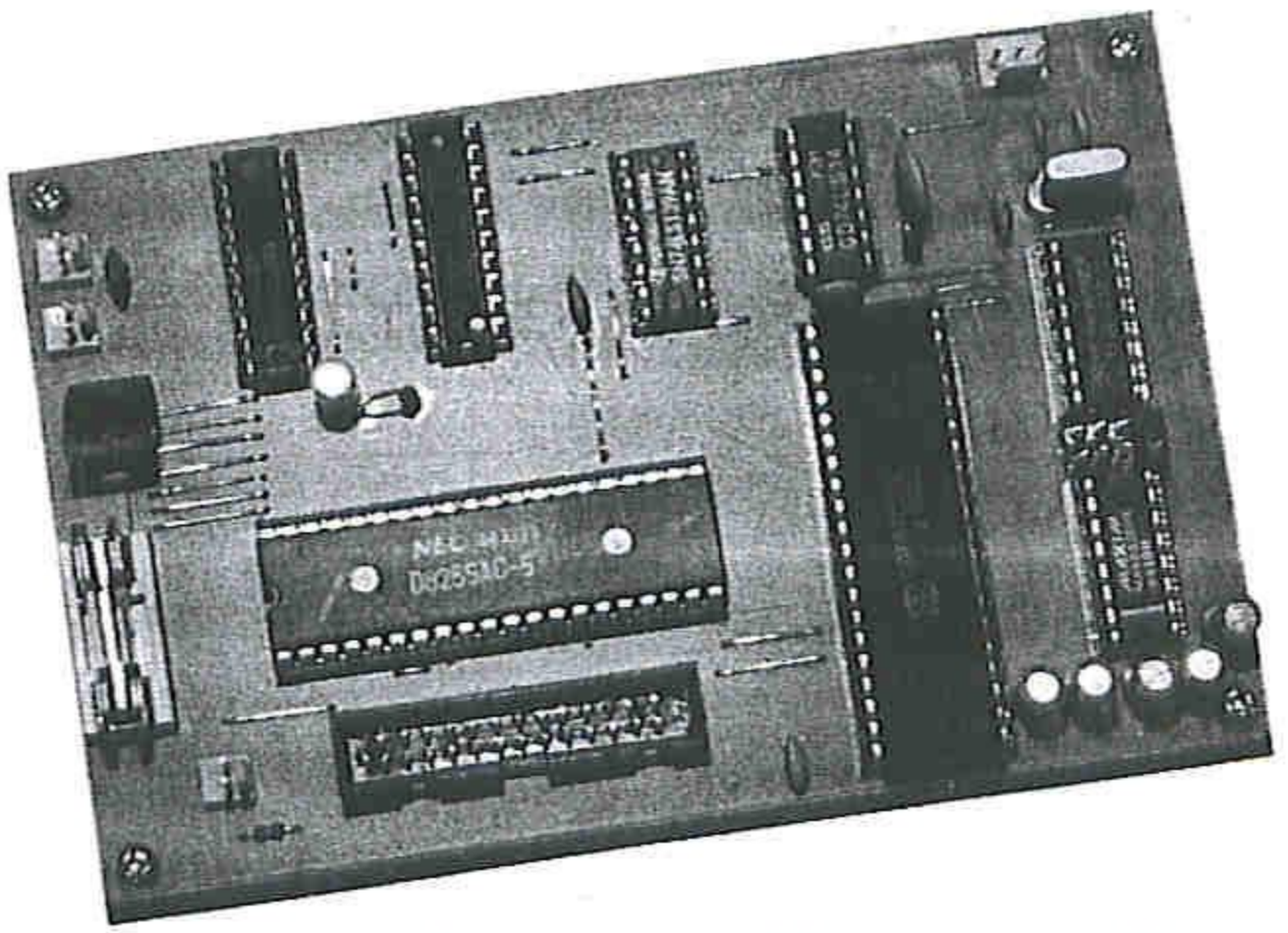
To make satellite position location more accurate the PoSAT carries a Global Positioning by Satellite (GPS) receiver based upon the Trimble TRANS-II receiver. The received data is decoded and filtered by the Transputer system to produce a highly accurate three-dimensional co-ordinate for the satellite together with information about its velocity and maintenance of the on-board time reference. With the aid of the GPS receiver, the satellite can generate its own orbital element set, providing scheduling and synchronisation to other computers, and allow groundstations equipped with a GPS receiver to experiment with applications for real-time differential GPS.

This GPS capability is potentially of enormous commercial importance since it would allow the construction of an 'asset monitoring' satellite, a satellite which could track from space vehicles, ships or planes that are equipped with simple transmitters. This satellite could tell the owners of such assets exactly where they are, thus improving efficiency and also enabling the police to track down, for example, a stolen lorry. Current GPS systems will only tell the vehicle driver where he is, they will not track the vehicle from a central control.

Since SSTL, its customers and its partners are keen to be involved in projects like IRIDIUM and Teledesic, it is not surprising that another experimental payload on PoSAT is a digital signal processing experiment. This consists of two Texas Instruments processors, a TMS320 C25 and a TMS320 C30 with associated ROM, RAM and I/O interfaces. These chips can be used as a programmable communications modem to modulate downlink data from, or demodulate uplink data to, the on-board computer, thereby enabling experiments in modulation techniques for LEO satellite mobile communications.

These experiments are aimed at trying to overcome some of the problems; for example: the varying communications path and link characteristics, the high Doppler shifts, and the problems associated with handing over from one satellite to another in the constellation of microsatellites. The DSP system on the PoSAT should enable engineers to look at ways of overcoming these problems.

As can be seen the SSTL microsatellites such as PoSAT are very sophisticated and complex systems and they provide a significant pointer to some of the applications for satellites over the coming years. It is good to see that despite the almost complete lack of any viable UK space program, at least one UK company is leading the world in space technology.



Versatile 24 line RS232 I/O Interface

Serial RS232 communications are an ideal way of linking a PC to a remote device. In this project Dr Pei An shows how to convert a serial input into a bidirectional I/O port

The RS-232C port is an industrial standard asynchronous serial data communication interface for the purpose of serial data communications between two devices and is available on almost all modern personal computers. It is frequently used for connecting printer, modems and mice to the computers. One advantage of the serial port is its simplicity in linking the computer to external devices. In total, there are nine lines. However, in most applications, lines are sufficient to perform a bi-directional communication. Another advantage is that the communication distance of the RS232 interface can be up to 100 feet. Unfortunately, unlike parallel ports which requires simple interface circuit, the RS232 port requires a slightly complicated interface circuit.

In this article, I am going to introduce a versatile, general purpose RS232 I/O card. This card is connected to the PC's RS232 port via three wires and it is able to provide 24 input/output lines. These lines are organized in four groups, namely, groups A, B, C1 and C2. Groups A and B have eight data lines and groups C1 and C2 have four lines. Each group can be configured as an input or output under the control of the RS232 port. Figure 1 shows the schematics of the card connected to a PC and the pin functions of the 26-way DIL socket from which external circuitry is connected.

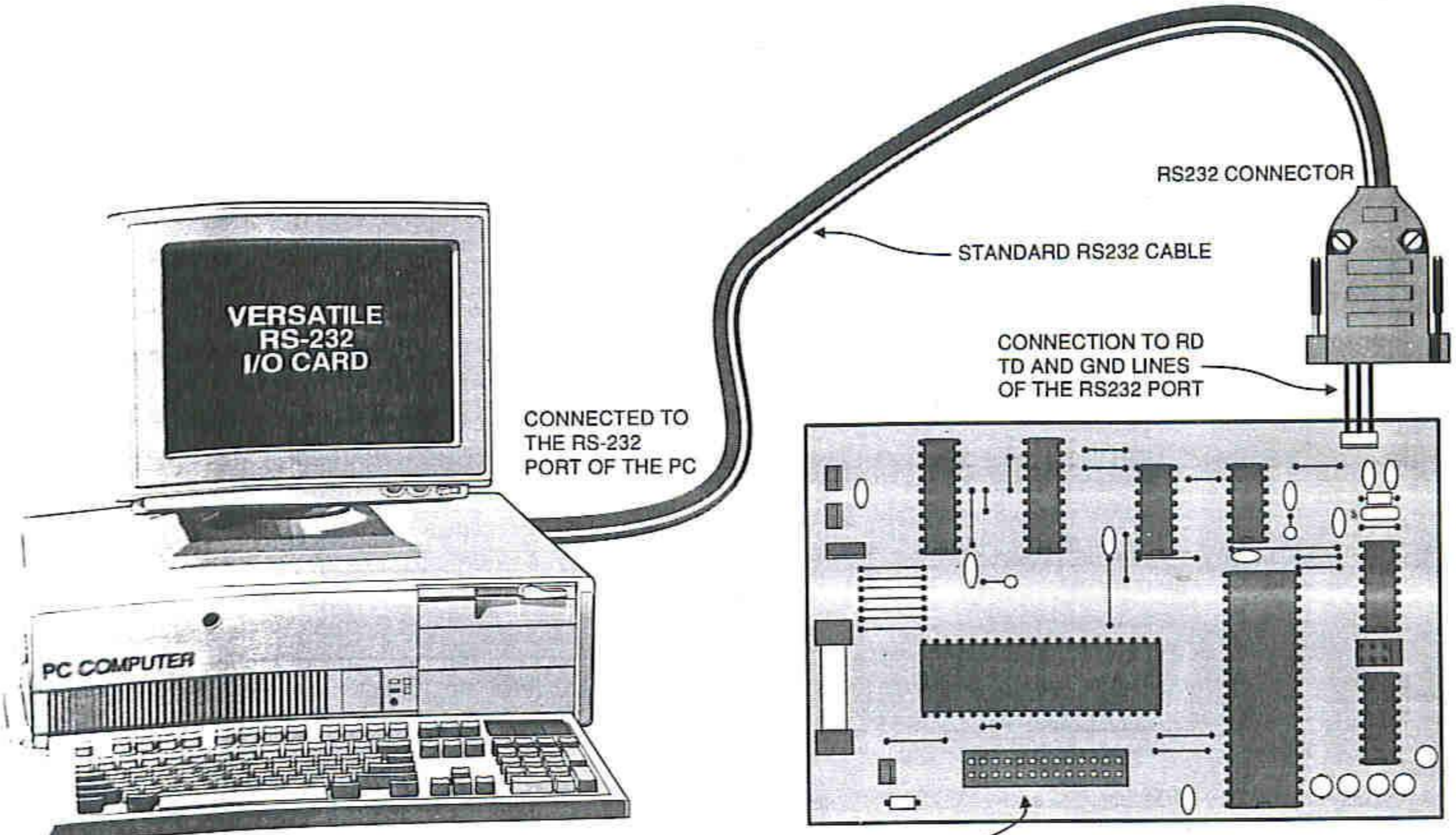
FUNDAMENTALS OF SERIAL DATA TRANSMISSION

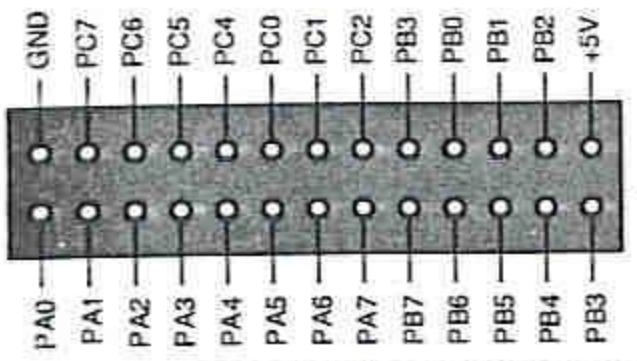
Serial data transmission

Unlike a parallel port, which normally has eight data lines and each time transmits an 8-bit byte in one go, a serial port only needs one data line to transmit the 8-bit byte. The byte is transmitted successively, or in another word, serially. There are two serial data transfer schemes: the synchronous and asynchronous transmissions. In the synchronous data transfer scheme, additional lines are required to transmit handshake or timing signals along with the data line to indicate when the next bit is to be transmitted on the line. The advantage of this data transfer is that the receiver is able to respond to the clock rate of the transmitter automatically. For asynchronous data transfer, the transmitted data themselves contain the information of synchronization and neither handshake nor clock signals is needed. The transmitted serial data comprises a Start Bit, which indicates the beginning of a data transmission. It is followed by Serial Data Bits and then Stop Bits indicating the end of the transmission. An optional Parity Bit can be added between the Serial Data Bits and the Stop Bit for parity checking. The receiver device detects the start bit and receives the subsequent data bits. This data transfer requires that the transmitter and the receiver must have the same clock frequency. Asynchronous data transmission is used in most personal computers. In practice, the asynchronous communication is facilitated by a family of industrial standard computer peripheral ICs known as the UARTs (Universal Asynchronous Receivers & Transmitters). Most PCs use 8250/16450 UARTs and the present card employed a 6402 UART.

SERIAL DATA FORMAT

The format of the serial data transmission generated by UARTS consists of four parts: a start bit, data bits, one parity bit, and at least one stop bit (see Figure 2). The functions of these bits are discussed in detail: When no data is sent, the transmitting data line is at high state (2V-5V TTL level). The beginning of a data





FUNCTIONS OF THE EXPANSION SOCKET

...ersatile RS232 I/O card connected to a PC

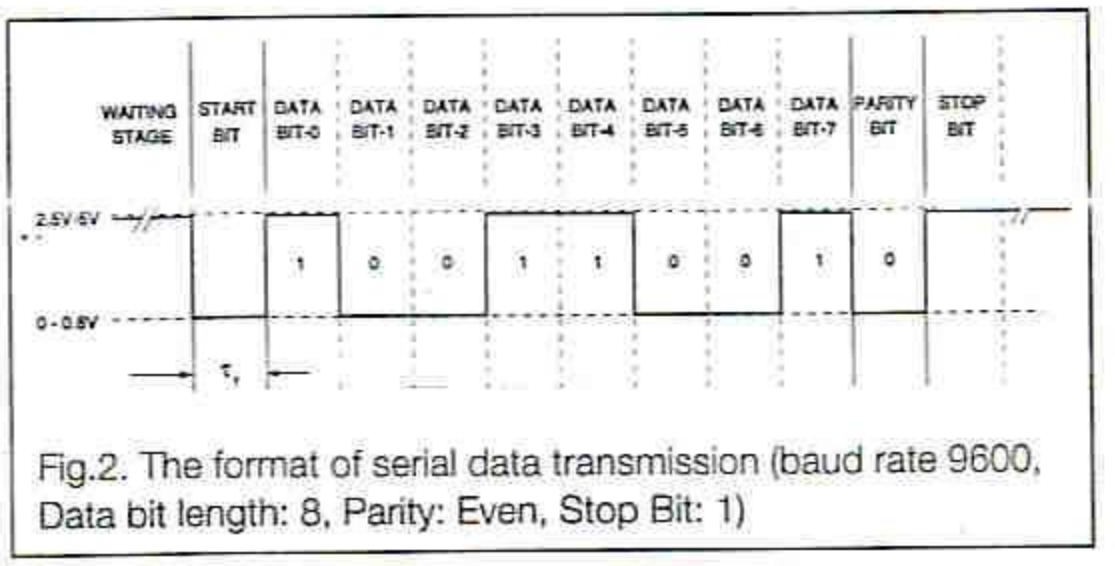


Fig.2. The format of serial data transmission (baud rate 9600, Data bit length: 8, Parity: Even, Stop Bit: 1)

...is indicated by the line going low (0-0.8V TTL ... This bit is the Start Bit. The Data Bits are ... one after another with the least significant bit ... The length of the data bit can be 6, 7 or 8. Following ... data bits is the Parity Bit,which is used to check for errors ... the data transmission. The last bits are the Stop ... The data line goes to high for at least 1 bit time to identify ... end of the data transmission. The stop bit can be 1, 1.5 ... 2 bit length.

The serial data transmission format is generated by the ... the transmitting UARTs. The electronics in ... detect the leading edge of the start bit. It then ... for one and a half bit times before reading the data bit. ... therefore come exactly in the middle of the ... waits for one bit time and reads the second bit. ... the reading comes exactly in the middle of the ... After completing reading the data bits, the ... detects the parity of the received data for error ... and resets itself during the stop bit to wait for the ... data transmission.

The rate at which data bits are sent is measured by Baud ... is defined as 1/(the time between the shortest signal ... see Figure 2). The standard Baud Rates ... are 110, 150, 300, 600, 1200, 2400, ... and 9200. Knowing the Baud Rate, the number of characters (or bytes) to be transmitted per second can be calculated. For example, if the serial data transmission has 8 data bits, one parity bit and 1 stop bit, the total length of serial data bits is 11. The transfer rate for characters is the Baud Rate divided by 11. For example, a Baud Rate of 9600 will transfer 870 characters per second.

Parity check can be ODD, EVEN or NONE. The odd and even parity checks indicate that the total number of ones ('1') in the transmitted serial data is an odd number or even one. This method is the simplest way for checking errors during data transmission. However, it is only reliable to detect single-bit errors. Errors with several bits cannot be detected this way. The parity bit is generated by the electronic circuit of the transmitting UART in such a way that the number of ones ('1') in the data bits plus the parity bit is odd or even as declared. For example, suppose we send a binary byte 01000011 (there are 3 '1's in the byte) from the computer to an external device and the parity has been declared as odd (the total number of '1' is odd). The parity bit will be set 0, since the number of '1' in these 9 data bits is already odd. At the receiver end, the receiving device must also be configured to have an odd parity check. The electronics in the receiver UART will count

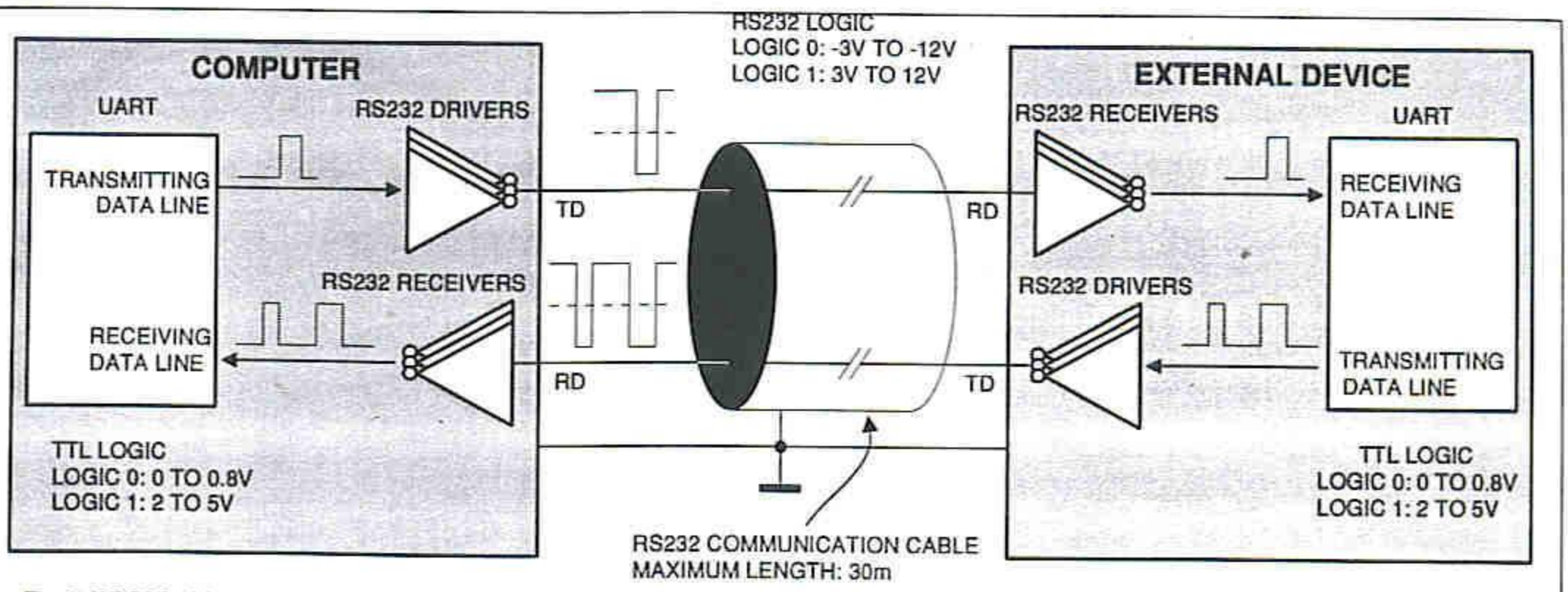
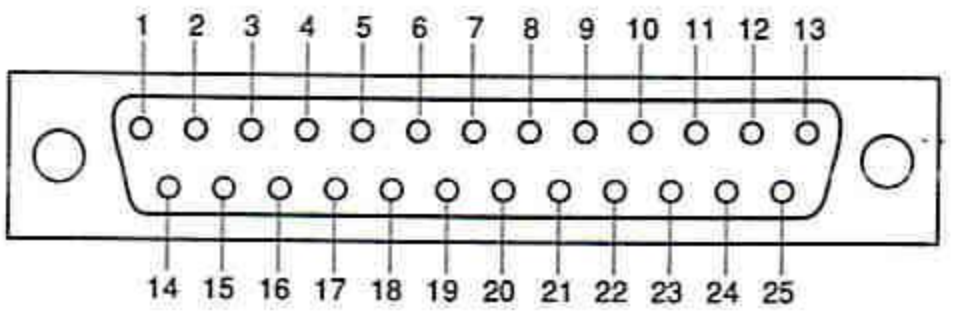


Fig.3 RS232 driver and receiver in PCs and external devices

(a) 9-PIN MALE SOCKET VIEWED FROM THE BACK OF THE COMPUTER



(b) 25-PIN MALE SOCKET VIEWED FROM THE BACK OF THE COMPUTER

Fig.4. Pin layouts and functions of the RS232 connectors of PCs

STANDARD PIN FUNCTIONS OF THE RS232 CONNECTORS

25 PIN	9 PIN	NAME	DIRECTION (FOR PCS)	DESCRIPTION
1		Prot		Protective ground
2	3	TD	OUTPUT	Transmit data
3	2	RD	INPUT	Receive data
4	7	RTS	OUTPUT	Request to send
5	8	CTS	INPUT	Clear to send
6	6	DSR	INPUT	Data set ready
7	5	GND		Signal ground (common)
8	1	DCD	INPUT	Data carrier detected
20	4	DTR	OUTPUT	Data terminal ready
22	9	RI	INPUT	Ring indicator
23		DSRD	I/O	Data signal rate detector

the number of ones in the received data. If the data does not have an odd parity, an error signal is given indicating that a transmission error has occurred. If the parity check is declared as NONE, the parity bit will not be generated and checked.

LINE DRIVERS AND RECEIVERS FOR RS232 LINK

The signal from the UARTs have a TTL level. A logic high corresponds to a voltage ranging from 2V to 5V and a logic low corresponds to a voltage between 0V and 0.8V. Signals of this voltage level can not be transmitted reliably over a long distance. To solve the problem, RS-232 drivers are used to boost up the voltage of the transmit line from TTL level to the RS-232 level which is substantial higher than the former. Receivers are used in the receiving device to convert the RS232 voltage level to the TTL level (Figure 3). This arrangement enables the RS-232 interfaces to communicated over a maximum distance of 100 feet. The voltage conversion between TTL and RS232 is facilitated by industrial standard RS-232 driver and receiver ICs. In the present application, a MAX232CPD RS-232 driver/receiver IC is used. It is noted that the RS232 drivers and receivers have an inverting action. A TTL logic high is translated to -3V to 12V at the RS-232 side and a TTL logic low is translated to +3 to 12V.

RS-232 INTERFACE ON PCS
PC RS-232 INTERFACE

A standard RS-232 interface is a 25-pin interface, which is housed in a 25-pin male D-type connector. A short version of the interface is also commonly used on PCs which is a 9-pin male D-type connector. The pin functions of the connectors are shown in Figure 4. The functions of mostly used pins are shown below:

Prot:	Protective ground line. It is connected to the metal screening in the cable and the metal chassis of the equipments
GND:	Ground line. It provides a common voltage reference for input and output signals.
TD:	Transmitting Data line on which serial data is transmitted.
RD:	Receiving Data line on which serial data from other RS232 devices is received.
RTS:	Handshake line. It indicates a RS232 device is Ready To Send data out.
CTS:	Handshake line. It indicates a device is ready to receive data transmitted to it.
DTR:	Handshake line. It is used to indicate that the RS232 transmit ting device is ready
DSR:	Handshake line. It is used to indicate that the RS232 receiving device is ready.

SOFTWARE CONTROL OF THE RS-232 INTERFACE

A PC supports up to 4 RS232 interfaces. They are labelled COM1, COM2, COM3 and COM4 with each controlled by an 8250/16450 UART. There are 9 I/O ports associated with each UART for accessing 9 internal registers of the UART. The base I/O addresses of these registers are shown in the following table:

Interface	Base Address	Addresses
COM1	3F8h	3F8-3FFh
COM2	2F8h	2F8-2FFh
COM3	3E8h	3E8-3EFh
COM4	2E8h	2E8-2EFh

Each UART internal register has a particular function. As an example, the functions of the registers for COM1 interface (base address=3F8) are listed in the following table:

Offset	Address	Function of the port	Description
00h	3F8h	Receiver buffer register	
Transmitter hold register		Holding the received data	
		Holding the data to be transmitted	
01h	3F9h	Interrupt enable register	Setting the mode of interrupt request
02h	3FAh	Interrupt identification register	Checking the mode of interrupt request
03h	3FBh	Data format register	Setting the format of serial data transmission
04h	3FCh	Modem control register	Setting the UART modem control logic such as RTS and DTR
05h	3FDh	Serialization status register	Containing information on status of the receiver and transmitter section
06h	3FEh	Modem status register	Containing the current signal status of DCD, RI DSR and CTS
07h	3FFh	Scratch-pad register	Acting as an one-byte memory

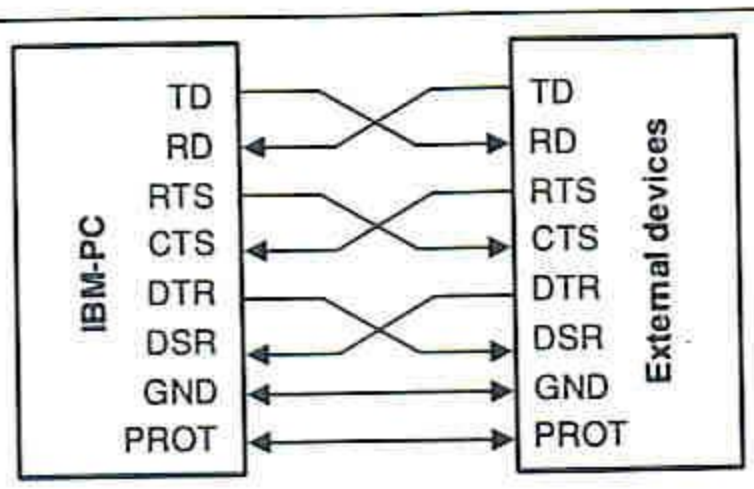
The functions of these registers are not discussed in detail in this article. Readers who are interested in this can read the manufacture's data sheet. In the present project, only two registers are relevant to us: the register accessed by Port 3F8h (offset 00) and the one accessed by Port 3FBh (offset 03). The former is the data receiving/transmitting register. If we write a data byte to the register, the byte will be transmitted through the TD (Transmitting data) line. If we read a byte from the register, the received serial data from the RD line is fetched. The other register defines the serial data transmission format, e.g. the Baud Rate, the number of data bits, parity check and the length of stop bits. We can write a byte to the register to define the serial data format. However, as we see below that we can use a DOS 'MODE' command to define the data format in a convenient way.

