An experimental paging unit

S. H. Lavington, D. J. Kinniment, and A. E. Knowles

Department of Computer Science, The University, Manchester 13

An Associative memory is described which has been attached to an ICL 1905E computer to provide hardware page-address translation. The resulting paged system is being used for experiments in connection with the design of a large research computer.

(Received March 1970)

1. Introduction

Paging is a technique for easing the store-management problems that arise on large multi-programming computers. In particular, the difficulties of ensuring protection between programs, utilising the maximum amount of physically-available fast store (e.g. core store), and integrating a backing-store into the memory hierarchy, are overcome more easily in a paged environment than in a conventional datum-and-limit machine. This paper describes the connection of an associatively-accessed address-translation unit to a standard ICL 1905E computer in order to produce a paged machine. A general discussion is first presented on the developments in the concept of a paged store, and from this some requirements are deduced for an experimental paged system. A new Associative Store is described which meets this need, and its incorporation in a standard 1905E is then considered.

2. General requirements for page-address translation

The convention adopted for translation is that each user-program generates virtual addresses, acting as though the whole of the addressing-field is available for his private use. An address-translation unit, or Associative Store, is interposed between the central processor and the fast store (see **Fig. 1**), and this associates each virtual-address reference with a page of real store. The size of the Associative Store is limited by cost, and is conveniently divided into lines known as Current Page Registers, or CPRs. The Virtual field of each CPR holds a user's virtual page- (or 'block')-address, to be associated with a real page-address which is held in the Real field of the same CPR. Hardware lockout is provided to distinguish between CPRs loaded with the same virtual address, but referring to different user-processes. Referring to Fig. 1, the operation is as follows. A user-generated address arrives from the CPU, and the most significant page bits are strobed into an interrogate-register. The least-significant bits, specifying the line within a page, bypass the translation process. Association takes place on the page bits and if equivalence is found with the Virtual field of a CPR, the corresponding real page-address is read out into the data-register. It is then concatenated with the line bits, and the complete real address is sent to the fast store. With a limited number of CPRs, a match may not be found during the association process. In this case a 'Non-Equivalence' interrupt is generated, and the operating-system takes steps to locate the whereabouts of the real-address being requested, via reference to Page Tables. A transfer from backing-store may take place. A CPR is eventually re-loaded with the desired virtual/real address combination, and translation may then proceed. The associative hardware is designed to operate much faster than the main core store's access-time.

The Associative Store used on Atlas, the first computer to implement paging (Kilburn, Edwards, Lanigan, and Sumner, 1962), is simpler than the arrangement implied in Fig. 1, in that the real field of each CPR points to a unique fast-store page. More flexibility is obtained, including the facility for users to share common areas of store, if the real field of a CPR may point to any page of real store. This is the case, for example, for an experimental paged 1904 computer produced by ICL West Gorton (Doré, 1968). Another improvement on the basic scheme is to incorporate a process-number in the virtual-address field, so that CPRs relevant to several user-programs may remain set up at any one time. This may be useful in a multi-programming environment, especially where on-line users are being serviced on a time-sharing basis. A 4-bit process-field has been incorporated in an experimental paged IBM 360/40 (Lindquist, Seeber, and Comeau, 1966; Hellerman and Hoernes, 1968). The Atlas system has a hardware marker that is set whenever a page is used, and is able to be reset by software; this assists the page-turning algorithm (Kilburn *et al.*, 1962). It would be advantageous if this marker could be extended so as to include information about whether the particular page has been *altered* or not. Other enhancements that can be made involve the provision of selective Read, Write, or Obey, permission lock-out on each CPR, and provision for detecting the fault-condition of Many Equivalences. These improvements have been incorporated on one or other of the two experimental systems referred to above.

Further experience (e.g., Corbató and Vyssotsky, 1965) has suggested two ways in which paging strategy, and by implication Associative Store hardware facilities, could be further improved. Firstly, it is difficult to choose a fixed page-size that is efficient for *all* job-environments. It would thus be convenient to have some means whereby the page-

size could be varied. The fixed size chosen for Atlas is 512, 48-bit words, whilst Doré chose 1,024, 24-bit words, and 4K bytes was chosen for the 360/40. The Associative Store for the Manchester 1905E is designed to allow page-sizes extending from 16 words to 64K words—(see Section 5). Secondly, it is often convenient for a user and/or compiler to notionally segment a program according to its routine-structure, etc. It would be advantageous if the virtual addressing conventions were able to reflect this segmentation. Such a convention exists informally at compile-time on the Atlas computer, because of the large size (2 million words) of the virtual address space. More formal conventions exist on other systems (e.g., Glaser, Couleur, and Oliver, 1965). In general, each segment may require an individual page-size for optimum efficiency. Further, some segments such as those holding library routines may be required to be shared between different user-processes, and the hardware should facilitate this. The Associative Store for the 1905E allows 64 segments per process, of which 32 may, if desired, be regarded as 'common' segments—(see Section 5). Each segment may have a different page-size. Factors determining the choice of these parameters for the 1905 Associative Store, and some general design-aims, are discussed below.

