Early Technical Computing at Rover Solihull
By Roger Mills

INTRODUCTION

This article describes the early use of computers to solve engineering problems within the Rover Engineering Research Department at Solihull and covers the period from when I joined the company in 1964 through to the mid-1970s when the Technical Computing Section was absorbed into a wider systems department, and dispersed within what had by then become BL Cars.

THE PRE-COMPUTER ERA

When I joined the Rover engineering research department in Solihull in September 1964 there was not a computer to be seen either in the department itself or elsewhere in engineering. The company did however have an IBM 1401 mainframe computer which had recently been installed in a custom-built computer hall at the Lode Lane end of the factory. At the time, this was used only for business – not technical – applications, such as stock control, production scheduling, payroll, etc. More on that later.

I was joining the company as a fresh Physics graduate, having just received my degree from Imperial College London. The engineering research department was a little unusual in that it employed a high proportion of university-educated staff compared with other parts of Engineering, most of whose staff had gained their qualifications via the HNC/HND route.

The research staff came from a number of different disciplines. Besides engineering graduates there were graduates in physics, mathematics, aeronautics and electronics, and possibly other subjects. The department existed primarily to innovate and think outside the box, coming up with concepts and ideas which could be incorporated into future models without having the immediate time pressure of working within a specific model development cycle. The department was also the first port of call from elsewhere in engineering when any knotty problems arose, either in design or with production vehicles – both of which needed the best brains in the company to solve them. Many inventions made by members of the department were successfully patented. In addition, some of my colleagues were occasionally called as expert witnesses in court cases when Rover cars had been involved in accidents where they were asked, for example, to estimate from the available evidence how fast the car must have been travelling immediately before the accident.

It is worth noting that, at the time when I joined the company, Rover was still an independent company – manufacturing Rover cars and Landrovers (and experimenting with gas turbine applications) – and was not part of any larger conglomerate.

Rover and its predecessors were well known for innovation – from the very early days at the end of the 19th century when they invented a revolutionary bicycle to replace the Penny Farthing, through to being the first company to build a number of prototype cars using gas turbine engines – starting with JET 1 in 1950. Rover had also invented and produced its own automatic(ish) transmission as fitted to the Rover 105R in the mid-1950s. Looking back from today's perspective, this transmission looks like a creation of Messrs Heath and Robinson, but it nevertheless enabled Rover customers to have an automatic transmission option several years before the Ford-designed Borg Warner DG transmission – as fitted to later P4 and P5 models – became available in the UK.

My first job was to join a project team working on Torque Converters. At the time, the P6 2000 model had recently entered production, with the automatic version of that using a Borg Warner 35 transmission with a nine and a half inch torque converter. Several prototype P7 cars had also been built, using a 6 cylinder 3 litre version of the 4 cylinder 2 litre P6 engine – mounted in what was essentially a P6 body. By default, a 3 litre engine would need a larger (eleven inch) torque converter to handle its torque. But the combination of the 50% longer engine and larger bell housing would have resulted in an unacceptable transmission bulge inside the passenger compartment. The aim of the project was thus to design a 9.5" torque converter which could handle the torque of a 3 litre engine, and to demonstrate to Borg Warner that this was feasible.

The scope of the project was to evaluate a number of possible configurations on paper, to build a few of the most promising ones, and to test them – firstly on the torque converter test rig and then in a car.

[As an aside, some reports claim that Managing Director William Martin-Hurst had killed the P7 project in late 1963 – before I joined the company. If so, he didn't seem to have told the people with whom I was working. My recollection is that P7 development continued well into 1965, and only finally died when Rover bought the rights to manufacture the lightweight Buick V8 engine].

