# Pegasus Personified - Simulation of an Historic Computer

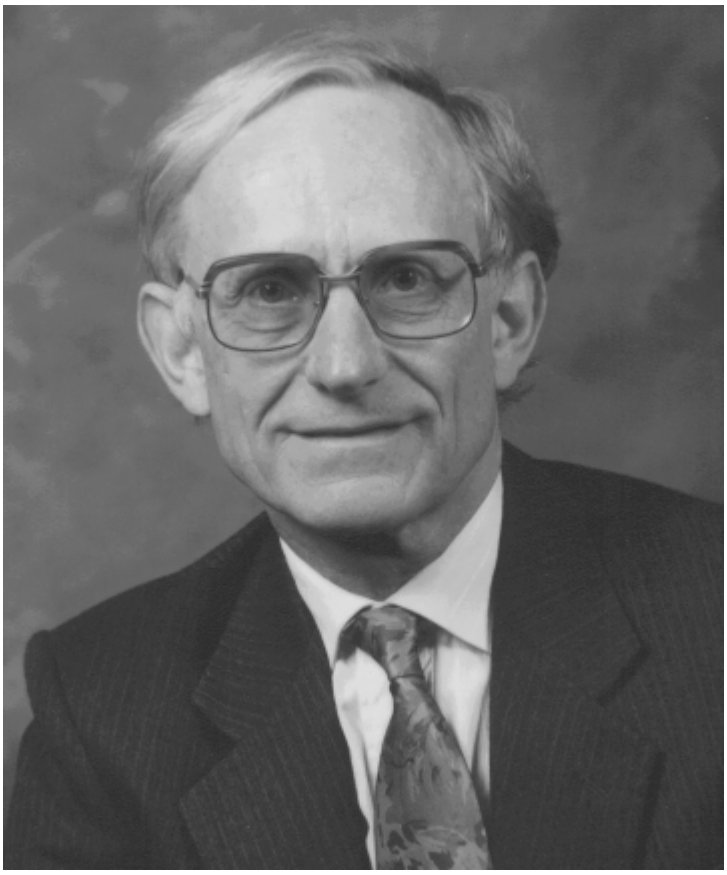
Christopher P. Burton, The Computer Conservation Society

Wern Ddu Fach, Llansilin, Oswestry, Shropshire SY10 9BN, UK.

*Tel:* +44 (0)1691 791 274 *Email:* cpb@envex.plus.com

*Web and email addresses updated – 6 Jan 2020*

## Author's biography



Christopher P. Burton has a BSc in Electrical Engineering from the University of Birmingham and an Honorary MSc from the University of Manchester. He is a Chartered Engineer, being a Fellow of the Institution of Electrical Engineers (IEE) and of the British Computer Society (BCS). He worked on computer hardware, software and systems developments in Ferranti Ltd and then ICT and ICL from 1957 until 1989, nearly all based in the Manchester area. He is a founder member of the Computer Conservation Society and led the team responsible for building a replica of the Manchester Small-Scale Experimental Machine. Other roles in the CCS include chairmanship of the Elliott 401 Working Party and past chairman of the Pegasus Working Party. For replicating the Manchester SSEM he was awarded an honorary degree by the University of Manchester, the first Lovelace Gold Medal by the BCS and a Chairman's Gold Award for Excellence by ICL. He lives in Shropshire, and spends a lot of time indulging in practical computer history matters.

**Abstract**

The vintage British computer Pegasus is briefly described, together with the work done to preserve examples in Science Museums. The development and use of a simulator program named PegEm which runs on a personal computer is then described. PegEm is considered to capture the "persona" of Pegasus.

## The Ferranti Pegasus

Readers of the Annals will by now be familiar with the fact that the Ferranti company based in the Manchester area in England was one of the pioneering firms in the world to get involved with the design and manufacture of computers[1]. Indeed the first computer in the world to be delivered commercially, in February 1951, was a Ferranti Mark 1. By 1950 the Ferranti company had sixty years of experience in innovative electrical and then electronic engineering, and its one-time slogan "First into the Future" was well-chosen. It had an early and continuing relationship with computer development in the University of Manchester. However it soon needed to set up a development and sales operation in the London area - many intellectual people attracted to the new business did not want to work in faraway Manchester. In Ferranti's London Computer Laboratory, in elegant Portland Place in London, a team led by Bill Elliott designed from scratch what became the Ferranti Pegasus computer[2]. Not quite from scratch - the concepts, technology and many of the team in fact emerged from Elliott Brothers Ltd. and its pioneering 401 computer exhibited in April 1953.

