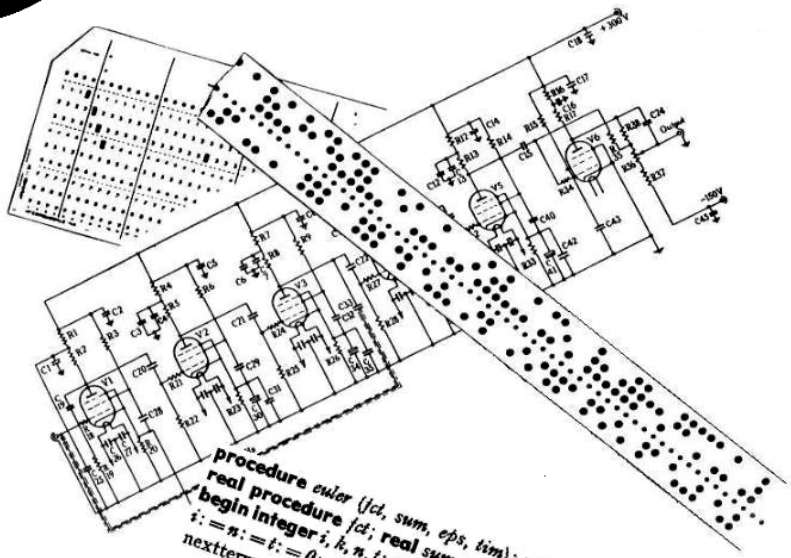
Issue Number 63 **Autumn 2013**

RESURRECTION

The Bulletin of the Computer Conservation Society



```
procedure euler (fct, sum, eps, tim); value eps, tim; integer tim;
real procedure fct; real sum, eps;
begin integer i, k, n, t; array m[0:15]; real mn, mp, ds;
    i:=n:=t:=0; m[0]:=fct(0); sum:=m[0]/2;
nextterm: i:=i+1; mn:=fct(i);
    for k:=0 step 1 until n do
        begin mp:=(mn+m[k])/2; m[k]:=mn; mn:=mp end means;
    if (abs(mn)<abs(m[n]))∧(n<15) then
        begin ds:=mn/2; n:=n+1; m[n]:=mn end accept
    else ds:=mn;
    sum:=sum+ds;
    if abs(ds)<eps then t:=t+1 else t:=0;
    if t<tim then go to nextterm
end euler
```

Computer Conservation Society

Aims and objectives

The Computer Conservation Society (CCS) is a co-operative venture between BCS, The Chartered Institute for IT, the Science Museum of London and the Museum of Science and Industry (MOSI) in Manchester.

The CCS was constituted in September 1989 as a Specialist Group of the British Computer Society. It is thus covered by the Royal Charter and charitable status of BCS.

The aims of the CCS are:

- ◊ To promote the conservation of historic computers and to identify existing computers which may need to be archived in the future,
- ◊ To develop awareness of the importance of historic computers,
- ◊ To develop expertise in the conservation and restoration of historic computers,
- ◊ To represent the interests of Computer Conservation Society members with other bodies,
- ◊ To promote the study of historic computers, their use and the history of the computer industry,
- ◊ To publish information of relevance to these objectives for the information of Computer Conservation Society members and the wider public.

Membership is open to anyone interested in computer conservation and the history of computing.

The CCS is funded and supported by voluntary subscriptions from members, a grant from BCS, fees from corporate membership, donations and by the free use of the facilities of our founding museums. Some charges may be made for publications and attendance at seminars and conferences.

There are a number of active projects on specific computer restorations and early computer technologies and software. Younger people are especially encouraged to take part in order to achieve skills transfer.

The CCS also enjoys a close relationship with the National Museum of Computing.

Resurrection

The Bulletin of the Computer Conservation Society

ISSN 0958-7403

Number 63
Autumn 2013
Contents

Society Activity

EDSAC Replica — *Andrew Herbert*

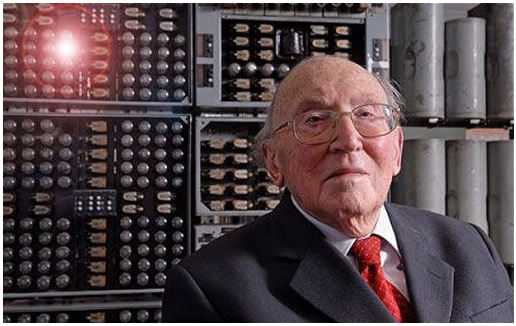
John Pratt has completed chassis 05 (Tank Decode Logic, part of the store addressing system).

Alex Passmore has completed chassis 14 (Rack Distributor) which transmits the selected address lines to the storage regeneration system.

Commercial manufacture of the one highly replicated chassis (type 01, Storage Regeneration) is under investigation. We need 42 of these (they make up about one third of the machine) and it would be a big drain on volunteer effort to make these ourselves.

A new volunteer, James Beer has joined the team and is investigating the "computer control" part of EDSAC: instruction fetch, decode and execute.

On 26th June the EDSAC Replica Project team put on an event at TNMoC to celebrate the 100th anniversary of Sir Maurice Wilkes, who led the construction of the original EDSAC in 1947-9.



Maurice Wilkes with Witch in 2009

The highlights of the event were the display of a rack containing seven working "new" EDSAC chassis and a talk by Wilkes' son Anthony on growing up in the era of early computers.

The chassis on display comprised:

- the clock pulse generator, which provides the central clock to the rest of the machine
- one digit pulse generator, which distributes pulses for individual digits within an EDSAC word
- one half adder, of which a total of four will be needed, two for the arithmetic unit, one for main control (instruction sequencing) and one for store access (major cycle counting).
- four further chassis making up a 4 bit store address decoder.

Anthony Wilkes described how his father would often take him and his sisters into the Mathematical Laboratory on a Saturday morning and let them lose to play in the workshops while he worked on EDSAC and other laboratory business. Anthony showed two souvenirs in his possession: his father's note book of suitable jokes for use in public speeches and a paper tape punched by EDSAC congratulating Wilkes on his election to a Royal Society Fellowship.

Anthony Wilkes with the EDSAC Replica chassis

The project continues to make good progress on other fronts. Peter Linington continues to progress nickel delay lines stores. Peter Tomlinson and Ian Parsons are working on the development of a semiconductor delay line emulator that can be used during commissioning. Bill Purvis has converted a Creed teleprinter to EDSAC code and demonstrated it working via a Raspberry Pi. His attentions are now turning to the paper tape reader, picking up from the research done earlier by John Deane. TNMoC are engaged in preparatory work to make space for EDSAC in the museum and the project team are accordingly working to firm up infrastructure needs, in particular power supply, monitoring and distribution. Reconciling 1940's practice with modern attitudes to electrical safety will inevitably require some compromises against authenticity, but we will do our best to disguise these as much as possible.

In financial terms the project is in sight of having spent the proceeds of the initial fundraising and the trustees are actively seeking further donors to secure the remaining funds required.

Harwell Dekatron — *Delwyn Holroyd*

Operation has been largely problem free with the exception of a trigger tube fault which necessitated replacement. At the beginning of this month we welcomed a large group from the University of Wolverhampton, including the Vice Chancellor, senior academics and students past and present. They listened to a talk about the history of the machine and the restoration process, culminating in a demonstration and a lively Q&A session. The event was covered by the Wolverhampton Express and Star newspaper, continuing a long list of articles it has published about the WITCH going back to the 1950s.

ICT 1301 — *Rod Brown*

The 1301 project volunteers still await arrival of the system at TNMoC. In the meanwhile members of the project continue to write software to decode the captured images of the magnetic tapes taken to date. This software needs to be viable before effort can be put into rebuilding a tape transport to capture further tapes.

The project's current supporter base continues to grow via the website and the launch of further links to the site are in progress.

Requested presentations on the London University, Senate House years are nearing completion.

Hursley Museum

One marked gap in our collection until June this year has been the PC-AT. At one time you couldn't move for these heavy beasts in IBM, now they seem to be as rare as hens' teeth.

Recently we spotted a PC-AT on eBay, and the three curators agreed we'd stretch to £30 and try our luck. The machine was in Essex, collect only, so we contacted the seller and got his agreement that if we won he would hang onto it until we could arrange collection. It sold for over £80. Mysteriously it went back onto eBay a couple of weeks later, this time the system unit and monitor separately. We contacted the seller again, to discover that some nice Spanish person had outbid everyone but then neither paid nor communicated.

This time the system unit sold for somewhat over £30. The seller got in touch the very next day, were we still interested, the Spaniard again! £30 later it was ours. Now to arrange collection. Our private pilot (Peter) decided a trip to Essex was as good as a trip to any other airfield, and there was a private strip close to the seller. He agreed to meet Peter there, and so the PC-AT was air-freighted from Thurrock via Popham to Hursley! It turns out that the monitor was also sold to a non-payer, so we got that for another fiver.

ICL 2966 — *Delwyn Holroyd*

The card reader mechanism has been stripped down and rebuilt, and the rollers treated with a cleaner to restore grip. It is now reliably feeding cards at 1000 card/minute and I have already read quite a few decks via the PC interface box.