3. Design-aims for the associative store

The Associative Store for the 1905E was designed in the context of a larger research project, namely the MU5 multi-processor computer, currently being developed in the Department of Computer Science (Kilburn, Morris, Rohl, and Sumner, 1968). The design-aims relate to the specification of the MU5 in four ways.

Firstly, the MU5 will use special-purpose associative circuit-elements (Aspinall, Kinniment, and Edwards, 1968) in the implementation of its CPRs, and in an operand Name Store, and in certain central machine associative buffers. It is thus essential that these new associative elements be proved before incorporating them in the hardware of the MU5. The Associative Store for the 1905E was designed to incorporate these special circuits, and thus to form a means for hardware evaluation.

Secondly, it is eventually required to connect the 1905E into the MU5 multi-processor complex, principally by allowing common access to a large Mass store and certain other local stores. This interconnection is facilitated if the 1905E is made to have an address-structure similar to the MU5. This relates to the third factor—the software of the MU5.

It is desirable that development of an operating system and compilers for the MU5 should proceed concurrently with the logic design. This necessitates a period of simulation or interpretive programming on an existing computer. This phase of development is simplified if the existing computer can be made to approximate the conventions of the MU5, especially with regard to virtual addressing structure. It was therefore desirable to convert the 1905E to a paged machine by the addition of an Associative Store possessing as far as possible the MU5 Virtual-field conventions. An Executive/Operating system for the paged 1905E could then be written that would form a trial run for the eventual MU5 system. Since this new 1905E Executive can be written in the system-program language SPG (Morris, Wilson, and Capon, 1970), notional transformation to the MU5 assembly language, where appropriate, can be relatively easy.

Finally, the store-allocation strategy to be adopted in the MU5 operating system cannot be worked out in detail until more is known about the effects of various page-sizes on different job-mixes, and on different classes of segment within a job. Existing work in this field is sparse (e.g., Joseph, 1970; Fine, McIsaac, and Jackson, 1966); and in order to gather more comprehensive statistics there is a need for a variable page-size system such as is provided by the modified 1905E, to act as a research-tool.