The engineering of Pegasus was outstanding[3]. It used a plug-in package technology, with each package containing two or three valves (vacuum tubes) and associated circuitry. Immediate access storage used acoustic delay lines mounted on the same type of package. Main storage was a magnetic drum, and input and output was via 5-track punched paper tape. Later, punched cards, magnetic tape and other peripheral devices could be added to Pegasus.

The objectives of the design team included that the machine should be highly reliable, easy to use and economical to manufacture. The plug-in technology contributed to the first and last. The functional design - the architecture - contributed to the second. The instruction set was particularly clean and consistent, with no unexpected exceptions. It was cited by Bell and Newell[4] as having, "….the nicest ISP processor structure…".

In many ways, Pegasus was the first “user friendly” computer. Unanimously, the hundreds of programmers and users of the machine have commented on the ease of programming and operation. The reader must remember that in the mid-1950s, a programmer wrote his program in machine code, or at best in the most primitive form of assembler. He typically took his program to the computer and debugged it personally, hands-on. So he needed an intimate knowledge of how the machine worked and how to operate the numerous control switches and display lamps. A fundamental part of Pegasus was what we might now call a primitive Operating System, a set of routines called Initial Orders which was stored permanently in a write-protected area of the drum. Pressing the “Start” key caused the Initial Orders to be executed, and they gave the programmer facilities for inputting programs and data, for debugging, for assembling large program systems from sub-sections and libraries, and so on. For routine maintenance of the machine, a large collection of Engineer’s Test programs was also held in a protected part of the drum and these could be easily run at any time to systematically exercise all parts of the machine so that no gate was left untested.

Unfortunately, only about forty Pegasus systems were sold, between 1956 and 1962. Manufacturing costs turned out to be greater than planned, possibly because the excellent engineering such as the “Rolls Royce” quality of the cabinet work was out of tune with mass production methods. In the same period, where Pegasus was delivered to 40 customers, IBM had delivered 1800 of the technically inferior 650 system. Figure 1 shows an early Pegasus computer in use at Vickers-Armstrong Aircraft Ltd.

**Pegasus and the Science Museums**

Pegasus is quite rightly regarded as a very important type example of early British computers. Originals are held by both the National Science Museum in London and by the Museum of Science and Industry in Manchester.

The London Science Museum Pegasus is serial number 25, and had a chequered career having been re-located at least eight times in its life, including a period in Stockholm, Sweden. It was acquired by the museum from University College London in 1983 and due to shortage of space at the museum was initially displayed at ICL’s works in Manchester. The author had his first contact with the machine there when he was allowed to maintain it in working order in his own time. After a couple of years, pressure of space

in ICL forced a move of the machine back to storage in London. Subsequently, with the formation of the Computer Conservation Society (CCS), a Pegasus Working Party was established to re-commission and demonstrate the machine at the museum. After a number of years of movement but continued maintenance, it was put on prominent display in the Computer Gallery in 2000 where it continues to be maintained and demonstrated. It is a tribute to the original engineering that it has survived this constant stripping down, moving, and re-erecting, albeit with some scratches on the highly polished paint-work!

The Pegasus held in the museum in Manchester is earlier, possibly serial number 6. It is on display and has been conserved, i.e. treated by the CCS to reduce deterioration. The museum policy is to preserve it in this original form and to not interfere with it further by restoring to working order, because the London Pegasus has that capability.

The early months of the Pegasus Working Party were very exciting because this was effectively the trial project of co-operation between the museum and external volunteer enthusiastic experts. The latter in particular had a lot to learn about curatorial practices. One issue raised early was the dilemma that the machine was being re-commissioned to working order, yet from a curatorial point of view one did not want all-comers to lay hands on it and use it. At about this time the question of offering a software model of the machine was discussed and the author felt that this was a suitable private project to tackle.

**Emulation vs simulation**

There is nothing new about emulating a target computer in software or hardware on some other host computer. There are numerous examples especially of early and obsolete microprocessor systems emulated on more modern platforms. Many emulators are focussed on a particular processing mechanism and work by sending and receiving data streams to and from the target model. Others embed the target's processing model within the host's environment so that the user is living with features of the host environment, for example the modern file system. That may not have existed on the original target. I will refer to such modelling systems as *emulators* of the target machine.