TNMoC — *David Hartley & Stephen Fleming*

A programme of relatively minor changes to the galleries has been started to free space for the EDSAC rebuild, to enable installation of a display of the current NATS air traffic control system alongside its predecessor IRIS which came out of West Drayton, and to accommodate a temporary display being developed with and supported by Google.

The Board of Trustees are reviewing their membership with a view to strengthening the governance of the organisation.

A new software gallery has been opened by Sir Charles Dunstone, Chairman of Carphone Warehouse and TalkTalk. The new gallery, sponsored by Insightsoftware.com, traces the development of computing software from its beginnings on huge computers to its presence in everyday household items.

The new software gallery, laid out in four quadrants, includes:

a wall-sized programming language timeline
an "exploded" PC showing its internal components
a robotics display
a computer language database, already containing 2,000 entries to which visitors can add
an early, single-purpose accounting software machine, the Burroughs L5000
a display demonstrating the pervasiveness of software in the home
a special programming challenge for visitors and other hands-on exhibits.

The gallery has been created by an assembly of TNMoC volunteers led by Jill Clarke and Bob Jones.

Software — *David Holdsworth*

We are making great strides with our LEO III resurrection. We have execution of the Intercode translator as far as reading the first line from the paper tape reader. The team's morale is holding up well, but I am keen to show more execution, as in the past it has proven to be a great motivator.

We are finding documentation inconsistencies and incompletenesses. I doubt other old systems have complete documentation either – perhaps IBM 360 is an exception.

LEO III is a stunningly idiosyncratic machine by today's standards, and pretty unusual by the standards of the 1960s. I am finding this effort to resurrect an unfamiliar machine to be quite revealing of the kind of pitfalls historians of our subject might meet in the future. As I learn more of the programming system, I am amazed that they ever got any software to work at all, but then that was a different world, one which we are steadily exploring.

It is also showing how impossible it would be to get old software to work without the aid of veteran programmers from the original machine. I suspected this to be the case, but took on this resurrection of a machine of which I had no previous non-trivial knowledge in order to test this hypothesis.

A couple of the LEO veterans have expressed surprise at how the project is going.

I hope to maintain enough momentum to record an outsider's view of the features of the LEO III computer, upon which all UK telephone bills were produced for many years.

We have a webserver upon which most of our developing material lives. There are links to it on the s/w preservation website at: *http://sw.ccs.bcs.org/leo*. CCS members are welcome to browse there.

The team's principal members are John Daines, Ray Smith, Geoff Cooper, Tony Jackson, Chuck Knowles, Ken Kemp, Dave Jones. We have one member in Bermuda, and another in Australia. I suspect that the rest live in the UK.

Meanwhile, at the other end of the country, Dik Leatherdale is fitfully progressing his emulator for Atlas 1 – still incomplete after nine years. Recently, source listings for the Brooker-Morris Compiler Compiler have come to light. The resuscitation of this significant piece of software has become the task's next objective. Iain MacCallum, one of the original CC team, is joining in, supported by Simon Lavington together with help from Manchester University. As an initial step, the concept of the *Supervisor Tape* which held all the compilers for Atlas, has been introduced to the emulator and a very crude compiler for loading store images expressed in octal (as per the original machine) has been written.

Manchester Baby Replica (SSEM) — *Chris Burton*

The machine is in quite good working order. There are two issues to be addressed:

a) The main store CRT is rather finicky and needs more than occasional tweaking to run reliably, i.e. for 2 or 3 hours. One team of volunteers is good at tweaking and will look at it when time allows. Meantime the dummy store is used. The Control CRT and Accumulator CRT are very reliable, though we think the 12" Accumulator is susceptible to interference – it is less well shielded than the other CRTs.

b) For much of the spring the subtractor has been unreliable. This is now thought to be due to drift in resistor values. By shunting one of the critical resistors the unit has been good and has been working without error. There will be a drains-up on resistor values over the next few weeks.

The site looks extremely tidy and safe due to work by MOSI to insulate exposed parts.

The training of volunteers to the hierarchy of levels is working well, with volunteers available at all levels. So far this year six new volunteers have joined and another two are in the pipeline. This enables demonstrations to be done two or three days a week plus some weekends.

Hartree Differential Analyser — *Charles Lindsey*

There have been recent troubles with the half-nut slipping on the leadscrew which drives one axis of the plotter, and which it is somewhat awkward to remove (we wasted much time trying to remove it the wrong way).

Since our plotter is in fact a modified input table (the original plotter being in London) we swapped the whole assembly for the one on our input table. This cured that problem, but the new assembly had a bent shaft which caused a nasty wobble to show up on the plots. So we are now engaged in reverting to the original assembly, but with the half-nut from the other one (which we now know how to remove properly).

We have recently taken delivery of an inverter to enable us to control the speed and direction of the main drive motor (which was a DC motor on the original machine). This will enable more realistic demonstrations of the machine, especially when we can gain access to the collection of gears etc. still imprisoned in off-site storage.

Elliott 803/903 — *Terry Froggatt*

On the hardware side, there has not been a lot to report recently. The 803 may be developing a paper tape station power supply fault, as it turns itself off occasionally, but this is not yet frequent enough to be a bother or to locate. The 903 paper tape reader light beam went out of alignment, and was readjusted, but the reader still only works intermittently. For now it has been replaced by an alternative reader. And after having the 903 on display for two years, with the racks merely resting inside the desk, we have finally got round to bolting them in.

On the software side, Andrew Herbert, who described the software available to 903 users in *Resurrection 61*, has also generously arranged for the most useful grey ring binders of 903 documentation to be scanned. The complete 2.5Gbyte archive of user manuals and paper tape images, from several sources, along with his own demonstration scripts and simulator, is all gathered up in a Dropbox, accessible from *tinyurl.com/903doc*.

Analytical Engine — *Doron Swade*

Work continues on the technical archive. This is primarily assessing the completeness of the drawings in respect of their fitness to serve as a specification for construction whether simulated or real. An academic research proposal has been submitted to investigate Babbage's formal description of the Engines and this has cleared the first phase of acceptance. With the passing in 2002 of Allan Bromley, who was the first to decode the designs in any detail, a great deal of unpublished and undocumented knowledge was lost. Given the likely length of the project, we have built in post-doc research student posts from the start by way of succession planning. As a separate initiative we are seeking sponsorship for a post-doc researcher to accelerate the initial assessments of the adequacy of the drawings to serve as an effective specification. In summary, a period of background undergrowth clearance, foundation-laying, and throat clearing.

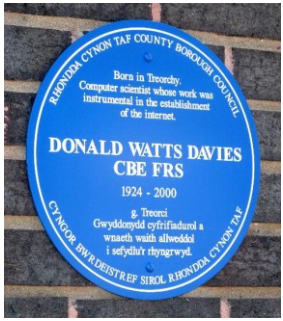
North West Group contact details

Chairman Tom Hinchliffe: Tel: 01663 765040.
Email: *tah25@btinternet.com*
Secretary Gordon Adshead Tel: 01625 549770.
Email: *gordon@adshead.com*

News Round-Up

In June the Queen Elizabeth Prize for Engineering was awarded to Louis Pouzin, Robert Kahn, Vint Cerf, Tim Berners Lee and Marc Andreessen for their pioneering work on the development of the Internet and the Web. Eyebrows were raised when it was noticed that the ground-breaking work of Donald Davies and his team at the National Physical Laboratory had apparently been overlooked.

A week or so later, amends were made when Vint Cerf hosted a large gathering at the London headquarters of Google at which Roger Scantlebury and Peter Wilkinson (NPL) together with Peter Kirstein (University College London) described the early development of the Internet at some length. Tilly Blyth gave a short presentation on the new *Information Age* (née *The Making of Modern Communications*) gallery at the Science Museum and David Harley held forth on the work of TNMoC. Google-produced films on EDSAC and LEO were shown too. Altogether a great evening.



The icing on the cake came at the end of July when a blue plaque for Donald Davies was unveiled in Treorchy where he was born.

101010101

In July the last public telegraph service was closed down in India. A technology which has served the world for 169 years has finally passed away. Abolished in the UK as long ago as 1982 it survived as late as 2006 in the US. Apparently in Brazil it is a requirement to resign from your job by telegram, though how this can be achieved is another matter.

101010101

In June alarm bells rang as rumours spread that NMSI (the parent body of the Science Museum and MOSI) might have to close one of its four northern branches if the government spending review was too severe. In the event it wasn't and it later emerged that Bradford's National Media Museum was the one in the firing line anyway. Non-story then.

101010101

Another year, another Apple I is auctioned (see also *Resurrections 53 & 59*). This time it realised £450K. If you have one in your loft, hang onto it for now. The way things are going you could be a millionaire in a couple of years.

Word arrives from the unlikely location of Stoke on Trent's Chatterley-Whitfield Mining Museum of an IBM 1401 (with 1403 printer and 1402 card reader) which seems to have been abandoned by the National Coal Board several decades ago. The 1401 was the first computer to gain widespread acceptance in the world of data processing as opposed to that of scientific computing. Over 10,000 were built between 1959 and 1971. Very few have survived.

Anybody with experience of these seminal machines is asked to contact John Adams, *john@adams.me.uk* who is keen to see whether it can be restored.