The four general design-aims indicated above were

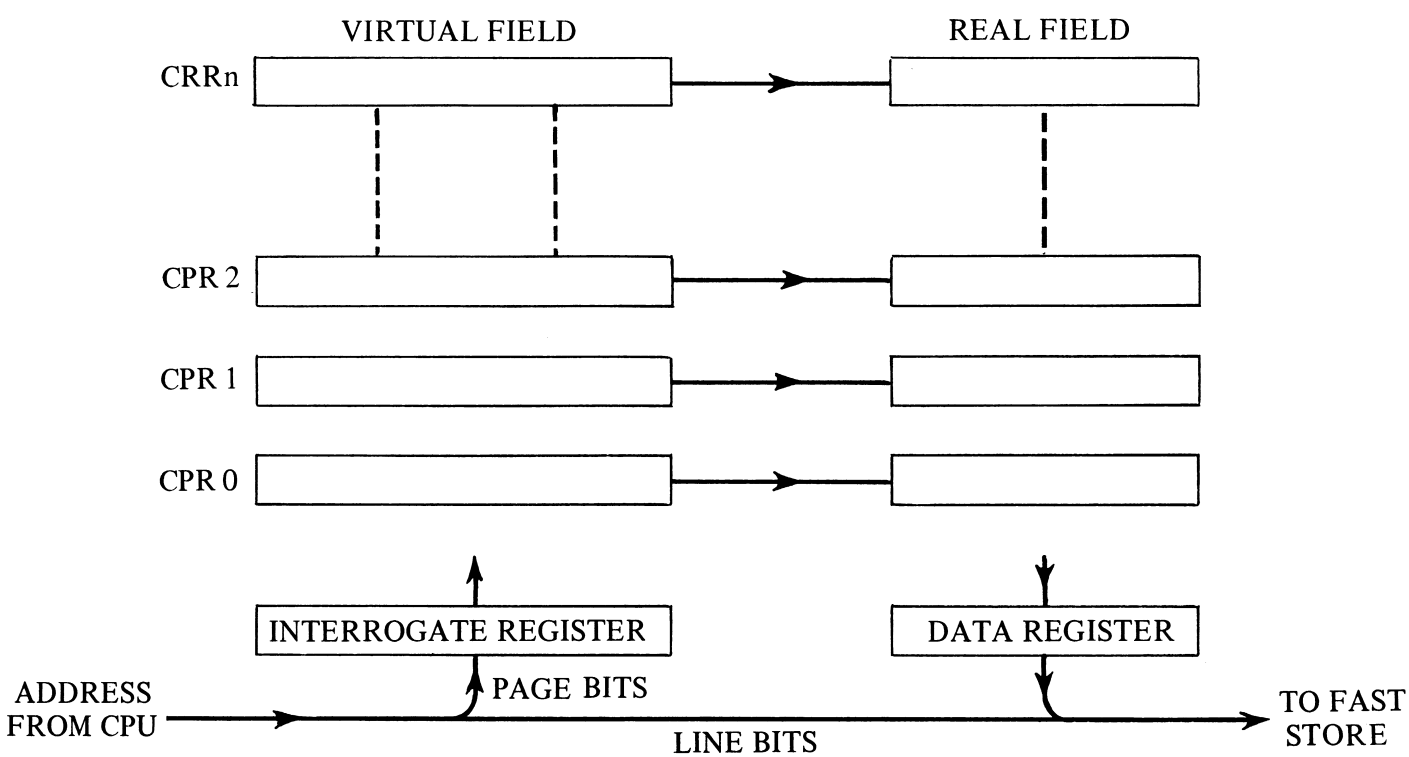

Fig. 1. General scheme for page-address translation

modified by certain practical considerations, as follows. The addition of the Associative Store to the 1905E should involve as little disturbance to the system as possible, so that existing software projects could continue uninterrupted. The central machine hardware modifications that *were* necessary were to be of such a nature that the 1905E could at all times be switched to 'unpaged mode', when it should behave as a standard datum-and-limit machine. To complete the discussion on design-considerations, it is appropriate to give a brief account of the ICL 1905E configuration on which these modifications took place.

The 1905E at Manchester is a 24-bit word-length machine, having 32K words of 650 nanosecond cycle-time core store. It has two exchangeable disc-drives, each disc-cartridge capable of storing 2 million words. Store-protection between user-programs is provided by a hardware datum-and-limit mechanism. The computer is provided with a hardware floating-point unit, and the usual range of basic peripherals. Transfers to and from disc may proceed autonomously, but all other peripherals cycle-steal via the Standard Interface Hesitation sequence. This hesitation sequence, once initiated, is organised by a hardware microprogram which makes reference to a simple control-word area in the Executive area of core store. Peripheral interrupt conditions force an entry to Executive, which examines bits in three special event-registers to determine the cause of interrupt. Other conditions such as 'Datum/Limit Fail' also cause entry to Executive. As is shown in Section 4, the Standard Interface mechanism is utilised for loading and unloading CPRs in the Associative Store, and the interrupt facility is used during action subsequent to a 'Non-Equivalence' condition.

4. Overall system configuration

A block-diagram of the system is shown in **Fig. 2**, in which it can be seen that the Associative Store is interposed between the 1905E central processor and its 32K core store. The Associative Store accepts a 22-bit Virtual address from the 1905E and performs the translation into a 15-bit Real address, assuming that equivalence is found with one of the Virtual addresses previously loaded into the CPRs. There are 32 CPRs in all, each one divided into fields specifying Process Number (IP), Segment (S), Block (B) etc., as described more fully in Section 5. The Virtual field of each CPR extends to 24 bits in all, and the Real field occupies 36 bits. The Real field includes digits specifying access-permission, page-size, etc., as detailed in Section 5. Each CPR may be accessed individually by means of a 5-bit address, denoted by the A-field in Fig. 2. Loading and unloading of the CPRs is performed by Executive, via the Standard Interface hesitation facility, at a maximum transfer rate of about one 6-bit character every 6 micro-seconds. Since the Standard Interface highway is only one character wide, the Associative Store contains a set of buffer registers to facilitate transfers. The sections of these buffer registers which correspond to the Virtual field are also used as an interrogate-register, during the association operation.