A DOS command 'MODE' can be used to set up the format of the serial data transfer for COM1 through COM4. The syntax of the MODE command is:

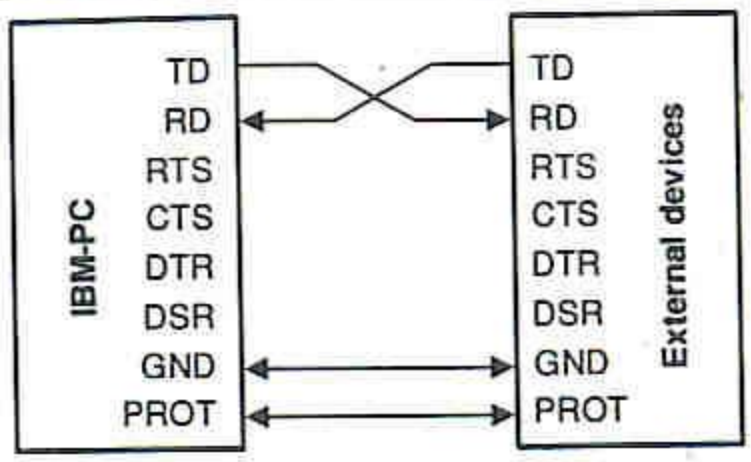
```
MODE COMm: baud=b, parity=p, data=d, stop=s,
retry=r
     or
     MODE COMm: b, p, d, s, r
```

Functions of the specifiers in the command are shown below:

COMm	specifies the number of the RS232 port, valid values for m are 1,2,3 or 4.
baud=b	specifies the first two digits of the Baud Rate. For example, to set a Baud Rate of 9600, b should be 96.
parity=p	specifies how the system uses the parity bit to check for transmission errors. The p value can be one of the following: n(none), e(even), o(odd), m(mark) or s(space)
data=d	specifies the number of data bits in the serial data transmission. Valid values for d are 5, 6, 7 and 8.
stop=s	specifies the number of stop bits. The valid values for s are 1, 1.5 or 2. if the Baud Rate is 110, the default value is 2.
retry=r	specifies the retry action to take if a time-out error occurs.

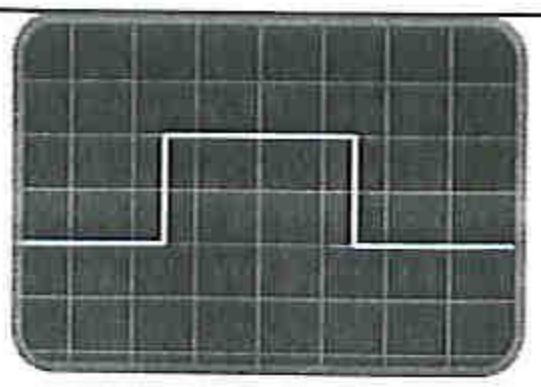


(a) Null modem for connecting two RS232 data terminal devices

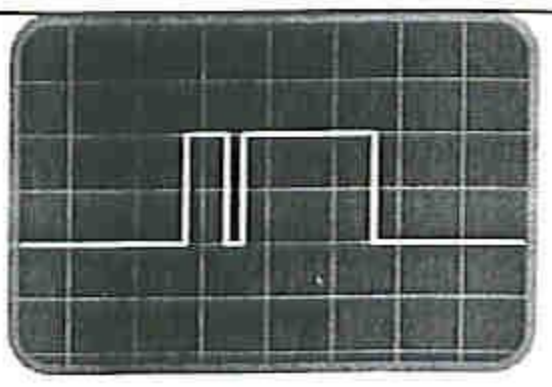


(b) RS232 port pin connections between two devices via three lines

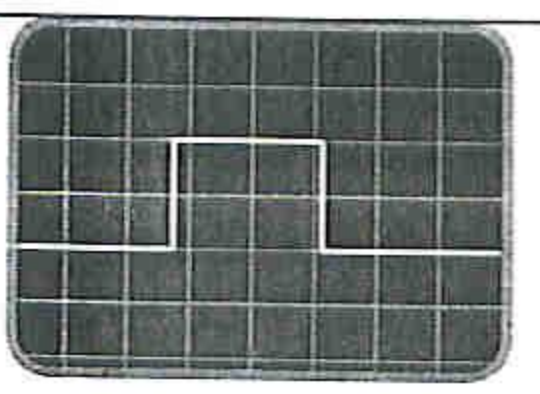
Fig.5 RS232 pin connections between PCs and external devices



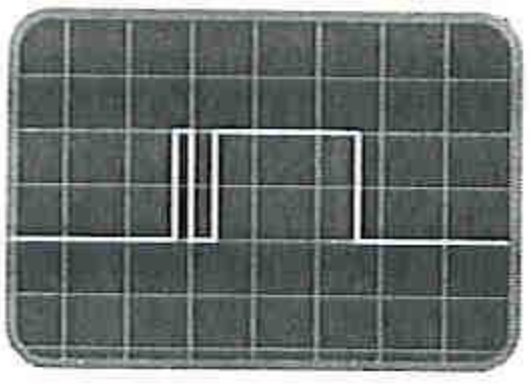
(a) Sending out a binary value of 0



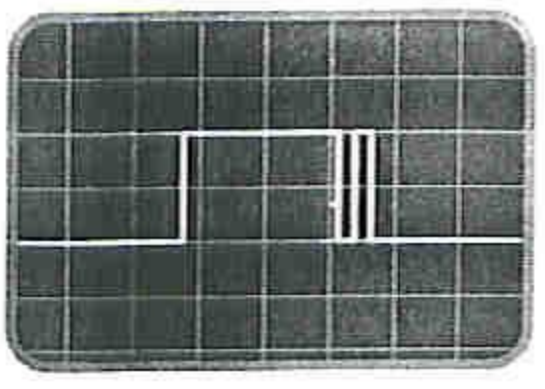
(a) Sending out a binary value of 2



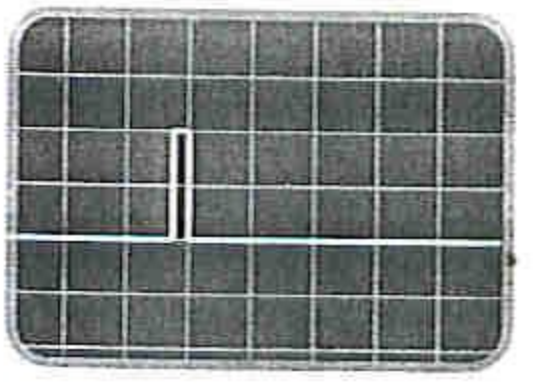
(a) Sending out a binary value of 128



(a) Sending out a binary value of 1



(a) Sending out a binary value of 64



(a) Sending out a binary value of 255

Fig.6 Waveform of TD signals shown on the oscilloscope when outputting a binary byte from the RS232 port.

To send data out of the COM1 interface, we need to write data to Port 3F8h. The following commands can be used:

```
OUT 3F8h, X              (in BASIC)
PORT[$3F8]:=X  (in Turbo Pascal)
```

In which X is the data to be sent (in decimal).

To fetch data from the COM1 interface, we read data from Port 3F8h and the following commands can be used:

```
    Y=INP[3F8h]              (in BASIC)
Y:=PORT[$3F8]        (in Turbo Pascal)
In which Y is the input byte (in decimal).
```

CONNECTION BETWEEN RS232 INTERFACES

Two common RS232 links are shown in Figure 5. The arrows in the figure show the direction of the data flow. Figure 5a shows a connection known as a null modem. In this arrangement, some lines of the two RS232 interfaces are used as handshakes between the two devices. Figure 5b shows a connection which only consists of 3 lines.

AN EXPERIMENT WITH THE RS-232 INTERFACE

Finally, let us conduct a simple experiment which will help us to understand how the serial data is transmitted. Try to use the knowledge we know to explain what we have observed. This experiment needs an oscilloscope. The procedure of the experiment is shown below:

Step 1: Type the following command under DOS prompt C:\
MODE COM1: 96,n,8,1
This command will configure the COM1 interface to the following specification: 9600 Baud Rate, 8 bit of data length, no parity check and 1 stop bit.

Step 2: Type one of the following programs into the computer. These programs will cause the COM1 to send out the specified value (Byte_Value) continuously.

(In Turbo pascal 6)

```
Program RS_232_Experiment;
var     Byte_Value:byte;
Begin
Byte_Value:=0;{Input the byte to be sent by COM1
which is set to be 0,1,2,64,128 and 255}
repeat
port[$3F8]:=Byte_Value;{$3F8 is the address of the
COM1 transmit data register}
delay(1);{Delay 1 ms}
until keypressed;{Press any key to stop the
program}
end.
```

(In BASIC language)

```
5  Byte_Value=0 ;Input the byte value to be sent
from COM1, which is set to be 0,1,2,64, 128 and 255
10 Out 3F8h, Byte_Value
15 delay(1)
20 goto 10
```

STEP 3: Connect the oscilloscope probe to the COM1 serial port
The negative pole of the probe is connected to the GND line of COM1 (Pin 7 for 25-pin connectors and Pin 5 for 9-Pin connectors) and the positive connected to the TD line (Pin 2 for 25-pin connectors and Pin 3 for 9-pin connectors).

STEP 4: Run the above program and observe the waveform of the TD signal on the oscilloscope.
The control knobs of the oscilloscope should be finely adjusted. TD waveforms on the oscilloscope are shown in Figure 6 when sending values of 0, 1, 2, 64,128 and 255 to COM1.

STEP 5: Try to explain these waveforms on the basis of the knowledge we have.

Next month
Dr Pei An will show how to build an RS232 to parallel interface board which will allow one to connect a wide range of circuits and projects to the serial port of any personal computer.

TRANSPUTER BASED SINGLE BOARD COMPUTER

In the fourth and final part of this project to build a Transputer based single board computer, Andy Papageorgiou and Mark Robinson take a look at programming the board

This month we explain in detail how the I/O of the transputer card is organised and present some useful multitasking software to illustrate the transputer's features. A basic familiarity with transputer assembly language is assumed so now is the time to read the literature. A summary of the transputer integer instructions is provided as a text file on the examples disk as a reference.

The I/O mapping

The I/O occupies the upper 128 bytes of the memory map (addresses 7F80H to 7FFFH) whenever the ROM is paged out. This area also contains the ROM page register, which is accessed using a special mode of the transputer and so is invisible to normal read and write operations. The complicated structure of the I/O area is the main reason for using a GAL in this application: a memory map like this would be nearly impossible to implement in discrete logic.

Figure 1 summarises the organisation of the top 128 bytes of memory as seen by "normal" memory accesses, i.e. those performed using the STNL and LDNL ('STore Non Local' and 'LoaD Non Local') instructions. Since the upper 64 bytes are an exact mirror of the lower ones, we will restrict our discussion to the area from 7F80H to 7FBFH.

The I/O space is divided into four word blocks by IC8 and IC9. The lower six blocks are allocated to the I/O port ICs, the seventh block is unused and the eighth is assigned to the port data direction register. All the I/O devices are 8 bit devices occupying the low byte of the 16 bit word. The upper byte is always read from or written to the RAM.

Write operations are straightforward; a STNL to any of the four words in a block will write to the port or register associated with that block. Reads are slightly more complicated. A read operation from the data direction register has no effect. A read from a port either reads from the port directly or from the port input latch depending upon which address is read from. Data is put into the input latch on the rising edge of the port input strobe signal.

The following fragment of code illustrates the I/O operations; it sets ports 0-2 to output and ports 3-5 to input, then reads from port 3 directly (i.e. not the latch) and writes the value to port 2.

The ROM page register also occupies the top 128 bytes of the memory map. The transputer has two instructions for accessing byte wide memory - SB (Store Byte) and LB (Load Byte). The sequence of signals on the control bus are different for a Store Byte to an odd address than for any other memory write operation. These signals are detected by the GAL and used to operate the ROM page register.

The ROM is paged out by executing an SB instruction to any odd address between 7F81H and 7FBFH; the top half of the memory will then be occupied by RAM and I/O. Executing an SB to an odd address between 7FC1H and 7FFFH will page the ROM in, the top half of memory will then be occupied by ROM. The following code fragment illustrates paging the ROM out.

```
LDC 0

LDC $7F81

SB                      ; Page ROM out
```

The Boot Loader

Last month, we tested the transputer card by using the boot-from-link ability. While booting from link allows programs to be developed and debugged quickly, it has a couple of snags. First, once one process is loaded and running no further processes can be loaded, and second, the program must be less than 256 bytes long and has to be loaded into a preset memory location.

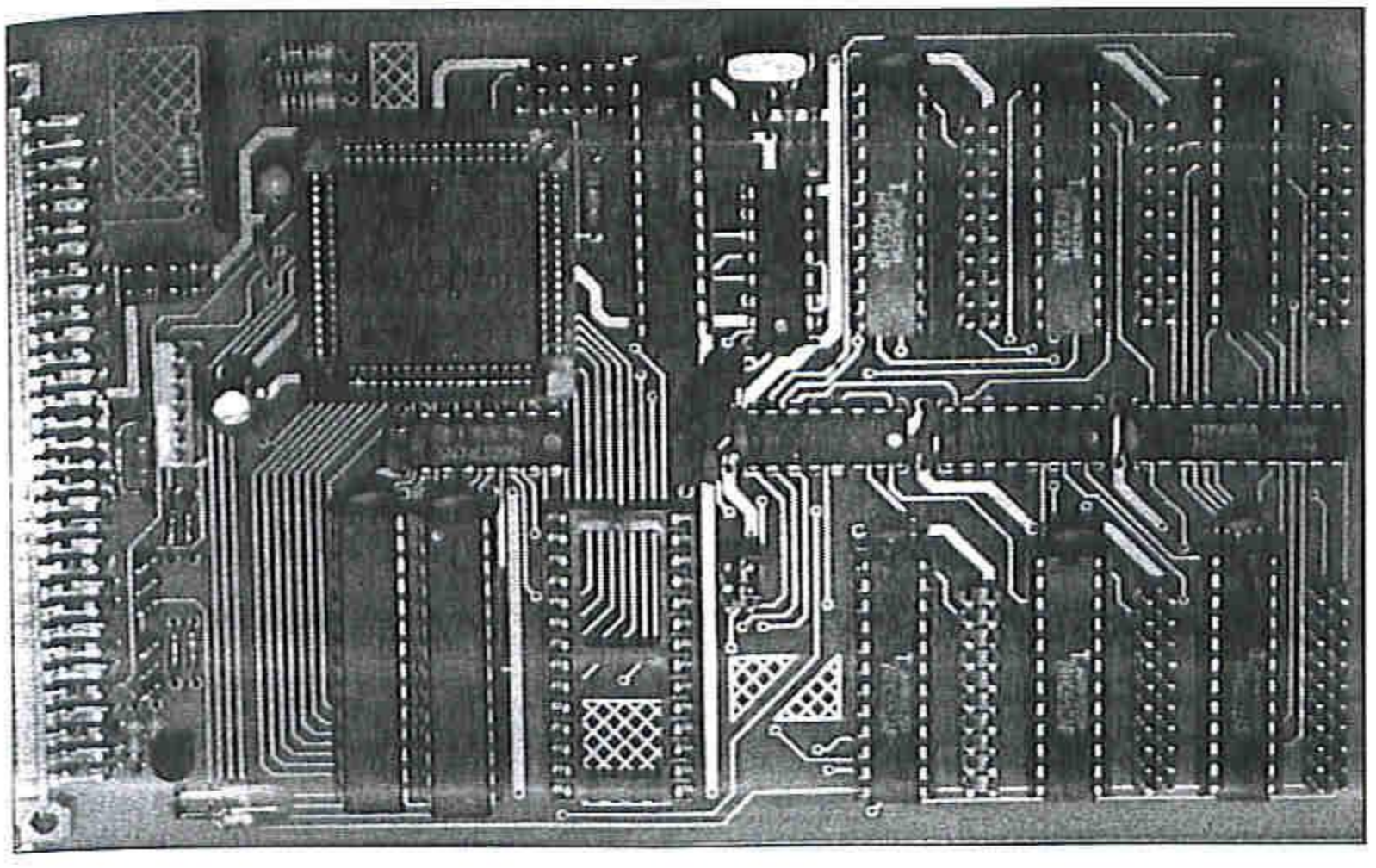
To solve these problems, we will now describe a small boot loader which will monitor a link and launch processes of any length continuously. This program, together with a program to run on the host PC, forms a rudimentary operating system, will give us our first taste of concurrent programming, and allow easy development of parallel programs. The code can also be used with some modification to program the EEPROM or NVRAM.

Listing 1 is the transputer boot code which launches the processes, (process queues) the program monitors the status of link3. When data arrives on link 3, it is taken to be program code and spawned as a child process. The sequence expected by the program is two bytes indicating the address to store the new process from, two bytes to indicate the length of the process and finally the program code.

The hard work of this program is done in the loop from getwords onwards. This code illustrates just how easy it is to communicate between processes and to launch child processes using the transputer. The loop performs two IN instructions, the first to fetch the two word header, which is placed in the workspace, and the second to fetch the program code. Once the code has been fetched, it is launched as a process using the STARTP instruction. A minor complication is that the address is absolute, but the STARTP instruction requires an offset address relative to the next instruction. A bit of simple maths is used to sort this out.

It is interesting to note that this program will be descheduled at the IN instruction, and will not be rescheduled until data arrives. Hence,

```
LDC     3F      ; ports 0-2 o/p ports 3-5 i/p
LDC     $7F83   ; address of the Data Direction
                Register
STNL            ; write to DDR
LDC     $7F80   ; Base of I/O area
LDNL    $0      ; 4*Port_no direct (4*Port_no+1 latched)
                ; Read port 3 and leave result on the
                stack
 LDC    $7F80   ; Base of I/O area
STNL     $02    4*Port_no, writes  result of
                previous read  to
                port 2
                ; Ports are numbered 0-5
```



apart from a hundred or so bytes of memory, the program uses no resources at all unless loading a process.

In listing 2 ,is a C program which is used to communicate with the loader. It is given as dump.c on the examples disk, and for those without a C compiler dump.exe is the executable. The program is basically identical to the boot loader tboot.c from last month, except that the program prompts for the address to load the process to, and sends the header words required by loader.asm.

The final link in this chain is listing 3. This is the error light flashing code from last month stripped down to its bare minimum. This code is called errlite.asm on the examples disk. Assemble the code, then send it to the loader using the command

```
dump errlite.asm
```

The program will ask you for an address to load the code to, type 8800 for now. Any address in RAM could be used except the lower 200 bytes or so where the reserved memory and the loader program are situated. Hopefully, you will be rewarded with the error light flashing.

Now for the real concurrent processing. Re-assemble errlite.asm with different on and off times and resend it to a different address (8900 for example) without resetting the transputer. Repeat this a few times and you should see the error light behaving very strangely as each process fights for control over it - the ETI chaotic LED flasher!

Using the Transputer Card.

The example programs given here are fairly trivial, and were mainly designed to illustrate the main features of the card. What to do with the card is obviously down to the requirements (or imagination) of the user. This card can be used anywhere where an embedded microcontroller is required, although in some applications it would be hard to justify using the transputer rather than an 8 bit micro. Where this board would find an advantage is real-time or semi-intellegent control; digital PID controllers for example.

The application where this card really stands out though is robotics. Autonomous robot vehicles need a reasonable amount of on board processing power to make sense of the world around them. Neural networks (usually simple single-layer perceptron types) are becoming popular in autonomous robots, where they allow the robot to 'learn' how to solve problems in the environment. This is a fascinating area of robotics with plenty of scope for the amateur to make breakthoughs. Neural nets are a particularly good application for parallel computing.

Machine vision and image processing are other areas where this card could find applications. Video cameras provide a huge amount of information and processing all this (often confusing) information requires a huge amount of power. Most robots with vision capabilities are not autonomous, but are connected to a powerful workstation using an 'umbilical cord'. The transputer wins here as well; the use of links for communication means that the umbilical cord can be just two wires rather than a thick ribbon cable. The lower cost and lower weight of this arrangement gives the robot more freedom from its host, while the bandwidth is sufficient to send digitised video or audio back to the host machine.

Another interesting possibility is to use the card as an I/O processor in a larger network of transputers. The 32 bit and floating point transputers are now becoming reasonably priced, and the T9000 promises phenomenal power. As for software, public domain operating systems (including a UNIX like environment) exist for the 32 bit transputers.

The prospects for this project are very exciting, and ETI will continue to support the card with add-ons and software articles in the future.

Acknowledgements

The authors would like to thank the Computer Science department at the University of Manchester, and especially Jim Garside, for invaluable assistance. Thanks are also due to the numerous Transputer users on comp.sys.transputer who answered my naive questions.

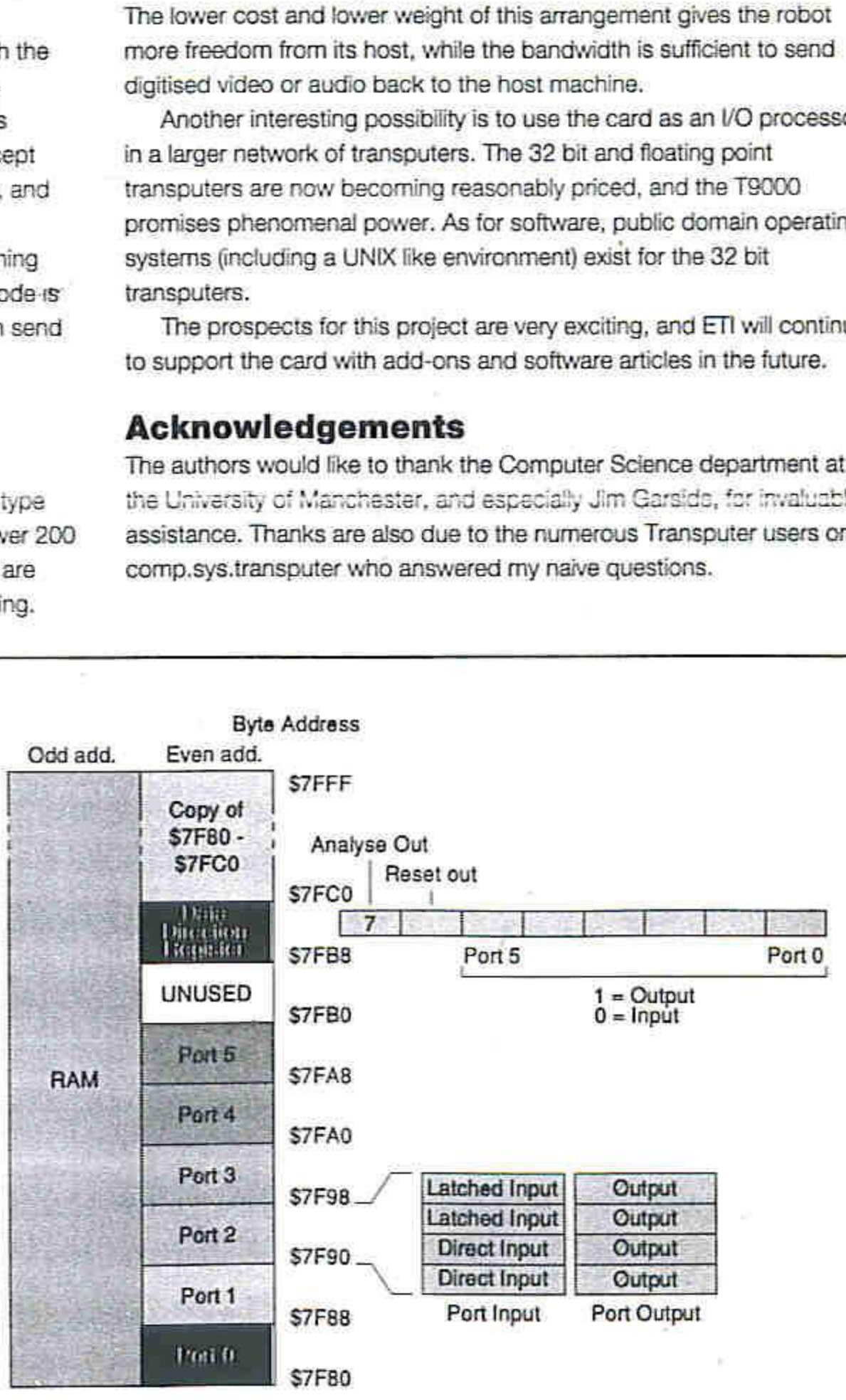


Fig.1. Memory map of top 128 bytes as seen by 'normal' memory accesses.

Listing 1 (LOADER1.ASM)

```
        ; Process Launcher (C) 1994 Mark Robinson.
        ; Waits for code to be sent down link 3 and launches it
        ; as a process. Messages take the form:
        ;   <address> <number of bytes> <data> <data> <data>...
        ; <address> and <number of bytes> are both 16 bit words.

buffer EQU 8
procoff EQU (here+$8024)
progstrt:       AJW $10                 ; move workspace up a bit
                MINT                    ; Minimum integer
                STLF                    ; Initialise queue pointers so that
                MINT                    ; multi-processing may be used
                STHF                    ;

                CLRHALTERR              ; Clear ''Halt on Error''
                TESTERR                 ; Clear error flag
                LDC 0                   ; Initialise loop control
                STL 0
                LDC 11
                STL 1
InitLp:         MINT                    ; Null value
                LDL 0                   ; Index
                MINT                    ; Base address of transputer
                WSUB                    ; Calculate ''channel'' address
                STNL 0                  ; and store null value in it

                LDLP 0                  ; Point at loop control block
                LDC InitLpEnd-InitLp; Offset to start of loop
                LEND                    ; Repeat

InitLpEnd:      LDC 0                   ; Initialise and start the clocks
                STTIMER                 ; which enables timeslicing

                LDC 0
                LDC $7F81
                SB                      ; Page out ROM
                LDC $3F
                LDC $7FB8
                STNL 0                  ; Set ports to O/P keeps CMOS happy

getwords:       LDC 0
                STL buffer
                LDLP buffer             ; Pointer to message buffer
                LDC $800E               ; Channel address for link 3
                LDC 4
                IN                      ; Get 2 words to buffer

                LDL buffer              ; Destination address
                LDC $800E               ; Channel address for link 3
                LDL buffer+1            ; Number of bytes to get
                IN                      ; Get message from link

                LDL buffer
                LDC procoff
                SUB                     ; Compute offset to new process
                LDL buffer
                LDL buffer+1
                SUM
                LDC $20                 ; Put new process workspace 20H
                SUM                     ; bytes above the last instruction
                STARTP                  ; Spawn the new process

here:           J getwords              ; Wait for next link activity
```

Listing 2 (DUMP.C)

```
#include <stdio.h>
#include <dos.h>
#include <time.h>
#define BASEADD 0x300                   /* Base address of the 8255 card */

#define PORTA (BASEADD + 0)
#define PORTB (BASEADD + 1)
#define PORTC (BASEADD + 2)
#define CONTROL (BASEADD + 3)
#define MAKEINPUT 152
#define MAKEOUTPUT 136
#define TIME_OUT 5                      /* wait 5 seconds before timing out */

#define CS 8
#define RW 4
#define OPINT 16

/* Wait for port C bit 4 to become active, indicates that the link
   is ready to accept a byte of data */
void wait_for_link()
{
time_t start;
start = time(NULL);
while (!(inportb(PORTC) & OPINT))
        if ((time(NULL)-start) > TIME_OUT)
                {
                printf("Link timed out\n");
                exit(-1);
                }
}

/* Write the byte contained in val to the link adapter register reg */
void write_register(reg, val)
int reg;
unsigned char val;
{
/* Put the required byte on port A */
outportb(PORTA, val);
/* Pulse the link adapter CS line low to write the byte */
outportb(PORTC, (unsigned char)(reg+CS));
outportb(PORTC, (unsigned char)(reg));
outportb(PORTC, (unsigned char)(reg+CS+RW));
}
```

```
unsigned char read_register(reg)
int reg;

unsigned char temp;

outportb(CONTROL, MAKEINPUT);
outportb(PORTC, (unsigned char)(reg+RW+CS));
outportb(PORTC, (unsigned char)(reg+RW));
temp = inportb(PORTA);
outportb(PORTC, (unsigned char)(reg+RW+CS));
return(temp);


/* Initialise the link adapter status registers */
setup_link_registers()

write_register(2, (unsigned char)3);
write_register(3, (unsigned char)3);


main(argc, argv)
int argc;
char **argv;

FILE *hexfile;
int i, j, fbit;
int num_bytes, num_rows, last_row, this_row;
unsigned int val, from;

setup_link_registers();
/* Make port A an output, since all link transfers are
writes */
outportb(CONTROL, MAKEOUTPUT);

if (argc!=2)
        {
        printf("Usage: %s <hex file>\n",argv[0]);
        exit(-1);
        }
if (!(hexfile = fopen(argv[1],"rt")))
        {
        printf("Can't find file %s\n",argv[1]);
        exit(-1);
        }

printf("Start Address ?\n");
scanf("%X",&from);


/* Count the number of strings in the file, including
addresses */
num_bytes = 0;

        {
        fscanf(hexfile,"%X",&fbit);
        num_bytes++;
        } while (!feof(hexfile));
/* Round odd bytes to the nearest word. If the hexfile
contains an odd
number of bytes, the last one will be sent twice */
if ((num_bytes%2)==1)
        num_bytes+= 1;
/* Rewind the file and compute how many actual data bytes
there are */
rewind(hexfile);

num_rows = (int)(num_bytes/17);
last_row = num_bytes%17;
num_bytes = 16*num_rows;
if (last_row!=0) num_bytes+=last_row-1;

printf("Writing %d bytes to %X\n",num_bytes,from);
wait_for_link();
write_register(1,(unsigned char)(from%255));
wait_for_link();
write_register(1,(unsigned char)(from/256));
wait_for_link();
write_register(1,(unsigned char)(num_bytes%255));
wait_for_link();
write_register(1,(unsigned char)(num_bytes/256));

/* Hex file consists of an address followed by 16 bytes on
each row.
   A minor complication is that the last row may have less
than 16 bytes */
this_row = 16;
for (i=0; i<num_rows+1; i++)
        {
        if (i==num_rows) this_row = last_row-1;
        if (this_row != 0) fscanf(hexfile,"%X",&fbit);
        for (j=0; j<this_row; j++)
                {
                fscanf(hexfile,"%X",&val);
                printf("%.2X ",val); fflush(stdout);
                wait_for_link();
                write_register(1,(unsigned char)val);
                }
        printf("\n");
        }
}
```

Listing 3. (ERRLITE.ASM)

```
; Example process to flash error light.
; (C) 1994 Mark Robinson.
; This is not boot code -- use the loader program

        LDC 3000        ; Error light on time
        STL 4
        LDC 3000        ; Error light off time
        STL 6

LLabel: SETERR          ; Turn on error
light
        LDTIMER
        LDL 4
        SUM
        TIN             ; Wait
        TESTERR         ; Turn off error
light
        LDTIMER
        LDL 6
        SUM
        TIN             ; Wait some more
        J LLabel        ; Do it all again
```

PC Clinic

This month in PC Clinic Nick Hampshire looks at the hardware and software which generates the display shown on the monitor, at the video adapter card, and at some of the other display technologies which are in use today

The very first computer that I owned, back in the late 1970s, was an Altair 8800b, a machine now widely regarded as the world's first commercial micro-processor based computer. The 8800b had no video terminal, no keyboard, no disk drives, no DOS operating system, no ROM based BIOS. In fact it had nothing but a processor, a couple of Kbytes of RAM, a serial interface to a Teletype, and a front panel full of switches and lights which allowed one to look at and alter the contents of any byte of memory.