*Simulation* by contrast will be taken to mean that the user is not aware of the host platform, and the whole of the target environment is represented on the host. Of course, the degree to which this can be approached in practice is limited - it is rather hard to

simulate a seven-foot tall mainframe on a fifteen-inch tall personal computer screen. Real aircraft and other vehicle simulators get closer, but in the context of simulation of old computers such physical realism is not justified.

For our modelling of Pegasus, a simulator was desirable, so that a user could feel he was operating a Pegasus rather than just getting a Pegasus program executed.

A prior example is the simulator of the Cambridge EDSAC by Martin Campbell-Kelly[5]. This impressive program presents a view of a monitor cathode ray tube of EDSAC together with various control switches and dials, but no attempt is made to represent an actual view of the panels of EDSAC. Indeed part of the host screen also shows controls and abstractions relevant to execution of an EDSAC program but not representing anything physical in the original EDSAC. The simulator is aimed at the EDSAC program writer and Campbell-Kelly gives good reasons why his simulator has evolved the way it has. At the time the Pegasus work was started, there were separate discussions in the CCS about issues of simulating old computers but the present author was not a participant and had not seen the EDSAC simulator. So any vestigial similarity is purely coincidental!

**The simulator program PegEm**

The Pegasus simulator, PegEm[6], was written before multimedia personal computers were commonplace, and Windows 3.1 was state of the art. The author aimed at potential host platforms with MSDOS running on a ‘386, and with a 16-colour 640x480 (i.e. VGA) screen. This gave the best chance of widespread availability of the simulator and corresponded with the capability of his own PC at the time. Little did he realise how rapidly host capabilities would progress, and how modest a capability he accepted! Nevertheless what was achievable was very satisfactory and welcomed by colleagues in the Pegasus Working Party and outside.

A simple strategy for writing PegEm was adopted. There would be a synthesised representation on the screen of the control panels of Pegasus, together with images of the punched-tape equipment. There would be an image of a metal filing cabinet with drawers containing rolls of paper tape, and a metal waste bin. The only other thing on the screen would be a realistic image of the user’s hand (well, the author’s hand actually) analogous to the hand cursor in a Windows system. The real user could move the hand about to manipulate the objects on the screen. All this represents the Man-Machine Interface

(MMI) of Pegasus. Figure 2 is a bird's eye view of the "cockpit" of a real Pegasus, and Figure 3 shows the representation used by the simulator, which is realistically coloured.

Behind the MMI of the simulator is a module which models the logic of Pegasus. This is quite detailed and is close to gate-level in some areas and high-level in others. The aim was to roughly model the gross timing behaviour of the original machine. For example, shifting a number in an accumulator in Pegasus is done by a setting-up phase, a multi-beat phase where the number is shifted one place at a time per machine beat, and a finish phase. Each of those steps is modelled in the logic module.

A third module permits off-line interaction with the simulator program to set up the assumed environment for running the simulator. About twenty things can be set to suit the taste of the current user. An example is the choice that output from the simulated tape punch is sent to the simulated teleprinter (realistic, as in Pegasus) or else is sent to the printer of the PC (unrealistic, but useful). A more trivial example is selection of the colour of the shirt cuff of the "user's hand".

All control of the simulated Pegasus is done by operating the simulated controls by using the hand cursor. Control of the hand in the first version of PegEm was done using the PC keyboard, though the current version uses the mouse. The arrow keys moved the hand about in such a way that it was constrained to always land on something sensible such as a switch. For example the left and right arrow keys would move the index finger on the hand from switch to switch in a row of switches, and the up and down arrows would press the switch up or down. Larger scale movements used the Page Up and Page Down keys. It sounds clumsy, but it rapidly became extremely natural and easy to manipulate the screen objects.

It took a few weeks to develop the MMI part of the simulator, writing in Turbo Pascal and making good use of its graphics library. The simulator was then shown to colleagues with just one Pegasus instruction modelled, an "ADD to accumulator" instruction . The instruction was set up on the simulated control switches, the "Manual" operation switch set, and at "Single Shot", or on "Run", the number counting up could be seen, in realistic binary, on the simulated cathode ray tube monitor. The demonstration was greeted with enthusiasm and encouragement to get on and model the rest of the logic. This took place over about nine months, using the very good documentation which exists for Pegasus. The accuracy of the logic model was due to the careful testing of the simulator by the late

Derek Milledge, one of the pioneers who helped to develop the Initial Orders and routine libraries for Pegasus. His detailed knowledge of the exact way each instruction should work with all combinations of weird operands was a key to getting the logic model right. Groups of instructions would be modelled using Pascal code, then Derek would take a copy of the simulator program as developed so far and check it for correctness. Note that his only view of the logic was as it behaved as seen through the MMI. He had no knowledge of the underlying Pascal coding.