Rover had lots of torque converter design and development expertise, dating back to the 105R. The design calculations – involving a lot of 3-dimensional trigonometry and fluid mechanics – were well documented. However, for a given basic size and geometry, there are almost an infinite number of possible combinations of blade angles for the three components – pump, turbine and stator. The only aid available for performing these calculations was a calculator similar to the one shown on the left. Although electrically powered, it was entirely mechanical inside, and would whirr away for seemingly ages for each step of the calculation, eventually printing the result of that step on something resembling a narrow till roll. Using this method, each possible configuration took 2 or 3 days to evaluate. [Some of my colleagues also used a Monroe mechanical calculator as shown on the right (nicknamed "Marilyn" for obvious reasons) which had to be hand-cranked to perform calculations].

INTRODUCTION TO DIGITAL COMPUTING

Towards the end of 1964 IBM, who wanted to encourage greater use of their computing equipment – they would, wouldn't they! – set up a Fortran programming course at their offices in Birmingham aimed at engineering use of the 1401. Being a relative newbie, I was not invited to attend this course, but the senior members of the department did attend – possibly in order to evaluate the potential rather than with any intention of becoming programmers themselves. Nevertheless I was able to pick up enough knowledge from them after the course to enable me to get started on programming.

This was my first encounter with digital computers. Looking back after 60-odd years, it seems strange that, in my degree course at one of the country's leading universities, there was no mention whatsoever of computers – all calculations being performed either with slide rules or log tables. This was despite the fact that computers such as the UNIVAC and LEO had existed since the early 1950s. However, I thought nothing of it at the time.

GETTING STARTED ON PROGRAMMING

Since the torque converter project was still very much alive, this was an obvious first candidate for having its calculations performed by a computer. I set about writing software to do just that, and I also managed to manipulate the equations to some extent so that not only could we calculate the performance of a particular configuration, we could within limits calculate a configuration to achieve a specified performance.

The IBM 1401 computer occupied a room somewhat larger than an average domestic lounge, having a number of tall cabinets housing the processing unit, memory and magnetic tape drives, and two very large boxes containing a line printer and card reader. Nevertheless, it had several orders of magnitude less memory and processing power than even a modest current day smartphone!

It could only perform one task at a time, and it was fully scheduled to run commercial applications until 10pm each evening. This meant that we could only use it **after** 10pm. We had been given basic training on how to operate it, and were left to our own devices after the regular operators had gone home. The modus operandi was that we would prepare our programs and data during the day, using specially formatted Fortran coding sheets. We would then submit these to the card-punching team located in the old admin block close to the computer hall. This enabled us to arrive at 10pm with a deck of punched cards ready to be fed into the computer. Our Fortran program would be converted into machine code by the Fortran compiler. A program called the Link Editor then had to be run in order to add all the necessary system routines and produce an executable program which – I think – was punched onto another deck of cards. Finally, the executable program followed by the data could be fed in and run. There were, however, limitations caused by the small amount of core memory

– only a few tens of **kilobytes**. Don't even think about Megabytes or Gigabytes! One of the system routines was the Formatter which processed the input data and made it ready for use by the program, and which formatted the output data ready for printing on the line printer. The Formatter occupied a vast chunk of memory, leaving insufficient memory for a program which involved a large number of calculations, such as the torque converter program. This meant that such programs needed to be split into 3 parts. The first part read in the data, and stored it unformatted on a magnetic tape. The second part read the data from the tape, performed the calculations, and wrote the results – still unformatted – onto another magnetic tape. This part did not need the Formatter. The third part read the output data from the magnetic tape and formatted it and printed it on the line printer.

Although this sounds like a long, drawn-out process, it nevertheless meant that a torque converter configuration could be evaluated in minutes rather than days. The fact that we were operating the computer ourselves also meant that if our program contained an error – such as a missing bracket or misspelt variable, we could quickly punch a replacement card and correct the error.

[As a bit of light relief, the regular operators had acquired some decks of cards containing 'fun' programs. One of these programs, called "Edith", produced images on the line printer, consisting just of printable characters, depicting a young lady in various states of undress. In the final one, where it was just starting to get interesting, she was holding up a modesty screen! Another program, by manipulating the frequency at which the printer's hammers operated, produced a sound which was recognisable as the Colonel Bogey March.]