As may be seen in Fig. 2, the Virtual field of each CPR is 24 bits wide, whereas the 1905E has provision for only 22 address digits. For this reason the Process Number is pre-loaded into the most significant 6 bits of the Associative Store's interrogation-register just prior to running a particular user's program. The minimum page-size possible with the system is 16 words, so that the least-significant

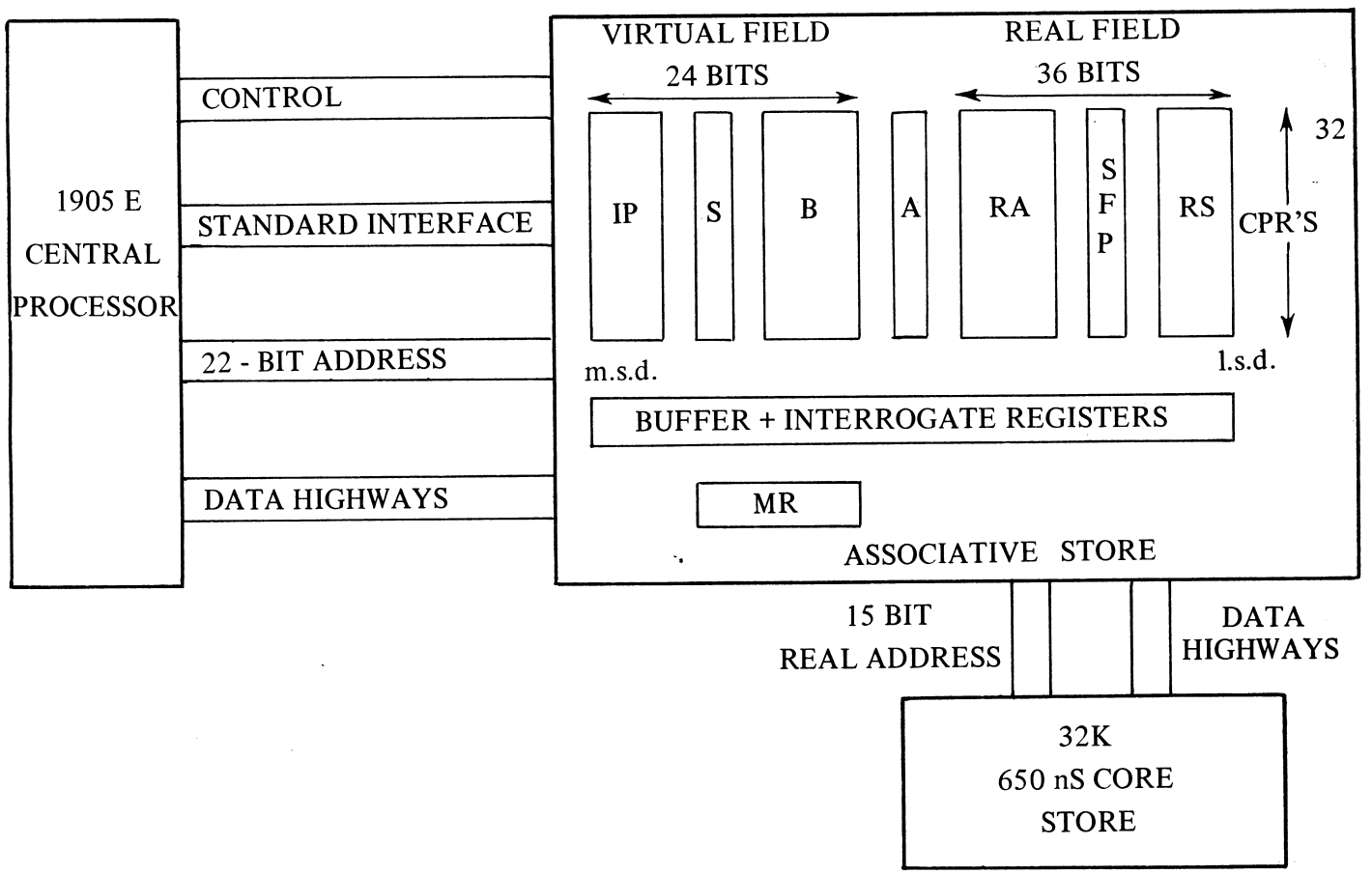

Fig. 2. Block-schematic of the associative store attached to the 1905E

four address-bits of the 22 sent by the 1905E are always treated as line digits and take no part in the association operation. Different segments of a process may have different page-sizes, and these are specified when a CPR is loaded. Thereafter, adjustment of the correct page-size for a particular address-translation takes place automatically, as described in Section 5.

