Manchester computer architectures, 1948 - 1975

S H Lavington,

Internal Report CSM-182, February 1993

Invited article prepared for a special Manchester Issue of the IEEE Annals of the History of Computing.

Department of computer Science

University of Essex

Colchester

CO4 3SQ

UK

Tel: 0206 872 677

e-mail: lavington@uk.ac.essex

Manchester computer architectures, 1948 - 1975.

S H Lavington,

Department of Computer Science, University of Essex, Colchester.

Abstract

Because of changes in computer technology and terminology, it is often difficult for present-day observers to judge the significance of early digital computer projects. In this paper we follow some architectural themes of interest, as they evolved in the design of three innovative Manchester University computers: the Mark I, Atlas and MU5. Themes such as operand address-generation, instruction formats and memory-management are traced during the period 1948-75. These themes are illustrated by a set of normalised diagrams which may act as an aid to further study of original references.

Keywords

Asynchronous design, caches, index registers, paging, pipeline, Williams tube storage, virtual memory.

1 Introduction

The purpose of this review is to explain in modern terminology the structure of the principal digital computer designs to emerge from Manchester University in the period 1948-75. There are three aims in this analysis:

(a) to examine the contemporary reasons for the introduction of various features which now appear curious or novel;

(b) to compare these features with contemporary developments in other centres of innovation;

(c) to assess the longer-term impact, where it exists, of early Manchester ideas whose influence might be detected on computer architectures of the 1980s and 90s.

The computer design group at the University of Manchester was responsible for five prototype machines during the period under review [1]. The names of these prototypes and their UK industrial derivatives are summarised in Table 1.

University prototype	date of first operation	Industrial derivative	date first delivered to a customer
Mark I	1948*	Ferranti Mk I	1951
Meg	1954	Ferranti Mercury	1957
Experimental transistor machine	1953	Metropolitan Vickers MV 950	1956
Muse (later Atlas)	1962*	Ferranti Atlas	1963*
MU5	1975*	ICL 2980*	1975

Table 1: summary of five Manchester computer projects. Asterisks denote qualifying comments in the text.

In this paper we concentrate on the first, fourth and fifth projects in Table 1 because these are the most interesting architecturally. For completeness, however, the other two machines are briefly described in Appendix A.

Before analysing the architectures of the three computers of principal interest, it is perhaps appropriate to hint at the motivations, personalities, resources, and working environment that characterised the computer design group at Manchester University during the period under discussion. The full story will be found in [1], but we may give a quick sketch that is sufficient to suggest where priorities lay.

From 1947 to about 1952 the Manchester group consisted of two or three key faculty (or equivalent), supported by an equal number of Ph.D. students. They worked with war-surplus components and an enthusiasm for electronic innovation which was inspired by Professor F.C. Williams' notable successes in war- time radar technology a few years' earlier. (See also the article by Mary Croarken in this issue of the Annals). From about 1952 to 1959 there were about four key people, who had by then developed links with the nearby Manchester electrical engineering company Ferranti; these links provided technical support in the form of components and facilities. During this period, systems software and high- level language considerations were beginning to exert an influence. From 1959 to 1968 the group of about eight key University people was supplemented by an equal number of Ferranti hardware and software design engineers who brought access to relatively lavish industrial resources compared with the resources available to other UK Universities. There were consequential pressures to maintain a reliable and responsive computing service to an ever-growing community of internal and external users. Another set of pressures on academic staff was introduced in 1964 when the computer group, which had up to then been part of the University's Department of Electrical Engineering, split off to form the UK's first Department of Computer Science. The first undergraduates arrived in October 1965.

Finally, the period 1968 to 1975 saw a gradual expansion of key academics from about eight to 16 by 1971, in which year the MU5 design team also included about 25 Ph.D. students and 19 engineers on secondment from ICL (the company who had taken over Ferranti's mainframe computer interests). In 1968 the computer design group obtained its first major research grant from public funds (an award of £630,446 from the UK's Science Research Council). Coincidentally, there was a divergence from 1974 onwards in the computer design visions of

the University and ICL. (The ICL viewpoint is ably recounted in [2]). 1975 also saw the effective end of the team leadership of Professor Tom Kilburn, who had arrived in Manchester in December 1946. Tom Kilburn's personal attributes of non-derivative thought, few words and much focussed hard work had characterised the computer design team's efforts for 25 years.