OTHER AREAS WHERE COMPUTERS WERE APPLIED

We continued to use the 1401, mainly for torque converter design work for several months. However, with the arrival of the V8 engine, the need for a high capacity 9.5" torque converter immediately disappeared.

I then turned my attention to accelerative performance predictions. My colleague Peter Stubbs had been performing these manually for several years, using what can best be described as a paper-based spreadsheet – with rows and columns of numbers. The basic data for this included the engine torque curve (as a tabulation of torque versus speed), gear ratios and efficiencies, final drive ratio and efficiency, tyre rolling radius, vehicle weight, aerodynamic drag coefficient and vehicle's frontal area, tyre rolling resistance, moments of inertia of rotating items (engine flywheel, road wheels, etc.) which enabled the effective mass of these components to be calculated and added to the vehicle mass. The calculation process involved step-wise integration, calculating the time taken to get from one engine data point to the next, and the speed change and distance travelled in the process.

Computerising this process not only meant that calculations could be performed much more quickly but also enabled a number of refinements to be added. For example, engine torque data could be entered at smaller speed intervals. Also, whilst it was assumed by default that changing up to the next gear would occur at the maximum power point, there were occasions when optimum performance could be achieved by changing up earlier if the power at the wheels had dropped to a lower value than that available in the next gear. The program was able to take account of this, and move the calculation to the next gear at the optimum point.

A little later, a version of the program was created for vehicles with torque converter automatic transmissions. This was more tricky than for manual transmissions because there was no longer a linear relationship between engine speed and road speed in a given gear due to the action of the torque converter. This needed an iterative process to be used, which would have been totally impractical to perform manually without a computer.

Over time, many more programs were added to the portfolio – including predicting roll and yaw angles for vehicles driving in circles around the steering pad at MIRA, calculating hill-climbing performance for vehicles towing various weights of trailer, particularly at high altitudes (such as in the Alps) where engine performance was reduced due to the lower barometric pressure.

CHANGE OF MAINFRAME

After a couple of years or so of late night computing (I forget the exact date), The IBM 1401 was replaced by an IBM 360, which was much more powerful – although not **that** powerful by modern standards. Sadly, we research engineers were not permitted to operate the new computer. Instead, we had to submit a deck of cards in the afternoon for running overnight, and to collect the printout the next morning. Whilst this

avoided the necessity for burning the midnight oil, it meant that if there was an error in a program, it would fail to run and we would have to submit a correction the next day, losing a day in the process.

In 1966, Stephen Crouch joined the department having just graduated at the Advanced School of Automobile Engineering, Cranfield (now part of Cranfield University), where he had acquired a vast amount of expertise in mathematical modelling of dynamic systems. Stephen's arrival coincided with the launch of a project to study the feasibility of producing an active suspension system which would prevent a vehicle from rolling when cornering without adversely affecting the ride quality. The motivation for this was the desire to keep the road wheels upright when cornering so that the tyres' cornering power would not be compromised by adverse camber angles. The conventional method of reducing roll by fitting very stiff anti-roll bars would have resulted in a very harsh ride over single wheel bumps. The plan was to start with a fairly soft suspension for maximum ride comfort, and to supplement that with a hydraulic system – with engine-driven pump, hydraulic rams and accumulators – which would transfer fluid from one side to the other, to raise the outer side and lower the inner side in response to lateral g forces. The initial experiments were carried out using a Citroen DS as a mule, because that already had a hydraulic suspension. Later, systems were created for P6-type vehicles. The project was a collaborative venture with Automotive Products at Leamington Spa, who provided most of the special hardware. The control system for this was purely mechanical, using damped pendulums to respond to the g forces, in turn controlling spool valves to direct the lateral fluid flow.

When designing such a system there are many variables to be taken into account, such as the amount of damping on the pendulums in order to provide a rapid response without overshoot, how far the spool valves need to move as a function of g force, etc. Using his Cranfield-acquired expertise, Steve was able to build a complex mathematical model of the whole system – including the vehicle itself – using a series of differential equations. These could be used to predict how the car would respond to (say) a step input from the steering wheel. Because the response was typically a high frequency waveform involving hundreds if not thousands of data points at intervals of a few milliseconds, these needed to be displayed graphically rather than as a sea of numbers.