Once most of the Pegasus instructions were coded, it was opportune to load the Initial Orders and the Engineer's Test Programs on to the simulated drum. There is a standard Pegasus bootstrapping procedure for this, copying the programs from binary paper tapes into the protected area of the drum. The procedure only needs to be performed once because the drum is non-volatile. The author owned a paper tape reader which he had interfaced to a serial port on his PC, and this enabled the program tapes to be read into the PC file system. From there they were bootstrapped into the simulated drum. In the simulator, operation of the appropriate switches would now run the Initial Orders, and more importantly, the Engineer's Test Programs. It was with great satisfaction that eventually the test programs gave the modelled Pegasus a clean bill of health! In the same manner, all the library paper tapes and numerous utility and application program tapes were copied into the file store for access from a simulated paper tape storage cabinet.

As paper tapes move through the input/output devices, they are animated realistically. A small image of a section of tape in a reader is shown adjacent to the reader. This is an authentic situation because it was often necessary for an operator to examine the punching on a tape during a debugging session for example. To increase realism the I/O devices were each made to click the PC speaker every time one of them dealt with a character, so that there is a suitable sound as tapes are read, punched and printed. Furthermore, the real Pegasus has a loudspeaker in the console which can be patched to any waveform in the computer. It is typically left connected to a particular waveform in the normalise instruction, and the analogous place in the logic module of the simulator was provided with code to click the PC speaker. Consequently, when the Pegasus music program is run on the simulator it produces the expected tunes.

Although no particular attempt was made to simulate correct timing, the similarity of the logic module to the actual hardware means that the various instruction times bear a reasonably correct ratio to each other. It turned out that on my moderately fast 386 PC the

simulator ran at near-correct real time. There is a simple slow-down facility for faster PCs.

**Tape editing**

For a user to do useful work with Pegasus he needs a facility to prepare and to interpret paper tapes - a tape editing set. Such equipment was provided by Ferranti, and typically comprised a desk with a keyboard teleprinter, a tape punch and a tape reader. These were all standard electro-mechanical telex devices, and have not been simulated in PegEm. The then leader of the Pegasus Working Party, the late John Cooper, wrote some suitable routines to translate standard personal computer text files into the form used by the simulator. These routines are used "offline" independently of PegEm. A program writing session may thus consist of coding up the program on coding sheets, typing the code using a text editor such as Notepad, converting the text file to the simulator format, testing the program using PegEm, and then either printing results on the local printer, or else translating the output tape file back to text for subsequent editing. This is a tedious process and a better integration of the activities is scheduled.

**Using the simulator**

The side panel illustrates a sequence of activities when running a program on the Pegasus simulator. The many former users of Pegasus who have tried using the simulator have commented that it does respond much like the real thing. It has been aptly said that it "captures the persona" of the original machine.

Although there is a certain amount of documentary material supplied with the simulator, including a tutorial on writing and running a Pegasus program, serious use of the system requires The Pegasus Programming Manual[7]. This 320 page hardback book written by George Felton was widely regarded in the UK as a model of how a user's handbook should be written, taking the reader through the principles of programming, the instruction set of Pegasus and how to exploit its power, to details of the Initial Orders and subroutine libraries. Regrettably, the book has not yet been transcribed to machine-readable form, and the very large page format (UK foolscap size) makes even photocopying rather troublesome.

**Future enhancements**

In a private project such as this, one is always tempted to go on improving and enhancing what exists. The main and urgent requirement is to improve the documentation as mentioned earlier. There are a few cosmetic improvements to the simulator envisaged, and a total re-write so that the simulator runs smoothly under Windows instead of in an MSDOS environment would make the system more easily accessible.

Longer term, the author would like to take the model further by embedding more hardware in it. He has inherited a real Pegasus control panel, and believes that the correct type of paper tape equipment is available. It would be a lot of fun to build a Pegasus using authentic I/O and control panels, and leaving all the functionality in software behind the scenes.