101010101

The Centre for Computing History has relocated from Haverhill to Cambridge and is now open five days a week. Readers are warned that, at the time of writing, their website (*www.computinghistory.org.uk*) is not fully up to date in respect of its location.

101010101

We hear of a nuclear industry installation in Canada which has declared an intention to continue using its DEC PDP-11 until 2050, some 80 years after the PDP-11 was first announced. If that sounds optimistic, it's worth reflecting that we're already more than half way there. Be that as it may, there are already staffing problems. PDP-11 Assembler skills are apparently hard to come by.

101010101

Nothing to do with the history of computers, but Jack Schofield's recent article about the musculo-skeletal problems associated with the use of laptops, tablets and mobile 'phones at *tinyurl.com/lapback* makes for alarming reading. Sometimes progress comes at a price, it seems.

101010101

In July the UK Government indicated that it would not oppose the *Alan Turing (Statutory Pardon) Bill* introduced into the House of Lords by Lord Sharkey who was taught mathematics at university by Turing's close friend Robin Gandy.

101010101

We have recently heard of the *Musée Bolo*, a Swiss computer museum with a large collection of computers, many of which are on public display at the *École Polytechnique Fédérale de Lausanne (EPFL)*. Its website is at *www.bolo.ch*.

101010101

A memorial to Bill Tutte the mathematician on whose work Collossus was based is proposed. See. *billtuttememorial.org.uk*.

101010101

To the Science Museum in June for a two day conference *Making the History of Computing Relevant*. Organised by IFIP's Arthur Tatnall and sponsored by Google, over 100 delegates assembled from every corner of the globe (Do globes have corners? [ed.]) to listen to 29 presentations including no less than eight given by prominent members of the Computer Conservation Society. Visits to TNMoC, Blythe House and an after-hours reception in the Science Museum's Turing exhibition also featured.

There were far too many contributions to list individually, but an arbitrary selection might include –

- Chris Avram and Barbara Ainsworth's presentation on the Monads Project in Melbourne. Various exhibits are displayed in a main circulating area within the University. *Resurrection 44* gives a flavour.
- Marie d'Udekem-Gevers described a 19th century project in Belgium in which an attempt was made to set up a library for all the world's books in all the world's languages. The founders aspired thereby to prevent wars. A noble aspiration which fell spectacularly short (not least in Belgium) but an idea whose time was yet to come with the development of the World Wide Web.

Special mention should also be made of Peter Onion who delivered Charles Lindsey's challenging presentation at very short notice and with great aplomb when Charles was briefly taken ill.

101010101

In March Ewart Willey, CCS's first chairman from 1989 to 1992 passed away. He was BCS President in 1974/5. Within BCS, he was the founding chairman of the Advanced Programming Specialist Group serving from 1959 to 1974. He was a great servant of the society. He worked for many years as a Manager in the DP department of the Prudential in High Holborn.

In 1990 the very first article in the first edition of *Resurrection* was penned by Ewart. In it he offered a vision for the newly-formed society. Almost a quarter of a century on, his vision remains surprisingly intact.

And we also have to report the passing of Polish Computer Scientist Wlad Turski one of the co-creators of the seminal ALGOL 60 programming language and a BCS Distinguished Fellow.

The Cambridge CAP Computer

Andrew Herbert

CAP was the last mainframe computer built by the University of Cambridge Computer Laboratory, with a direct lineage reaching back through Titan, and EDSAC 2 to the pioneering EDSAC of 1949. In contrast to its predecessors CAP was not built to provide a central computing service to the university – that role had passed to the IBM 370/165 mainframe that had replaced Titan in 1971-3. CAP was an experimental machine arising from an academically driven research agenda: CAP did however go on to provide an internal computing service to support subsequent research in the laboratory, especially the Cambridge Distributed System based on the Cambridge Ring local area network.

The gestation of CAP was a consequence of the re-structuring of the Computer Laboratory: by the late 1960's the provision of a central university computing service on Titan and the IBM 370/165 required a dedicated large scale professional organisation whereas the growing teaching and research activities of the laboratory demanded a similar growth in academic staff. Thus in 1969 the Cambridge Computing Service was split off as a separate entity, led by David Hartley, reporting, through Maurice Wilkes, to the University Computing Syndicate. The teaching and research side, also under Wilkes, functioned as an academic computer science department within the Faculty of Mathematics.

At the time of the split some of the systems people who had worked on Titan chose to remain with the service (with Barry Landy following in 1971) and devoted their energies to developing Phoenix, the multi-access system that significantly improved and extended IBM's MVT operating system and TSO (time-sharing option).

For those who went across to the academic side, including Roger Needham and David Wheeler, the need was to find a challenging fundamental problem to work

on, but which hopefully would have practical implications for future systems of the kind that would be used by the Computing Service. Fortunately there was an obvious topic to go for, and one in which there was widespread interest across all the leading research laboratories internationally, namely "protection", that is the means by which user programs are prevented from corrupting one another, or the operating system, and indeed how the different parts of the operating system could be protected from one another. This need had arisen from the experience of developing the first generation of multi-access and time-sharing systems such the Cambridge Titan system, ICT's George 3, IBM's OS/360 and similar systems. They all had in common the fact they were programmed by large teams, using assembly code, and turned out to be quite fragile because an error in one part of the system could easily corrupt the memory used by some other part of the system, leading to a failure. There were also difficulties experienced in preventing user programs corrupting the operating system itself either through inadvertent blunders or deliberate "hacking" that bypassed inadequate checking of system call parameters leading to inappropriate use of operating system privileges.

In the research community three attacks had been launched on this problem: some looked to writing operating systems in high level languages in the hope that modularisation, strict data typing and array bounds checking would trap coding errors; some looked to using formal methods to rigorously prove the absence of defects in their system; and a third group, including Cambridge, looked towards hardware solutions to the memory protection problem.

When the CAP project started in 1970 there were two approaches to hardware protection. The first to be investigated was the use of protection rings and access control lists for system resources by the General Electric (later Honeywell) Multics system. The principal concept was that the operating system would be divided into concentric rings of increasing privilege, and access to hardware resources such as memory, physical I/O devices, page tables and processors would only be accessible to software running in a specified ring, or an inner ring (and therefore with even higher privilege). User programs ran in the outermost ring. To provide finer granularity, system resources such as files were associated with access control lists stating the privileges held by different users towards the resource. Thus, for example, to open a file a user program would call the file system which would be running in an inner ring and the file system would look up the file name, check the user's access privileges, and, if acceptable, make the resource available.

Experience with building the Multics system had exposed two limitations to this architecture. Firstly, it was hard to decompose an operating system into a strict hierarchy of levels: often independent sub-systems would end up in the same ring and therefore not be protected from one another. Secondly, for shared objects, access controls lists could get very long and so additional features had to be added to represent groups of users, public access and so forth.

The alternative approach to that followed by Multics was the use of "capabilities". Here the concept was to divide up the operating system and user programs into hardware protected "domains", where each domain held a set of unforgeable tickets, called capabilities, for the resources it required to perform its function. Capabilities could be passed from one domain to another (for example by inter-process communication) to enable sharing of resources.

From an abstract point of view access controls lists and capabilities are "duals" – both essentially encode an "access control matrix" with "subjects" as its rows and "objects" as its columns – each cell indicating the privileges an individual subject holds towards an individual object. The argument of the capability enthusiasts was that their encoding of the matrix (by row) was more practical than the by column encoding inherent in access control lists.

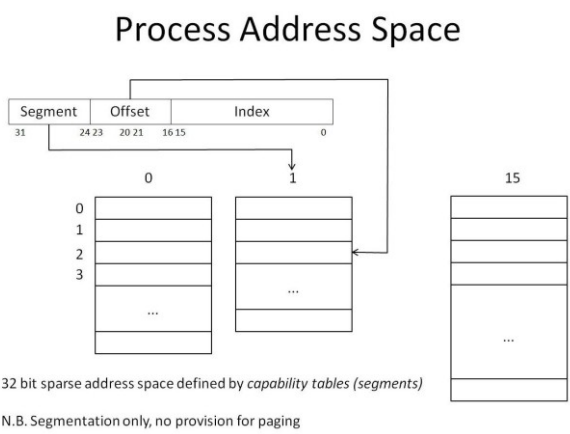
At the time work on CAP started, the capability concept was mostly theoretical: there had been some early work at MIT, Berkeley and Chicago but nobody had build a capability machine, let alone written an operating system for it. Wilkes had been aware of the American work and persuaded Needham and Wheeler that this would be a good project for the newly founded Systems Research Group. They divided the work between them along lines established during the Titan development: Wilkes mapped out the basic concepts and went about securing funding for the project; Needham took responsibility for the system architecture and Wheeler the hardware design. Construction was undertaken by the Computer Laboratory Electronic and Mechanical workshops. When the hardware was completed in early 1975, work commenced on writing the operating system which itself was completed in 1977. A major update was undertaken in 1978 to make CAP an integral part of the Cambridge Distributed System (CDS) and it remained in use thereafter as a departmental machine used for cross platform development of many of the microprocessor based servers that made up CDS. Unlike its predecessors, CAP was never formally "turned off": it was shut down when a fire in an adjacent department threatened the Computer Laboratory and never switched back on. It now survives as a static exhibit in the Computer Laboratory's new home in the William H. Gates III building at West Cambridge.