When one switched the machine on one could not simply power it up and use it, one first of all had to laboriously enter a small bootstrap program into memory using the front panel switches. This program was no more than 20 or30 bytes long. Using this bootstrap loader, one could then load in a small monitor program stored on punched paper tape via the Teletype's paper tape reader. Only then was it possible to use the Teletype keyboard and printer to communicate with the computer in a meaningful manner.

Within a year or two of the launch of the Altair 8800b, other manufacturers were producing microprocessor based systems with no front panel switches and display, with the bootstrap and monitor code stored in ROM, and programmes loaded from cassette tape - systems that were a lot easier to use and which no longer required the slow, cumbersome Teletype with its paper tape reader and punch.

The Teletype was very quickly replaced by what became known as the 'Glass Teleltype', a simple stand-alone, text-only video display terminal that exactly emulated the old Teletype and so could be used as an interchangeable replacement. Thus the text scrolled up the page in the same way that it was printed on paper, and the serial communications used the same data transfer speeds and protocols.

The problem with this type of terminal was that the data transfer rate was quite slow on account of transmission, and terminal, hardware limitations. Thus, a typical video terminal of this period, the late 70's early 80's, displayed text only in 40 character lines and 25 lines per screen, data being transfered at rates of between 300baud, or 30 characters per second, and 9600baud, or 960 characters per second.

Although this type of data transmission speed is ample for user data input from a keyboard it is obviously totally inadequate for generating high quality text and graphics displays. Hence, this use of a separate terminal for input and output was abandoned quite early on in the development of personal computers. It was abandoned in favour of the currently used fully integrated system where computer, keyboard and monitor were all part of the same unit.

This means that the terminal and the computer have been merged together. The video display has been connected to special high speed circuitry that interfaces directly with the processor bus and thus allows for the rapid generation of high resolution colour displays, including both text and graphics. The revolution that made the Apple II such an enormous success.

In video display terms, the first IBM PCs that were sold in 1981were rather disappointing compared to the Apple, since they lacked both colour and graphics capability. It supported the monochrome MDA card with its 80x25 character text display. IBM saw the PC as being used in business, and thought that in consequence there was no need to support either colour or graphics displays. How wrong they were!

The first true graphics cards to be made available for the PC were not produced by IBM but by another independent company, thereby setting a trend for third part innovation which has helped power the PC to a dominant position in the world personal computer marketplace. This card was the Hercules and it rapidly became the standard for high resolution graphics with its 720x350 pixel image area. Although only a monochrome display, one can still find them in use.

The success of both the Apple and the Hercules prompted IBM to introduce their Color Graphics Adapter, or CGA card. This offered a resolution of 640x200pixels with 2 colours or 320x200 with 4 colours. Although very primitive by todays standards, it is still being used in some current portables and notebooks. By the mid-1980's IBM upgraded the graphics on the PC to the Enhanced Graphics Adapter or EGA card. It offered the user 16 colours from a palette of 64 and had a much improved graphics resolution of 640x350 pixels. It is rarely found on any of today's current systems but is still widely used on older machines.

Today's most popular video display standard is the Video Graphics Adapter, or VGA card, which was first launched by IBM in 1987. This was quite a revolutionary design since it not only offered users a 640x480 pixel resolution with 256 colours, fast 60 or 70Hz refresh rates and far clearer square pixels with a 4:3 aspect ratio, but it was also the first design to dispense with the old RGB monitor interface and move to the modern analogue monitor interface.

VGA is probably the most widely used PC display technology in use today, but is now superceded by SVGA systems which can offer resolutions of 1024x768 pixels or more, and up to 16million colours. These systems work at very high speeds, so high that they can no longer work within the bandwidth of the

PC bus. To allow them to work, special high speed local bus systems have been developed by the motherboard designers, systems such as VESA and PCI.

The availability of such high resolution, high speed, colour graphics displays means that we now live in a full colour computer graphics dominated world. There is almost universal use of graphical user interfaces such as Windows and the complex applications software which makes use of it. Now with multimedia we are going even further with the use of displays that are fast enough to display computer-generated full motion video. Indeed, in the future there will be very little to differentiate between computer display technologies and the new digital interactive TV technologies.

To interlace or not to interlace?

At higher resolutions some adapter cards and monitors run in what is known as interlaced mode. What this means is that the adapter makes two passes to display a complete image; in the first scan it builds up half the image using the odd numbered lines and in the second scan fills in the rest of the image using the even numbered scan lines.

This of course means that with on an interlaced display the image will take twice as long to display as it would on a non-interlaced, or sequential, display with the same frame rate. The advantage is that by taking longer to display an image the system bandwidth can be reduced, and with high resolution displays this can give rise to a considereble reduction in display hardware cost.

Although with an interlaced display the image takes twice as long to generate the actual refresh rate stays the same, the only difference being that only every other raster line is refreshed. For most people this avoids any of the problems of having a jerky looking screen, something which would certainly be noticed if the farme rate were reduced to say 25 or 30Hz.

Having said this many people do experience visual problems when looking at interlaced displays. To such people the image appears on the screen in waves. An effect which can lead to eye strain, watering eyes, headaches, and in some exceptionaly sensitive people an actual physical feeling of nausia somewhat akin to motion sickness.

For these reasons it is preferable to use a display which generates a noninterlaced image, at high resolutions it is will cost more, but the additional expenditure is well worth entailing in order to avoid the unpleasant side effects for users.

Upgrading PC video.

One of the best ways to upgrade a system so that it can handle the needs of modern graphical interfaced software is to upgrade the display hardware. While an EGA display may be fine for use with DOS based software, one really needs a standard VGA as the minimum level of display quality for Windows, whilst if you want to run multimedia applications there is really no alternative to a very fast local bus SVGA display.

Considerable care should be taken when choosing a new adapter card. It is, for example, no good buying a high speed VESA card if your motherboard does not have a VESA slot. Unless you also want to buy a new monitor it should be compatible with your existing monitor. Go for a card which has the highest speed for the desired number of colours and graphics resolution. The speed of a video card can be tested with the aid of the graphics benchmark program that was included on last month's ETI front cover disk. But do not forget that it should be tested on a PC with similar specifications to your own.

Having chosen a new video adapter card, installing it in a system is quite straightforward. The first step is to check the amount of video RAM it has. If you want a lot of colours at high resolution you may want to add more memory at this is best done at this stage.

If you want to add more memory you will first of all need to check what type of video RAM is used; it is usually DRAMs or SIMs, memory speed, etc. This data can usually be obtained from the manual that comes with the board. Failing that, have a look at the RAM chips/modules which are already installed. One must then acquire and insert the appropriate number of the appropriate chips or modules.

When inserting DRAM chips it is very important that they are correctly oriented in the sockets on the board. First check that the notch or dot at one end of the DRAM IC is at the same end as the notch indicated on the board legend. Then line up the pins on one side of the chip with the correct side of the socket and gently ease the chip in, rotating it slightly to bring the other line of pins into alignment with the socket. Then carefully verify that all the pins are aligned in the socket before firmly pushing home.

When inserting ICs, one should first make sure that any static electricity charge in your body is discharged and hold them by the ends rather than the pins. Also take particular care not to bend a pin under the body of the IC when pushing it into the socket.

If the video adapter uses SIMs, and these are almost universally used on modern systems, then first of all determine the correct orientation of the SIM to the socket. Having done this, insert the edge connector of the SIM into the socket and then push it back into the socket until both the tabs at the end of the

DC000-DCFFF	free		
D8000-DBFFF	free		
D4000-D7FFF	free		
D0000-D3FFF	free		
CC000-CFFFF	free		
C8000-CBFFF	8514/A		
C4000-C7FFF	8514/A	nonPS/2 VGA	EGA
C0000-C3FFF	8514/A	nonPS/2 VGA	EGA
BC000-BFFFF	EGA/VGA text & LoRes	Hercules Page 2	CGA
B8000-BBFFF	EGA/VGA text & LoRes	Hercules Page 2	CGA
B4000-B7FFF	MDA	Hercules Page 1	
B0000-B3FFF	MDA	Hercules Page 1	
AC000-AFFFF	EGA/VGA High resolution display		
A8000-ABFFF	EGA/VGA High resolution display		
A4000-A7FFF	EGA/VGA High resolution display		
A0000-A3FFF	EGA/VGA High resolution display		

socket snap into place and hold the SIM tight.

Having added the additional memory, if any is required, to the video adapter board, the next step is to configure the adapter and the motherboard. Installing extra video RAM may require the resetting of a jumper or DIP switch on the adapter board; for details about this, carefully consult the manual that comes with the board and set the jumpers or DIP switches accordingly.

The video adapter card should now be inserted into the appropriate bus slot, putting 8bit adapter cards into 8bit slots, 16bit adapter cards into 16bit slots and of course VESA/PCI bus cards into slots with the appropriate local bus connector. Secure the adapter card with a screw, and connect to the monitor using the appropriate cable.

Memory configuration and device drivers

Once the board is installed it should work perfectly under the DOS environment, but to get optimum use from it with applications such as Windows you may need to reconfigure memory usage and also install special drivers.

Some of the memory used by a video adapter lies in the processor's upper memory area, or UMA. This means that if the adapter has more memory than standard then unless steps are

VESA Video Modes Table

Mode # (Hex)	Text or Graphics	Resolution	Colours	Memory	Card types
0	text	40x25	2	1K	CGA/EGA/VGA
1	text	40x25	16	4K	CGA/EGA/VGA
2	text	80x25	2	2K	CGA/EGA/VG
3	text	80x25	16	8K	CGA/EGA/VGA
4	graphic	320x200	4	16K	CGA/EGA/VGA
5	graphic	320x200	2	8K	CGA/EGA/VGA
6	graphic	640x200	2	16K	CGA/EGA/VGA
7	text	80x25	2	2K	MDA
D	graphic	320x200	16	31K	EGA/VGA
E	graphic	640x200	16	63K	EGA/VGA
F	graphic	640x350	2	27K	EGA/VGA
10	graphic	640x350	16	109K	EGA/VGA
11	graphic	640x480	2	38K	MCGA/VGA
12	graphic	640x480	16	150K	VGA
13	graphic	320x200	256	63K	VGA
100	graphic	640x400	256	250K	VESA
101	graphic	640x480	256	300K	VESA
102	graphic	800x600	16	234K	VESA
103	graphic	800x600	256	469K	VESA
104	graphic	1024x768	16	384K	VESA
105	graphic	1024x768	256	768K	VESA
106	graphic	1280x1024	16	640K	VESA
107	graphic	1280x1024	256	1.28M	VESA
108	text	80x60	16	2K	VESA
109	text	132x25	16	2K	VESA
10A	text	132x43	16	3K	VESA
10B	text	132x50	16	3K	VESA
10C	text	132x60	16	4K	VESA

Table of some common PC display synchronisation frequencies

Display type	Frame rate(Hz)	Horizontal scan(KHz)
MDA (monochrome)	50	18.4
Hercules	49	18.1
CGA (RGB)	60	15.7
EGA colour	60	21.8
VGA 640x350	70.1	31.5
VGA 640x480	60	31.5
SVGA 800x600	58	36
SVGA 1024x768	40	32.1
SVGA (VESA)	72	48
8514/A	60	48
1280x1024	60	66

taken to prevent it, this memory could be used by other applications.

To prevent this situation, it is necessary to use the memory manager EMM386 or equivalent. This will entail editing CONFIG.SYS to make sure that the EMM386 command includes an X switch that will exclude the area of memory used by the video adapter. The resulting CONFIG.SYS command could look something like this:

```
DEVICE = C:\DOS\EMM386.EXE NOEMS X=A000-CBFF
```

The PC memory map with respect to most common video adapters is as follows:

You will probably have to check in the video adapter manual to make sure which memory blocks it uses. Also, before you change CONFIG.SYS, do not forget to keep a copy of the original version just in case the change causes problems.

If you intend to use the video adapter exclusively for DOS applications then there should be no need for any special device drivers, but such drivers will probably be necessary if you will be running Windows or any other GUI. A selection of drivers will probably be provided on disk together with the video adapter board. To install the selected driver, follow the steps shown in the video adapter card manual, or install under Windows using SETUP from the DOS Windows directory.

Testing

The only way to really test a video card is to start up the PC and run a few applications. The following are a few common faults and some remedies:

Blank display - check that the monitor is turned on and connected properly to the power supply and to the video adapter. This type of fault is invariably caused by a bad connection.

Multiple images on the display under Windows - this is probably because the monitor and adapter are not using the same video modes. Read the manual for the monitor and the adapter; you may well be able to reset the modes using a switch on the monitor so that they both match. Alternatively, switch to a video mode that is supported by them both, thus some monitors do not support 800x600 mode with some adapters but will support 1024x768 with the same adapter. If problems with Windows persists then try running Setup with either the VGA or SVGA drivers.

Improving performance under Windows

Video speed is one of the most important factors in determining how well a system will run under Windows; it is even more important than screen resolution and number of colours. So if you intend to run Windows you will need to make sure that your video adapter card is running as fast as possible. The following are a few ways of enhancing the speed:

* Unless you really need 256 or more colours, use 16 colour mode.

* Unless you really need very high resolution then use a lower resolution and trade resolution for speed

* Carefully check for any memory conflicts, this may be the cause of the system slowing down and producing occasional strange error messages. To test for this, try running Windows from DOS using the WIN/D:X switch which prevents Windows from using any memory in the UMA. If the problem goes away then there is probably a conflict and one should use EMM386 to exclude all video memory from use by Windows

RGB displays

The simplest colour monitor drive technique, found on CGA and EGA systems, is known as RGB. This refers to the red, green, and blue electron guns that are used in every colour CRT. By turning the three guns off and on in various different combinations it is possible to generate a range of eight colours. This can be seen in the folowing table:

Colour		R	G	B
black	0	0	0	
blue	0	0	1	
green	0	1	0	
cyan	0	1	1	
red	1	0	0	
magenta		1	0	1
yellow	1	1	0	
white	1	1	1	

Note that to generate white all the guns are turned on, whilst they are all turned off to generate black.

By changing the intensity of the on/off signals from the video adapter to the monitor it is possible to increase the range of colours available. Thus on a CGA system a fourth wire is used to indicate whether the intensity is full or half normal (where full is about one volt on the electron gun and half is half a volt). Thus the fourth Intensity line can be used to toggle between, say, red and a kind of pink, thereby increasing the number of colours that can be generated to sixteen.

On EGA systems the concept of having an additional line to control intensity has been expanded, and there are three intensity lines, a secondary red, green and blue. With this system it is possible to generate displays using 16 colours drawn from a palette of 64 different colours.

This is the practical limit to this technique for generating different intensities.

It should be noted that on both CGA and EGA systems the wires connecting the video adapter and the monitor are all at standard TTL voltage levels - +5V and GND.

Analogue displays

With the VGA adapter launched in 1987, IBM introduced a new way for the video adapter card to send image data to the monitor, the analogue interface. A way which allowed a lot more colours to be displayed on the screen.

The original RGB simply turned the electron guns of the colour monitor CRT on or off, all the different permutations of the three guns gave a range of eight colours. Then with CGA an intermediate gun intensity level was introduced thereby increasing the number of colours to 16, a range that was stretched to 64 in EGA displays.

However, there is no reason why each gun should not have an infinite number of different intensity levels thereby giving rise to an infinite number of displayed colours and thus allow the computer display to show images which have all the range of colour seen in everyday life. To do this the voltage of each one of the three lines to the electron guns needs to be varied with great precision. This is done using special digital to analogue converter circuits on the video adapter, hence the reason why such boards are refered to as analogue display boards.

On VGA displays the digital to analog converters can produce up to 64 different intensity levels on each of the three guns, thus enabling a VGA display to have 64^3, or 262,144 different colours. Although to do this will require eighteen bits of storage for each pixel, six bits for each gun, or just under 700Kbytes of video RAM for a standard 640x480 pixel display.The analogue technique has been expanded even further with SVGA where the guns can be driven at up to 256 different levels thus giving rise to a potential 16.7million colours, assuming memory is available at 24bits per pixel, or 2.4million bytes of video RAM. Most of the adaptors sold today have at least 1Mbytes of video RAM, this should be checked carefully when buying either a new system or adapter board, it is important to make sure that can add more video RAM to increase the capability of the display system.

Note: in most cases RGB monitors will not work with analogue video adapters and analogue monitors will not work with RGB adapters. However, some more expensive monitors come with sockets for cables from both types of video adapter.

Composite video

A handful of PCs, such as the Commodore range of machines, also have a composite video output as well as the conventional RGB output. This type of output combines the three colour signals and thr synchronisation signals into a single signal that can be sent down a piece of coax wire with a phono plug. This signal is essentially the same as that which comes into the aerial socket of a conventional TV.

The reason for including the composite video output is simply that it allows the system to use a TV set as a monitor, a very useful capability in some applications. Primarily, however, its use was a hang over from the home games computers such as the Commodore 64, and it is now rarely encountered.

If you do need to use a TV set as a monitor then there are a number of special VGA to TV converter cards on the market. But it should always be born in mind that the current generation of computer monitors have displays which are far superior to that found on the average TV. So do not expect a TV based monitor to have the same quality of display. Having said that, a TV output does allow a PC to utilise equipment such as projection TV displays.

Light pen connectors.

A light pen is not normally considered as a PC pointing device, the mouse has almost exclusively taken over this role. But this has not always been so, and on many EGA cards there is a special light pen connector, and in the PC BIOS, routine Int 10h Function 04 will return the position of the light pen.

The light pen connector is invariably a 6 pin SIL socket on the board, often labeled the P-2 connector. The pin connections are as follows:

Pin	Function
1	+light pen input
2	not used usually acts as socket key
3	+light pen switch
4	ground
5	+5 volts
6	12 volts

The light pen is not implemented on any VGA or PS/2 systems.

NEXT MONTH
in PC clinic we will take an in depth look at moderns and communications devices

PROMs VERSUS LOGIC

We tend to think of EPROMs as simply being a form of non-volatile memory for computers. Graham Reith shows that they can also be used to solve a great many circuit design problems

Many practical problems are best solved nowadays by using an electronic system of some sort. A choice of which system then has to be made, and in particular the precise components to be used. One is looking for the best tool for the job, in terms of practicality, convenience and cost, One of the choices likely to face the problem-solver is whether to use a system of logic gates, or whether to go for something more sophisticated like an EPROM. This article is largely concerned with the considerations which may influence that choice. Some everyday, practical examples are offered to illustrate the factors involved.

Logic gates are easily understood and widely used in electronic circuits. Various types offer the user a variety of functions, and for certain purposes they can be the ideal tool for the job. But there are severe limits to what any one logic gate can do, as can be seen from the small truth table each one has. Logic gates usually need to be used in combination with each other in order to help solve even fairly modest tasks. Such combinations can become very complicated, and the circuit becomes bulkier, more expensive and more difficult to wire. For the more challenging tasks, therefore, an alternative is required.

EPROMs (standing for Erasable Programmable Read Only Memory - there are similar products known EEPROM, E2PROM, or PROM but in this article they will be called simply EPROMs) can be made to behave like the more complicated logic systems. For example the 2716 EPROM can replace a logic system which has any truth not greater than 8 bits wide or 2048 lines long. EPROMs are more expensive than logic chips, but one EPROM can replace many gates and so an EPROM is often the more economic choice. EPROMs also require less wiring than complex logic systems, although they have to be programmed. EPROMs (and EEPROMs) can be erased and reprogrammed many times, and so their behaviour can be altered with little effort at no added cost - unlike logic systems which are hard-wired, i.e. their behaviour cannot be changed without changing wires.

Technically, a third option is available in the form of programmable logic, such as PEELs (Programmable Electrically Erasable Logic Arrays) or PLDs (Programmable Logic Devices). These call for their own particular programming techniques. It is not clear that they offer a better solution than EPROMs do. They and their programmers seem to be less commonly available than EPROMs and their programmers. Programmable logic components are not therefore considered further in this article.

In general, the theme adopted in this article is one of finding easily available, economical solutions to everyday practical problems. That, after all, is the essence of the contribution which the science of electronics has made, and continues to make, to human invention and development. This article goes on to take as examples three common or garden problems and to discuss how these can be solved using logic gates or EPROMs as appear most suitable.

The first problem chosen will be how to control the automatic heating and cooling of a greenhouse in differing conditions where it is either night and day and where ambient temperatures rise and fall. Of course one might choose to solve such a problem manually, using paraffin heaters or whatever, but a fully automatic system is available to those who want it by using electric heaters, electric window fans, and an electronic control system. The article describes how this might best be done, using a simple logic gate system.

The second example tackles a more complex problem where traffic management is needed at a busy T-junction. In this country the old method of manual direction by a traffic policeman has long been abandoned in favour of the cheaper and more efficient system of traffic lights. Various methods of phasing the sequence of traffic lights have developed over the years. This article suggests that a good method is the use of EPROMs.

The last example takes a more modern, more sophisticated, problem which is that of how to communicate information in alphanumeric displays. The use of displays and the way in which characters are presented on them have been evolving and one cannot exclude further development with the exercise of imagination. The design of characters in a display is normally manufactured within the display driver. However, there remains scope for individual experimentation. This article suggests a sample range of characters, and suggests that in this area of electronics there is no practical alternative to the use of EPROMs.

This article not only discusses the merits of logic gates and EPROMs in each case, but offers a detailed account of how circuits might be set up, and of the truth tables involved. A start is made with the common or garden greenhouse.

PRACTICAL: A TEMPERATURE CONTROLLED ENVIRONMENT

Thermostats are fairly simple devices, commonplace around the home. Most thermostats turn a heater on when it gets too cold and turn it off when it gets to the right temperature. However, some people may want a more sophisticated thermostatic system which can maintain a more complicated environment. A practical example would be to equip a greenhouse with the means to keep the indoor temperature between specified

boundaries by heating and cooling the air when required during the daytime, and similarly by heating during the night (the need to cool during the night being unlikely). For example, there could be a thermostat which kept the temperature between 16°C and 20°C during the daytime, and greater than 10°C during the night.

There are various ways of building a system to meet these requirements. The system designed here is perhaps one of the simplest ways of doing it.

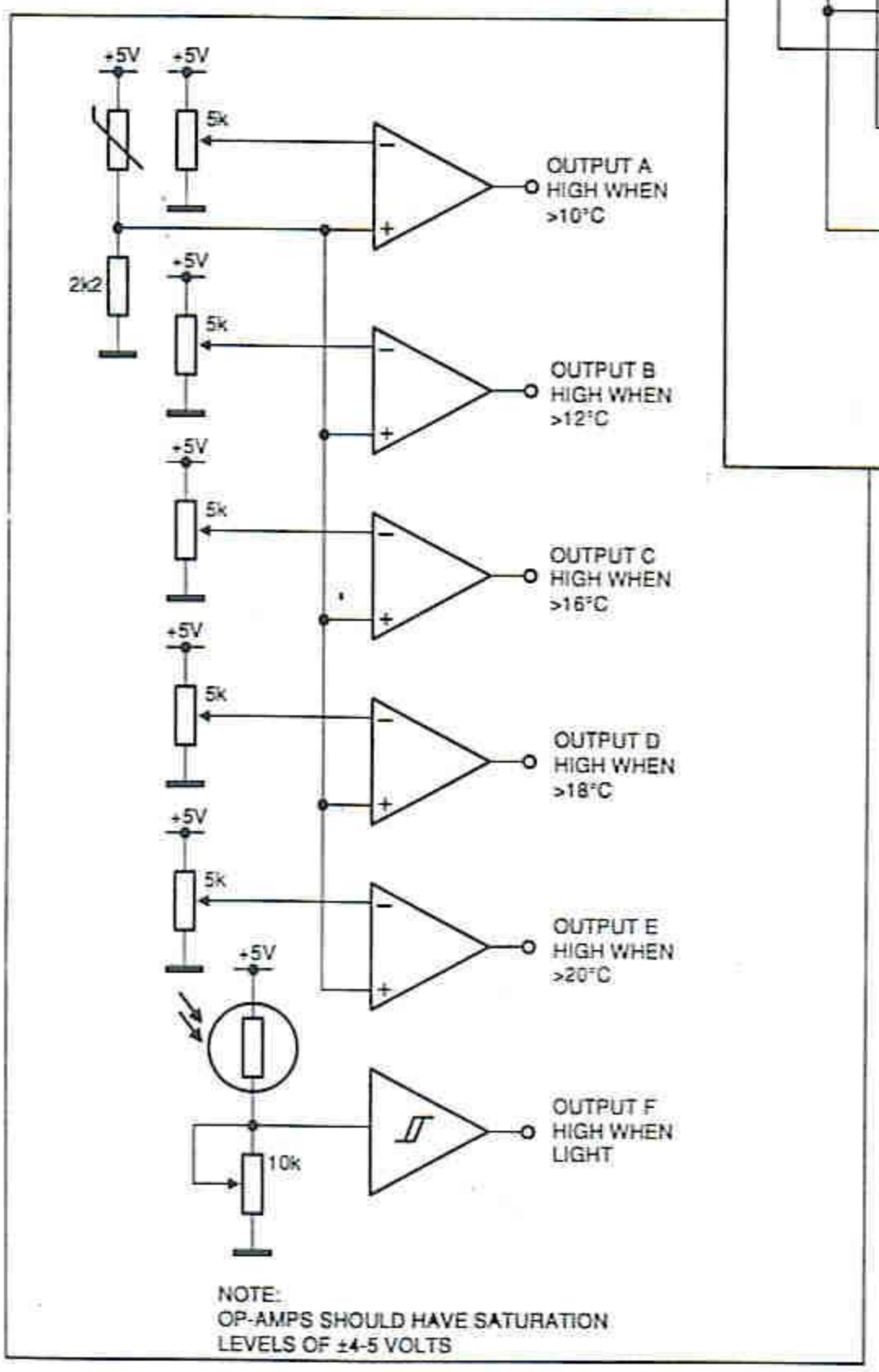
There has to be a system of temperature and light sensors to determine the conditions inside the greenhouse. Using only one of each type of sensor has the advantage that the system is more reliable in the sense that any two temperature sensors will not be identical and variations in their performance would cause undesirable complications. Their sensor system, as set by the owner using a therometer, should indicate when the temperature is >10°C, >12°C, >16°C, >18°C, and whether it is light or dark. The idea is to have a heater switching on if it is dark and <10°C until the temperature becomes >12°C, and switching on if it is light and the temperature is <16°C until it becomes >18°C. A window fan to expel hot air switches on if it is light and >20°C until it is <18°C. The point of having it switching on at one temperature and off at another is to prevent the appliances from switching on and off very rapidly around one temperature threshold.