The Real field of each CPR contains provision for up to 22 real-address digits, of which at least four will be line digits, as mentioned above. With the existing 32K 1905E core-store, only the least-significant 15 real-address bits have any relevance; the remainder have been provided for use in the eventual multi-processor environment, when other, remote, stores will require to be accessed. The data-highways, which at the present are connected directly to the 32K store, will likewise be required to be routed to other devices in the proposed multi-processor system.

The highway denoted as 'control' in Fig. 2 contains signals for performing three tasks. Firstly, information is provided about the type (i.e., 'read', 'write', or 'execute') of the current store-access request, so that the Associative Store may check for permission violation and also update the use-bits. Secondly, there is a group of interrupt-signals, used for denoting the occurrence of a CPR non-equivalence, permission-violation, etc. Thirdly, there are signals indicating that the current store-access is required to bypass the CPRs—that is, the address is to be treated as absolute and no translation is necessary. The conditions for which absolute addressing is required are as follows. (*a*) When the 1905E is in Executive-mode. This is a manually switchable option, so that Executive-mode routines can be paged, if eventually desired. (*b*) If the 22-bit virtual address in Executive mode presented to the interrogation-register is less than a manually-selectable amount. This option is provided so that if the main areas of Executive *are* paged, then at least the first few addresses which contain the hardware entry-points for interrupt and extracode sequences etc., can be accessed absolutely. (*c*) When the microprogram uses core locations as temporary dumping registers, during the execution of certain long orders. The locations involved are all within the first 16 words of physical core. (*d*) During all peripheral transfers. These proceed independently of the current process, and hardware provision is made in the 1905E microprogram to facilitate transfers to and from non-contiguous areas of core store.

5. Details of the associative store

Referring to Fig. 2, the 60 digits of each CPR are divided up as follows:

IP : 6 bits :
Four assigned as the Process-number (the 1905E only provides identification for a maximum of 16 current processes); one spare; one used as an Ignore-bit, for CPR lock-out purposes.

S : 6 bits :
Virtual segment-number.

B : 12 bits :
Virtual page (or 'block') number. These digits are each implemented by two associative elements. One element is interrogated when the corresponding virtual address bit is a one and masked when it is a zero; the other is interrogated for a zero and masked for a one. In this way the two elements can be set to give equivalence for both a one and a zero, and allow for 'don't care' cases when page sizes of greater than 16 words are in use. These 'don't care' cases are determined at CPR load time by the setting of the RS register. The maximum possible page-size is therefore 64K words.

RA : 18 bits :
Real page-address.

SFP : 6 bits :
Two bits are assigned for statistical purpose; S_u is set to logical 1 the first time the particular page is used; S_a is similarly set if the page has been altered (written to). F is one marker-bit, employed during the 'Find' operation—see below. The three remaining digits give the desired permission-information, as follows: P_r is set to logical 1 at CPR load-time if *reading* from that page is allowed; P_w is similarly set if *writing* is allowed; P_x is similarly set if processes are permitted to *obey* that page.

RS : 12 bits :
This is loaded by Executive with the Real page size which must be a power of two, in the range 16 words to 64K words. RS is used as a mask when reading out the contents of RA, subsequent to successful association.

In addition to the CPRs there is a 6-bit CPR address-register, A, with one digit spare, and a set of buffer/interrogate registers as described in Section 4 above. Finally there is an 18-bit register, MR in Fig. 2, used as a variable mask during the 'find' operation. The CPR fields of Fig. 2 closely resemble those for the MU5. The main differences are that in the MU5 the segment-bits are increased from 6 to 14, and the real-address field is increased from 18 to 24 bits.

During normal operation there are six possible events which cause the Associative Store to interrupt the central 1905E. These are: (*a*) CPR Non-Equivalence; (*b*) CPR Many Equivalences (implying a system error or hardware fault); (*c*) Read-violation (implying that permission for a read-access had not been assigned to the particular address for which a 'read' was requested); (*d*) Write-violation; (*e*) Execute-violation; (*f*) Store-access counter overflow. This last interrupt comes from hardware that counts the number of paged store-accesses, for statistical purposes. The counter can be manually switched to overflow for counts in the range 128 to 4096, in powers of two. The six causes of interrupt are or-ed together to form a signal known as Pagefail. The central machine action on receipt of a Pagefail is described in Section 6. The cause of interrupt is staticised in the Associative Store, and is cleared upon translation of the next successful paged store-access.