System Architecture

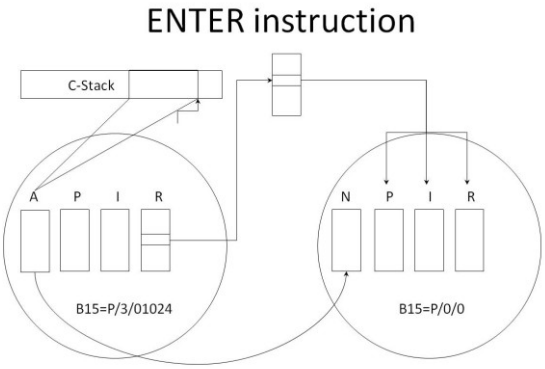
Wilkes' original concept, following the work by Fabry at Chicago, was for a "capability register" machine in which there would be special registers to contain capabilities and programs would access system resources through these registers. A capability would either denote a region of store (in terms of physical base address, size and access privileges) or a protection domain. Ordinary instructions would access data via registers holding store capabilities, much like segment registers in other systems, and control could be passed from one protection domain to another by referencing a domain capability for the target domain. There would be privileged segments containing capabilities (distinguished by having "read capability" and "write capability" privileges as opposed to read, write and execute privileges for ordinary data). Programs could load capability registers from these segments using special instructions as required. Privileged parts of the operating system would have "write capability" access to capability segments in order to set up new capabilities for dynamically allocated resources.

A practical design was developed along these lines, and indeed was taken as the basis for the Plessey System 250 computer following a visit of Plessey to Cambridge. However there were concerns expressed by colleagues working on high-level language systems (Martin Richards, BCPL and Steve Bourne, ALGOL68) that managing capability registers as well as the normal general purpose registers was an unacceptable burden on compiler writers. Ultimately a design due to Robin Walker, one of Needham's PhD students, was adopted in which capability addressing was implicit and therefore transparent to compiler writers.

Walker's idea was that capability segments could be used to describe address spaces. In his model an address consisted of three parts: table number, capability number and offset. The table number selected one of 15 possible capability segments, the capability number one of up to 256 capabilities in

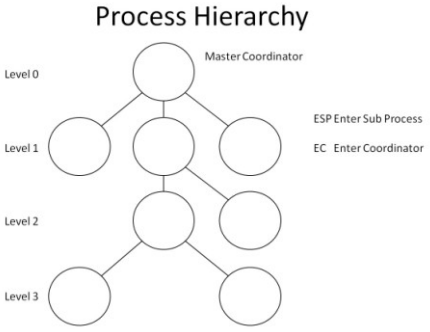


that segment, and then if it was a store capability, the offset selected the location addressed relative to the base of the region of store described by the capability. This turned out to be very flexible: an address space could be set up with adjacent capabilities defining a contiguous flat address space or it could be sparsely configured with separate data structures in their own isolated segments to provide hardware bounds checking. To implement this, Walker envisaged a hardware capability cache acting like a page table cache (translation look-aside buffer) in a paged virtual memory system.



In Walker's design a protection domain was called a "protected procedure" and a protection domain capability was called an ENTER capability, named after the special ENTER instruction used to transfer control from one procedure to another. Each protected procedure was defined in terms of three capability segments that held capabilities for the resources unique to the procedure: these segments became tables 4,5 and 6 of the address space following the ENTER instruction. The ENTER instruction would then transfer control to address 4/0/0, the "entry point" for called procedure.

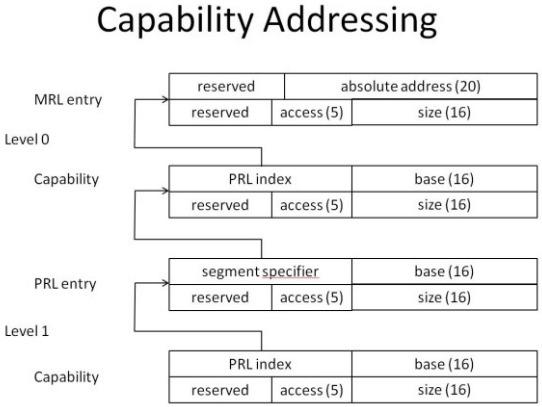
To facilitate passing capability arguments there was a further special instruction called MAKEIND. This would create a brand new, empty table 3 in the address space. Capabilities could be moved between tables using a special MOVECAP instruction. On ENTER the calling procedure's table 3 would appear as table 2 in the called procedure.



Each address space in CAP corresponded to the conventional concept of a process. There was a process hierarchy with a master coordinator at the top level and successive layers of sub-processes beneath it. Any process at any level could launch a sub-process using

the ESP (enter sub-process) instruction. This would be passed to a segment containing the details needed to create the junior process as a separate address space. This consisted of four data structures: the process base, c-stack, resource list and capability tables. The process base provided the system with space to dump registers and other state information. The c-stack provided the space used by the MAKEIND instruction to create new capability tables.

A process's capability tables were seen as data by its coordinator process and as capabilities by the junior process. The senior process would lay out the capabilities it wanted to grant to a junior process by setting up a process resource list referencing its own capabilities by their address in the senior address space.



The junior process capabilities could be set up to reference the senior capability in its entirety or they could be set up as a "refinement" restricting access to a subspace of the senior capability with possibly reduced access privileges.

The mapping was done in two steps: each capability in each capability table referenced an entry in the process's resource list and the resource list entries contained the address of the senior capability – this allowed a senior resource to be mapped to several junior resources so that, for example a single store segment at the higher level could be carved out into separate program, stack and heap segments at the lower level.

This structure was fully recursive and could in principle be expanded to any depth. In practice only two levels were used and the CAP hardware had an implicit limit of depth 64. At the top level, the resource list of the most senior process contained physical store addresses so that ultimately a capability address in a junior process would resolve to a physical memory address. A junior process could return control to its immediate superior using the EC (enter coordinator) instruction which would resume execution in the coordinator process at the instruction following the ESP instruction that had started the junior process running.

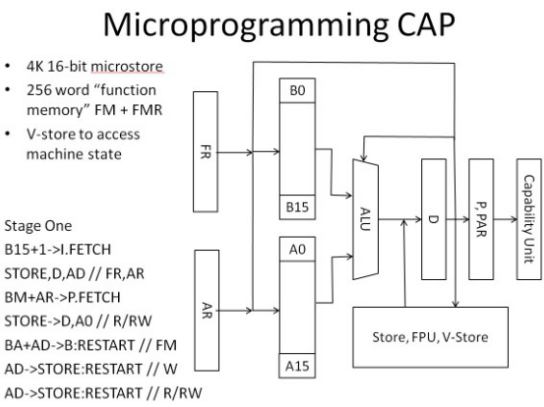
Hardware interrupts and traps forced a return to the top-level (master) coordinator with a transfer of control to an address specified in its process base.

While it was possible for protected procedures running in a single process to pass ENTER capabilities back and forth it was not possible to pass them between separate processes at the same level or between senior and junior processes. This was not seen as a limitation, but it was an issue that had to be addressed by the operating system.

In all other respects the architecture was conventional and heavily based on Titan. Store was word addressed, words comprised 32 bits. There were 16 general purpose B registers. BO was a constant 0. B15 was the program counter. B14 was conventionally used for subroutine return links. A typical instruction was BBPS Ba Bn N, meaning take the contents of the store location addressed by adding N to the contents of register Bn and add them to the contents of register Ba. Thus, in this example, Ba is acting as an arithmetic register and Bn as an index register.

CAP Hardware

The CAP hardware was designed by David Wheeler and was his first excursion into the use of integrated circuits: the two EDSACs having used thermionic valves and Titan discrete transistors. In the Cambridge tradition it was a microprogrammed machine. There was 4K of 16 bit micro-store and 16 additional A-registers, additional to the B registers available to the microprogrammer.



The internal state of the machine could be accessed via a separate "V-store": reading and writing to a V-store location would access or set the corresponding flip-flop, buffer or register as appropriate.

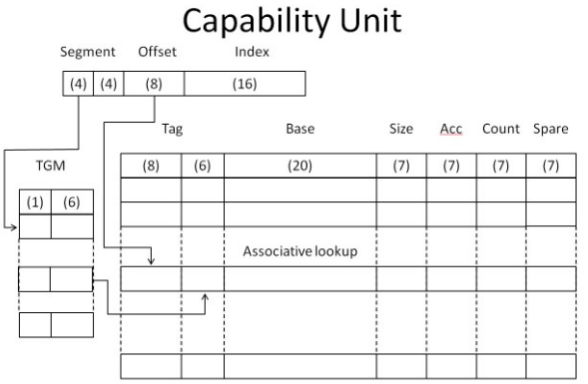
Micro-instructions comprised register-to-register transfers, register to micro-store, register to V-store and control transfers: for example, the microinstruction B15+1->I.FETCH would copy the contents of B15 (the program counter) to the

accumulator, increment it and both send the result back to B15 and to the address decoding system with a "fetch instruction" access request (as opposed to "read data" or "write data"). A subsequent STORE->D,AD would gate in the result from the store to the accumulator, known to the microprogrammer as the D register. The instruction would stall until data was available from the store.