**Conclusions**

The original aim of PegEm has been achieved - a system is available for users to write and run Pegasus programs and study how the machine behaves at the user's level. It has been useful within the Pegasus Working Party on the one hand to carry out investigations and enhancements to the Engineer's Test Programs and also in developing or resurrecting various application and demonstration programs. It is envisaged that it will also be used to provide visitors to the Computer Gallery with an animated visualisation of the running machine, when the Pegasus is otherwise unattended.

It is striking that since PegEm was written there have been remarkable advances in developing Virtual Reality (VR) systems. The author would claim that PegEm is an approach to VR, to immerse the user in an environment where he really feels he is operating the target system. It raises the question whether, if the simulation is good enough, it is necessary to preserve original artefacts at all. But that is a discussion for another place[8].

**Acknowledgements**

The author is very much indebted to all those colleagues in the CCS who have encouraged this development. In particular, the late Derek Milledge greatly facilitated progress by patiently and meticulously checking the accuracy of the simulated machine logic and by proposing many valuable enhancements to make operation more realistic.

**References**


1. Geoffrey Tweedale, "*A Manchester Computer Pioneer: Ferranti in Retrospect"*, IEEE Annals of the History of Computing, Vol 15, No. 3, 1993, pp 37-43.

2. Simon H. Lavington, "*The Pegasus Story - A history of a vintage British computer*", The Science Museum, London, 2000.

3. William S. Elliott, Charles E. Owen, C.Hugh Devonald and BrianG. Maudsley, "*The Design Philosophy of Pegasus, A Quantity Production Computer*", Proc.IEE., Vol. 103, Part B, Supp. 1-3. 1956, pp. 188-196.

4. Gordon C. Bell and Alan Newell, *Computer Structures: Readings and Examples*, McGraw-Hill, New York, 1971, pp170.

5. Martin Campbell-Kelly, "*Past into Present: The EDSAC Simulator*", in R. Rojas and U. Hashagen, eds., *The First Computers - History and Architecture*, MIT Press, Cambridge, Mass. and London, 2000.

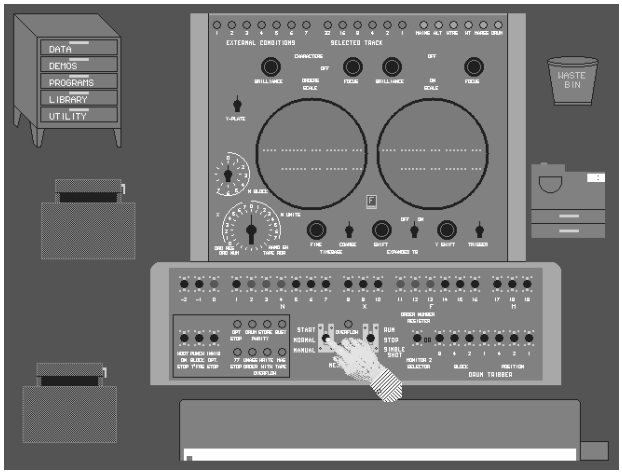
6. The simulator PegEm version 1 can be downloaded from the Computer Conservation Society's web site at
   www.computerconservationsociety.org and following the "Software" menu item
   Version 2 which uses the mouse is easier to use but contains some minor bugs.

7. George E. Felton, "*The Pegasus Programming Manual"*, Ferranti Ltd., London, 1962.

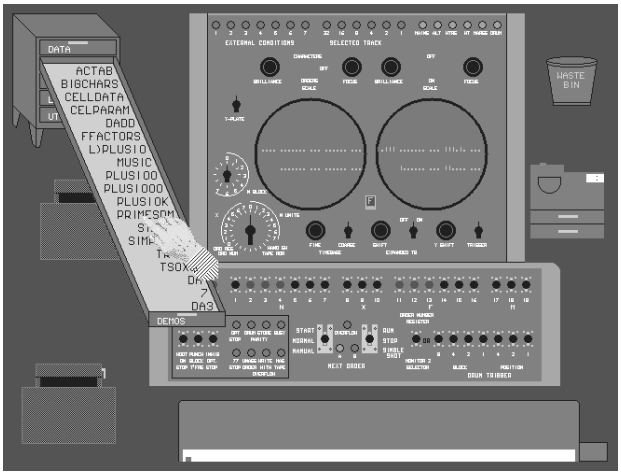
8. Doron Swade, "*Virtual Objects - Threat or Salvation?*", in S. Lindquist, M. Hedin and U. Larsson, eds., *Museums of Modern Science*, Nobel Symposium 112, Science History Publications, USA, 2000, pp. 139-147.

Side Panel showing a sequence of operation of the Pegasus Simulator.