THE NEED FOR GRAPHICAL OUTPUT

However, the IBM computers which we had been using had no provision for graphical output. This didn't matter too much for (say) torque converter calculations because the results contained a limited number of data points, so plots of torque ratio and capacity factor versus speed ratio could easily be produced by hand from the tabulated output data. But in this case, it **did** matter and a solution was sought. The solution adopted was to use an external computer bureau, the equipment at our end comprising a teletype with keyboard, printer and paper tape reader/punch, along with a Calcomp pen plotter. A dial-up connection to the bureau was made and, once connected, the telephone handset was inserted into an acoustic coupler in order to act as a crude sort of modem. Communications took place at the breakneck(!) speed of 110 Baud. When start, stop and control bits were taken into account, the actual throughput was 10 bytes per second. Yes, you heard that right – not kilobytes, not megabytes, but BYTES! To add insult to injury, there was no error correction so, in the event of noise on the line, the printed output would contain some gibberish. If it was not too bad, we could usually work out what it should have said but, in the case of the plotter – which received binary data – errors due to phone line noise could cause the pen to jump all over the place. Nevertheless, as if by some miracle, the system worked well enough to fulfil its purpose – receiving binary data and plotting it in the form of dynamic response curves.

THE LATE 1960s

This pattern of working continued through the late 1960s, with a number of research engineers writing their own programs and running them either on the IBM mainframe computer or via an external bureau. Although, by then, The Rover Company had been acquired by Leyland Motors, who already owned Triumph, it was still "business as usual" within Rover Research despite the imminent a merger between the Leyland Motor Corporation and British Motor Holdings (comprising BMC and Jaguar) which would create British Leyland.

CREATION OF TECHNICAL COMPUTING GROUP WITHIN ROVER RESEARCH

In 1970 a decision was made to create a computer group within Rover Research, which would have its own computer and which would develop and run software systems to support other research engineers, who would then be free to concentrate on the engineering aspects of their work.

And so “Technical Computing” was born. Stephen Crouch, who by then had left the company, was brought back to lead it. The Research Department was housed in a wooden hut – known as WH3 – located between the engine test houses and the WW2 air-raid shelters, In order to accommodate the new section, two Portakabins were installed at the north end of WH3 – one to house the computer, and one as office and media storage space. The Portakabin housing the computer was air-conditioned – making it a popular congregating place during heat waves!

Our first computer was an IBM 1130 which, although physically smaller than either of the mainframe machines, was specifically designed for engineering and academic use. Its default high-level programming language was Fortran unlike the commercial machines which had Cobol as their default language, with Fortran being an “also ran”. The 1130 was thus optimized for running Fortran and was much more efficient for this than the mainframes. We were able to use essentially the same Fortran programs as we had run on the mainframes but with the advantage that this machine was dedicated to Technical Computer and could be used as needed rather than only at dead of night.

The standard equipment included a control desk with a golfball Console printer, a line printer and a card reader and punch. In addition, it had removable disk storage in the form of IBM 2315 disk cartridges and also a rebranded Calcomp digital pen plotter. Each disk cartridge consisted of a plastic casing about 15” in diameter and 1.5” high, and contained a single 14” oxide-coated disk platter. It had a massive(!) capacity of 2.5 **megabytes** of program or data. Contrast that with the hard drives on today’s laptop computers which are 2.5” in diameter, and hold Gigabytes – or even terabytes – of data!

The plotter enabled graphical output to be produced, largely eliminating the need to use external bureaux.

My recollection was that it was the smaller of two available models, plotting on 11” wide paper. It only had a single pen so, to slightly mis-quote Henry Ford, you could plot in any colour you liked as long as it was black!