Two 256 word read slave stores (i.e., cache memories) and one 32 word write slave store were provided, indexed by physical addresses.

Floating point arithmetic and integer multiplication and division were provided by an autonomous floating point unit, with its own micro-engine. The unit held numbers as a 64 bit mantissa and 8 bit exponent. This was the one part of the machine where Wheeler was given complete freedom and he went to town. Only Wheeler understood the inner workings of the unit and others relied on him supplying scripts for performing commonly required functions.

Wheeler's approach to documentation was idiosyncratic. Until Chris Slinn (one of Needham's Ph.D. students) wrote a comprehensive manual, the primary documentation for the microprogrammer was a double sized sheet of A4 comprising handwritten fragments of timing diagrams, logic diagrams, circuits and tables (e.g., names of V-stores). Wheeler assumed that this sheet and a reading of Walker's thesis were sufficient to understand CAP.



The most unique part of the machine was the "capability unit", a 64 way cache for holding capabilities and performing address translation. The cache was searched by matching on the table number and capability number from an address and, if matched, the corresponding entry would contain the physical base address, size and access rights of the corresponding capability. Capability search was automatically carried out as part of store addressing and so transparent to the macroprogram and microprogram. If no match was found, the size limit exceeded or access right violated a microprogram trap occurred. (The capability unit had a two level

lookup structure so that on an address space change only the contents of 16-way "tag memory" had to be invalidated as opposed to the entire 64-way cache.)

Initially CAP accessed peripherals via a CTL Modular One with a teleprinter, paper tape reader and punch, line printer and fixed and exchangeable discs. (The fixed disc was only used by the Modular One operating system). The CAP was connected to the Modular One via a fast link accessible to the microprogram via the V-store. Later the Modular One was replaced by a Cambridge Ring interface. The microprogram implemented a "basic block" (i.e., packet) level interface to the operating system that in turn implemented higher level protocols for access to the Cambridge Distributed System file server, printer server and terminal server etc.

Thus principal job of the CAP microprogram, in addition to implementing all the regular instructions, was to manage the contents of the capability unit as the makeup of an address space was changed by ESP, EC, ENTER, RETURN, MAKEIND and MOVECAP instructions, and to manage input/output. In general terms each of these took about one third of micro-store and it required very tight coding to fit the complete microprogram in the available 4K.

(CAP also had a dedicated high speed tape reader and teleprinter connected: these were used by the microprogram to read in micro bootstraps and print diagnostics. The most useful diagnostic for the microprogrammer was PMDUMP which produced a lineprinter listing of the full internal state of the machine including slave stores, registers and V-store).

Initially CAP had 192K of main store, comprising two 32K word units of Plessey 2 microsecond core store salvaged from Titan and 128K of slow Philips 10-12 microsecond core store. Around the time of the Ring connection, the core store was replaced by 256K words of Intel memory using 16K memory chips. Later still these were replaced with 64K memory chips obtained from Acorn who bought in bulk and were able to get them more cheaply than the university could, taking the store capacity up to a magnificent 1024K.

The mechanical construction of CAP was quite novel. Each distinct functional unit (e.g., capability unit, microprogram) comprised two "pages" of circuit boards each measuring approximately 3 foot by 2 foot attached to a hinge at the rear of the machine. Each page was wire-wrapped on the outside faces and the logic mounted in sockets on the inside faces.

Each pair of pages snapped together and could be swivelled from left to right like pages of a book. As pages were moved a baseplate beneath then was automatically shifted left or right to ensure that cool air blown up from the base of the machine was vented over the logic. To many visitors this simple mechanism drive by a motor pulling a cord and micro-switches was the most intriguing part of the machine and featured in any demonstration.

Operating System

The CAP operating system was architected by Needham and various parts given to research students as PhD projects. The present author for example wrote the system generation program as his first year project. Andrew Birrell was hired as a research assistant to pull together all the contributed code into a coherent whole. He also wrote the bulk of the filing system. Birrell had been a Ph.D. student of Steve Bourne working on ALGOL68 compilation, and he and Steve persuaded Needham to use ALGOL68C as the CAP implementation language. In terms of programmer productivity this was an excellent decision and ALGOL68C proved itself to be a good systems programming language, but it subtlety undermined the original project aim of using hardware to defend against programming blunders since the strict type checking of ALGOL68 removed many of these: it was hard to evaluate afterwards how much relatively speaking the programming language or the hardware checking had contributed to the rapid speed with which the operating system was built and debugged.

The structure of the operating system echoed Titan. There was a master coordinator responsible for allocating resources to and scheduling a set of processes, some of which were systems processes and others of which were user processes, corresponding to an offline user job or an online user session at a terminal. A user process could communicate with the systems processes through a set of protected procedures of which each user process had a copy. For example, the file system was accessed through a procedure called DIRMAN (directory manager). A freshly started user process was granted an ENTER capability to an instance of DIRMAN endowed with a capability for that user's home directory. To DIRMAN the directory was just a disc file. DIRMAN had the

capability (literally) to send messages to the Virtual Store Manager process to map disc files in and out of its address space as the user traversed the file directory structure. Similarly each user process had an Input/Output Controller (IOC) protected procedure. This could be requested to create an input or output stream connected to a device or a file. Each such stream was a dynamically constructed protected procedure holding the required capability to speak to the appropriate device process or disc file. (Output devices were generally spooled).

Given the architectural limit of a store segment to 64K, a windowing mechanism was available to allow programs to read or write files longer than this. Through the STOREMAN (store manager) procedure a user process could ask to have a region of length up to 64K of a disc file to be mapped as a store segment into its address space. Further calls to STOREMAN could be used to move this window up and down the file.

Generally when a user program obtained a capability for a disc file it was delivered as an invalid store capability containing a reference to the "system internal name" (SIN) of the disk file. (A SIN was the equivalent of a Unix inode). The first time a program tried to use the capability there would be a trap in the microprogram that would result in a software trap to the master coordinator which in turn would then force an entry to the user process's FAULTPROC protected procedure, where the trap would be inspected. On detecting the "outform" capability, FAULTPROC would then ask the virtual store manager process to "inform" the disc file as a store segment. The virtual store manager would in turn invoke the real store manager to allocate physical space and then the user capability could be fixed up and the failing instruction retried.

Unfortunately CAP (like Titan) didn't do paging and so whole segments had to be swapped, which never performed well. Despite many valiant attempts, Needham and the author never succeeded in devising a segment-swapping algorithm that did better than "random".

There was a similar virtualisation mechanism for protected procedures. A further protected procedure called "MAKEPACK" could be instructed how to compose an outform protected procedure in terms of the structure of the address space, the files to be bound into it (i.e., program) and any dynamic storage to be created (e.g., stack and heap). MAKEPACK wrote this information to a disc file where it was known as a "procedure description block" (PDB). By wrapping up code and data as a PDB a user could package up a "protected sub-system" that others could call by ENTER/RETURN but without the ability to peep inside. A typical PDB was that for the ALGOL68C compiler that had capabilities for the code of the compiler, the libraries it needed and the workspace required.

If a user fetched a PDB entry from a directory (via DIRMAN), an outform ENTER capability would be set up referencing the SIN of the PDB. When the user attempted to exercise the ENTER capability there would be a trap, ending up in FAULTMAN which in turn would invoke the LINKER protected procedure to read down the PDB and instantiated it as an inform, i.e., hardware, ENTER capability.

Evaluation

By the time the CAP operating system was complete, it was generally accepted that operating systems should be written in a high-level language, and if one of appropriate strictness was used, many of the errors of the kind that originally motivated the project would be removed. However current mainstream operating systems are mostly written in C and C++ which are not that strict and so problems of system crashes persist, especially for systems like Microsoft's Windows which have to tolerate third party code (i.e., device drivers) deep in their internals. To address this issue Microsoft is increasingly turning to automated verification techniques, for example the SDV (static device driver verification) toolkit based on Byron Cook's work on proving program termination. From this perspective the two alternatives to hardware protection working together can achieve the original goal without hardware support.

However in terms of system structure, capabilities forced a very strong modularity on the CAP operating system and narrow interfaces between those modules. This turned out to be very influential on the next generation of operating systems that were designed for multi-processor and network-based distributed systems and where cryptographic techniques could be used to make unforgeable "network capabilities". The division of the operating system into a simple coordinator, set of system processes and "in-process" brokers to access them influenced the design of microkernel-based operating systems, such as Mach from CMU which underpins the contemporary Apple MacOS.

From a performance point of view CAP was pedestrian, although compared to other capability systems of its time, which lacked the same level of hardware support, it was quite nimble. A null ENTER/RETURN sequence cost about 50 times that of a normal subroutine call. A process switch was several hundred times as expensive as a procedure call. Much of this can be attributed to the limited size of the capability unit (just 64 entries) that in comparison to a current translation look-aside buffer is puny. That said, running as a four user departmental system it compared favourably to the level of service offered by the central IBM 370/165 or of the first generation of home (i.e., personal) computers that began to appear towards the end of its life.