A possible sensor system is shown below. The outputs A - F will be inputs to a logic system. The potentiometers allow the temperatures at which each output changes to be adjusted.

The state of the outputs in relation to temperature and light is shown in the table below. The table also shows what the required output of the logic system is, i.e. when the heater and fan are to be turned on and off.

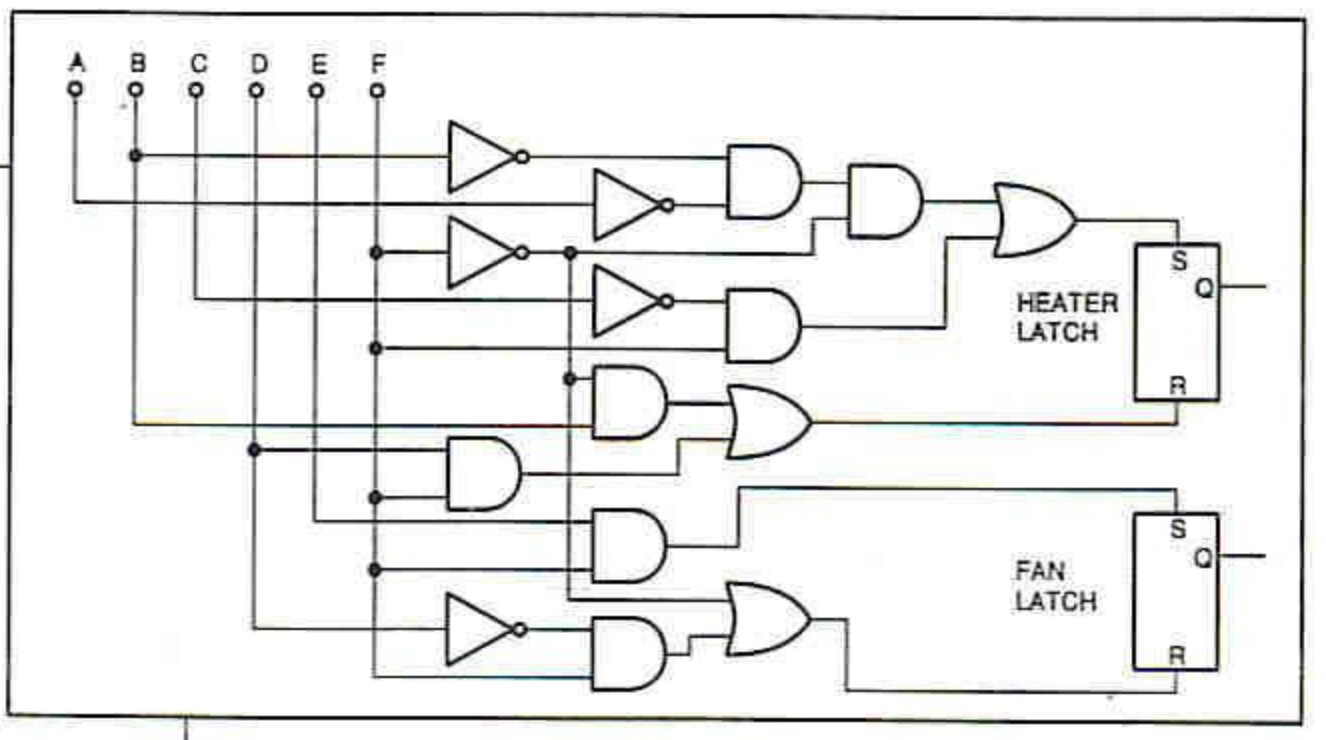
A suitable logic system to perform this would be:

								Heater		Fan	
	°C	A	B	C	D	E	F	S	R	S	R
Dark	<10	0	0	0	0	0	0	1	0	0	1
Dark	10-12	1	0	0	0	0	0	0	0	0	1
Dark	12-16	1	1	0	0	0	0	0	1	0	1
Dark	16-18	1	1	1	0	0	0	0	1	0	1
Dark	18-20	1	1	1	1	0	0	0	1	0	1
Dark	>20	1	1	1	1	1	0	0	1	0	1
Light	<10	0	0	0	0	0	1	1	0	0	1
Light	10-12	1	0	0	0	0	1	1	0	0	1
Light	12-16	1	1	0	0	0	1	1	0	0	1
Light	16-18	1	1	1	0	0	1	0	0	0	1
Light	18-20	1	1	1	1	0	1	0	1	0	0
Light	>20	1	1	1	1	1	1	0	1	1	0



Note: Op-Amps should have saturation levels of ±4-5 volts.

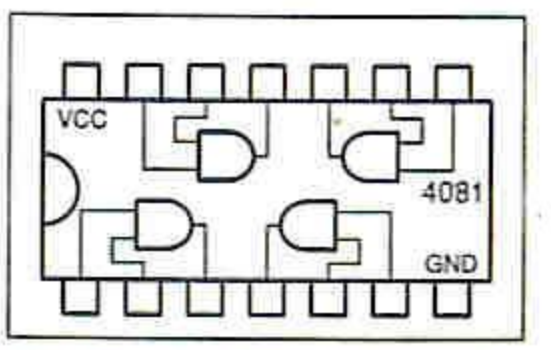
5 NOT gates



7 AND gates
3 OR gates

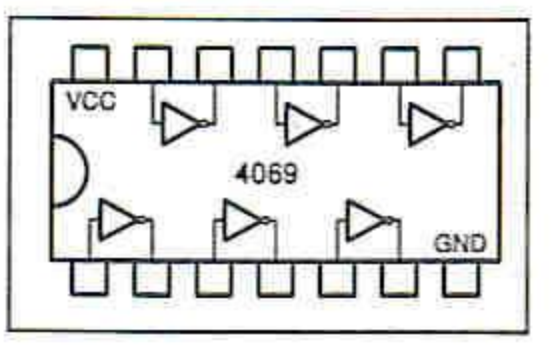
Wiring up Logic:

Logic gates tend all to be in the same format on chips. The drawing below shows how gates are arranged on a chip.
For example:
Quad AND gate:



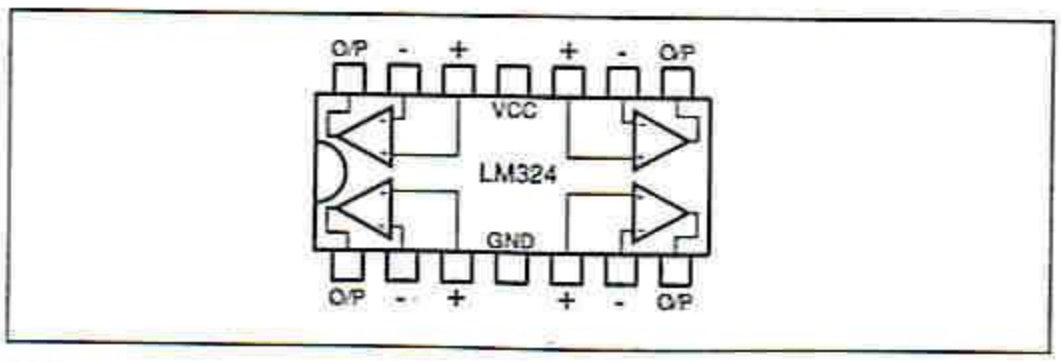
Wiring up Op-

Hex NOT gate:

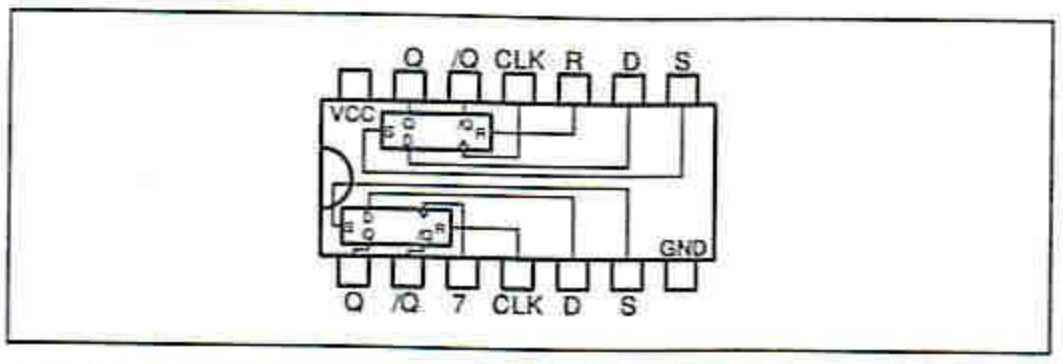


Amps:

This depends on the Op-Amps used. A suitable cheap quad Op-Amp chip is the LM324.



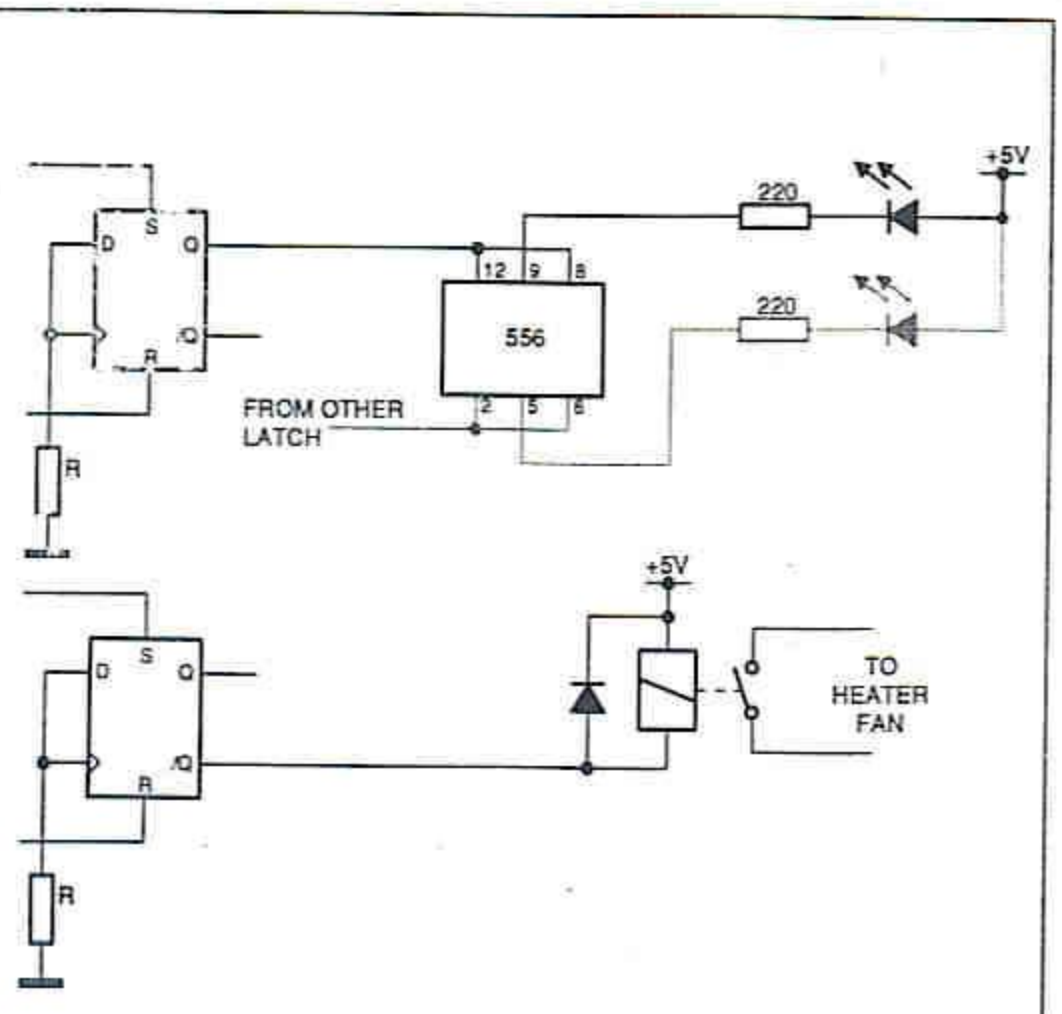
Wiring up set-reset latches, eg 4013:



This logic system consists of 15 logic gates which, when linked together, will consist of 4 chips: 1 hex NOT gate, 2 quad AND gates, and 1 quad OR gate. These chips cost less than 25p each and so the total cost of the logic system is around £1. The cost of an EPROM is substantially more and although the system could easily be made with an EPROM, requiring much less wiring, on balance the money saved makes logic the best option. The behaviours of the system can be changed without changing the logic, simply by adjusting the potentiometers which change the switching thresholds of the fan and heater. There is little that would require the logic to be changed, so the versatility of the system is not affected.

To run a cooling fan and a heater off system, relays need to be used to buffer the output from the latches. The relays used depend upon the kind of heater and fan used and how much current they draw. Illustrating the behaviour of such a logic system does not actually require a fan or heater - LEDs could be used instead to indicate whether conditions were too hot or too cold, i.e. when the heater or fan should be on. A 556 buffer is probably the easiest way of buffering the outputs of the latches to drive the LEDs. The 556 is an inverting buffer, so it would be sinking the current.

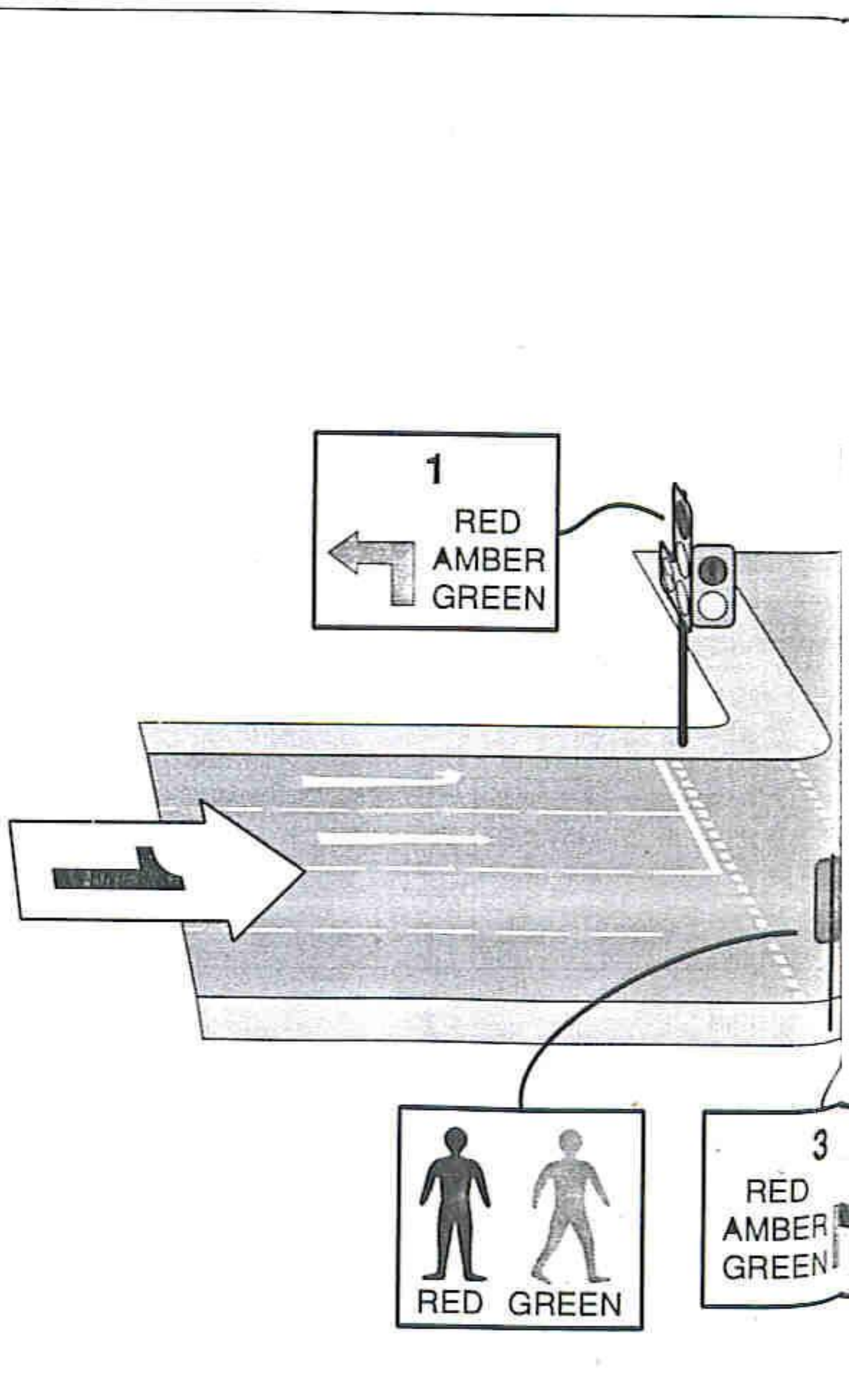
That example deals with the comparatively simple requirements of heating or cooling a greenhouse, which can economically and efficiently be met by a cheap logic system. We can now move on to discuss a more complex set of everyday requirements, such as would be found at a street corner in any town. The management of traffic light controls possess a number of practical problems which are unlikely to be best solved by logic systems.



PRACTICAL

Automatic T-junction traffic lights with pedestrian crossing

The following diagram shows how the traffic lights would be found at most actual T-junctions. The numbers beside the lights indicate which road (marked 1, 2 or 3) the traffic light would be directed towards. Sets of lights beside the same number operate in unison. The letters R, A and G denote red, amber and green, and there are appropriate symbols for the red and green man.



The first thing that must be done is to work out the sequence of the traffic lights and how long each light should be on for. The sequence is shown below, together with possible timings. '1' indicates a traffic light is on, and '0' indicates it is off.

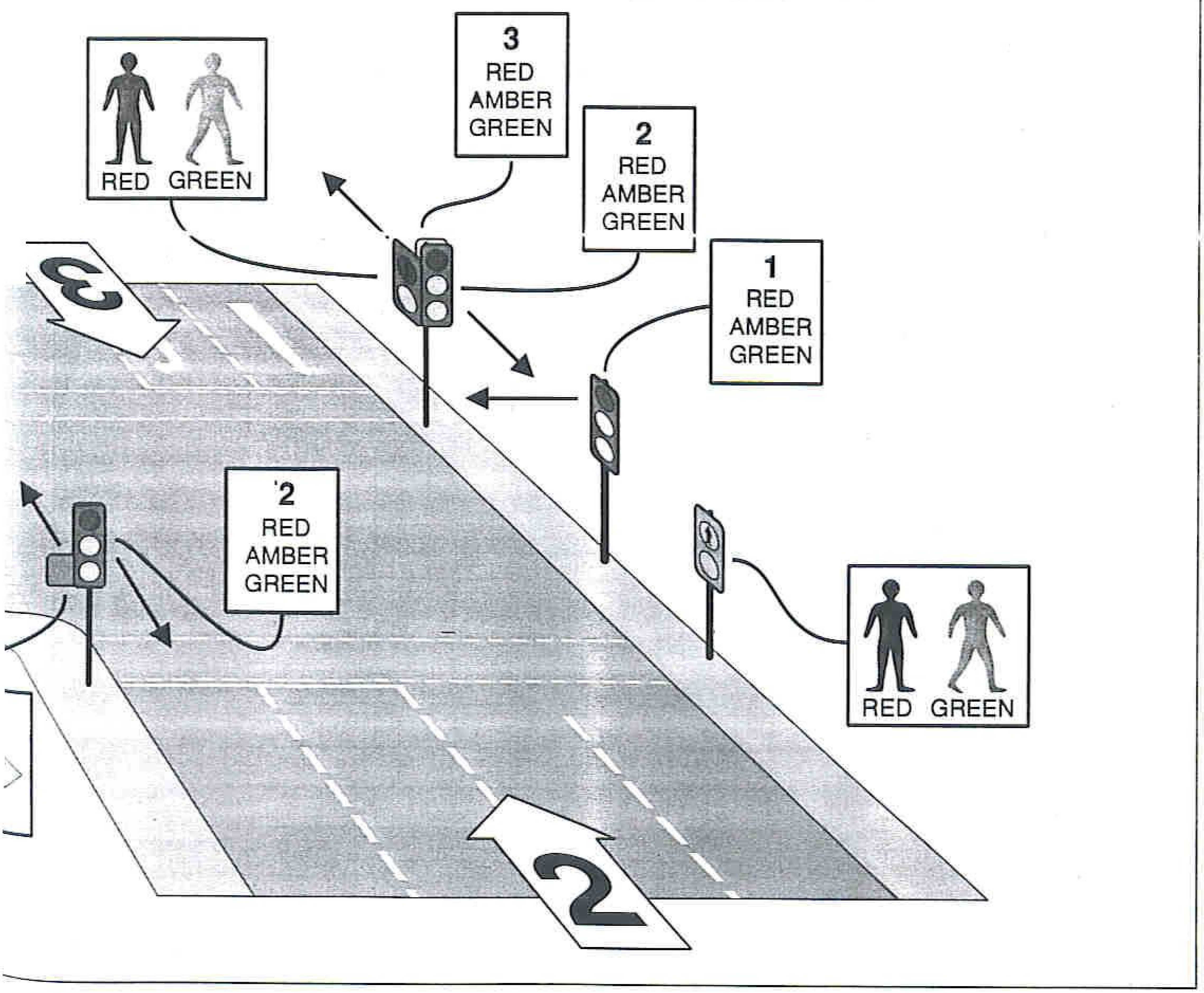
Notes: The square-bracketed lines apply only if a pedestrian button has been pushed.

The total time for the cycle is 78 seconds including pedestrian time.

This system could be made using logic gates. A counter could be made to count up to 78 (seconds), and logic could be used to turn the light on and off at the appropriate times. This, however,

1				2			3				Pedestrian			
RED	AMBER	GREEN	⇦	RED	AMBER	GREEN	RED	AMBER	GREEN	⇨	RED	GREEN	RESET	TIME
1	0	0	0	1	0	0	1	0	0	0	1	0	0	2 secs
1	0	0	0	1	1	0	1	1	0	0	1	0	0	2 secs
1	0	0	0	0	0	1	0	0	1	0	1	0	0	20 secs
1	0	0	0	0	1	0	0	0	1	0	1	0	0	2 secs
1	0	0	0	1	0	0	0	0	1	0	1	0	0	2 secs
1	0	0	1	1	0	0	0	0	1	1	1	0	0	14 secs
1	0	0	1	1	0	0	0	1	0	0	1	0	0	2 secs
1	0	0	1	1	0	0	1	0	0	0	1	0	0	2 secs
1	1	0	1	1	0	0	1	0	0	0	1	0	0	2 secs
0	0	1	0	1	0	0	1	0	0	0	1	0	0	14 secs
0	1	0	0	1	0	0	1	0	0	0	1	0	0	2 secs
1	0	0	0	1	0	0	1	0	0	0	1	0	0	2 secs
1	0	0	0	1	0	0	1	0	0	0	0	1	0	6 secs
1	0	0	0	1	0	0	1	0	0	0	0	0	0	1 sec
1	0	0	0	1	0	0	1	0	0	0	0	1	0	1 sec
1	0	0	0	1	0	0	1	0	0	0	0	0	0	1 sec
1	0	0	0	1	0	0	1	0	0	0	0	1	0	1 sec
1	0	0	0	1	0	0	1	0	0	0	0	0	0	1 sec
1	0	0	0	1	0	0	1	0	0	0	0	1	0	1 sec
1	0	0	0	1	0	0	1	0	0	0	1	0	1	—

The shaded area only applies if a pedestrian button has been pushed.
Total time for the cycle is 78 seconds, including pedestrian time.



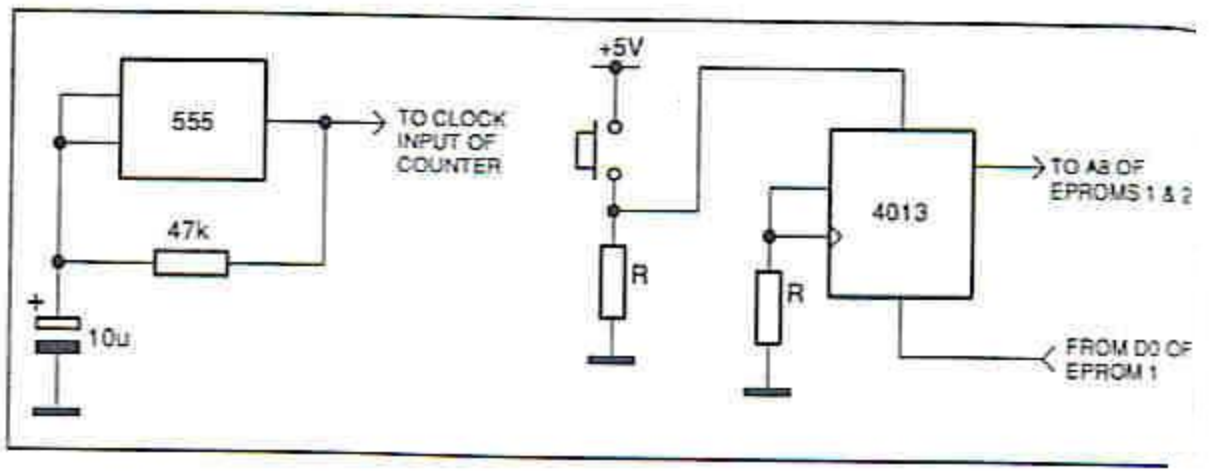
would be very large, complicated, and expensive. Indeed, it would be very inconvenient to use logic because of the subsystem which is accessed whenever a pedestrian wishes to cross, requiring more logic gates to be added to the system. If the traffic lights needed to cope with a Monday to Friday rush hour, under a logic system a different circuit would need to be introduced during set times. If there were to be a change in longer-term traffic characteristics, such as a new supermarket built in the area, the entire circuit of a logic system would need to be redesigned. In short, a logic system is too cumbersome for a task like this requiring flexibility.

An alternative is to build the system out of EPROMs. This would be substantially cheaper than a logic circuit, and any changes required in the system such as pedestrian use and a rush hour could all be catered for in the same program, inside the same chip. If there were to be a long-term change in the traffic characteristics, the sequence and timing could be altered by changing the program inside the EPROMs. Such adjustments would take a fraction of the time and expense that constructing a different logic system would involve. EPROM chips are erasable and re-programmable, so the very same chip could even be used again.