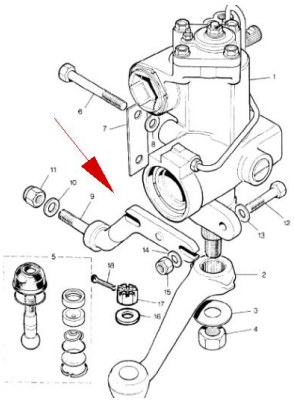
As stated above, the creation of the Technical Computing section represented a slight change in direction of travel whereby those of us employed in the section were creating and running software applications in support of other engineers rather than being directly involved in engineering projects themselves.

OCCASIONAL CONTINUING VEHICLE PROJECT WORK

Nevertheless, we still liked to keep our hands in with the occasional project. One which I remember concerned a problem with the Range Rover shortly after it entered production. It was discovered that even a relatively small amount of imbalance on one of the front wheels would cause a very nasty steering wheel torsional vibration. The Range Rover still had a Landrover-type live axle, but now had a coil spring suspension rather than leaf springs – giving it a much smoother ride but losing the inherent friction damping which comes with multi-leaf springs. Lateral location of the axle was by means of a Panhard rod, with the chassis end attached to a forging bolted to the side of the chassis, and extending below it. A few inches away, on the other side of the chassis box-section was a conventional steering box whose output shaft also extended down below the chassis.

On studying the dynamics of the steering system, it became apparent that a single wheel imbalance would cause that wheel to bounce up and down on the springiness of the tyre. This caused a small periodic rolling motion by the axle. The resulting gyroscopic torque caused the wheels to oscillate about their steering axes which wound up the steering system, causing the steering wheel to oscillate. The early Range Rovers did not have power steering, and they had rather heavy steering wheels with a large moment of inertia. It’s probably easiest to envisage what was going on by considering a simplified model consisting of a torsionally flexible shaft with a flywheel at each end. If one flywheel is subjected to a periodic torsional excitation, the other flywheel will oscillate at an amplitude which is at its maximum when the excitation frequency matches the resonance frequency of the system.

We set about producing a mathematical model of the steering system in order to evaluate how the vibrations could be reduced by changing certain parameters. This required us to measure the physical characteristics of the components – including the torsional stiffness of the steering system. This required building a rig with the help of the research department's fitters. This was based on a Range Rover chassis with the front wheels clamped to prevent them from steering. The steering wheel was then rotated and torque versus rotation angle was measured.



When winding up the steering in this way, I noticed that the chassis was distorting, with the end of the Panhard rod bracket moving in one direction and the steering box extension moving in the other. We carried out an experiment with a fabricated tie-bar which tied these components to each other. This significantly increased the torsional stiffness of the steering system, and the modelling showed that this would greatly reduce the steering wheel vibration by increasing the resonant frequency such that a resonance would only occur above the top speed of the vehicle. This was borne out in real life, and a production version of the tie-bar was designed and fitted to all subsequent vehicles of that type. I have always been proud of my contribution to Range Rover refinement! [I noticed that this same tie-bar was fitted to Landover Defenders right up until the end of their production – although it probably would have been unnecessary after the introduction of power steering.]

We continued to use the IBM 1130 for about 3 years, after which time it could no longer meet all our requirements. In particular, a requirement had arisen to be able to process analog data recorded onto magnet tape – mostly generated by in-vehicle test equipment. Initially, we had been able to take the tape recorders and tapes to IBM's Birmingham office to be processed by one of their larger 1800 computers, but this was inconvenient and again could only be done late at night. We therefore needed an in-house computer which could perform real-time processing.

In 1973 the IBM 1100 was replaced with a Hewlett-Packard 2100 which was more powerful, and had a real-time processing capability. It came with two different operating systems – DOS for every-day use and RTE (Real Time Executive) for real time applications, such as capturing and digitising analog data at the rate at which it was replayed from an analog tape recorder. TheHP2100 still had only 16k 16-bit words (32 kB) of core memory because this was very expensive. I seem to recall that 1k words of core memory cost £1000! Perhaps this was not surprising when you consider that each BIT of core memory needed its own individual ferrite core – so that 16k words needed over a quarter of a million cores. Contrast that with current day solid state computer memory, which comes in multi Gigabyte chunks and only costs pence per megabyte.