As has been mentioned, loading and unloading of the CPRs take place via the Standard Interface. To facilitate this, and to aid maintenance, 14 Associative Store system commands have been assigned, as follows:

1. Load CPR (*n*)
2. Load Process-number register
3. Set the Ignore-bit for CPR (*n*)
4. Read the SFP bits for all CPRs
5. Load the SFP bits for all CPRs
6. Find. (Set up a pattern in the IP, S and B registers; load a mask in MR; then associate on IP and as much of S and B as is selected by MR. The 'Found' bit is then automatically set for all CPRs which give equivalence.)
7. Resume Normal Association—(issued subsequent to Standard Interface activity)
8. Load the A-register
9. Select the CPR whose address is currently in the A-register, and read its contents

10. Read all working registers
11. Read the interrupt-digits (i.e., Non-Equivalence, etc.)
12. Standard-Interface status-request
13, 14. Spare.

Orders 8 to 11 are for engineering maintenance purposes. In this respect Orders 4 and 6 are also useful, for they enable the Associative Store to be tested on an unpaged machine, using the normal manufacturer's software—(see also Section 6).

The Associative Store provides certain manually-selectable options, as already mentioned. These include switches for inhibiting interrupts, selecting the upper bound to the size of unpaged Executive store-area (see Section 4), and selecting the size of the store-access count. There is also a facility denoted as 'allow common segments'. If this option is switched on, then Process 0 is automatically forced for all virtual addresses which contain a most-significant digit of 1. Thus, the first 32 segments of a user's process are private to his program, and the remaining 32 segments refer to a common area known as Process 0, which could contain library routines, etc. Any user's process may therefore speedily refer to this common area. Finally, the Associative Store is provided with a remote test facility, for off-line engineering maintenance.

The associative memory-elements which form the basis of each CPR consist of special-purpose integrated circuits, each chip holding 8 bits (Aspinall *et al.*, 1968). The remaining logic-elements are implemented in the ECL integrated-circuit technology. The cost per bit of the associative circuits is comparable to that of conventional flip-flops. The performance of the Associative Store, as judged by the translation delay it introduces to core-store references, varies slightly for different combinations of virtual address digits. In the worst case, the elapsed time between receipt of a 22-bit virtual address at the Virtual Field and the sending of a corresponding real-address to the data-register is about 130 nanoseconds. This is illustrated in **Fig. 3**, which shows the Oscilloscope traces for this worst-case translation. When in absolute-address mode (no translation required), the Associative Store introduces a delay of 20 nanoseconds. The effect of these delays on the 1905E instruction-times is discussed below.

6. Interface with the 1905E

(i) *Hardware considerations*

The main group of modifications necessary to the standard 1905E in order to implement paging, concerned the fixed-store microprogram unit. This fixed-store, of 1,024 48-bit words, is responsible for sequencing all machine-code instructions, and for controlling peripheral and interrupt activity. Modifications were performed to ensure that all instructions could be re-started if interrupted by, for example, a CPR Non-Equivalence occurring on an operand-fetch. This involved postponing any irrecoverable changes to central registers, e.g., resetting carry, until such time in each microprogram sequence that the particular order was bound to run to completion. Special problems occurred in the case of some multiple-operand orders, such as the MOVE instruction. This order copies up to 512 words from one area of core to another. With permitted page-sizes as small as 16 words, such orders become impossible to implement directly, if only 32 CPRs are available. Thus, during paged mode, these long orders were made to cause entry to Executive, where they are implemented as extra-codes. Executive itself may still obey the long orders directly, since it is assumed that a prior software check will have been made by Executive to ensure that the transfer areas are in core store. Pagefail Interrupts from the Associative Store, such as Non-Equivalence, were made to cause hardware entry to a microprogram tidying-up sequence, similar to the one entered for a Datum/Limit fail on the standard machine.