Constructing the EPROM system:

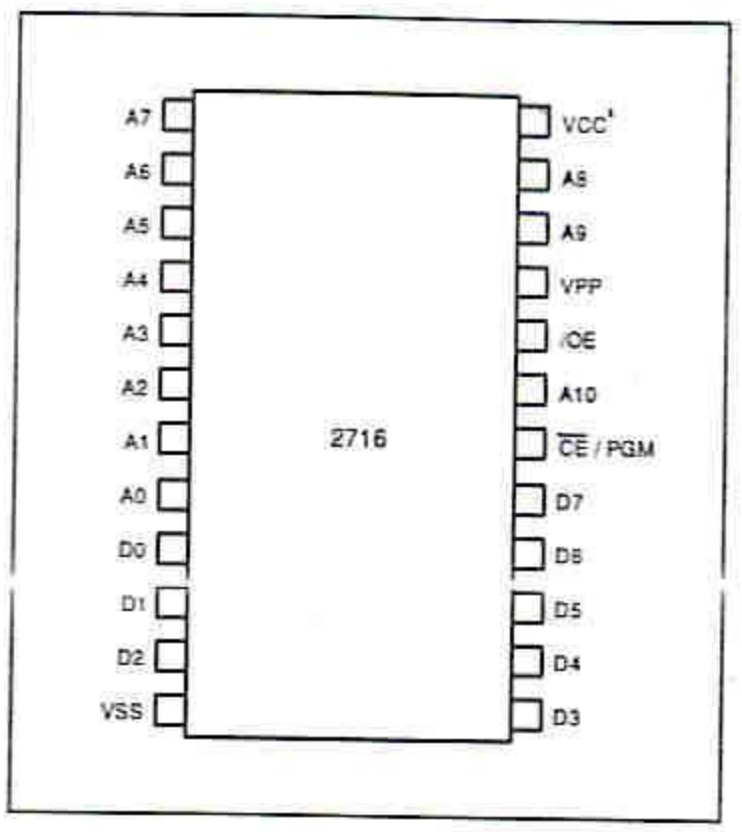
The system needs at least 14 outputs, and needs enough address lines to allow 78 different stages as well as settings for pedestrians and potentially a rush hour, which means a minimum of 9 address lines. As there are 14 outputs to the system, and standard cheap EPROM chips have only 8 output lines, the system needs to have 2 EPROMs. The type of EPROM most suitable for the purpose is the 2716 EPROM which has 11 address lines and 8 data lines. Two 2716 EPROMs can give a combined effect of one EPROM with 11 address lines and 16 data lines. (Remember that the same address needs to be fed into both EPROMs).
The block diagram for the system is shown below:

Wiring up the Clock and Latch:



Connecting up the 2716 EPROM:

Pin out diagram:



For normal operation of a programmed chip:

Vcc - +5V

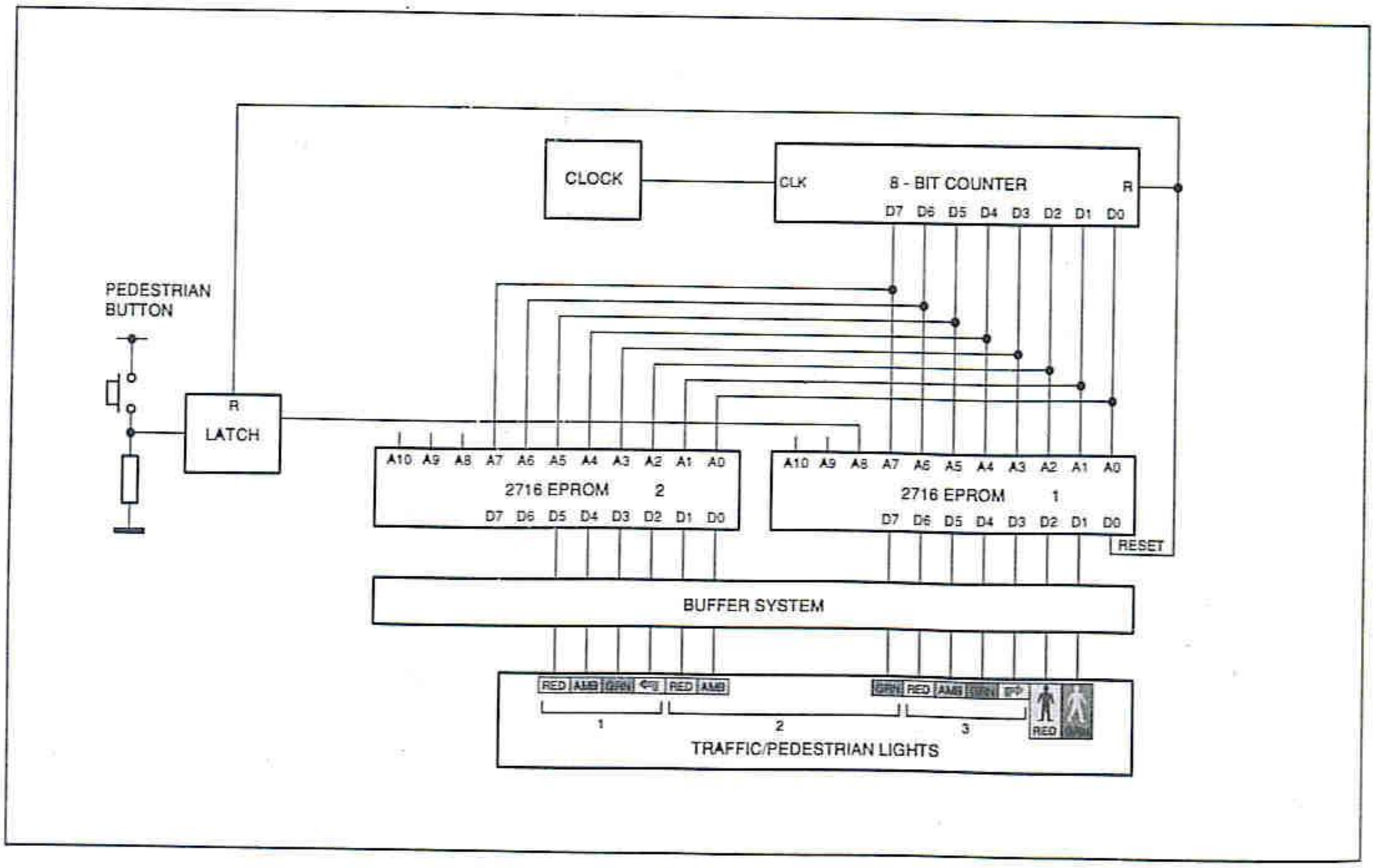
Vpp - 0V

OE - 0V

CE/PGM - 0V

Vss - 0V

The address is fed into the address pins A0 to A10, holding any unused pins at 0V. The data stored at that address will then



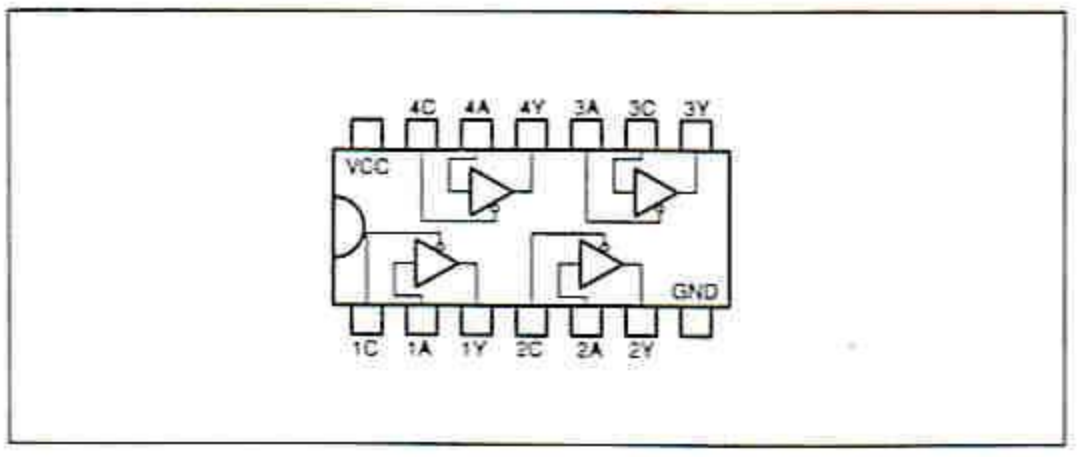
appear at the data outputs D0 - D7.

The pedestrian lights:

These can be made from coloured LEDs, with protective resistors of around 220 ohms. The filter lights could be made using a marker pen and a little artistry.

The buffer system:

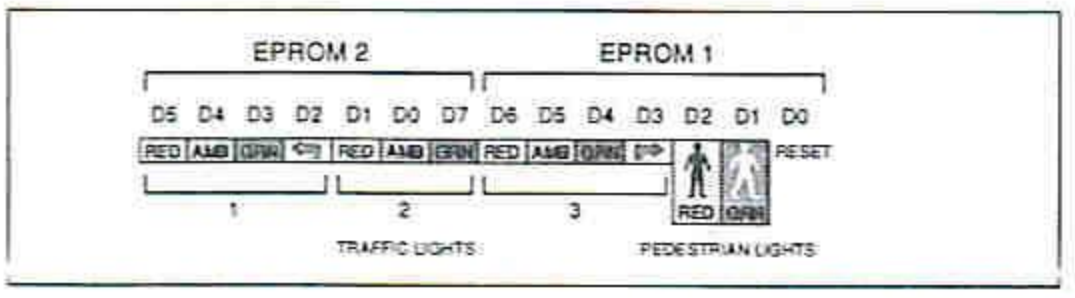
The buffers do not have to be very complex as they are only driving LEDs. The 74LS125 contains four buffers, and four of these chips will suit the purpose.



The EPROM Program:

Address Pins: A0 - A7 from counter
A8 from latch
A9 - A10 further expansion.

Data Pins:



Traffic Lights

Most EPROM programmers require the data to be fed in hexadecimal. Therefore the data needs to be converted from binary into hexadecimal to program the EPROM. The data D7 - D0 will appear in hexadecimal aş two characters. For example, 10100111 in binary equals A7 in hexadecimal. Remember that when converting, the last four figures are taken first, in this case 0111, and put as 7, and the next four, in this case 1010, are put as A. Thus multiple of four digits is required, and where fewer appear, zeros need to be added at the beginning to make up the required number eg 111001 should be written as 00111001.

The EPROM programs for the traffic light sequence are shown opposite:

The program works by following through the same basic sequence 000 - 040, at the end of which the counter is re-set back to 000. So, if uninterrupted, the traffic lights would be following the exact same sequence ad infinitum. However, if there is an interruption - a pedestrian wishing to cross the road - the system defers action until an appropriate point in the sequence has been reached, i.e. when all lights are red.

Only then does it put on the green man. In order to achieve this, a different program is used which incorporates the green man as part of its sequence. When the pedestrian pushes the button, a latch is set and the address changes 0-- to 1--. The sequence 000- 03F is exactly the same as the sequence 100 13F, and so pushing the button will not cause any immediate change of sequence. However, it would normally reset back to 000 at address 040. Instead, at 140 the pedestrian signals run their sequence, and the counter (and latch) are reset to 000 at address 14E.

There is room for expansion of the program. A timer could be used to run a different program at rush hour. During rush hour a 1 could be put to Address pin A9 on both EPROMs and the different sequence could be stored in Addresses 200 - 2FF. Remember that the pedestrian sequence during rush hour would be stored between 300 and 3FF.

In taking traffic lights as a practical example in the use of EPROMs, the EPROMs have been used as a means of proceeding through sequences stored in them. However, EPROMs have more varied uses, and another example worth looking at is that of holding look-up tables.

A look-up table is a table of data stored in an EPROM from which the user, or another electronic component, can summon up a specific piece of data for a specific function. The next example uses a look-up table to operate an alphanumeric display.

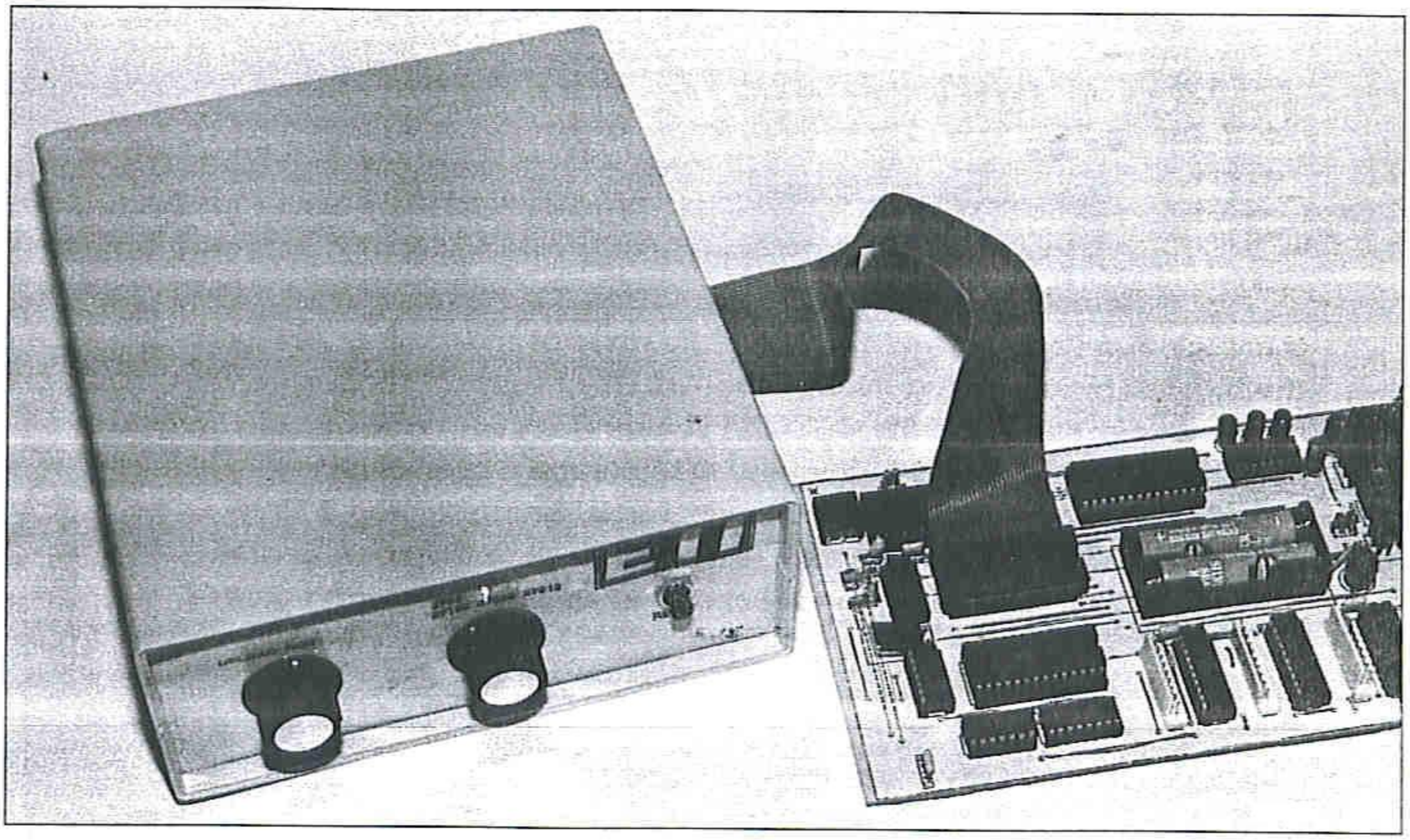
ADDRESS	EPROM 2 DATA	EPROM 1 DATA	ADDRESS	EPROM 2 DATA	EPROM 1 DATA	ADDRESS	EPROM 2 DATA	EPROM 1 DATA
000	22	44	01D	26	1C	03A	0A	44
001	22	44	01E	26	1C	03B	0A	44
002	23	64	01F	26	1C	03C	0A	44
003	23	64	020	26	1C	03D	0A	44
004	20	94	021	26	1C	03E	12	44
005	20	94	022	26	1C	03F	12	44
006	20	94	023	26	1C	040	22	45
007	20	94	024	26	1C	041-		
008	20	94	025	26	1C	OFF	FF	FF
009	20	94	026	26	1C	100-	same	as
00A	20	94	027	26	1C	13F	000	, 03F
00B	20	94	028	26	1C	140	22	44
00C	20	94	029	26	1C	141	22	44
00D	20	94	02A	26	24	142	22	42
00E	20	94	02B	26	24	143	22	42
00F	20	94	02C	26	44	144	22	42
010	20	94	02D	26	44	145	22	42
011	20	94	02E	36	44	146	22	42
012	20	94	02F	36	44	147	22	42
013	20	94	030	0A	44	148	22	40
014	20	94	031	0A	44	149	22	42
015	20	94	032	0A	44	14A	22	40
016	20	94	033	0A	44	14B	22	42
017	20	94	034	0A	44	14C	22	40
018	21	14	035	0A	44	14D	22	42
019	21	14	036	0A	44	14E	22	45
01A	22	14	037	0A	44			
01B	22	14	038	0A	44	14F-		
01C	26	1C	039	0A	44	1FF	FF	FF

Next month

Graham Reith will continue his look at the creative use of EPROMs with the development of an alphanumeric display system.

PROJECT

EPROM EMULATOR



To accompany the EPROM Programmer featured in last month's ETI, Paul Stenning has designed a versatile EPROM emulator which will remove a lot of the effort from system development

This EPROM Emulator was designed to complement the EPROM Programmer published last month. An EPROM Emulator can save a vast amount of time when developing software, compared to programming and erasing EPROMs. This is because it is a RAM-based memory that just looks like an EPROM to the system into which it is plugged.

The unit presented here will emulate the standard 27 family of devices, from 2764 to 27512, and can be used with an IBM PC compatible computer. The smaller 24 pin devices in this family (2708, 2716 and 2732) are almost obsolete and are now more expensive than a 2764. I did not feel it necessary to accommodate these devices in this design since they would significantly increase the complexity.

This design uses readily available components to reduce the likelihood of obsolescence. The unit is powered from either an external PSU or the circuit under development - the supply requirement being 5V @ 100mA. The power supply design published last month for use with the EPROM Programmer will also power this unit if an external supply is needed.

The emulator itself is dumb and is controlled by the host computer via the RS232 serial port (COM1 or COM2). Device selection and operation mode is set by front panel switches.

The accompanying software is available on disk from the author. The software will operate on any PC running MS-DOS or PC-DOS version 3.0 or later and having at least 512K of RAM and one RS232 serial port. A hard disk and a colourmonitor are strongly recommended.The software is written in Microsoft QuickBASIC V4.5, and the full source code is supplied for those wishing to enhance or modify it. This source code is also compatible with QBASIC, as supplied with MS-DOS 5.0 and later. You do not need QuickBASIC or QBASIC to use this disk. The disk also contains the software for the EPROM Programmer published last month. A description of the software operation is

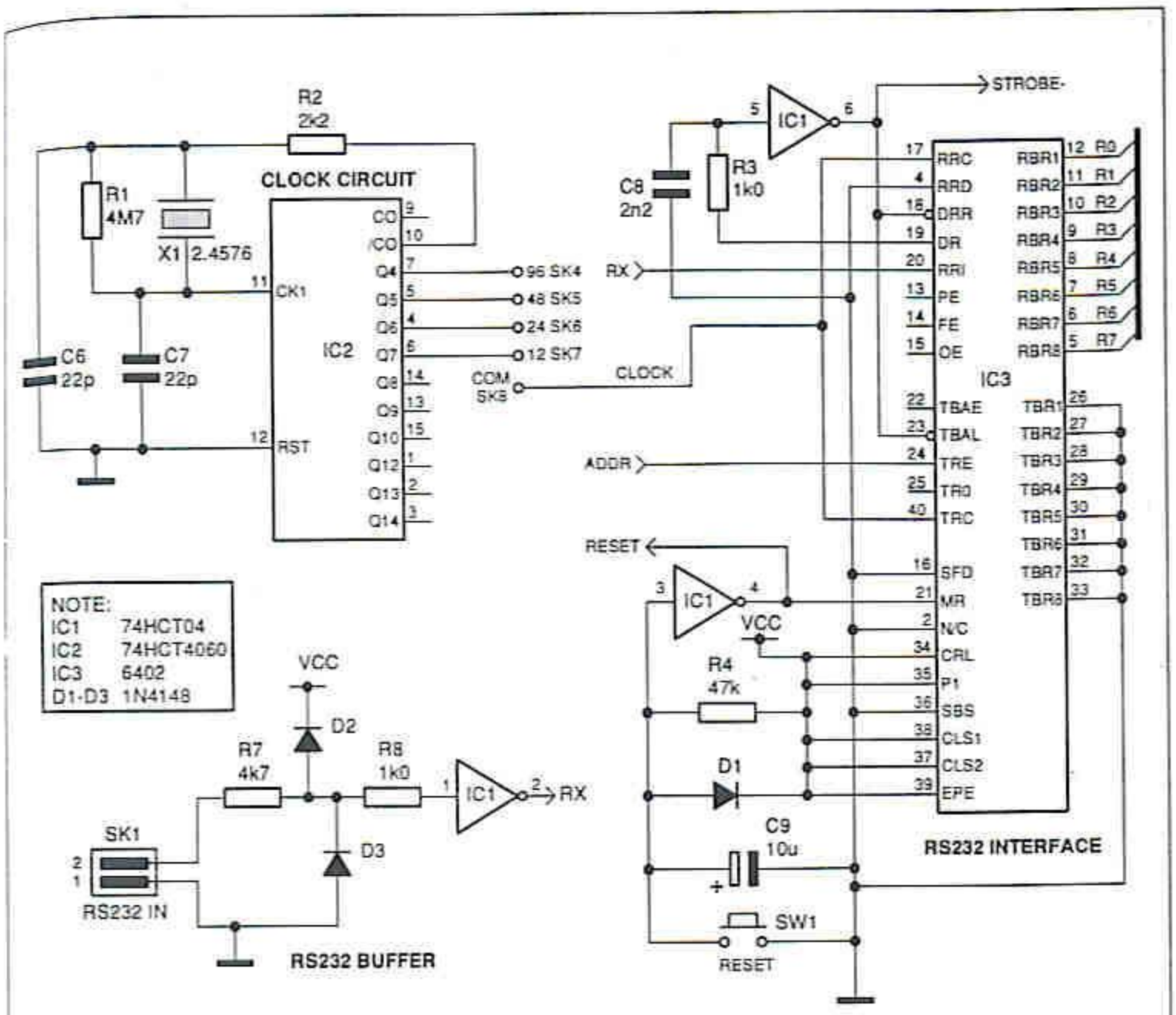


Fig.1. Emulator RS232 interfacing and clock circuitry

given later.

The unit may also be suitable for use with other types of home computer having an RS232 serial port, although this has not been tested and no software is available. It will definitely not work with Commodore Amiga computers; due to a peculiarity in the serial port handling.

How it Works

The circuit diagram is spread over a number of illustrations. Although it may initially look complex, it is in fact relatively straightforward. When a "-" follows a signal name (for example STROBE-), this shows that the line is active low. When a number is followed by an "h" this indicates that the number is hexadecimal.

The RS232 (serial) interface, buffering and clock are shown in figure IC3 (6402) is a UART (Universal Asynchronous Receiver/Transmitter) which basically converts serial data to parallel and vice-versa. The device supports most common serial data formats, in this application it is configured to give eight data bits, one stop bit and no parity checking. The data rate is set by the crystal-controlled clock circuit (IC2), which in most cases will be set to 9600 Baud. Since this unit is designed to

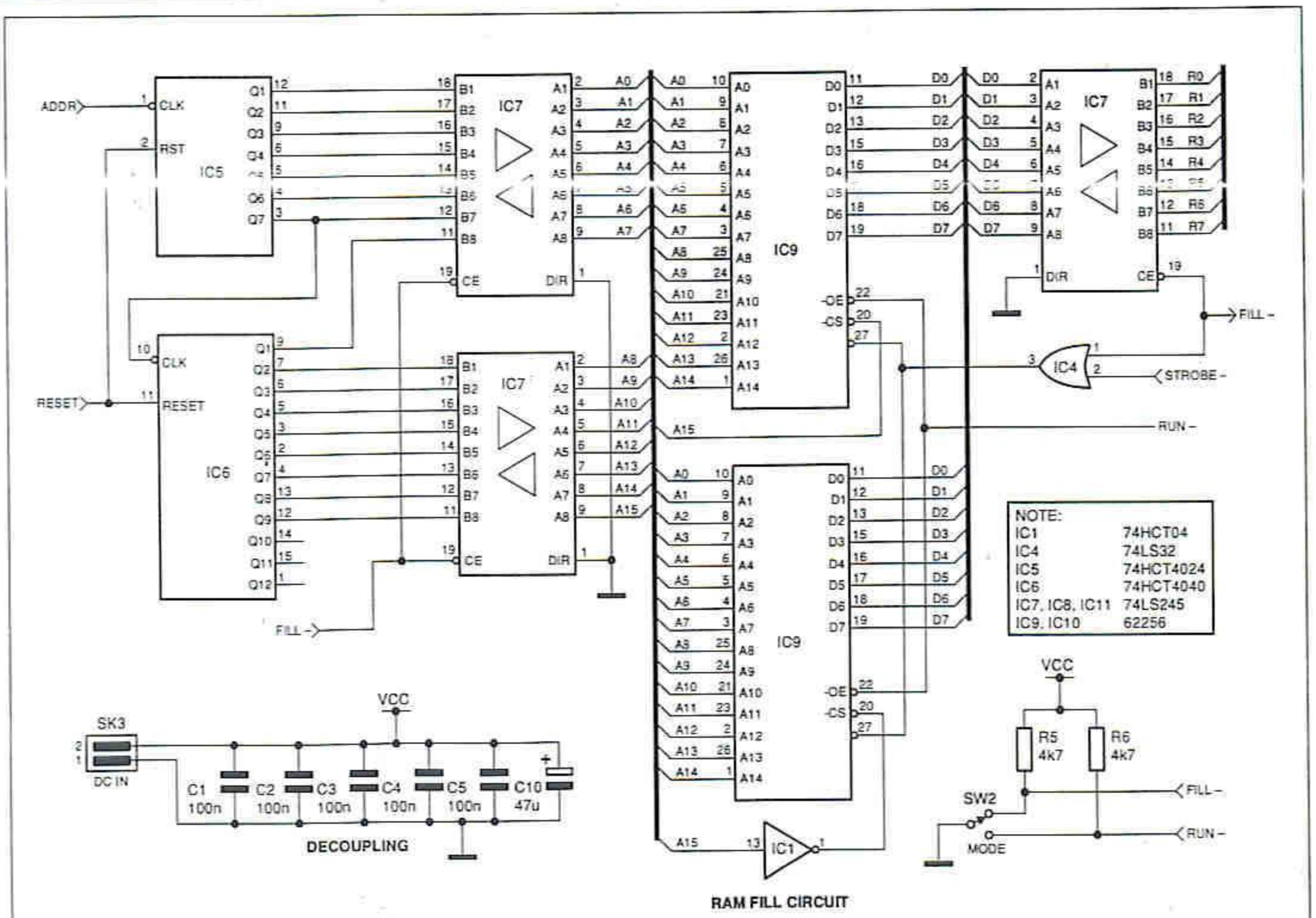


Fig.2. EPROM Emulator RAM fill circuit

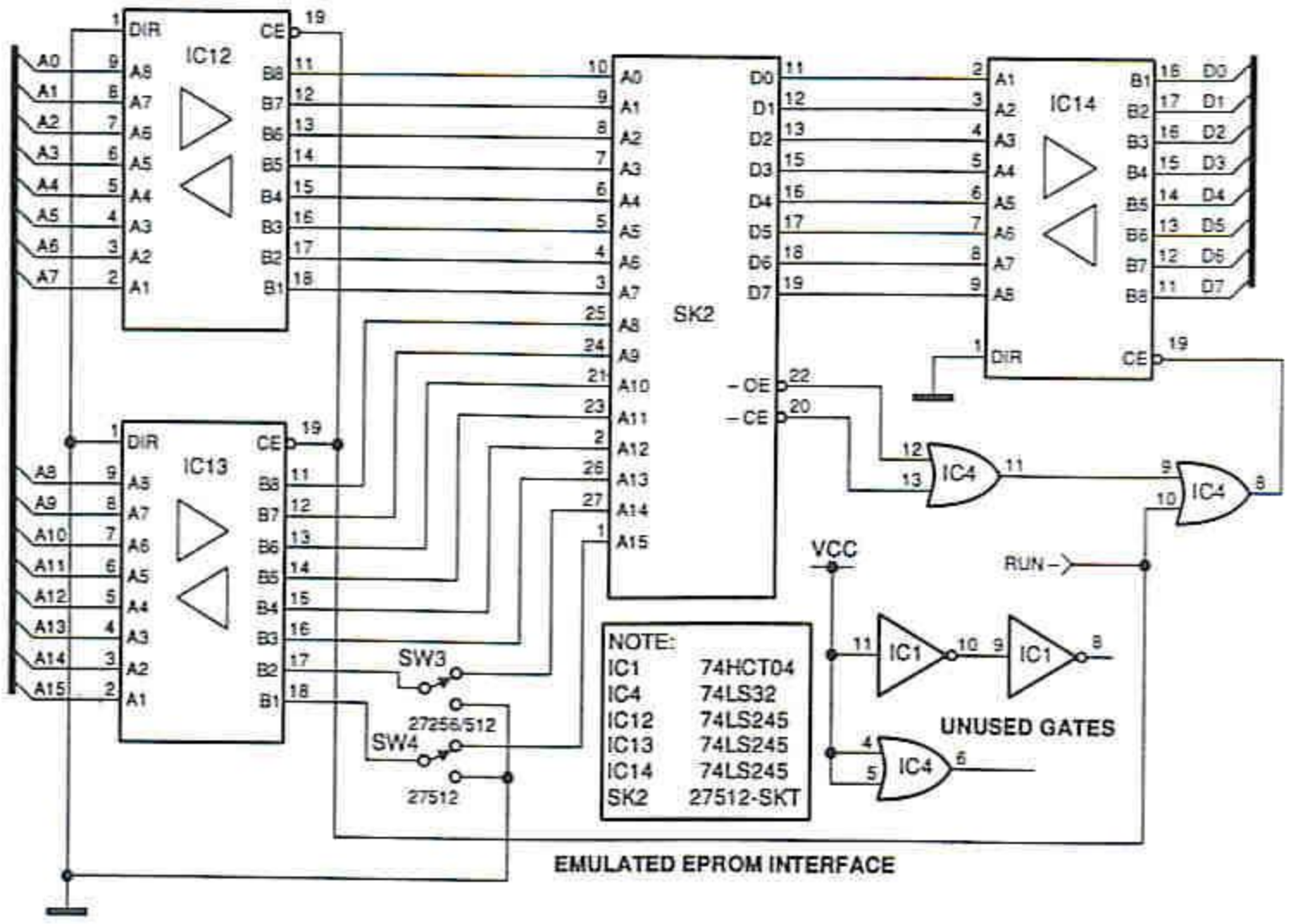


Fig.3. Emulated EPROM interface

receive data only, the parallel input lines (pins 26 to 33) are held low and the serial data output line (pin 25) is not connected.