The HP2100 used similar disk cartridges to the 1130 – still holding only 2.5 MB each. There were two drives, so the operating system and application programs would be on one, and the data on the other. Compiled Fortran programs were very efficient in those days in order to fit into the memory space available – there was no room for today's 'bloat-ware'!

Also attached to the HP2100 was a DEC GT40 Vector Graphics Terminal which was used for many graphics based applications and was the first terminal we had had which could display anything other than alphanumeric characters. This contained a DEC PDP 11/10 computer, and was able to perform many graphics manipulations independently of the 2100

We also had a much better pen-plotter than before – able to produce plots up to about 30" wide in **multiple colours**. In addition to being used for our own research applications, this was popular with other parts of Engineering. For example, David Searle and his colleagues from (vehicle) Programme Timing Department were often to be seen printing PERT and GANTT charts produced by their Project Management software.

By then, we had a team of 6 or 8 programmer/analysts – some of whom continued to develop the type of applications which had been run on the IBM 1130, while I led a team specialising in real time applications – both capturing real-time data and performing various types of statistical analysis on it. One type of analysis

calculated the amount of fatigue damage suffered by suspension and transmission components while performing instrumented road and pavé testing. Whilst the fatigue life of various materials is well documented, the underlying fatigue tests are invariably based on constant amplitude fatigue cycles, resulting in a graph showing number of cycles to failure versus stress amplitude. But, in real life, components are subjected to complex cycles, where multiple small cycles are superimposed on larger ones. We developed a suite of programs based on so-called 'Rainflow Analysis' (an explanation of which can be found on Wikipedia) which, in effect, extracts the small cycles from the large ones and constructs a table of the effective number of cycles measured at each of many amplitude ranges. The total amount of fatigue damage suffered can then be aggregated.

AUTOMATED ENGINE TEST BED

One real-time application which took up a great deal of my time in 1974/5 required a dedicated computer of its own. One of the nearby engine test beds was being refurbished in order to perform automated endurance tests – particularly on the Triumph-derived 2.3 and 2.6 litre engines destined for the Rover SD1.

Hitherto, all the test beds had used manually controlled water dynamometers, where the operator needed to adjust one handwheel to control the load and another one to keep the balance arm horizontal so that the torque could be read on a spring balance. Control of the engine throttle was also manual. To make the adjustments in order to establish the desired speed and torque was not a quick process, and was totally unsuited to running cycles in which each step was of relatively short duration.

The refurbished bed was fitted with a Heenan & Froude F-type dynamometer in which the torque was reacted by a load cell rather than a spring balance and where the load was remotely controlled by a servo controller. There was also a remote servo-controlled engine throttle actuator. The dynamometer and throttle actuator together could be operated in various modes – for example to maintain a constant speed regardless of load as the throttle was opened and closed, or to control both throttle and dynamometer to achieve a specified speed and load. All of the necessary control electrics [I hesitate to say electronics because it was largely a big heap of relays providing relay logic and a few operational amplifiers for the 3-term PID (proportional, integral, derivative) servo controllers] were housed in a control cabinet located just outside the test bed. The control panel on the cabinet also had many switches and push-buttons, potentiometers for setting speed and load, and analog speed and load displays. For steady-state testing, the load and speed could be set infinitely more easily than with the previous purely manual equipment. However, it would still have been difficult to run an endurance cycle from the cabinet – hence the need for a computer to provide sequencing and other functions. Technical Computing was given the job of providing the computer hardware and software.