Certainly in terms of the subsequent careers of those who worked on it, the CAP project was entirely beneficial: CAP was widely respected and many of us who worked on it went on to significant roles in academia and industry.

A lesser-known story about CAP relates to C++. Bjarne Stroustrup was Wheeler's PhD student. There was much debate in the software team about the relative merits of message-based operating systems and Hoare monitor based systems. Needham with a colleague Hugh Lauer at Xerox PARC where Needham was a consultant published a paper that showed that theoretically a system of one form could be automatically transformed to the other, and so the only substantive question was that of performance of the different primitives. Stroustrup decided to investigate this by simulation and so he built a model of the CAP operating system in Simula 67 for the IBM 370/165. Unfortunately the university only had a teaching version of Simula and Stroustrup's program was too large. He had to rapidly change tack. To his good luck Martin Richards had just built a co-routine system for BCPL to support his work on a new operating system called Tripos and this provided Stroustrup with the basics of a simulation engine so he converted his (object-oriented) Simula to BCPL and ran his simulations on CAP. On securing his PhD, Stroustrup left Cambridge to join Bell Labs. His first paper published there was on how to add co-routines to C, followed by a next paper on an object-oriented front-end for C and from thence to C++.

Footnotes

There have been three commercial capability machines. The Plessey System 250 as mentioned in the text, an unsuccessful Intel system, the iAXP432, and IBM's very successful midrange System/38.

An excellent out of print book by Hank Levy comparing and contrasting both academic and commercial capability systems can be downloaded from *tinyurl.com/capabook*.

A book on CAP was written by Wilkes and Needham and published by North Holland in 1978. Also out of print, it can be downloaded from *tinyurl.com/rneedham*.

Further details of CAP can be found in the proceedings of the 1977 ACM Symposium on Operating Systems in the ACM Digital Library.

This is an edited version of the presentation made by the author on 18th April 2013 at the London Science Museum. Andrew's email address is andrew@herbertfamily.org.uk

Homebrewing Computers: Then and Now

Oscar Vermeulen

> We are accustomed to the mindset that the world of computers is divided neatly into two camps – wardrobe-sized machines built from valves or transistors which excite our interest and the rather more prosaic little boxes we have on our desks or carry around with us – with processors contained within a single integrated circuit. So it may come as something of a shock to realise that, with the passage of time, these "modern" computers have a history stretching back nearly 40 years. There is a parallel world of computer history developing out there relying less on artefacts and more on "culture". It is a world with which we might profitably have more contact. Here is an introduction to that world. [ed.]

Hobbyists played a key role in the birth of the microcomputer. Collectively referred to as "homebrewers", they helped shape and define a computer category that mainstream computer manufacturers regarded as utterly irrelevant even as recently as 1980. Today, building your own computer in the spirit of those 1970s homebrewers can offer unique insights in the early years of microcomputing – and provide a great learning experience. This article provides an overview of homebrewing's role in microcomputing, and may convince you that it is a trip worth embarking on – even if you are 40 years late to the game.

Technology develops at break-neck speed. Today, there is no sensible use for 8-bit, 64 kilobyte computers with less processing power than a mobile phone. Nevertheless, after the birth of the microprocessor in 1971 it would take almost a full decade before microcomputers outgrew even that minimal specification. No wonder that mainframe computer manufacturers – IBM and the 'BUNCH' (Burroughs, UNIVAC, NCR, Control Data Corporation, and Honeywell) – showed no interest at all in these little chips. Instead, it was pretty much left to hobbyist "homebrewers" to define what could be done with these new but woefully underpowered microprocessors.

A Home-Brew N8VEM system

Yet the dream of owning a "personal" computer was a strong drive for many – there was indeed, a nascent commercial market here. Even before single-chip microprocessors existed, John Blankenbaker built the Kenbak-1 in 1970 – in his garage. Constructed from discrete TTL logic chips, this programmable machine offered 256 bytes of memory – but it was a computer nevertheless. At the unimaginable price of $750. The garage would soon prove to be an important place for microcomputers. Steve Wozniak famously developed his Apple I (now a prized collectors' object) in his parents' garage in 1976. In the five years that separate the Kenbak-I from the Apple I, the purpose of microprocessors and microcomputers was defined largely by tinkerers wondering what these things could actually do. Even Intel didn't quite seem to know. Originally intended to power calculators, they hired part-time tinkerer Gary Kildall to work on software development for their microprocessor on his days off as a lecturer at a Postgraduate School. He looked closely at what software made respectable minicomputers like DEC's PDP-8 tick and proved something similar (his CP/M operating system) could be run on microprocessors. It proved to be a defining moment: microprocessors could do serious computing work.

With the benefit of almost 40 years of hindsight, you no longer need to be a wizz-kid to understand – or even build – these very early microcomputers. In fact, there is a lively "retrocomputing" community these days that studies these early-days computers, designing their own systems from a "bag of chips" and an improvised circuit board. Hobbyists without any background in electronics somehow picked up the required skills and now share their homebrewing experiences online. Although some of their creations are stunningly exotic, most people actually build very simple machines: they take a CPU, add RAM, ROM, a serial port plus maybe a hard disk for storage. And most of them run either Basic (like the 1980s home computers) or use a "vintage" operating system like CP/M. Running CP/M, in fact, is a very nice target to work towards: lots of excellent, historical software ensures that your homebrew computer can do something interesting once it is built. CP/M has the benefit of being very simple. A few days of study are enough to run it on your circuit board.

Still, one challenge remains: if homebrewing is to be an enduring hobby instead of a one-off project, there should be some perspective beyond putting together a minimal computer and switching it on. But working all on your own, taking the next steps can get progressively more difficult: projects such as building graphics subsystems or using exotic processors.

This is where the N8VEM group comes in. In 2006, Andrew Lynch published his own single-board CP/M design with the express intention to engage and involve others. The N8VEM (named after his ham radio license) was intended to be built from 1970s electronic components – but also to be expandable with add-on cards and soon, an informal collaborative effort emerged. Builders with a wide range of skills got involved – from well-known 1970s systems designers to curious beginners who bought Andrew's $20 circuit board and then ordered the handful of required electronic components plus soldering iron online from an electronics distributor. Two days of wielding the soldering iron results in owning a nice CP/M computer and a full understanding of its inner workings. Builders who catch a more severe infection of the homebrewing virus, can even expand it into a powerful (we use the term lightly here) multiprocessor system with "blinkenlight" front panels, hard disks, graphics subsystems and various historical operating systems.

N8VEM, though, is certainly not about providing soldering kits. It is about experiencing the old homebrewing challenge, picking up skills along the way. Skills that range from reading schematics, down to debugging a computer card that does not do what it was supposed to. The learning curve may be steep at times, but in the N8VEM mail group, expert help is at hand. Nothing prevents you from plugging in your own CPU board design, but if you do you're not forced to then also develop all the other expansion boards on your own. And as the novelty of designing a simple SBC (single-board computer) wears off, maybe you prefer to focus your energy on exploring graphics systems, or ways to hook 8 bit machines up on the internet. Or jump into systems software development and share the fruits of it with a few hundred others.

First Steps: the Single-board N8VEM Computer

The N8VEM Single-Board Computer

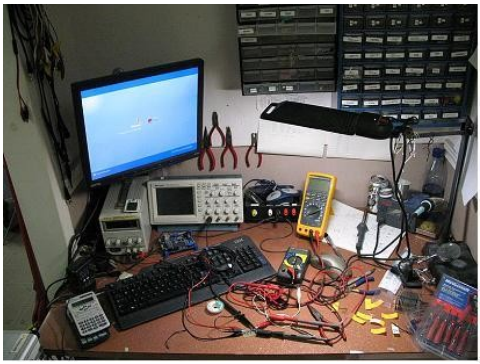
At a size of roughly four by six inches, the N8VEM computer does not look particularly impressive. Yet, it provides all the capabilities of a commercial microcomputer of the late 70s – in fact, thanks to CP/M it is software-compatible with them. That provides access to a range of still very useful historical software – programming languages including Basic, C, Pascal and, of course, assemblers. Some of the earliest word processors, spreadsheets or databases can be experienced.

The N8VEM is so small due to one concession to modern-day electronics: it uses a single, high capacity RAM chip as opposed to many dozens of old, small RAM ICs. But all the other electronics are components that would have been used "back in the day": simple 74LS logic chips, plus a Z80 and classic interface chips. Memory is backed up by a battery, and therefore the RAM disk is a practical storage mechanism. Especially because a ROM disk comes with most essential software installed. You use the N8VEM either with a 1970s serial terminal, or (more likely these days) with a PC terminal program. The Xmodem protocol allows transfer of files to and from the N8VEM.

Core to the expansion options of the N8VEM is the ECB bus. The N8VEM can be plugged into a "backplane" and access about a dozen or so peripheral cards that have been created so far. However, the first expansion option is actually not an ECB card, but the $5 PPIDE mini-board, which allows the use of an IDE hard disk or Compact Flash card. Costs are minimal: even an old 256MB drive offers more storage than can reasonably be filled with CP/M software.

The N8VEM is supported by a very effective toolchain. Software can even be developed from the comfort of a modern PC, tested on an emulator and then copied onto the real machine.