A gate in IC1 is used as the RS232 buffer, converting the +/- 10V RS232 standard data from the PC to a 5V pulse train suitable for the UART. The diodes restrict the RS232 voltage to within the 0-5V range, the remainder is dropped across R7.

A byte of serial data arriving on pin 20 of IC3 will be converted to parallel data, which will appear on pins 5 to 12. Pin 19 will go high, and no further data can be received until pin 18 is taken low momentarily. One gate of IC4 causes this to happen, R3 and C8 create a slight delay giving a STROBE- pulse with adequate width for clocking other devices.

R4, D1, C9 and one gate of IC4 produce the power-on reset pulse for IC3, IC5 and IC6. SW1 allows the circuit to be re-set again as required.

Assume SW2 is in the upper position (Upload), so Fill- is low and Run- is high. IC12, IC13 and IC14 are disabled, so the 'EPROM' socket SK1 is isolated from the RAM. IC7, IC8 and IC11 are all enabled, effectively connecting the RAM (IC9 and IC10) to the address counters (IC5 and IC6) and the UART.

When a byte of data is received, the Strobe- line will pulse low as described previously. This is connected to the Write Enable- (pin 27) on the RAM IC's so the received data will be written to the current memory location.

The Strobe- signal is also connected to pin 23 of the UART. This will cause the transmission section of the device to go through its motions (even though nothing is listening!). As the data is sent, pin 24 will pulse, incrementing the address counters IC5 and IC6. The section of the UART is used to merely apply a suitable delay before the address is incremented, to ensure the data is correctly stored.

When the next byte of data is received, it will be loaded into the next memory location, and so on. As long as the user presses the Reset button when prompted by the software, the data will arrive in the correct places. The software simply sends the data to the emulator one byte at a time; you could alternatively simply copy a binary image file to the COM port.

Two 256K x 1 Bit (32K x 8 Bit) RAM devices are used, to obtain the full 64K addressing range of a 27512 EPROM. The CS lines are used in conjunction with an inverter on the A15 line to select which device is used at a particular time.

Now assume that SW2 is in the

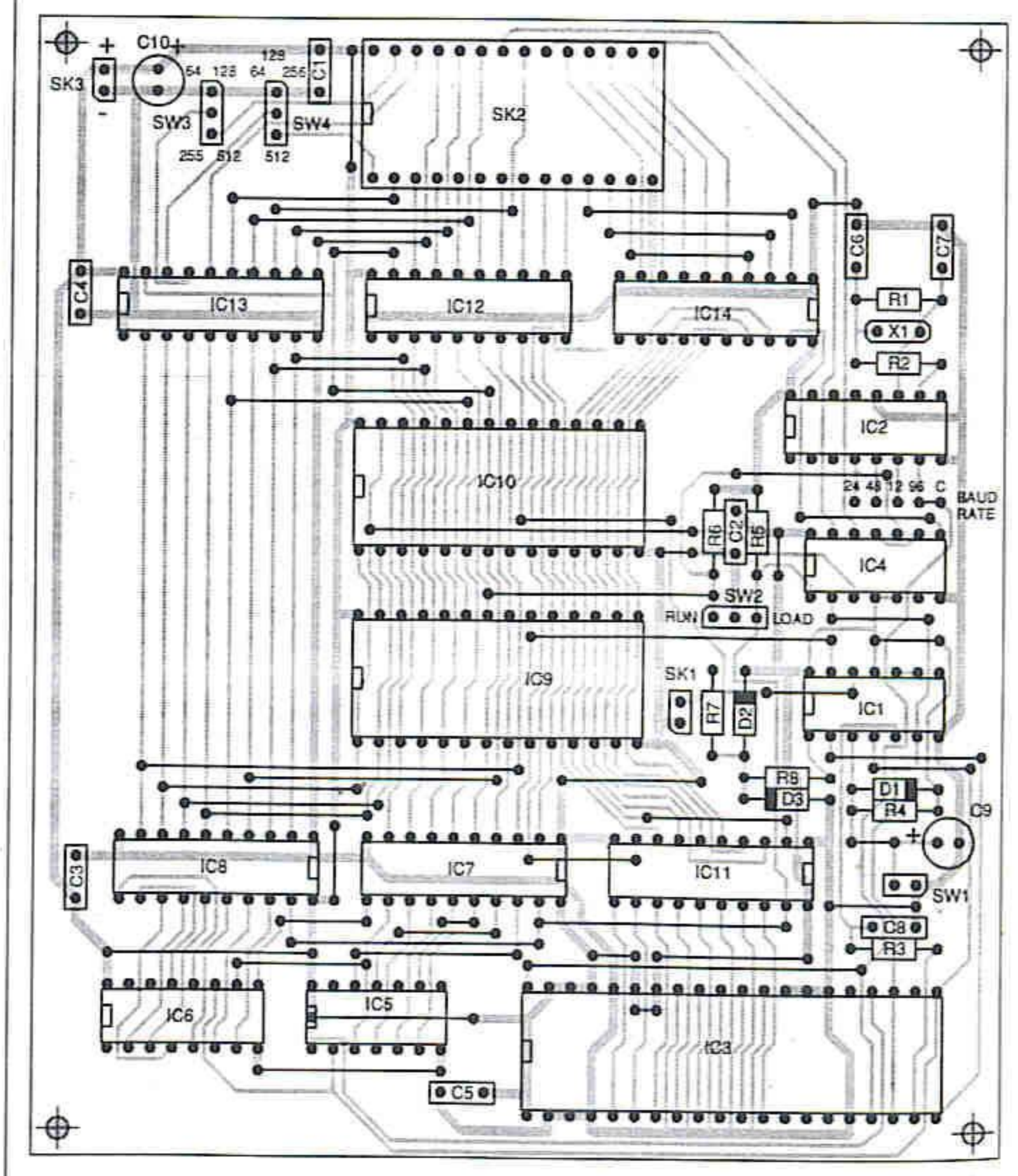
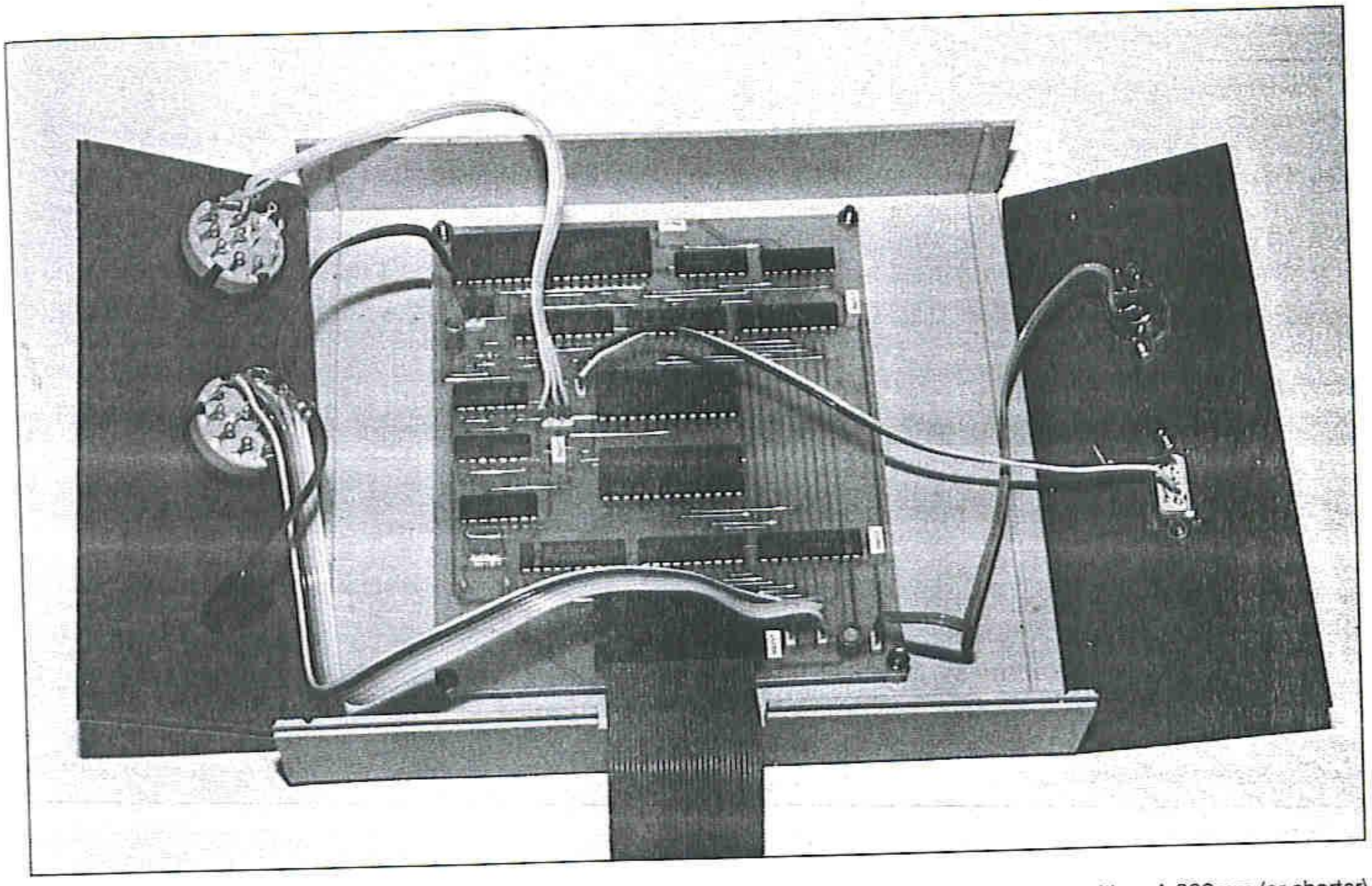


Fig.4. EPROM Emulator PCB component overlay



lower (Run) position. IC7, IC8 and IC11 are now disabled, isolating the RAM from the upload circuitry. IC12 and IC13 are enabled, effectively connecting the RAM address bus to the 'EPROM' socket. SW3 and SW4 select the EPROM type, and set any unused RAM address lines low. When pins 22 and 24 of the 'EPROM' are both low, IC14 is enabled (via IC4) connecting the RAM data lines to the 'EPROM'.

The prototype has not yet been tested on a speed-critical microprocessor system. The interface electronics should add no more than 20ns to the access time of the RAM devices used. Therefore with 100ns RAM chips, the unit should be able to emulate a 120ns EPROM, and will definitely emulate a 150ns device. If speed is critical, fit the fastest RAM chips you can get - 35ns devices are available if you are prepared to pay for them!

Construction

The unit is assembled on a single sided PCB, which is available from the ETI PCB service. The copper track layout is shown in figure and the component overlay is shown in figure

A number of wire links are required, which should be fitted before any components, since some pass underneath ICs. I would suggest that the resistors are fitted next, followed by the ICs, then the capacitors, then the remaining parts. Fit a link wire between COM and 96 to set the Baud rate to 9600. Fit SIL header strip or Veropins for the off-board connections.

I would suggest that sockets are used for the RAM chips and the UART, in view of their cost. It would also be a good idea to use sockets for IC4, IC12, IC13 and IC14, since these interface to the outside world and could be damaged if there is a problem on the system being tested. The sockets allow for ease of changing if the worst happens.

If you do not plan to emulate 27512 EPROMs, you could save a few pounds by omitting IC10 (it can always be added later). The unit will then emulate up to 27256 devices, so move the stop on the Device switch to prevent the 27512 option being selected.

Fit a 28-way IC socket in SK2 position. A 300mm (or shorter) length of 28 way ribbon cable should be fitted with a 28 way DIL connector on each end. These can be readily pressed together in a vice or WorkMate, if three thicknesses of Veroboard are used to protect the connector pins. Take care when doing this, as it is not easy to get the connector apart again if something goes wrong. One end of this cable plugs onto SK2; make sure the edge of the cable with the different colour goes to pin 1.

When you receive these connectors from your supplier, please make sure the two parts are not pressed fully together before removing from the polythene packing. If they are, send them back. The two I purchased from Maplin were fully assembled and I managed to break one while disassembling them.

The interwiring is shown in figure This should be carried out at this stage, since it is necessary for testing. After testing, the board can be fitted into the case.

The connections for both 9 and 25 way serial connectors, use whatever matches the socket on your computer. On the prototype a 9 way D connector (serial) and a 6 way DIN socket (DC input) were fitted to the case. The rotary switch connections are shown by giving the pin number or letter marked on the switch body.

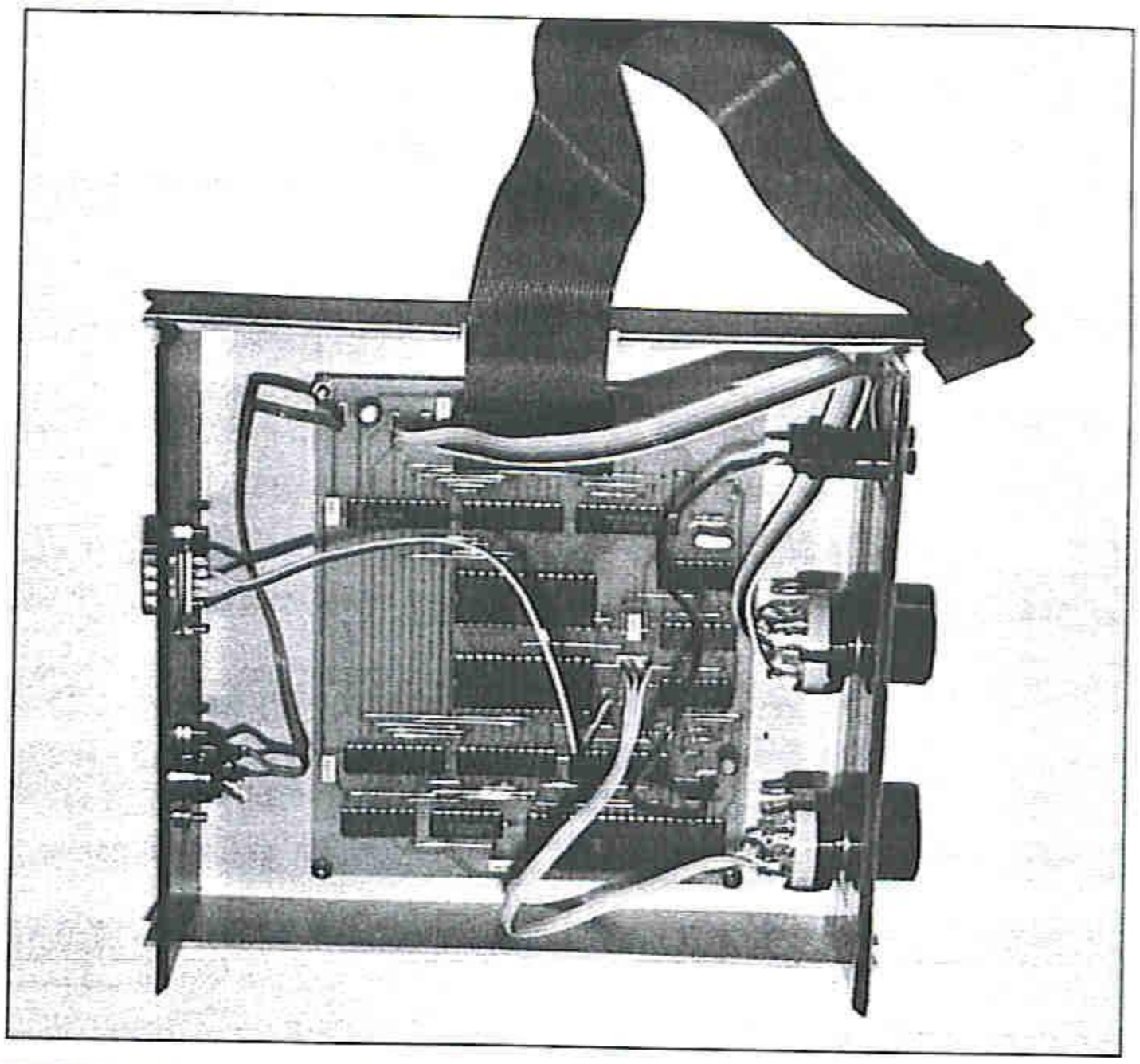
Testing

Connect the unit to a 5V supply and the serial port on your PC, and run the program "SER-TEST.EXE" on the software disk. When prompted, type "1" or "2" followed by Enter to say which serial port you are using. The program does nothing more exciting than wait for you to enter a 2-digit hex number (followed by Enter) and then sends it to the emulator. It then attempts to read back a number; if it's successful it prints the number otherwise it prints "***". Since the emulator does not transmit any data, the software will always respond with "***" To exit just press Enter on its own. All the responses in this section will be shown with quotes ("") around them - just type the

number (followed by Enter); don't type the quotes.

Switch the emulator to 27512 and Upload, and press Reset. Type "00" on the computer, and the software should respond with "**". Using a logic probe, test meter or oscilloscope, check the logic levels on pins 9, 10, 11, 13, 14, 15, 16 and 17 of IC9. They should all be low. If you type "01", pin 9 should be high, and the others should remain low. Now enter "02", "04", "08", "10", "20", "40" and "80" in turn. After each entry, check the logic levels on the data pins; only one should be high in each case - 10, 11, 13, 14, 15, 16 and respectively.

Press Reset on the programmer. Press Enter on its own to quit the software then run "ADR-TEST.EXE", which is also on the disk. Since the address counters are incremented when a byte is sent, it would take a long time to get the count to 65535 manually! ADR-TEST does it automatically, and pauses at four points to allow you to check the logic levels. Follow the instructions on screen. The table below shows the expected logic levels on the address pins of the IC9 at the four pause points.



Addr Line	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
IC9 Pin	20	27	26	2	23	21	24	25	3	4	5	6	7	8	9	10
Count	Expected Logic Level															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21845	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
43690	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
65535	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

If these readings are OK, and the preceding tests were also successful you can be fairly confident that the unit is working correctly. It is not so easy to check emulation part of the unit, but providing the unit was carefully constructed the chances of problems are unlikely.

If you have a microprocessor system and a suitable hex file , you can try emulating a known good EPROM using the information given shortly.

The Case

The prototype was constructed in a plastic case, 190mm 165mm * 68mm, see parts list for details. A suitable overlay for the front panel is shown in figure *. Two photocopies may be taken (enlarge to 162mm * 64mm), one can then be used as a drilling template while the other may fixed to the front panel with clear self-adhesive vinyl.

A suitable notch should be cut in the right side of the case for the ribbon cable to pass through. The power and RS232 connectors mount on the rear panel.

Software

The software for this project is supplied on one 3.5" 720K disk. Please note that this disk also contains the software for the EPROM Programmer, published last month.

Since the various programs extend to over 2500 lines of BASIC source code, it would make boring reading to print it in the magazine. A printed listing is not available since it would be more expensive than a disk to produce.

The software is supplied 'as-is' and neither Paul Stenning or Electronics Today International can accept any liability for any loss or damage however caused. The source code is supplied so that you may modify the software for your own use only. The software may not be redistributed in either its original or in a modified form. If you cannot accept these conditions please do not order the software disk. That's the legal bit done!

A batch file is supplied on the disk to simplify installation. Insert the disk in the drive, type "A:" then "INSTALL", and the batch file will make a \EPROM directory on your drive C:, and copy the software to there. If you do not have a hard disk, make a working copy of the disk using DISKCOPY, then put the original away. Do not write-protect your working copy or the software will not work.

If you are using Windows, suitable icon, PIF and group files are supplied on the disk. Some parts of the software will operate much slower under Windows, particularly the initialisation when the serial port is opened. However it will run in the background (probably very slowly) if you are using 386 Enhanced Mode.

The main software of interest for this project is spread over two programs, "EMULATE.EXE" and "HEX-CONV.EXE". The first of these is the main control software for the emulator, while the second converts various industry standard hex file formats to and from the EPROM emulator format.

Additional programs on the disk are "PROGRAM.EXE" which controls last month's EPROM Programmer, and "SPLIT2.EXE" & "SPLIT4.EXE" which divide Intel hex files into 2 or 4 files for 16 and 32 bit systems respectively. Since the full BASIC source code is given for all of these programs, it would be possible to create one large program containing all the facilities - if someone had more time than me! I would be interested to see

PARTS LIST

Resistors

R1	4M7
R2	2K2
R3,8	1K0
R4	47K
R5,6,7	4K7

Capacitors

C1,2,3,4,5	100n
C6,7	22p
C8	2n2
C9	10u
C10	47u

Semiconductors

IC1	74HCT04
IC2	74HCT4060
IC3	6402
IC4	74LS32
IC5	74HCT4024
IC6	74HCT4040
IC7,8,11,12,13,14	74LS245
IC9,10	43256-10
D1,2,3	1N4148

Miscellaneous

XT1	2.4576MHz
SK1	9 or 25 way D connector
SK3	DC INPUT
SW1	Push to Make
SW2	4 pole 3 way rotary
SW3 and SW4	one 4 pole 3 way rotary

PCB, Case, knobs, wire, ribbon cable, two 28 way DIL ribbon connectors

any enhanced versions.

When "EMULATE.EXE" is started, the Device Selection Menu will appear. From here you choose the type of device you will be using, either 2764, 27128, 27256 or 27512.

Once you have chosen the device required, the Main Menu will appear. Option 1 allows you to upload hex data to the emulator. The hex data is saved and loaded ASCII-Text format which is peculiar to this software. "HEX-CONV.EXE" will convert to and from this format.

When Option 1 is selected, you will be told where to set the switches on the emulator. You will then be asked for a filename; simply enter

8 the alpha-numeric characters - the extension is fixed to .HEX and does not need to be typed. Now sit back and wait; the progress will be shown on the screen.

Option 2 allows you to change the EPROM type as previously. Option 3 lets you run the Hex File Convertor program, "HEX-CONV.EXE" and option 4 lets you access a DOS Shell; type "EXIT" to return to the emulator. To quit the emulator, press Escape.

The Hex File Convertor was described in detail last month, so I won't bore you by repeating it here!

Emulating

The emulator may be powered by the microprocessor circuit being tested if there is sufficient capacity in the power supply. This will happen by default, via pins 14 and 28 of SK2.

If you need to power the emulator separately, you will need to isolate pin 28 of SK2 from the circuit under test, to prevent the two power supplies conflicting. This is easily achieved by removing pin 28 from a spare IC socket, and then fitting this onto the free end of the ribbon cable before plugging it into the test circuit.

The 300mm length of ribbon cable should not cause any problems unless the microprocessor system is very fast. In this case, try making another cable just long enough to reach.

The ICs interfacing this unit to the outside world are 74LS TTL devices, since these are somewhat more robust than 74HCT, and the inputs are not so static-sensitive For speed-critical systems you may need to use a different logic family here.

When the unit is not in use, it's a good idea to plug the end of the ribbon cable into a piece of anti-static foam. This helps protect the electronics from static, and prevents the pins from being bent.

The plastic case used for the prototype is made by Bafbox, and is available from RS/Electromail, stock no 506-788.

The software for this EPROM Emulator and last month's EPROM Programmer are available on one disk from the author. ' Please send a blank PC formatted 720K 3.5" disk; the price for the software is £10. If you do not have a suitable disk, send £12 and one will be supplied.Please add an extra £1 if within Europe, or £2 elsewhere, to cover the additional postage.

The author can also supply PCBs for the EPROM programmer, the PSU, and next month's project, the EPROM emulator. The EPROM PCB is priced at £12, the PSU PCB at £5, and the Emulator PCB at £15, or if you want to buy all three then they are available at £30. Do not forget to add P&P: £1.50 in the UK and £2.50 elsewhere.

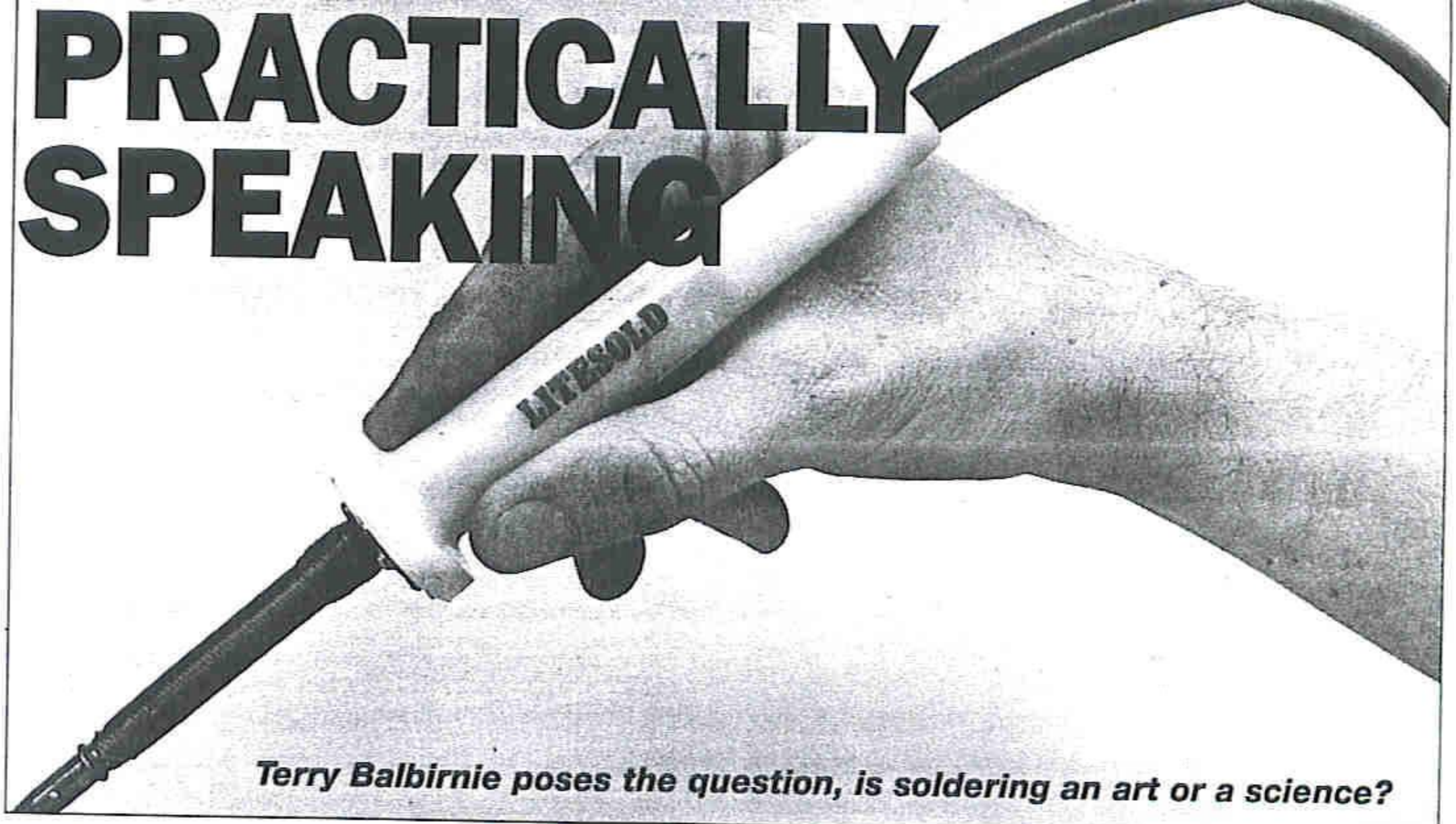
If you would like to purchase either the software or the PCBs then send a cheque or postal order for the appropriate amount made payable to "Paul Stenning", plus a return address label to the following address:

Paul Stenning,1 Chisel Close,Hereford,HR4 9XF.

f ordering from outside the UK, please make sure your cheque is in Pounds Sterling and is drawn on a UK bank. The author will be making up PCBs in batches so please be patient, cheques will not be cashed until the boards are dispatched.