In Section 2 we establish the technology framework within which the main themes of the paper can be presented. Amongst the architectural themes which are then discussed in Section 3 are: operand address-generation, number and use of central registers, instruction formats, context-switching, and memory management. In Section 4 we examine what, if anything, has carried over from 1970s to influence the design of present-day computers.

2 Technology overview

The overall characteristics of the three computers under review are summarised using modern terminology in Table 2. Although the Atlas and MU5 were rated amongst the fastest in the world at the time, the performances are today seen as quite modest. Indeed, the small amount of available RAM for each computer today seems scarcely credible. The Mark I described in Table 2 is the April 1949 machine rather than its June 1948 predecessor which was even smaller; the so-called 'baby machine' that first ran a program on Monday 21st June 1948 had just 32 words of 32 bits each for its main memory, no index registers, manual Input/Output, and a simple seven-function instruction set. The 1949 version of the Mark I given in Table 2 had two B-lines (general-purpose index registers), 128 words of primary memory, and 1024 words of drum backing memory. The production Mark I which was first delivered in February 1951 extended the capacity to eight B-lines, 256 words of primary memory, and 3.75K words of drum storage. The Atlas described in Table 2 is the first production version as inaugurated at Manchester in December 1962; subsequent production Atlases had more RAM. The MU5 is the one-off machine as it existed at Manchester in 1975. Note that, especially in the case of MU5, it is somewhat misleading to give a single value for some of the parameters in the Table. For example, as is shown later, MU5 instructions could be 16, 32, 48 or 80 bits long.

	Mark I	**Atlas**	**MU5**
Peak MIP rate (fixed-point add):	0.0006	0.6	20
Peak FLOP rate (hardware flpt add):	-	0.6	4
Principal computational word length, bits:	40	48	64
Principal instruction length, bits:	20	48	16
Instruction format for most instructions:	1-addr	1-addr	1-addr
Primary memory (RAM) size on prototype, Mbytes:	0.00064	0.096	0.128
Range of directly-addressable locations, Mbytes:	1	6	4096
Virtual memory (hardware addr. translation):	no	yes	yes
Number of general-purpose registers - (see text):	2	128	1

Table 2: overall characteristics of three Manchester University computer designs, expressed in modern terminology. (Note: the production Mark I had eight general-purpose registers).

Each machine in Table 2 occupied a floor-area equivalent to a large room, putting it in the modern category of 'large mainframe'. The photographs of Figures 1 to 3 show the scale. The technology of each machine is now only of specialised interest, but highlights are given in Appendix B for completeness. As always, however, there is no substitute for reading the original papers.

References [3] to [17] give a selection of technical papers for the Mark I, Atlas, and MU5. A modern explanation of the Mark I and Atlas instruction sets will be found in [18]. Atlas is discussed in Bell & Newell [19], along with a reprint of [8]; Bell & Newell introduce Atlas as: 'one of the most important machines described in this book' and comment that 'Atlas was about the earliest computer to be designed with a software operating system and the idea of user machine in mind'. The MU5 hardware and software are covered in considerable depth in the book by Morris and Ibbett [20]. A readily-accessible summary will be found in [21]. The architecture of the ICL 2900 series is described by Buckle [21]; whilst there are some fundamental differences, the architecture of the 2900 series owes much to and has a great deal in common with MU5 [1].

We now discuss five architectural themes of interest, using modern terminology where possible. The accompanying diagrams have been 'normalised' to highlight the development of concepts over the 30-year review period.

3 Architectural themes

3.1 Separate registers and ALU dedicated to address-generation.

The Manchester Mark I used random-access electrostatic storage both for primary memory and for most CPU registers. (The cost per bit was in general cheaper than flip-flop registers). From an engineering view, the natural sub-unit or 'building block' for the Mark I consisted of a Williams Tube (storing say 32 words) and a serial adder. Variations on this basic sub-unit were used for four purposes within the Mark I's CPU (see Figure 4):

the main accumulator and ALU;
a fast multiplier unit;
the collection of index registers (so-called B lines);
the Program Counter and Instruction Register (so-called Control).

This physical separation of computational activity and organisational activity, initially on technological grounds, probably influenced later thinking when it came to providing facilities in Atlas and MU5 for the generation of operand addresses.