How to Get Started: Books and Tools



A typical hobbyists' electronics workbench with the tools of the trade

The challenge with homebrewing is mostly to find out how to do things. Once you know, most steps are straightforward. And that is why homebrewing as a group makes a lot of sense. Still, two pieces of background information will prove indispensable for any builder: understanding basic computer hardware and having an understanding of assembly language. Reading up on these topics will not only make things easier, but will also add to the understanding of what you are putting together. Some free literature suggestions are at the end of this article.

Only a few tools are really necessary – although for many, building up an electronics lab is part of the fun. A good soldering iron, and a multimeter are absolute requirements. An old secondhand oscilloscope is a useful extra. Lastly, at some point you will need an Eprom programmer, unless you want to depend on others to burn eproms for you.

How to go Further

The obvious first step is to add the $20 backplane, and put the N8VEM plus backplane into a case (options range from DIY woodwork to buying a standard 19" card cage). From here on, the choice is a personal one. A few highlights:

- VDU cards: Adding a video card frees you from the use of a PC and brings that warm glow of old CRTs into the room. Using 1970s CRT chips – still easily available today – is also very interesting from a software perspective.
- Adding hard and floppy disk drives increases the "vintage" qualities of the N8VEM. But in practice, you will probably not use the floppy disks very much.
- "Frankenstein systems", blending old and new technology can be very interesting. For instance, a modern Propeller microcontroller card adds mass storage in the form of a modern SD card, but at the same supports VGA graphics and PS/2 keyboard.
- Blinkenlights: a front panel with dozens of switches and flashing lights was highly desirable in the 1970s. The ECB Bus Monitor card looks like a classical front panel, but actually is a much more sophisticated debugging tool.

Multiprocessor Systems

Another option is the 6x0x board, which deserves special attention. It adds a second processor and operating system to the N8VEM. Builders can choose to plug in a 6502, 6802 or 6809 microprocessor and let it run DOS/65, Flex or Cubix – all operating systems that played an important role defining computer systems in the late 70s.

Having a card rack full of peripherals operating the Big Three 8-bit microprocessors, running everything from CP/M to Flex, the N8VEM becomes a true fetish object for computer history enthusiasts. It can take years to master its universe of hard- and software, with plenty of manageable yet satisfying projects still waiting to be done.

In the last few years, development efforts have also broadened out towards the well-known S-100 bus, made famous by the original 1975 Altair computer. New S-100 cards provide everything from mass storage devices, video cards, to 80286 and 68000 processor cards. With that range of options, builders can provide a new lease of life for historically significant S-100 systems.

Keeping Things Together

With such a broad array of projects, and involvement from people with widely differing skill sets, it is remarkable how the N8VEM mail group binds all these builders together. The youngest builder is 14 years old; it would be impolite to mention the age of the oldest. But computer historians will be quite likely to encounter some of the people that designed those early 1970s machines on the N8VEM mail group. This loose organisation of builders means that it is quite easy to join in – whether homebrewing is a full-time hobby or just a one-off desire to own a $20 self-built computer for fun.

Suggested Literature, Available Online:

- Steve Ciarcia, Build Your Own Z80 Computer.
- Rodnay Zaks, Programming the Z80.
- N8VEM mail group: groups *groups.google.com/group/n8vem*
- N8VEM depository: *n8vem-sbc.pbworks.com*
- S-100 board development: *www.s100computers.com*

Oscar Vermeulen is an economist by education, and worked in quantitative investment analysis for the last 25 years. Witnessing the revolutionary impact (for better or for worse) of computing in the financial sector, his early passion for computing machinery evolved into a lively interest in computer history later on. He lives in Switzerland and can be contacted at o.vermeulen@altis.ch.

CCS Website Information

The Society has its own website, which is located at *www.computerconservationsociety.org.* It contains news items, details of forthcoming events and also electronic copies of all past issues of *Resurrection*, in both HTML and PDF formats, which can be downloaded for printing. We also have an FTP site at *ftp.cs.man.ac.uk/pub/CCS-Archive*, where there is other material for downloading including simulators for historic machines. Please note that the latter URL is case sensitive.

Letter to the Editor

Ian Cottam

Back around 1970, together with a fellow student (B. Jones), I worked on my first systems programming project. The goal was to produce a batch processing system for student-sized programs written in ALGOL 60 on an Elliott 803B computer. We called our program BRIAN, for "Batch Running In ALGOL Now".

The Elliott 803B was already long in the tooth, and that was one reason we never expected our system to see serious use, even though it worked. The other reason that made this endeavour truly 'academic' was that we had no card reader, only 5-hole paper tape. If you think about it for a while, batching jobs up on paper tape – such that no operator intervention was required to run them – is not very practical. (Splicing individual punched tapes together would take longer than the operator running them manually.)

The 803B was a (small) room sized computer that in modern terms was essentially a single-user, programmable calculator. There was no operating system, and I have always been amazed that it supported ALGOL – well perhaps I wasn't amazed at the time, but looking back I am. The compiler was one of the early ones that included much of the ALGOL 60 language with a few extensions for input/output. The compiler team had been led by Tony Hoare, already famous for his Quicksort invention and these days is Professor Sir Tony Hoare FRS who, in his academic retirement, works for Microsoft Research in Cambridge.

Another amazing thing was that Elliott's did not have (to my knowledge), or provide to their users, a conventional symbolic assembler. (I think one was written by a UK college in the south of England. I know we had access to one, but the Elliott ALGOL compiler was not written in such a pleasant language.) Besides ALGOL, one could code for the 803B in machine code or you might call it a non-symbolic assembler. It wasn't binary, but it was one step up from that: you wrote the instructions (load, store, etc) in their octal code and operands and addresses in decimal. That was it. I guess, on the positive side, the code to convert such programs to binary and load them was just a handful of instructions.

We did have a source code listing of the compiler: an incredibly thick, and heavy, bound volume. One clever thing that Tony Hoare had done was make the compiler team first write each of the recursive procedures that in aggregate formed the translator, in a pseudo ALGOL dialect. It was "pseudo" because there was no compiler for it (obviously) and also, as was seen in many subsequent software engineering efforts, it was a design language rather than an

implementation one. When the bound volume was opened, the pseudo ALGOL was on the rightmost page and the hand-written machine code to match it was on the corresponding left hand page.

What the debugging activity for it must have been like is hard to imagine. I experienced a little of it, as we opted to write our batch monitor system, which would invoke the compiler, in machine code too. I think we drew a top-level flowchart but then coded it straight into the aforementioned octal/decimal form. In those days one tended to use desk checking to both convince yourself that a (machine code) program was correct, and also to find errors when subsequent testing showed that it wasn't. The painful memory I have is that it took something like two solid weeks of desk checking the code to excise the most subtle bug and convince ourselves that it worked.

I decided that I never wanted to spend two weeks like that again (although on reflection it has been close in some projects). When I got the chance to do something similar a few short years later, with another colleague, our effort was much more like engineering.

The compiler this time was ALGOL 68R written by the Royal Signals and Radar Establishment at Malvern, and the computer was an ICL 1900 series running a batch system: George2. The task we took on as a spare time project was to make this compiler accept source programs already on the disk filing system, rather than from cards as it was coded to do. I know this sounds incredible when now, universally, at a command line interface you just type

```
compiler < diskinputfilename
```

but on the ICL 1900 with George2 it was a complex endeavour; particularly as this time we did not have the source of the compiler. This turned out to be a blessing in disguise: we could separate the various concerns and solve them one at a time. The new code was written in a symbolic assembler (PLAN) and tested with a stub for the compiler. The assembler could produce a form of machine code that another program could use to apply patches to existing binaries. We used this to patch the new disk handling code into free space within the compiler and link it in by a patch jump from the obvious card input instructions.

There was no fraught two weeks of desk checking, and it was an activity worth every minute to see our user's face when we told him we had solved his problem without access to the compiler's source code. He was a traditional engineer and thought we had used some kind of magic or voodoo.

40 Years Ago ….
From the Pages of *Computer Weekly*
Brian Aldous

IBM awarded Soviet Intourist contract Whilst Intourist has forbidden contract detail disclosure, IBM competing with ICL, CDC and Univac has won the contract. Washington believes IBM has bid a 370/155, one of the largest computers ever to be sold to the Soviet Union.

Woolworth's buy British Beating off strong competition from IBM, ICL has won one of its biggest-ever commercial orders, believed to be worth not less than £1m, and involving a 128K 1904S, a 32K 1902A and 100 VDUs.

Colour VDU to cost £2000 Small Lancashire company Terminal Display Systems to announce colour VDU costing £2000 with keyboard and believed to be the first such system to offer the user a choice of both background and foreground colour.

Fifth operating system for PDP-11 soon from DEC The fifth and smallest O/S for the PDP-11 range has been announced by DEC's Laboratory Data Products Group. Called RT-11, it is expected to be available in the UK by September, at a cost of £380.

Light pens cut prices at Argos New "catalogue showrooms" come into operation, whereby customers make their selections from a catalogue. Shop assistants have identical catalogues, with added barcodes which are read by Plessey Light Pens and sent direct to the warehouse.