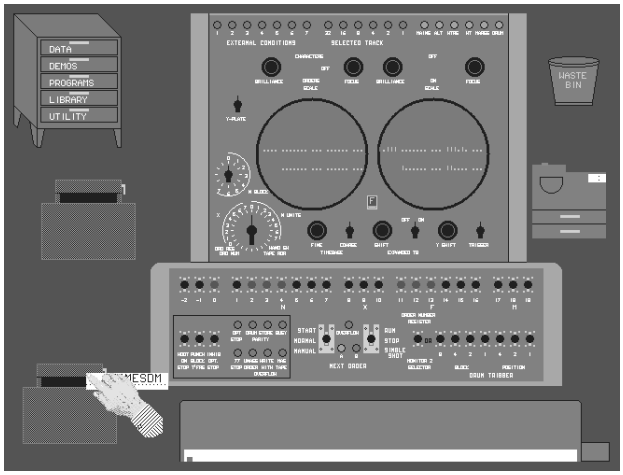
# PegEm in action



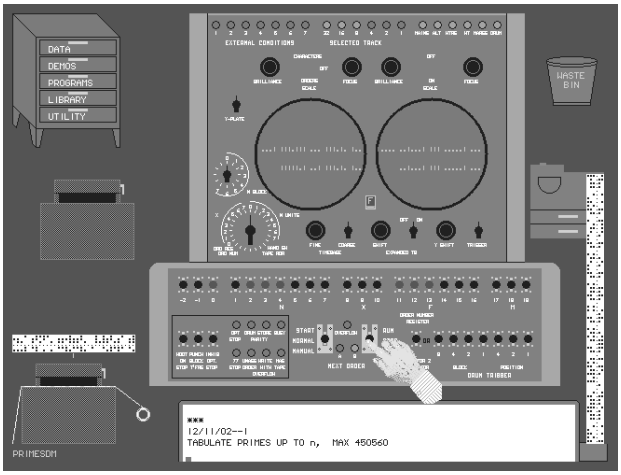

Starting situation, with the hand hovering over the Start and Run switches. A paper tape storage cabinet and two tape readers on the left, waste bin and output tape punch on the right, and the offline teleprinter nearest to the reader.



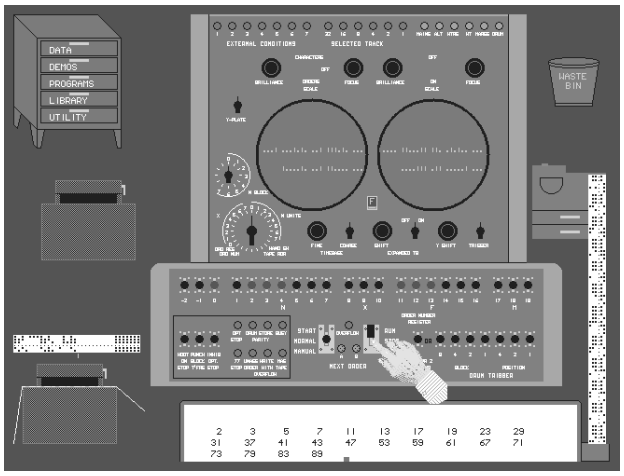

Tape cabinet drawer opened to extract a program tape. A program to generate prime numbers has been chosen.



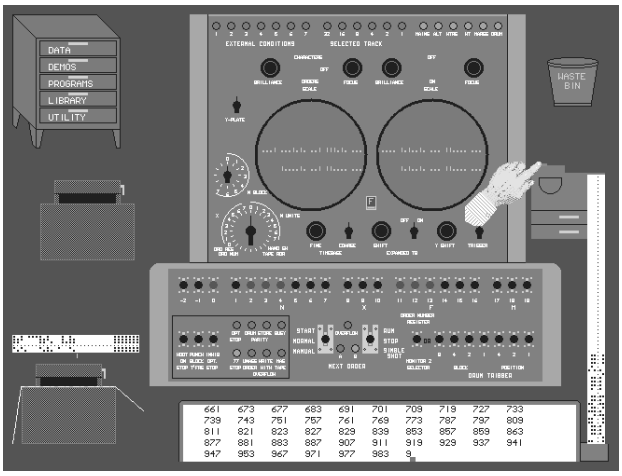

Hand carries the tape down to the tape reader and is ready to set it in the reader.