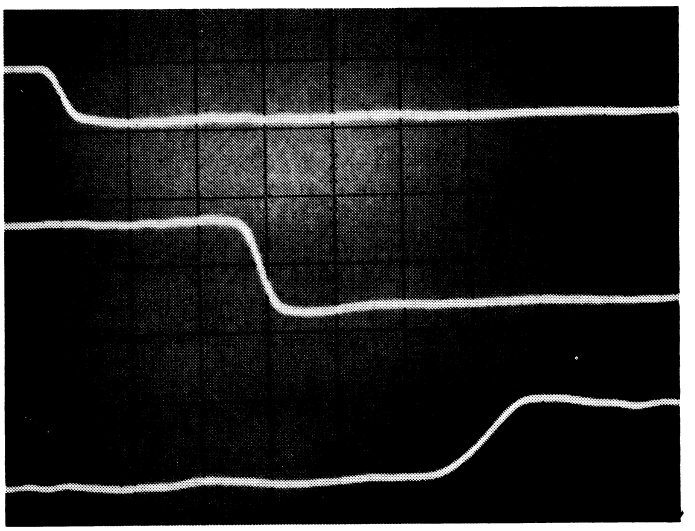

Fig. 3. Address-translation time

The upper trace (*a*) shows a digit of the interrogate-register changing; Trace (*b*) shows a CPR signalling equivalence. Trace (*c*) shows the input of the data-register (real-address) changing. The vertical scale is 1 volt per division, and the horizontal scale is 20 nanoseconds per division.

In addition to the usual preservation of control, floating-point accumulators, etc., a Pagefail causes the virtual address which produced failure to be preserved at absolute core location 1. The Pagefail interrupt microprogram sequence then forces entry to Executive at absolute location 16. All substantial alterations to the fixed-store miroprogram were implemented as conditional branches, entry to them being determined by the state of a manual 'Paging on/off' switch mounted on the main 1905E console. Thus, for 'Paging off', the 1905E performs as a standard datum-and-limit machine. The state of this switch can be examined by Executive.

The other group of engineering modifications necessary to implement paging, concerned the provision of special control signals between the 1905E and the Associative Store. As indicated in Section 2, these provide information about the type of store-access, whether the access is to be paged or treated as an absolute reference, and the type of Pagefail interrupt. Each of the six possible causes of Pagefail appears as a digit in the special CPU event-register SR129, where it can be examined by Executive.

The timing of the 1905E CPU can, with a slight modification, be made independent of core-store access time and so the additional delay introduced by the Associative Store was able to be accommodated. The total delay was effectively greater than the 130 nanoseconds quoted above in Section 5, because of about 60 feet of interconnecting cabling and because of the need to convert between 1905E and ECL logic-levels. These two additional factors produced a further 130 nanoseconds delay. The timing of certain CPU sequences had to be slowed down so as to allow the Pagefail interrupt to travel back to the CPU and force microprogram interrupt-entry at the correct time. The degree to which all these delays will affect the instruction-speed of the paged 1905E depends upon the amount of overlap occurring

between core-store references and central machine activity. For some orders, e.g., the ADD instruction, very little overlap is achieved. Other orders, such as BRANCH on the state of the accumulator, exhibit much overlap. There is a third category, such as DIVIDE, where central machine activity dominates the instruction. The total percentage increase in instruction-time for these three orders, as caused by CPR translation-delay, cable-delays, logic-conversion delays, and central machine re-timing, has been measured to be:

ADD	+ 59%
BRANCH	+ 21%
DIVIDE	+ 7%

A Gibson Scientific-mix test shows that the average instruction-time for the paged 1905E has been increased by about 53%. It should be remembered that Executive and peripheral activity, being in general unpaged, will be slowed down to a lesser degree.

(ii) *Software considerations*

It was required that development of the Associative Store should be overlapped with the use of the 1905E as a standard datum-and-limit machine. Additions had thus to be inserted into the manufacturer's Executive to enable the Associative Store to be controlled as a Standard Interface peripheral. The additional Executive routines were made of a general nature, so that future experimental peripherals could also be handled. The resulting software package (Collins, 1969) has been incorporated into the manufacturer's E6BM Executive. This has enabled a Normal-Mode CPR test-program to be written. To test the detailed paging facilities, an Executive-mode program with dummy Normal-mode processes is available.

The degree to which the Associative Store hardware could have eased the software tasks involved in CPR management, was limited by two original design-criteria, namely: the engineering modifications to the central 1905E should be kept to a minimum; experimental flexibility should be maintained. For these reasons the CPRs cannot be addressed directly, as in Atlas (Kilburn *et al.*, 1962). It was also not appropriate to implement a hardware algorithm for choosing a suitable CPR for re-loading, nor was it desirable to provide automatic table-searching on receipt of a Pagefail interrupt. Thus the Associative Store is intentionally not integrated too closely into the 1905E architecture, and is regarded mainly as a means for facilitating software experiments.