The above offers from the author are only valid until the end of March 1995. After that date, please contact him first before sending any cash.

PRACTICALLY SPEAKING



Terry Balbirnie poses the question, is soldering an art or a science?

After spending some months looking at the setting-up of workshops, it is now time to consider more constructional matters. This time we shall examine the soldering equipment which is available. Over the next few months, we shall continue with this topic then examine soldering technique and the tools needed to make a good job Soldering is a skill which all electronics hobbyists must master in order to make a permanent circuit. A properly-made soldered joint will last indefinitely and provide a low electrical resistance combined with the mechanical strength to hold the connection together.

Solder itself - at least the type used in electronics - is an alloy (a mixture of metals) usually consisting of 60% tin and 40% lead. This has a melting point of 188oC - less than either of its constituents. Solder used for electronics work contains cores of flux which run up the centre. Flux prepares the surfaces to receive the solder and without this, it would be difficult to make a satisfactory joint.

Solder is supplied in lengths or in reels by weight - often 500g and 2kg. The thickness is usually 18 SWG (1.22mm) or 22 SWG (0.71mm) although 24 and 26 SWG are available. It is much cheaper to buy reels (500g will cost around £7). 500g of 22 SWG solder contains about 175m so it will last a long time!

Tools of the trade

As well as solder, you will need a small soldering iron. These are inexpensive so it is not worth using an old hand-me-down. The chief points to consider are the size of its bit (although this is often interchangeable) and the power rating (wattage). After that, choose by the feel of the handle, balance, weight and price. There are some "own brand" bargains for less than £10. Maplin supply their Beginners Soldering Iron for £3.95.

For building amateur circuits, a 12W to 18W type will be suitable such as the Litesold LA12 (12W) or LC18 (18W) or Antex 12W Type M, 15W Type C or 17W Type CS. Antex kits include the soldering iron itself, bench stand, solder and a booklet about soldering - useful for those wanting to get started. Note that users of the mains-free workshop described in previous parts of this series must buy a 12V soldering iron rather than a mains model. Both the Litesold and Antex CS models mentioned above may be purchased in 12V versions or buy the Auto-Repair kit (no stand with this one). Many soldering irons can be bought with a silicone rubber lead at slightly greater cost. These are much less likely to suffer damage if they touch the hot tip. Replacement silicone rubber wire is also available from mail-order suppliers.

Established users should consider buying a thermostatically-controlled iron. These have a higher power element - usually 50W to 60W - so they heat up much more quickly. However, when the correct temperature is reached, the supply is controlled. Some models feature proportional control which supplies power to the element at the correct rate to maintain the temperature. Others have on-off thermostatic action like a domestic iron. These soldering irons do not tend to cool down when making many joints in quick succession as is the case with standard irons. Also, the bit stays in good condition for longer. Of the mains-voltage temperature-controlled irons available, some have a fixed operating temperature such as the Antex A718. The Litesold EC50 and Antex TCS240 have in-handle temperature control. An iron like this will cost between £40 and £50. No temperature-controlled irons appear to be available in a 12V option.

Station pick-up

Some soldering irons are used as part of a soldering station. These have a 24V supply provided by a transformer in the base unit. Basic models use low voltage simply for safety - the base station will cost about £50. The user will then choose a 24V iron to go with it. More expensive stations have a temperature control on the base unit These use the soldering iron supplied. This will have a temperature sensor in the tip and a connection to the electronic control system in the base unit. Some have a meter which shows the actual operating temperature. Soldering stations sometimes have a spring stand built-in and a small tray for the sponge used to clean the bit.

NEXT MONTH

Next month we shall pursue this topic by looking at self-contained soldering irons and soldering technique.

PARABENDER

Barry Porter shows how to build a stereo parametric equaliser in modular form which may be configured to suit your precise requirements. Fully balanced inputs and outputs with professional facilities and performance mean that no audio system should be without one ...

An equaliser is a glorified tone control, and has one purpose in life - to modify frequency response. There are three main reasons for doing this - to correct for inaccuracies elsewhere in the system, to get rid of some unwanted part of the frequency range, or to obtain a particular effect which requires an uneven response in order to get the required sound.

Recording and broadcast studio mixing consoles invariably have an equaliser as part of each input channel, but due to space or cost limitations, these are often quite basic and inflexible. It is quite common to find console equalisers with a fixed degree of sharpness for the peaking sections, or high and low frequency shelving zones with fixed turnover frequencies.

To be classed as Parametric, an equaliser should have continuously variable control over its main parameters; namely sharpness of its bandpass or bandstop sections (normally termed "Q") operating or turnover frequency and lift or cut amplitude. At the operating frequency extremes, the equalisation characteristic should be capable of being selected to either a peaking or shelving response shape (these terms will become

Fig.1. Parabender block diagram

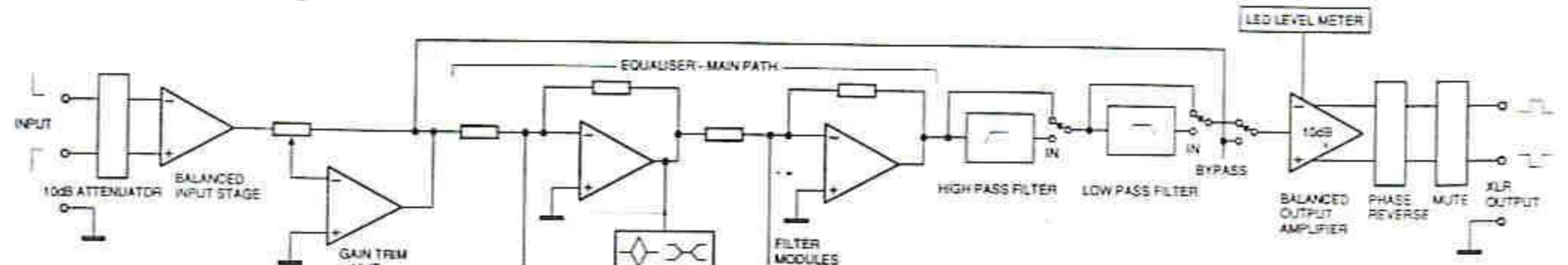


Fig.2. Parabender input stage

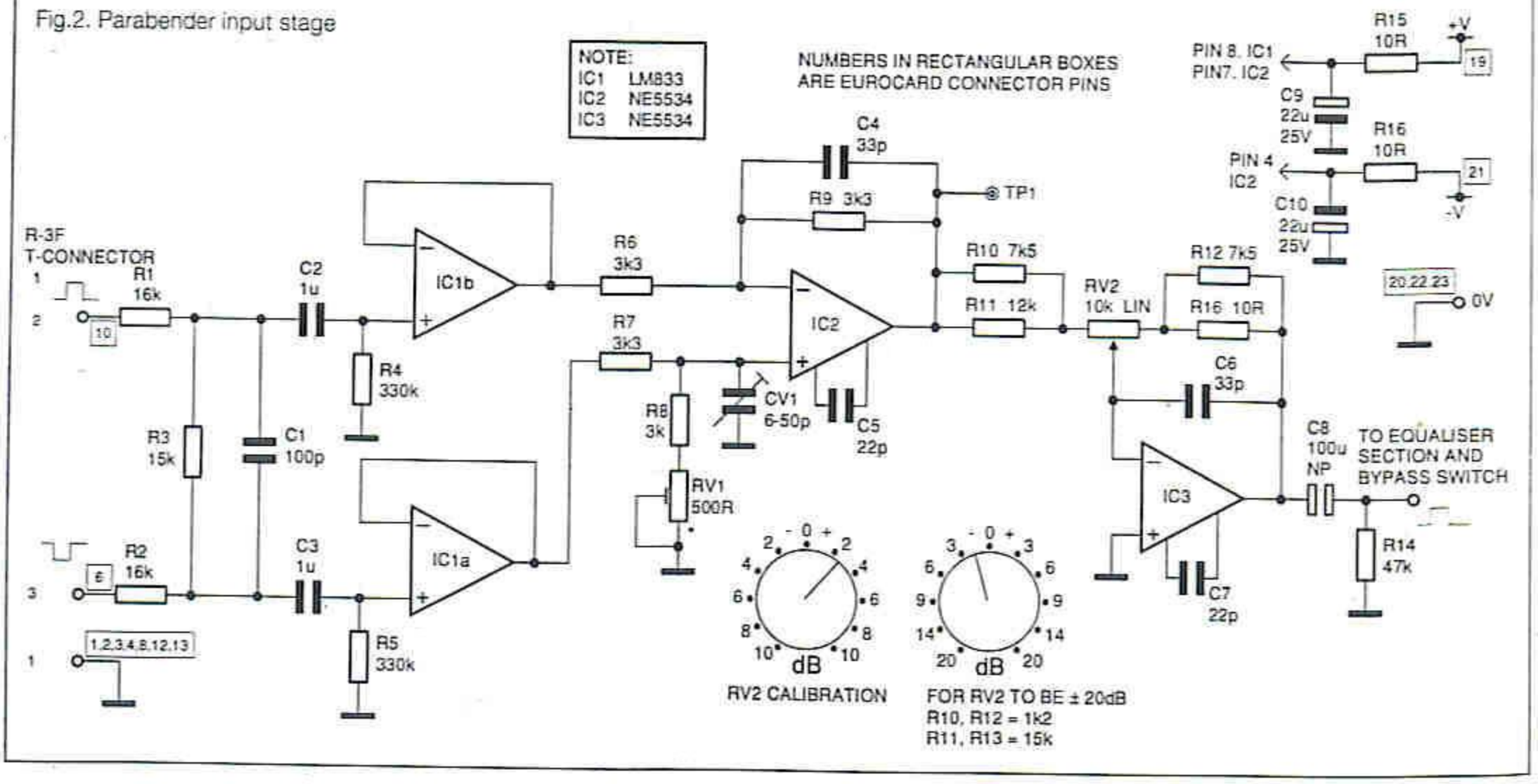
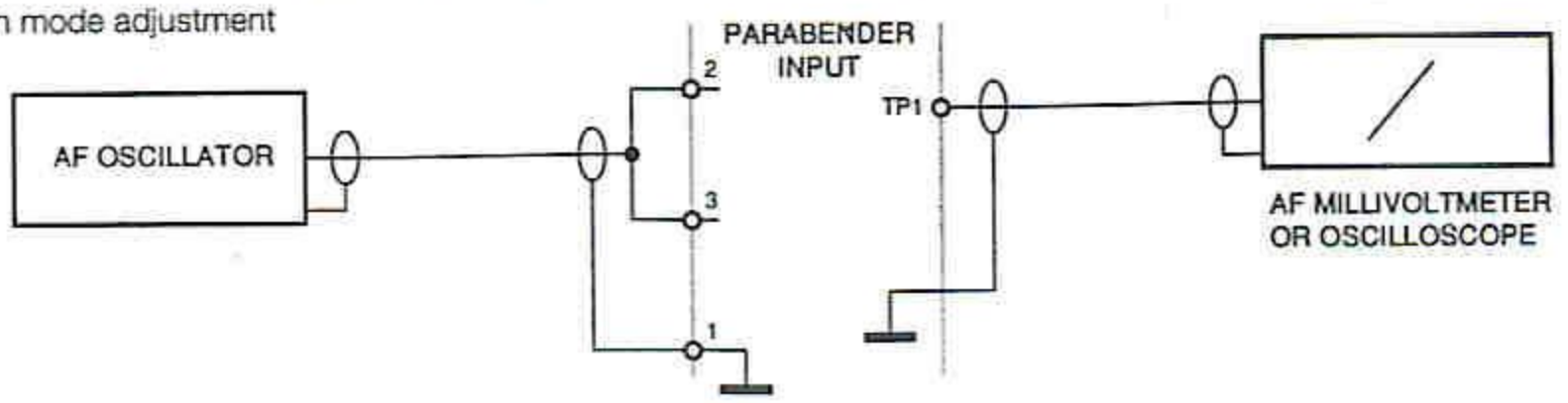


Fig.3. Input common mode adjustment



STEP 1	ARRANGE EQUIPMENT AS ABOVE
STEP 2	APPLY 100Hz SIGNAL AT APPROXIMATELY +10dBu (2.5v RMS)
STEP 3	ADJUST RV1 FOR MINIMUM OUTPUT
STEP 4	CHANGE FREQUENCY TO 10kHz
STEP 5	ADJUST CV1 FOR MINIMUM OUTPUT
STEP 6	REPEAT STEPS 2 TO 5 UNTIL OPTIMUM ADJUSTMENT IS ACHIEVED
STEP 7	APPLY LOCKING COMPOUND (NAIL VARNISH ?) TO RV1 AND CV1 ADJUSTMENT SCREWS
STEP 8	REPEAT WHOLE PROCESS IF ANY INPUT STAGE COMPONENTS ARE CHANGED

clear later) and it is usual to have variable high and low pass filters in the signal path.

The amount of available equalisation is overtly the subject of heated debate among audio engineers. The purist will usually maintain that he never uses equalisers, but if such things have to exist at all, they should have no more 6dB of lift or cut, as there can never be any sane reason for using more.

At the other extreme, the "pop" engineer, who normally uses equalisation to obtain particular effects, will complain that the customary 15dB variation is nowhere near enough, and when he has it explained that a wider range would seriously endanger the operating headroom, is likely to gaze blankly at the ceiling, shrug his shoulders and say that he doesn't see the problem.

Not everyone needs incredibly complex, expensive equalisers. For example, if a small amount of top lift is required to correct for loudspeakers with falling high frequency response, only a single band will be necessary, and to have more would not only be a waste of money, but could compromise both noise and reliability by introducing redundant circuitry into the singal path.

What is really needed is an equaliser that can be tailored to meet individual requirements, ideally by the use of plug-in units - say "Hello" to the ParaBender ... (Maybe it should be called the PluginBender!)

Basic Principles

Input Stage:

A block diagram of the ParaBender is shown in Figure 1.
The signal input is electronically balanced and is preceded by a 10dB attenuator in order to maintain an input overload margin of 30dB.
The following stage gives +/- 10dB of gain trim, so that the signal path through the equaliser section may be kept at approximately -10Bu for input levels between -10Bu and +10Bu. Details are given later if you feel that a wider control range is necessary.

Equaliser Section

The signal path through the equaliser section consists of two inverting amplifiers. The output from the first of these feeds the modular bandpass, or shelving filters and the filter outputs are directed to the input of either the first or second amplifier according to the position of the amplitude controls.

In the "Cut" position the filter adds frequency selective negative feedback around the first amplifier, thereby reducing the gain at frequencies corresponding to the filter output. In the "Lift" position, the same frequencies by-pass the input resistor of the second stage, giving additional gain at the frequencies in question.

The advantage of this equalisation system is that in both the Lift and Cut positions of the amplitude control, the filter output is added to the main signal at a virtual earth summing point, so any number of filters can be used without interaction between separate bands. It also allows different types of filter to be used, which is necessary in order to obtain both peaking and shelving response characteristics. An additional benefit is that when the amplitude control is in its central, flat position, the filter output is shorted to ground by the potentiometer centre tap, and therefore can add no unnecessary noise to the main signal.

Equaliser Filters

The individual filters are contained in separate plug-in modules, so any number up to six can be used in a stereo ParaBender. The limit is purely mechanical, and up to 14 can be employed with a single input/ouput module to build a mono unit.

Each module contains two filters - one giving a bandpass response, the other having a high or low frequency shelving characteristic.

The band-pass filter uses a triple op-amp "State Variable" configuration, which allows the centre frequency and "Q" to be independently varied without interaction between the controls.

The shelving filter is a single-pole, high or low pass type with variable turnover frequency.

In order that a single circuit board may be used to cover the complete audio frequency band, provision has been made for a range of frequency selective capacitors, and the shelving filter may be linked to operate in the high or low pass mode.

It is suggested that two types of filter are built - one operating between 20Hz and 1kHz, the other from 1kHz to 20kHz. The recommended Q range is 0.7 to 5, with an amplitude control giving +/- 10dB of variation.

Signal Path Filters

Following the equaliser section are double-pole high and low pass filters, both with variable operating frequencies and In-Out switching.
The high pass is of the equal value type in order to keep the adjustment potentiometer tracks of the same resistance, and the response has a Butterworth, maximally flat characteristic. The suggested frequency range is 10Hz and 200Hz. The low pass filter also has a Butterworth 12dB per octave response, and is adjustable between 5.5kHz and 21kHz.

It would be possible to have steeper roll-off filter slopes,

such as 18 or even 24dB per octave, but it is generally accepted that the two-pole variety sound better, giving adequate attenuation without introducing the hardness and ringing often associated with the more complex variety. The control pots are also considerably cheaper, and it amazing how often this can contribute towards improving the sound of something!

Output Amplifier

The output of the ParaBender is electronically balanced, and must provide gain of 10dB to counteract the input stage attenuator. The differential phase action of the output stage automatically doubles the signal voltage, so an adjustable gain buffer gives an additional 4dB, adjustable over a further 4dB range to enable the overall unit gain to be accurately set.

The output will operate unbalanced, but should have the unused leg connected to ground, when the gain of the opposite side will increase to compensate.

The output is equipped with a phase reversal switch, which simply swaps the output connections, and a by-pass facility which removes the equaliser section and filters from the signal path to allow the effect of any settings to be judged.

An output mute switch disconnects and grounds the ouput connections.

Level Meter

A 10 segment LED meter is driven by the output amplifier, and will be found useful in avoiding overload if excessive equalisation is used. Rather than use a standard meter I.C. such as the LM3915, discrete circuitry has been employed, as this allows access to the voltage reference and divider chain so that the steps can be set to suit individual requirements.

Circuit Details

Returning to the input stage, this time in detail, Figure 2 shows the relevant circuit.

The input attenuator is formed by resistors R1, R2 and R3. Taking into account the slight effect of R4 and R5, the actual attenuation is given by:

20 log (Rx) = 10.05sdB

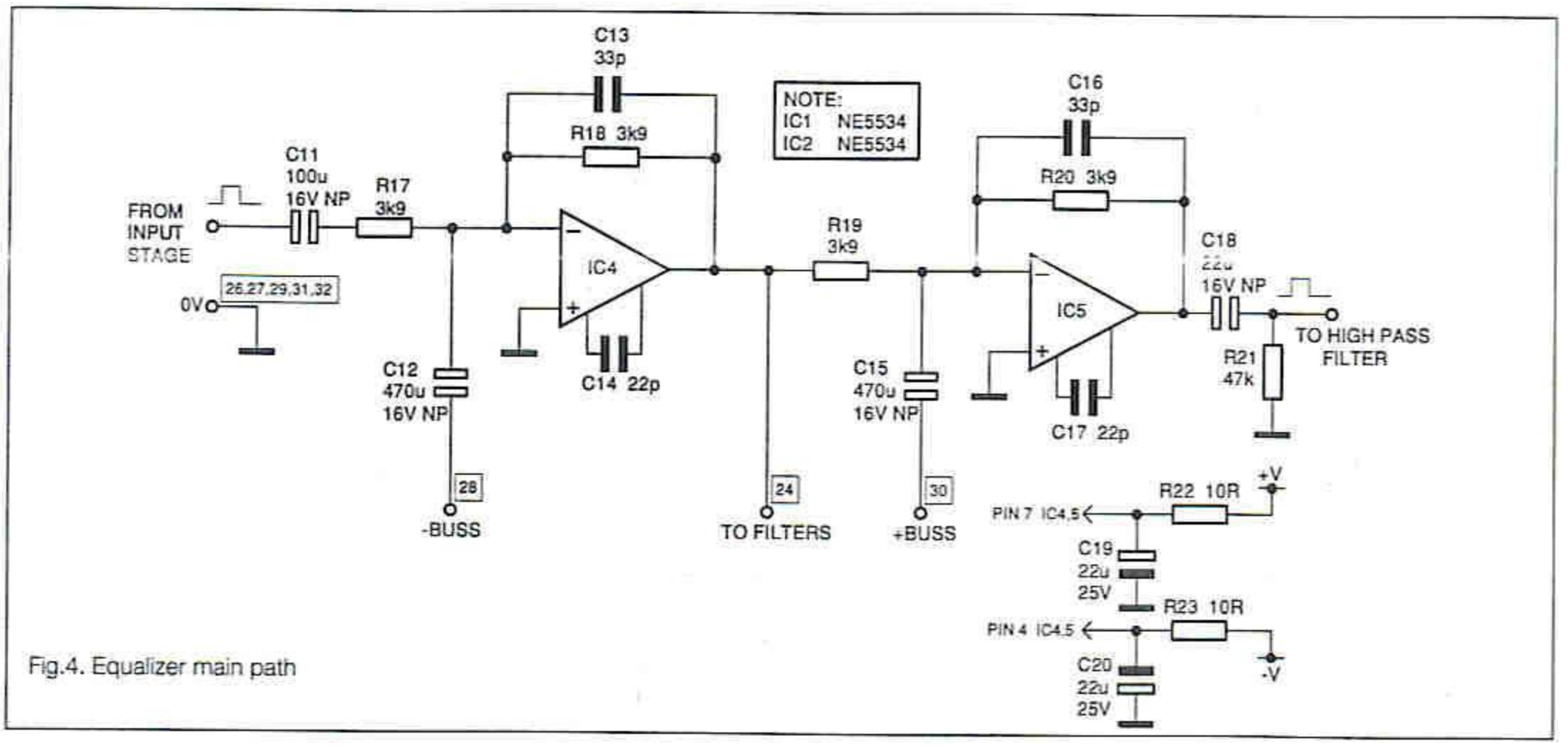


Fig.4. Equalizer main path

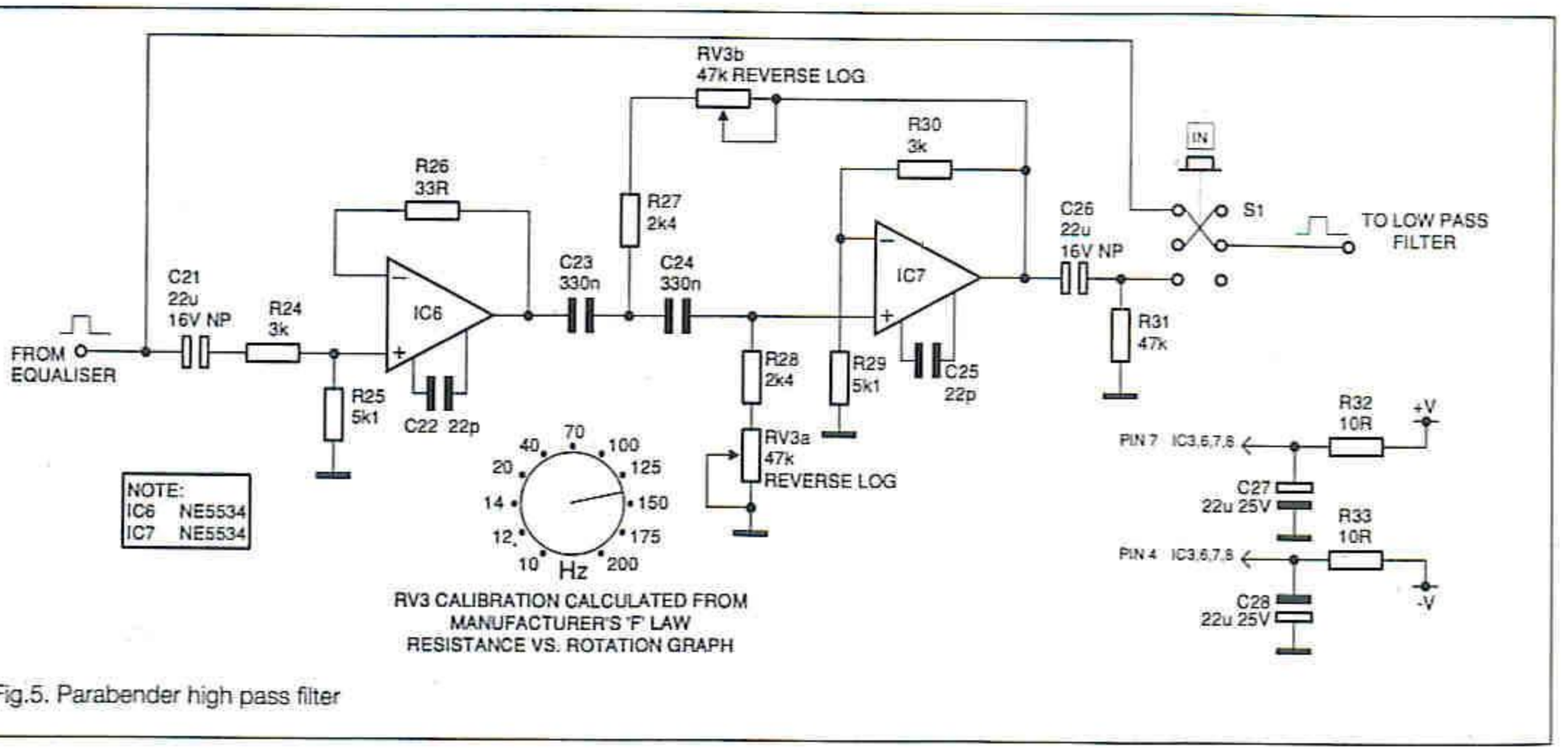


Fig.5. Parabender high pass filter

$$\frac{\text{-----------}}{(Rx + R1 + R2)}$$

$$\frac{R3 * (R4 + R5)}{\text{-------------}}$$

Where Rx = R3 + R4 + R5

Frequencies above 200kHz are rolled off by capacitor C1, and the low frequency turnover and phase shift are set by capacitors C2 and C3 with resistors R4 and R5, the -3dB point being:

$$(2\_C2R4) -1 = 0.48Hz$$

This leads to a 20Hz response of:

$$\frac{20 \log (2\_RCF)}{\text{------}}$$

$$(\_((2\_RCF)2 + 1) = 0.0025dB$$

With a phase shift of: 90 - tan - 1 (2_RCF) = +1.88°.

The incoming signal is buffered by unity gain stages IC1a and IC1b and then unbalanced by differential amplifier IC2. The common made rejection of this stage may be adjusted at both low and high frequencies by VR1 and VC1. The method of doing this is shown in Figure 3.