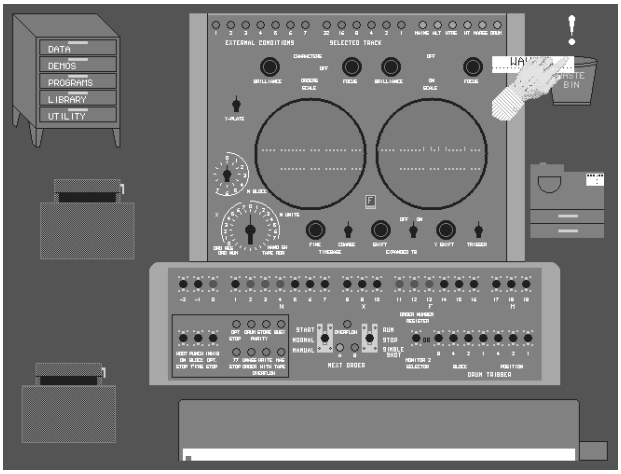

Start and Run switches operated, Initial Orders copies header from tape to punch. Tape from punch then moves to teleprinter for printing.




Computer running and calculating prime numbers. The numbers are punched as they are calculated, and the output tape is then printed.




Program has finished calculating, the "Stop" lamp is lit. Last few numbers on output tape are printed by pressing the runout button on the punch, thus providing enough slack in the tape for it to reach the teleprinter. This behaviour is authentic.




Output tape no longer needed so it is torn off and deposited in the waste bin.

Figure 1.

A Pegasus in use for aircraft design in 1957.

*Courtesy of ICL and Judith Milledge*

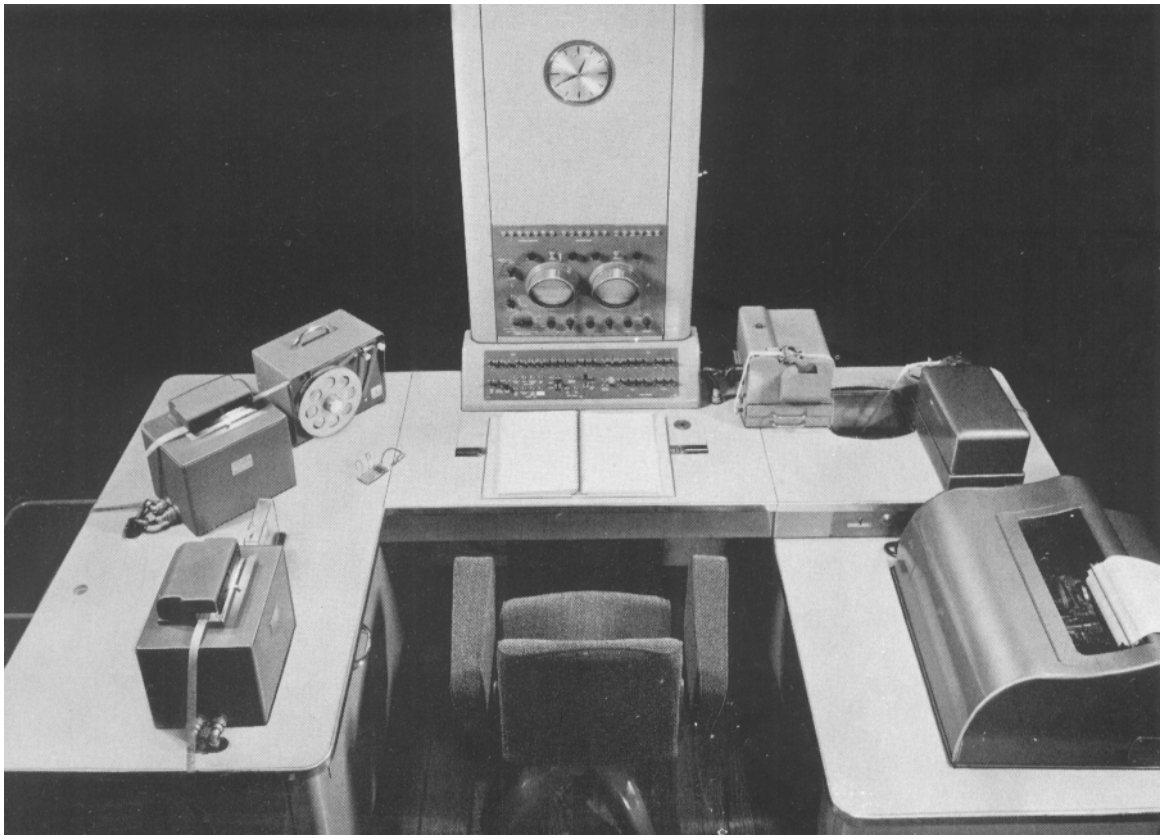


Figure 2.