A Texas Instruments 960 was selected for the job. The processor and memory had a similar capability to that of the HP2100, but it was much more compact, and capable of sitting on a desktop. It was specifically designed for process control operations, and had lots of analog and digital inputs and outputs. The operating system provided the framework for many software modules each to run at timed intervals or in response to interrupts. The 960 had no mass storage as such, but it had a terminal with built-in cassette drives and thermal printer. The cassettes looked like audio cassettes, but they had been adapted to store digital rather than analog information. We acquired a similar terminal and connected it to the HP2100, enabling the two computers to communicate by transferring cassette tapes from one to the other.

For efficiency both of memory usage and speed of operation we opted to programme the 960 in Assembler Language – which is, in effect, a symbolic representation of machine code. This still needed an 'assembler' program to convert human-readable assembler code into binary machine code (0's and 1's). Texas Instruments gave us an assembler program – written in Fortran – and

designed to run on a mainframe computer. One of our brightest programmers was tasked with adapting this to run on the HP2100, and did a brilliant job. We were thus able to write and assemble our code on the HP2100 and then transport it to the 960 on cassette. We also wrote software on the HP2100 which could compile engine endurance cycles, creating data tables which could be transferred to the 960 on cassette and correctly interpreted by its software.

We developed a suite of about 50 software modules on the 960 which not only put the engine through its paces but also implemented a sort of 'black box flight recorder' which would store speed, throttle position, torque, engine temperatures, etc. in a circular buffer at sub-second intervals so that, in the event of an emergency shutdown, these data could be analysed to determine what had gone wrong. In addition, the software constantly monitored speed, torque and temperatures, and alerted the operator if any parameters were outside of tolerances defined in the test schedule.

The interface between the 960 and the control cabinet was a collaborative venture between Technical Computing, the Rover Research Electronics Department and Heenan & Froude. I spent many happy hours at H&F's Worcester premises thrashing out what additional relay logic and analog and digital inputs and outputs their cabinet would need to enable the test bed to be safely controlled by the computer. The Electronics Department designed a small control panel with switches and status lights to be used by the test bed operator when starting, interrupting and continuing endurance tests – not requiring any direct interaction with a computer. They also designed a safety device which switched the cabinet to manual control if did not receive frequent 'I am still alive' signals from the computer.

It is sobering to think that everything we did on the 960 could probably be done today with a Raspberry Pi or Arduino – but such devices didn't exist in the 1970's!

TRANSITION TO BL SYSTEMS/ISTEL

By the mid to late 1970's, the effects of becoming part of BLMC were being more strongly felt, and various reorganisations and rationalisations were taking place. Of the greatest significance to Rover Technical Computing – and effectively marking the end of it as we knew it – was the arrival of Malcolm Silvester who had been brought in from Rolls Royce to head up all Engineering Systems within BL Cars. We found ourselves no longer a part of Rover Engineering but rather a part of the BL Cars Systems Department, albeit one which faced Engineering. This new department included many Cowley-based staff involved in systems work relating to Body Engineering, and a number of Solihull and Longbridge-based Systems Development Managers who had previously supported Engineering from the outside. Many of my Technical Computing colleagues were relocated to BroadOaks – an office block in central Solihull known as the 'Chocolate Box' because of its tinted windows.

At about the same time, John Leighfield was brought in to be the Supremo of all systems operations – both technical and commercial – within BL Cars. He was known to favour moving all systems activity into a subsidiary company which could serve not only BL Cars but also take on external customers, and he had the ear of chairman Michael Edwardes regarding this enterprise. This came to fruition in 1979 when the BL Board approved the establishment of BL Systems Ltd as a wholly-owned subsidiary.

In 1980, most BL Systems staff moved to offices away from the BL manufacturing sites. Most of the staff from Rover Technical Computing – including those who had temporarily been at BroadOaks – moved to Coventry Point, a high-rise office block in the centre of Coventry, where we both continued to support BL Cars remotely and supplied systems such as Catia CAD software, and the IBM RISC workstations on which to run it, to external customers. Some teams developed software to run on DEC PDP/11 and VAX computers both for BL Cars clients and external customers. I personally concentrated on being an Automation Consultant – assisting clients to select hardware and software for automated test systems, and writing interface specifications so that mechanical equipment could be computer-controlled.