The gain trimming stage, IC3, follows IC2. This has a range of +/- 10dB with the values shown. For other ranges, resistors R10, 11, 12 & 13 may be changed, the desired value being given by:

$$\frac{R\ Pot}{\text{-----}}$$
$$A - 1$$

and the nearest resistance being obtained by selecting the

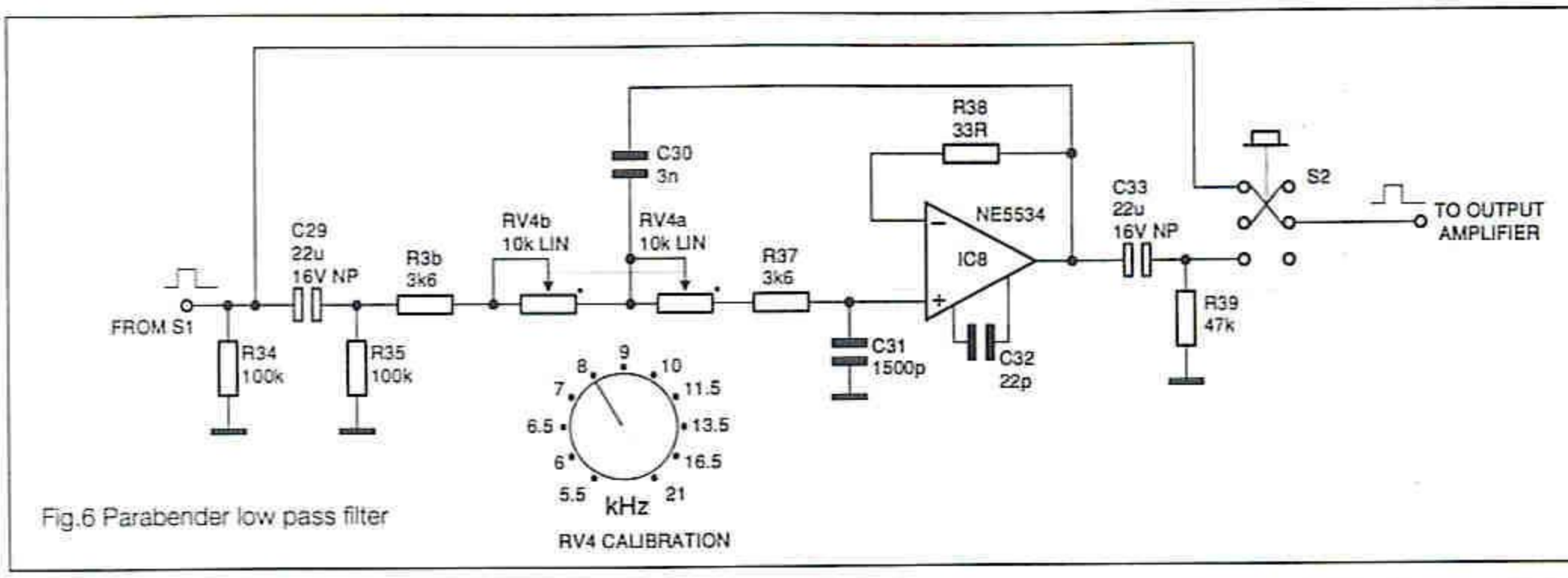


Fig.6 Parabender low pass filter

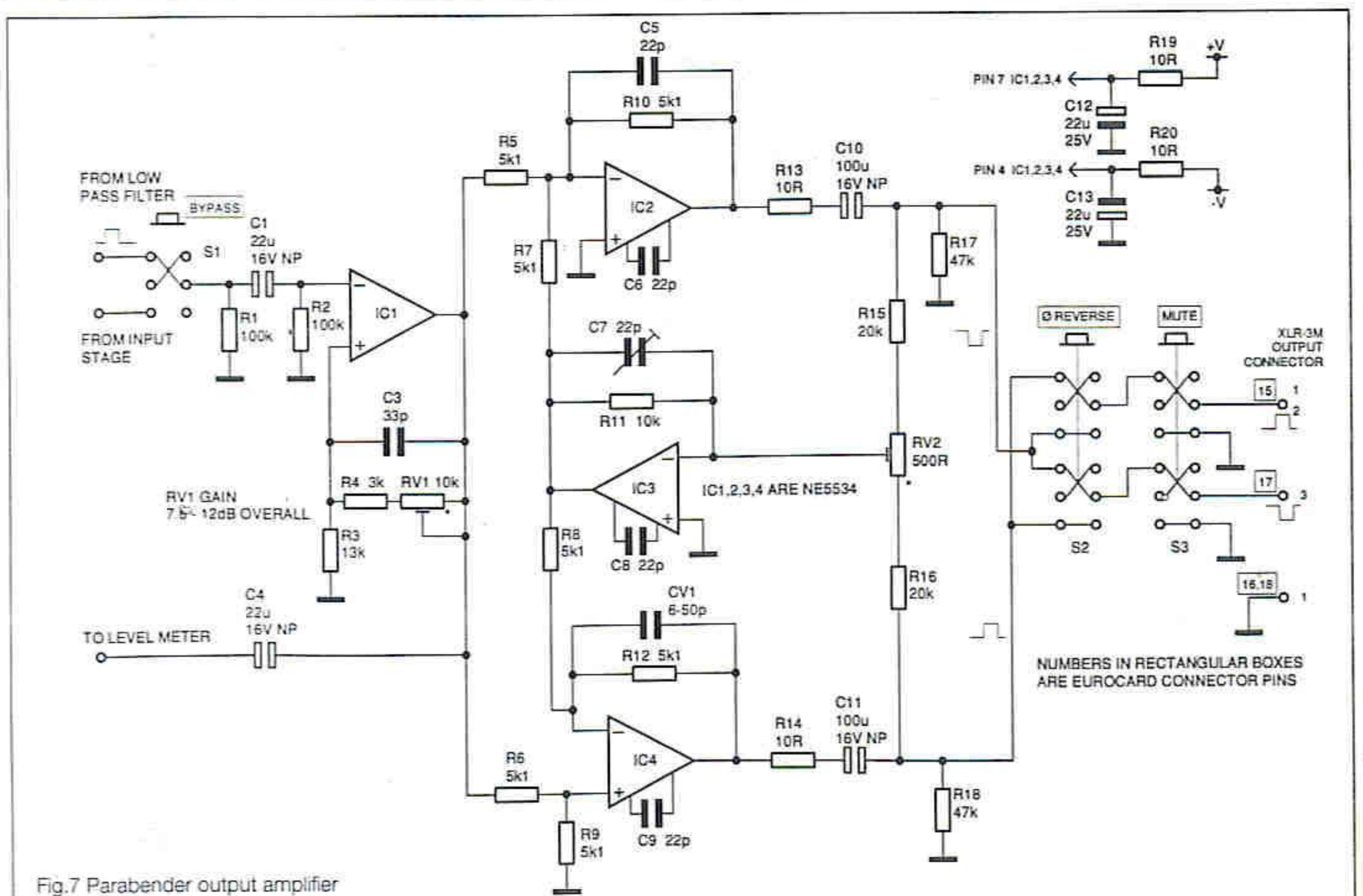


Fig.7 Parabender output amplifier

parallel combinations. For example, to obtain ±20dB range, each parallel pair of resistors should be:

```
                10k
                ---
                10 - 1          = 1.111k
  By trial and error, it is found that 1.2 * 15
                                                -----


                                                1.2 ÷
15      = 1.111
```

so if R10 and R12 are changed to 1k2 and R11 and R13 becomes 15k, the control will give ±20dB of level adjustment.

Note that as the gain trim stage is inverting in operation, a phase reversal has to be introduced to counteract this, which is why the wiring between the circuitry and input connector may appear to be wrongly shown.

The output of IC3 is A.C. coupled by capacitor C8 to enable the by-pass switch to be operated without introducing clicks onto the signal. For the same reason, the input of the equaliser is coupled by C11.

The dual invertor equaliser is formed by IC4 and IC5. (Figure 4) The value of coupling capacitors C8 and C11 may, at first sight, appear much larger than necessary. Remember that they are in series, and therefore must be treated as a single component of 50uF. This, when loaded by 3k9, has a 20Hz response of -0.0072dB, but more importantly, introduces a range of +2.34°.

The output of IC4 feeds the filter modules, and the amplitude control tracks are connected to the inverting inputs of IC4 and IC5 by capacitors C12 and C15. Again, the value of these may appear somewhat on the large side, because the low frequency response of the equalised signal is a function of capacitor C12 or C15 with the total filter output resistance. For 10dB of lift or cut, each filter has a series output resistor of 1k8. With C12 and C15 being 470uF and six filter modules, the effective resistance is therefore 300R.

This leads to the response at 20Hz being -0.014dB with a phase shift of +3.23°.

Okay, so it is totally unrealistic to attempt to use the ParaBender with every band set to maximum lift or cut, but it is usually best to play safe and design for 'worst case' situations, regardless of whether you think that no-one in their right mind will do a particular thing.

If you should decide to build a single channel ParaBender, fully stacked with 14 filter modules, the worst case situation becomes: 20Hz response: 0.075dB

20Hz phase shift: +7.5°

This performance is just about acceptable, so the size of C12 and C15 is justified, even though it is unlikely that anyone could hear the effect of reducing them to 100uF or thereabouts.

The output of the equaliser section is A.C. coupled to the high pass filter, again to prevent clicks as the filter is switched in and out.

Figure 5 shows the relevant circuit.

To keep the control potentiometer sections of equal value, the op-amp must provide some gain which is decided by the filter "Q". The amount of gain is given by:

Gain = 3 - (1/Q)

For a Butterworth response, the Q must ve 0.7071, requiring a gain of 1.5858 (4.005dB).

The resistor values given will result in this characteristic, but a bit of experimenting may lead to some worthwhile results. (See how you like the sound of a filter with a Q of 1.5 or 2).

In order to maintain unity gain through the filter, an input attenuator comprising resistors R24 and R25 reduces the signal level by the same amount as the filter gain - providing R24 = R30 and R25 - R29 of course. A unity gain buffer stage, I.C.6, ensures that the attenuator is not loaded, and provides the required low impedance source to drive the filter which is based around I.C.7.

The filter turnover frequency is given by our old friend:

```
F-3 = (2_RC) - 1
```

The values shown give a range of 9.76Hz. (For argument's sake, we'll call it 10 to 200!) but note that the frequency adjustment potentiometer must have reverse logarithmic, or "F" law tracks, otherwise the calibration will become extremely cramped at one end of the control rotation. As an example, if a linear potentiometer is used, the central position will give a turnover frequency of 18.6Hz, which means that the lower frequencies are very spread out, and those nearer 200Hz become impossible to select, particularly by anyone with a slightly shaky hand, and that means about 95% of all known audio engineers.

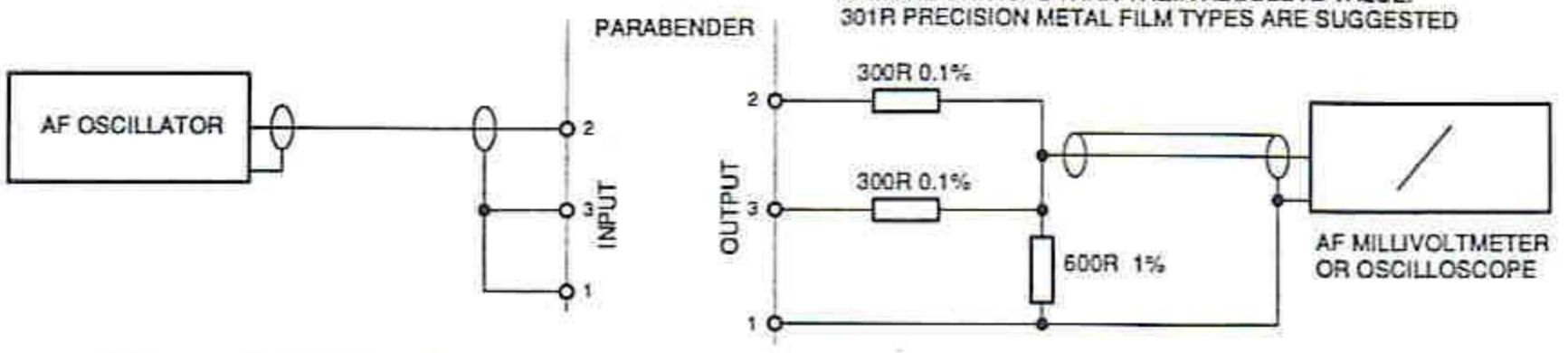
The output of I.C. 7 is coupled by C26 to the high pass filter selection switch, S1, from where the signal passes to the low pass filter, I.C. 8.

As shown in Figure 6, the high frequency roll-off is variable between:

```
                                  _2
                                  --
                        2_RC      = 5.517kHz and 20.840kHz
Where R = (R36 + VR4b) and C = C30
```

A bit of calculation will show that VR4 should have linear

Fig 8 Parabender Input unit PCB component overlay



STEP 1	WITH UNIT LINKED AS SHOWN, SET OSCILLATOR OUTPUT TO 100Hz AT +20dBu (7.75v RMS)
STEP 2	ADJUST RV2 FOR MINIMUM OUTPUT
STEP 3	CHANGE OSCILLATOR FREQUENCY TO 10kHz
STEP 4	ADJUST CV1 FOR MINIMUM OUTPUT
STEP 5	REPEAT STEPS 1 TO 4 UNTIL OPTIMUM ADJUSTMENT IS ACHIEVED
STEP 6	APPLY LOCKING COMPOUND (NAIL VARNISH ?) TO RV2 AND CV1 ADJUSTMENT SCREWS
STEP 7	REPEAT WHOLE PROCESS IF ANY OUTPUT STAGE COMPONENTS ARE CHANGED

tracks, as a reverse log law would result in a centre frequency of 16.3kHz, (with a log control it would be 5.9kHz) whereas with a resistance of 5k it becomes 8.72kHz and, as the calibrations shows, there is no excessive crampling at any point of the control rotation.

From the output of I.C. 8 the signal passes to switch S2, the low pass filter selection switch, and then on to the output amplifier which is shown in Figure 7.

Balanced Ouput Amplifier

The input to the final stage normally comes from the output of the low pass filter, but operation of the Bypass switch, S1, selects the output of the gain trim stage instead. (By the way, the apparent duplication of component numbers is not a deliberate attempt to confuse everyone - it just means that two similarly numbered components are on different circuit boards.)

Input buffer amplifier I.C.1 has gain, adjustable with VR1, of 1.8dB to 6.02dB, allowing the overall ParaBender gain to be accurately set.

Inverting amplifier I.C.2 and non-inverting stage I.C.4 drive the balanced output in anti-phase. Providing both output legs carry identical level but phase reversed signals, I.C.3 will receive no input, and play no part in the proceedings. However, if the output is unbalanced by one side being shorted to ground. I.C.3 will provide positive feedback to the opposite amplifier, increasing its gain by just under 6dB to counteract for the lost output voltage.

Ideally, the gain of the working amplifier should be increased by 6.02dB, but an analysis of the gain structure will show that this turns the output stage into an oscillator.However, with a 5.95dB increase, everything remains stable, and the missing 0.07dB is hardly enough to cause problems, even though there are, no doubt, many audiophiles who have had their lives ruined by considerably less!

The accuracy of the output balance is initially adjustable at low frequencies by VR2, which sets the inverting input of I.C.3 at the exact null point between the two output legs. At high frequencies, where the balances becomes affected by small phase differences and the outputs are not in precise anti-phase, VC1 is used to trim the balance to an optimum point. The method of doing this shown in Figure 8.

The two outputs are A.C. coupled via capacitors C10 and C11, which at 100uF each will drive loads of 1k and above with less that 10° of phase shift at 20Hz. Luckily, the days of 600R input impedances are just about over, bridging loads normally being 10k or more, so the output stage is quite capable of driving all modern equipment up to its maximum level of +28dBu (19.5V RMS) with less than 1° of phase shift at 20Hz.

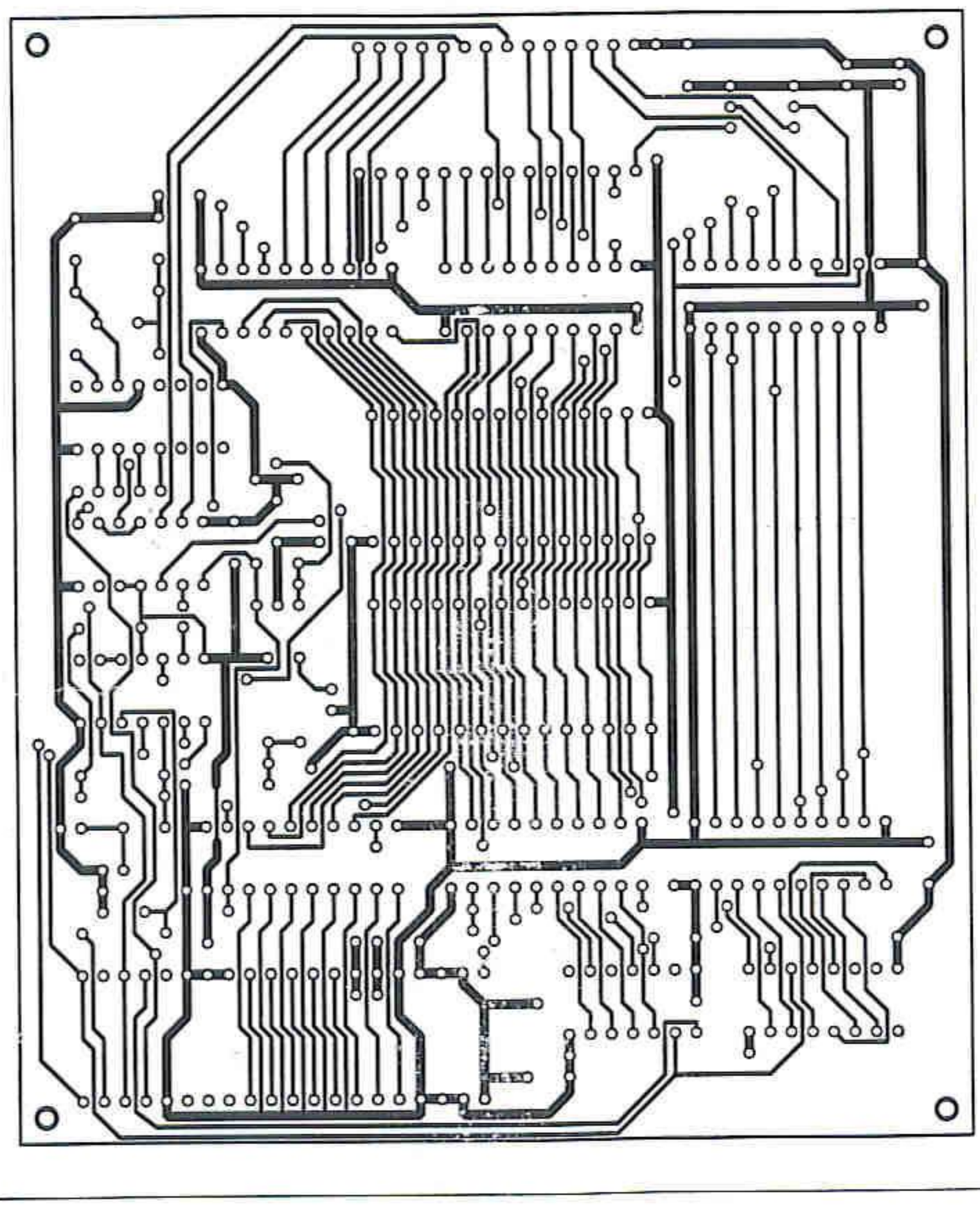
A phase reversal switch, S2, changes over the output lines, and can be used to correct for wrongly wired XLR interconnection leads, to check the phase of a channel by making a listening comparison, or, when both channels are reversed, changing the absolute phase of a signal to counteract other equipment which incorporates a reversal.

An output Mute switch, S3, is provided. This disconnects the output connector from the ParaBender circuitry, grounding the input of the following equipment in the process.

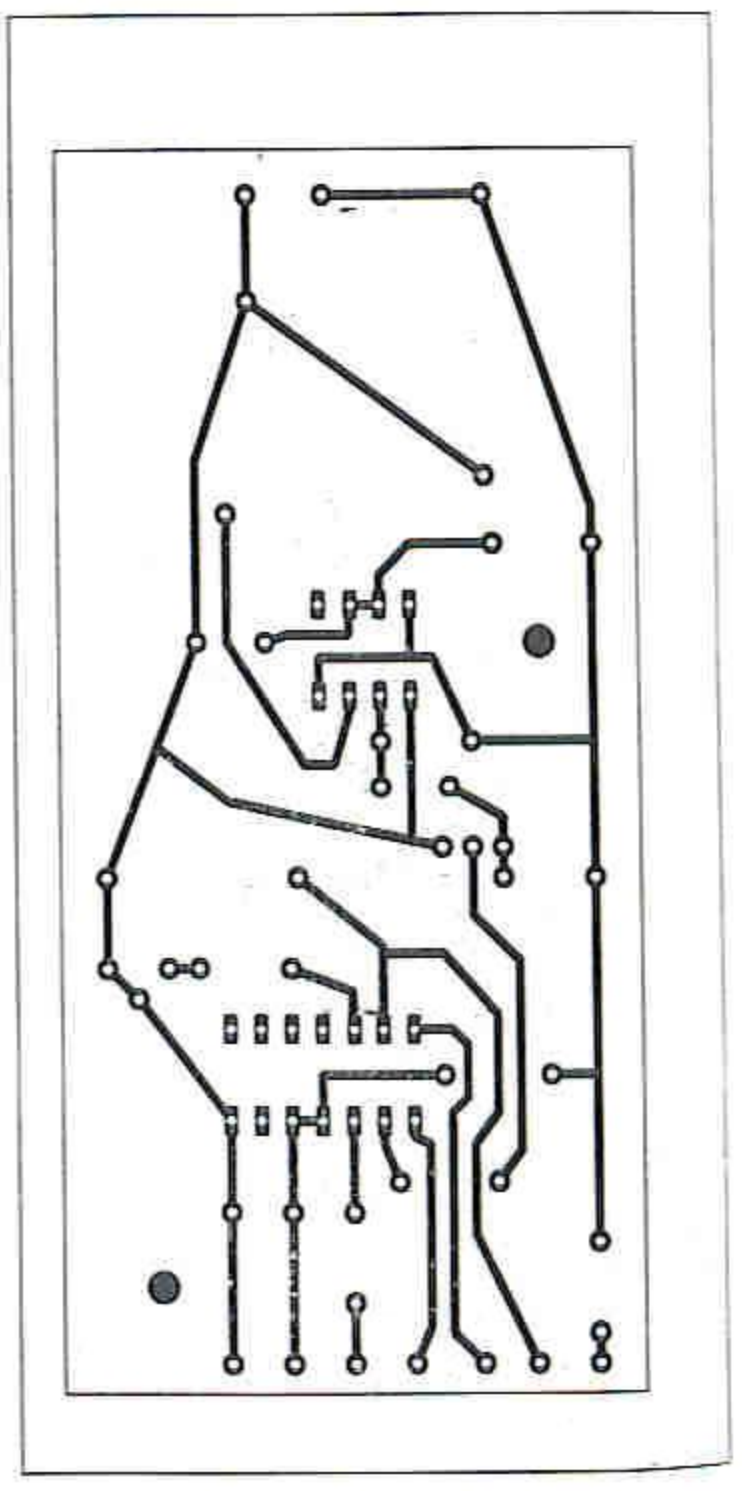
Next month

Next month Barry Porter concludes this project with the construction of the Level Meter, the Filter Modules, and the power supply. He also looks at the final construction and use of the equaliser.

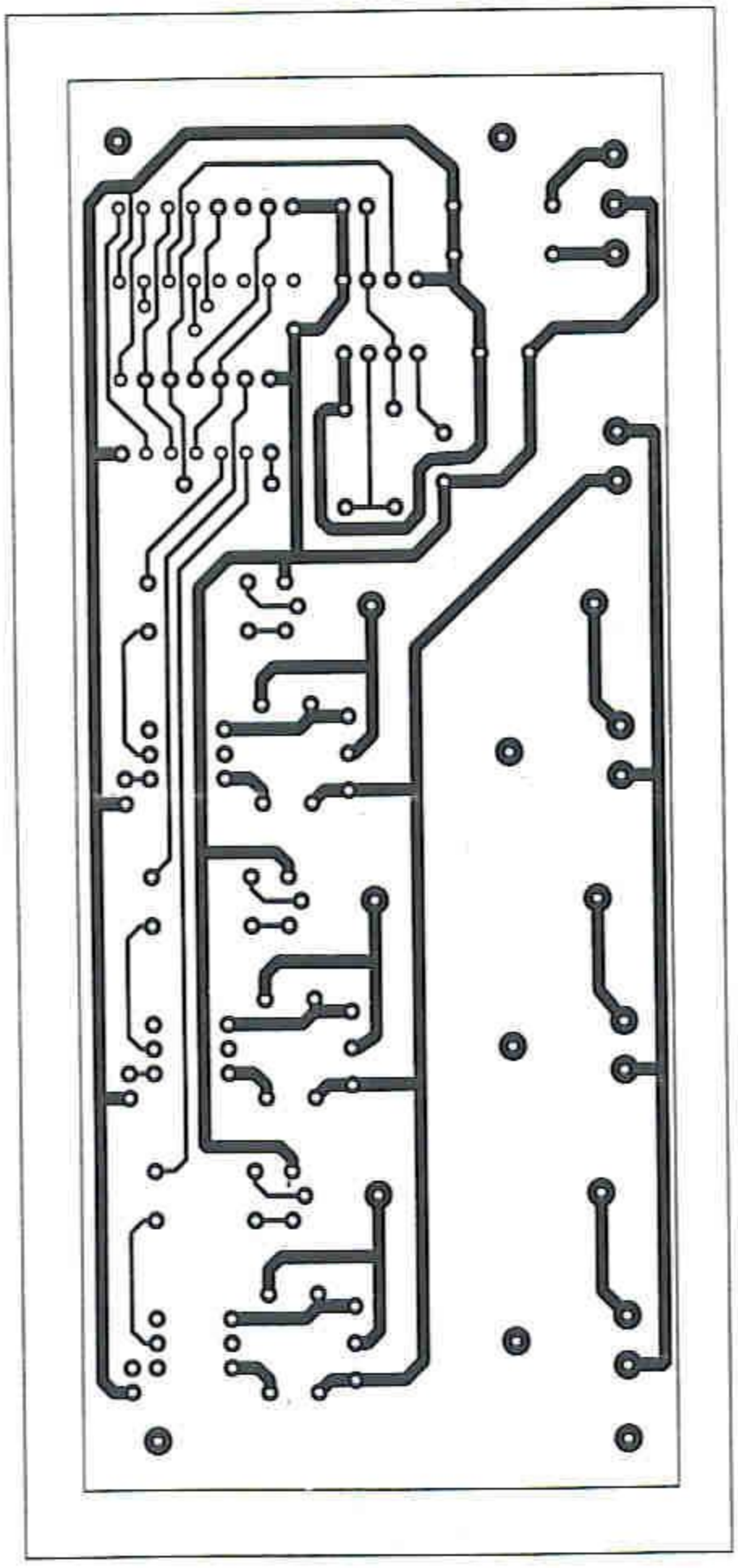
Foils for this issue



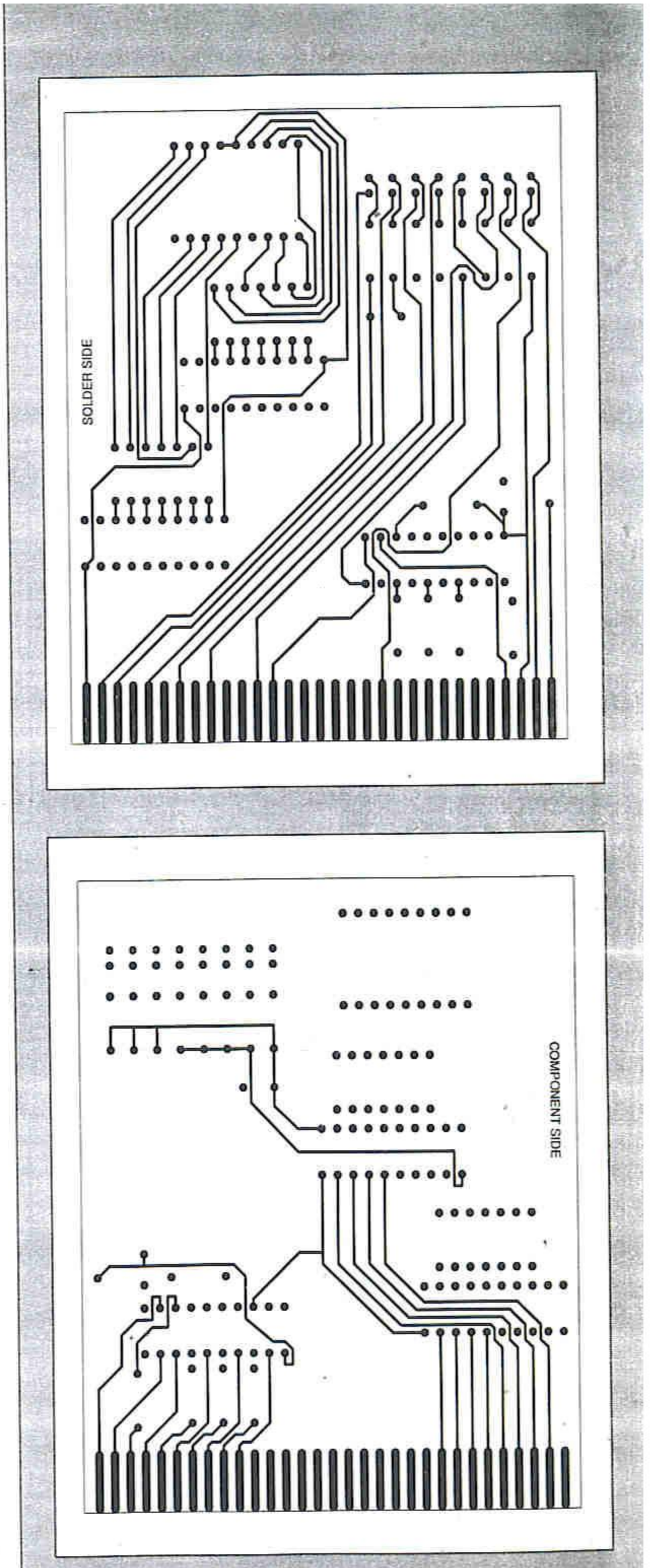
Eprom Emulator



Mini Enlarger Timer



Christmas Lights Flasher



Corrections To Foils For Last Month's Post Card Project