Dorset lathe programmed over link with Ohio A time-sharing link between Dorset and Ohio via Intelsat is helping a Powell Duffryn Group division to compile the programs to run its new £70,000 Heynumat 2000 numerically-controlled lathe.

Talking to the machine The latest claim to have successfully communicated with a computer in spoken English has been put forward by a team in Newton, Massachusetts, working on a research project financed by the US Navy.

1906S and Ferranti systems for Babs Contracts worth nearly £2m in total have been placed by British Airways Board with ICL and Ferranti for computer equipment for use in the Babs complex, which will replace the existing BOAC Boadicea and BEA Beacon systems. The contract includes the largest 1906S ordered to date, to work alongside an existing 1904A, replacement of another

1904A with an ICL 2070, and over 20 new Argus 700T to complement an existing 50 Argus computers.

Digitising by touch A new digitising surface, a sheet of glass which can be activated by finger-touch, has been announced by Intertrade Scientific of West Drayton, Middlesex.

Second phase of NPL network in operation The second phase of the National Physical Laboratory's general purpose packet-switched network is now operational, and is in service over the NPL's sites at Teddington and Feltham, Middlesex. This upgrade eliminates the limitation of the earlier system in which the filestore could only be accessed by one computer at a time, by adding a software packet-switching interface on the network and matched to an interface on the filestore.

Plessey get £1m Post Office order Post Office places £1m order with Plessey for a duplicated twin-processor Plessey 4660 to run the UK's first computer controlled International public telex exchange. The system is in fact, an Astrodata 4660 system for which Plessey signed a marketing agreement in 1972, and involves four General Automation 32K SP16/85 minicomputers.

GEC 4080 to aid Nimrod Rutherford High Energy Laboratory has ordered a £36,000 GEC 64K 4080 system for data collection in connection with the Nimrod proton synchrotron project.

Ferranti's flicker-free display Using a plasma neon matrix display system, Ferranti has announced a flicker-free display, claimed to be three to 10 times brighter than comparable displays.

EPSS contract for PO won by Ferranti Ferranti has won a £750,000 order for 13 48K Argus 700E computers for three exchanges, London, Manchester and Glasgow, for the Post Office's Experimental Packet Switching Service.

Am-Ex now looks at credit card terminals American Express is evaluating a retail-type terminal from Addressograph-Multigraph Data Systems, which can be installed in shops, hotels and restaurants, for online credit card authorisation via the card's magnetic encoding.

Post Office opts for GEC Mark II The Post Office has chosen the GEC MKIIBL processor as the first step in building the new System X all-electronic telephone exchanges to replace the TXE-4 exchanges in which electro-mechanical switches are controlled by hard wired equipment.

Forthcoming Events

London Seminar Programme

27 Sep 2013	100 Years of IBM	Terry Muldoon
	IBM Hursley	David Key
	+ Visit to the IBM Museum at Hursley Park (now fully booked)	
17 Oct 2013	Annual General Meeting at 14:00 then:	
	Ada Lovelace	James Essinger
21 Nov 2013	The EDSAC Replica	Andrew Herbert
12 Dec 2013	The Antikythera device	Tony Freeth

London meetings normally take place in the Fellows' Library of the Science Museum, starting at 14:30. The entrance is in Exhibition Road, next to the exit from the tunnel from South Kensington Station, on the left as you come up the steps. For queries about London meetings please contact Roger Johnson at *r.johnson@bcs.org.uk*, or by post to Roger at Birkbeck College, Malet Street, London WC1E 7HX.

Manchester Seminar Programme

17 Sep 2013	Early Use of PCs for Business Purposes	Lee Griffiths
15 Oct 2013	Enigma, the Untold Story: the Intelligence Chase for V1, V2 and V3	Phil Judkins
19 Nov 2013	Misplaced Ingenuity; Some Dead Ends in Computer History	Hamish Carmichael
21 Jan 2014	The CDC 6600 Computer	Dik Leatherdale

North West Group meetings take place in the Conference Centre at MOSI — the Museum of Science and Industry in Manchester — usually starting at 17:30; tea is served from 17:00. For queries about Manchester meetings please contact Gordon Adshead at *gordon@adshead.com.*

Details are subject to change. Members wishing to attend any meeting are advised to check the events page on the Society website at *www.computerconservationsociety.org/lecture.htm*. Details are also published at in the events calendar at *www.bcs.org.*

Museums

MOSI : Demonstrations of the replica Small-Scale Experimental Machine at the Museum of Science and Industry in Manchester are run each Tuesday between 12:00 and 14:00. Admission is free. See *www.mosi.org.uk* for more details.

Bletchley Park : daily. Exhibition of wartime code-breaking equipment and procedures, including the replica Bombe, plus tours of the wartime buildings. Go to *www.bletchleypark.org.uk* to check details of times admission charges and special events.

The National Museum of Computing : Thursday and Saturdays from 13:00. Situated within Bletchley Park, the Museum covers the development of computing from the wartime Tunny machine and replica Colossus computer to the present day and from ICL mainframes to hand-held computers. Note that there is a separate admission charge to TNMoC which is either standalone or can be combined with the charge for Bletchley Park. See *www.tnmoc.org* for more details.

Science Museum : Pegasus "in steam" days have been suspended for the time being. Please refer to the society website for updates. Admission is free. See *www.sciencemuseum.org.uk* for more details.

Other Museums : At *www.computerconservationsociety.org/museums.htm* can be found brief descriptions of various UK computing museums which may be of interest to members.

Contact details

Readers wishing to contact the Editor may do so by email to dik@leatherdale.net, or by post to 124 Stanley Road, Teddington, TW11 8TX. Queries about all other CCS matters should be addressed to the Secretary, Kevin Murrell, at kevin.murrell@tnmoc.org, or by post to 25 Comet Close, Ash Vale, Aldershot, Hants GU12 5SG.

Sharp-eyed readers will have noticed a change to the front cover of *Resurrection*. The Society has, at last, adopted its own logo. Here's the full version in all its glory

Committee of the Society

Chair: **Rachel Burnett FBCS**: *rb@burnett.uk.net*
Secretary: **Kevin Murrell MBCS**: *kevin.murrell@tnmoc.org*
Treasurer: **Dan Hayton MBCS**: *daniel@newcomen.demon.co.uk*
Chairman, North West Group: **Tom Hinchliffe:** *tah25@btinternet.com*
Secretary, North West Group: **Gordon Adshead MBCS:** *gordon@adshead.com*
Resurrection Editor: **Dik Leatherdale MBCS**: *dik@leatherdale.net*
Website Editor: **Dik Leatherdale MBCS**: *dik@leatherdale.net*
Meetings Secretary: **Dr Roger Johnson FBCS**: *r.johnson@bcs.org.uk*
Digital Archivist: **Prof. Simon Lavington FBCS FIEE CEng**: *lavis@essex.ac.uk*

Museum Representatives

Science Museum: **Dr Tilly Blyth**: *tilly.blyth@nmsi.ac.uk*
Bletchley Park Trust: **Kelsey Griffin**: *kgriffin@bletchleypark.org.uk*
TNMoC: **Dr David Hartley FBCS CEng**: *david.hartley@clare.cam.ac.uk*

Project Leaders

SSEM: **Chris Burton CEng FIEE FBCS**: *cpb@envex.demon.co.uk*
Bombe: **John Harper Hon FBCS CEng MIEE**: *bombeebm@gmail.com*
Elliott: **Terry Froggatt CEng MBCS**: *ccs@tjf.org.uk*
Software Conservation: **Dr Dave Holdsworth CEng Hon FBCS**: *ecldh@leeds.ac.uk*
Elliott 401 & ICT 1301: **Rod Brown**: *sayhi-torod@shedlandz.co.uk*
Harwell Dekatron Computer: **Delwyn Holroyd**: *delwyn@dsl.pipex.com*
Computer Heritage: **Prof. Simon Lavington FBCS FIEE CEng**: *lavis@essex.ac.uk*
DEC: **Kevin Murrell MBCS**: *kevin.murrell@tnmoc.org*
Differential Analyser: **Dr Charles Lindsey FBCS**: *chl@clerew.man.ac.uk*
ICL 2966: **Delwyn Holroyd**: *delwyn@dsl.pipex.com*
Analytical Engine: **Dr Doron Swade MBE FBCS**: *doron.swade@blueyonder.co.uk*
EDSAC: **Dr Andrew Herbert OBE FREng FBCS**: *andrew@herbertfamily.org.uk*
Tony Sale Award: **Peta Walmisley**: *peta@pwcepis.demon.co.uk*

Others

Prof. Martin Campbell-Kelly FBCS: *m.campbell-kelly@warwick.ac.uk*
Peter Holland MBCS: *p.holland@talktalk.net*
Pete Chilvers: *pete@pchilvers.plus.com*

Point of Contact

Readers who have general queries to put to the Society should address them to the Secretary (see page 36 for contact details). Members who move house should notify Kevin Murrell of their new address to ensure that they continue to receive copies of *Resurrection*. Those who are also members of BCS, however, need only notify their change of address to BCS, separate notification to the CCS being unnecessary.

Resurrection is the bulletin of the Computer Conservation Society.
Editor — Dik Leatherdale
Printed by — BCS the Chartered Institute for IT