The E6BM Executive, and related operating systems, do not constitute a suitable environment for the range of paging experiments which it was desired to perform. A new system called VIPER—(Virtual Processor Executive Routine)—has thus been written (Morris and Detlefsen, 1969; Detlefsen, Frank, Lane, and Sweeney, 1970). Briefly, this is a forerunner for the Executive on the MU5 computer, and embodies the same general conventions with regard to the use of the common-segment facility, communication between user-processes, and handling of input and output. Viper is written in 1900 SPG (Morris, Wilson, and Capon, 1970), which allows the source-text to resemble the MU5 assembler-conventions whilst retaining 1905E object-code efficiency. At present, page-sizes in the range 128 words to 1K words are permitted. The VIPER system is enabling data to be gathered on the effect of various page-sizes, and the exploitation of a segmented store-structure.

7. Conclusions

An Associative Store has been added to a medium-sized standard computer in such a way that, when used in conjunction with the VIPER operating system, an extremely flexible paged machine results. This paged system is sufficiently close to the MU5 in its virtual storage conventions to be able to be used for MU5 software development and experimentation. In addition, hardware evaluation has taken place concerning the performance of the associative memory elements; this has led to further developments which will enable the time for address-translation to be improved by between 30% and 40%. Finally, the paging unit is facilitating connection of the 1905E into the MU5 multi-processor complex.

8. Acknowledgements

This project was carried out with the aid of an SRC research grant. The authors would like to thank Professor T. Kilburn and all their colleagues in the MU5 design-team for the many helpful discussions that have taken place. Thanks are also due for the co-operation obtained from ICL and Ferranti Ltd.

References

ASPINALL, D., KINNIMENT, D. J., and EDWARDS, D. B. G. (1968). Associative memories in large computer systems, and An integrated associative memory matrix, *Proceedings IFIP Congress*, Edinburgh, pp. D81-D90.

COLLINS, R. J. (1969). Some System Programs for an Experimental Paged Machine, M.Sc. Thesis, University of Manchester.

CORBATÓ, F. J., and VYSSOTSKY, V. A. (1965). Introduction and overview of the multics system, *Proc. Fall Joint Computer Conference*, pp. 185-196.

DETLEFSEN, G. D., FRANK, G. R., LANE, R., and SWEENEY, T. J. (1970). VIPER/1900: Programmers' Reference Manual. (Available from the Department of Computer Science, Manchester University.)

DORÉ, J. (1968). 1904E Paging Feature—Paper No. 1—ICL, West Gorton internal memorandum.

FINE, G. H., MCISAAC, P. V., and JACKSON, C. W. (1966). Dynamic program behaviour under paging, *Proc. 21st National Conference, Assoc. Comp. Mach.*

GLASER, E. L., COULEUR, J. F., and OLIVER, G. A. (1965). System design of a computer for time sharing applications, *Proc. Fall Joint Computer Conference*, pp. 197-202.

HELLERMAN, L., and HOERNES, G. E. (1968). Control storage use in implementing an associative memory for a time-shared processor, *I.E.E.E. Trans. on Computers*, Vol. C-17, No. 12, pp. 1144-1151.

JOSEPH, M. (1970). An analysis of paging and program behaviour, *The Computer Journal*, Vol. 13, No. 1, pp. 48-54.

KILBURN, T., EDWARDS, D. B. G., LANIGAN, M. J., and SUMNER, F. H. (1962). One-level storage system, *I.R.E. Trans. on Electronic Computers*, EC-11, No. 2, pp. 223-235.

KILBURN, T., MORRIS, D., ROHL, J. S., and SUMNER, F. H. (1968). A system design proposal, *Proceedings IFIP Congress*, Edinburgh, pp. D76-D80.

LINDQUIST, A. B., SEEBER, R. R., and COMEAU, L. W. (1966). A time-sharing system using an associative memory, *Proc. I.E.E.E.*, Vol. 54, No. 12, pp. 1774-1779.

MORRIS, D., and DETLEFSEN, G. D. (1969). A virtual processor for real-time operation, *COINS Symposium*, Miami, Florida.

MORRIS, D., WILSON, I. R., and CAPON, P. C. (1970). A system program generator, *The Computer Journal*, Vol. 13, No. 3, pp. 248-254.