The control desk of Pegasus. On the left are two photo-electric tape readers operating at 200 cps, and an automatic tape spooler for winding up long paper tapes. On the right is the output paper tape punch operating at 30 cps. Tape from the punch is led across to an electro-mechanical tape reader directly connected to the teleprinter, which operates at 7 cps. The tape loop between punch and reader acted as an output buffer. In the centre are the monitoring cathode ray tubes, numerous indicator neon lights and the operating switches.

*Courtesy of ICL*

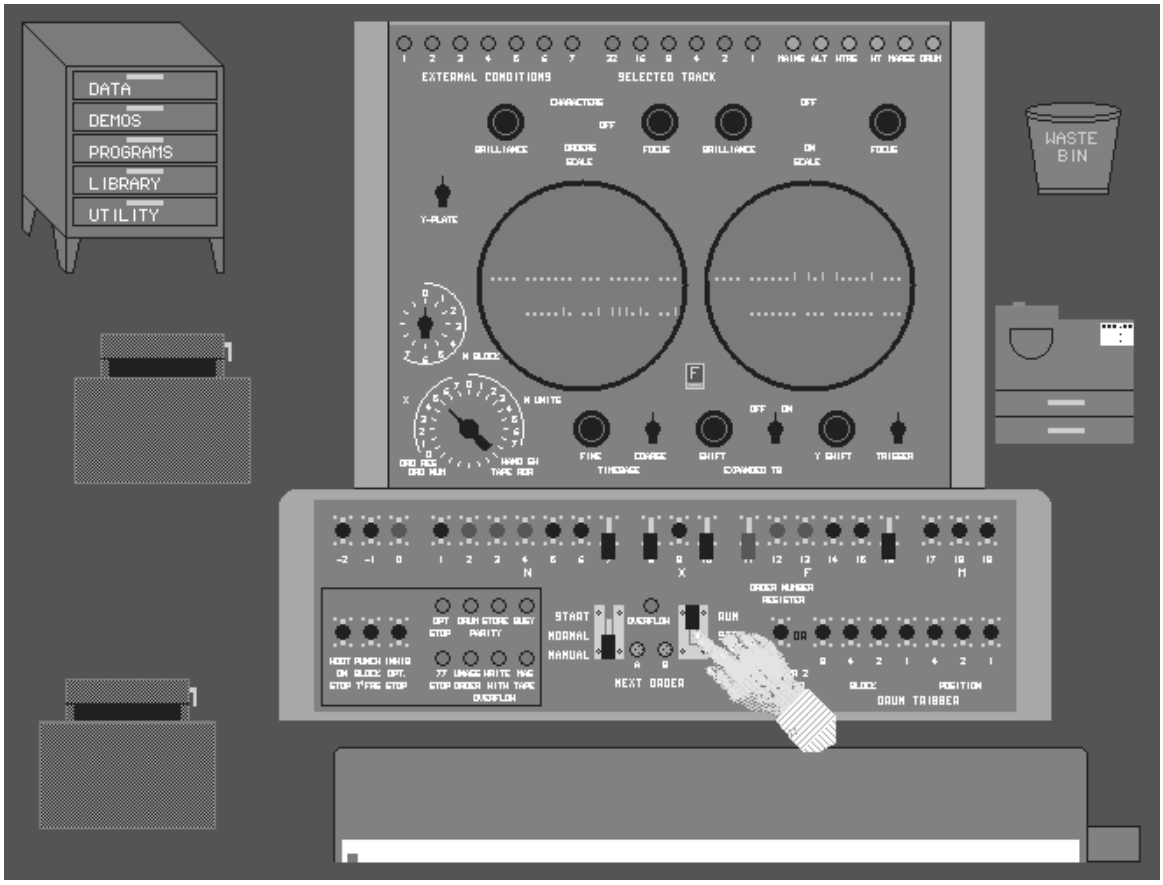



Figure 3.

The screen view of Pegem. In this view, a manual instruction to add 1 to accumulator 5 has been set up on the hand-switches, and the machine set to “Run” at “Manual”. The instruction repeats continually, and the incrementing binary pattern can be seen on the left-hand monitor, which has been selected to show accumulator 5.