In Atlas, programmers could make use of up to 90 of the 128 general- purpose 24-bit registers (B lines) for addressing purposes. It was envisaged that a program might wish to keep the base-addresses and offsets (and bounds) of several data-structures and working areas in a number of registers. The one-address instruction format allowed for double modification: that is to say, the final operand address was the sum of the address-bits in the instruction and the contents of any *two* of the 90 general-purpose registers; (B0 always contained zero). The address arithmetic was carried out in a special fixed-point B-arithmetic unit, whose operation could be overlapped in time with the main 48-bit computational ALU. The physical arrangement is shown diagrammatically in Figure 5. The Atlas B-arithmetic unit could, of course, also be used in its own right for 24-bit fixed-point computation via a set of B-arithmetic instructions.

MU5 also had a separate B-arithmetic unit, used both for address- generation purposes and

32-bit fixed-point computation. However, MU5 took a very different view of general-purpose registers - see also Section 3.2. Address-generation for scalar variables (including the descriptors of structured data) was the responsibility of a Primary Operand Unit (PROP, see Figure 6), which made use of a limited number of implicitly-invoked dedicated registers which held the addresses of the Name Segment, current Name Base, Stack Pointer, etc. For structured data, the MU5 B-arithmetic unit worked in cooperation with a Secondary Operand Unit (SEOP, see Figure 6) which contained registers for holding the current descriptor(s), buffers for holding the elements of a structure, etc. All the main MU5 sub-units illustrated in Figure 6 operated asynchronously. The Instruction Buffering Unit(IBU), PROP, and SEOP all had their own fully-associative caches and were all pipelined internally - (see also Apendix B). Each of the three units also had certain arithmetic and logical facilities appropriate to its function. For example, SEOP was responsible for array bound-checking and for string operations on blocks of data (e.g. string compare, table look-up). Taken overall, it was possible for as many as about 40 MU5 instructions to be in part-completion concurrently, spread out across the IBU/PROP/SEOP/ALU chain.

3.2 Number of central registers and format for instructions.

In common with two-thirds of the fifteen notable early computer projects in Britain and America [23], the Manchester Mark I used a single- address instruction format. The Manchester tendency to separate out address-generation facilities from the main (floating-point) computation (see above) probably helped to keep the Atlas design on this track. However, by the mid-1960s, when the design-requirements for MU5 were being formulated, there were several examples of high-performance 2- and 3-address computers in the market place. The MU5 design team considered several options. The factors that led the team to choose an augmented one- address format are discussed at length in [20]. Some of the factors - eg maximising address-space; minimising the number of operand- and instruction-fetches; aiding efficient pipeline design - apply equally to today's designs - be they CISC or RISC. Other MU5 factors, particularly the need to minimise size of object-code and to ease the high-level compiler-writer's task, are not given so much emphasis today. Since MU5 was asynchronous, popular overall metrics such as number of clock-cycles per instruction did not apply directly. Instead, the designers tended to concentrate on ensuring a good overlap of ‘organisational’ and ‘computational’ activity, as illustrated below for the scalar product loop.

Perhaps the point at which the MU5 designers differed most from today's architects is in their views about the number and purpose of a computer's central registers. Experience with Atlas had suggested that, to quote a catch-phrase of the time, there were only three sensible choices for the number of fast registers in a computer: zero, one, or infinity. For computational purposes, each MU5 instruction specified one 'register' (an accumulator, or sometimes the top-of-stack). For address-generation purposes, each instruction used one of a limited number of special 'registers' (eg Current Name Base; Global Name Base; top-of-stack) *implicitly*, in order to locate a primary operand; this primary operand could be a scalar variable, or it could be one of an 'infinite' number of named structure-descriptors. The most-frequently-used primary operands, be they scalars, descriptors, or the top few locations of the stack, were automatically cached in a 32-line fully-associative fast cache. This employed special CAM chips - (see also Appendix B). Once the address of a secondary operand such as an array-element had been calculated, the eight most-frequently-used long words (128 bits) were cached in another fully-associative 'vector' cache within the SEOP unit in Figure 6. The overall effect of these

arrangements was to lift from the compiler- writer the burden of optimising register use, whether for addressing purposes or for computational purposes.

This may be illustrated by the MU5 code compiled for a scalar product (dot product) sequence written in the block-structured language Algol:

```
FOR i := 1 STEP 1 UNTIL n DO
sum := sum + x[i] * y[i];
```

```
        ACC = 0
        ACC => sum        clear the scalar 'sum'
        B = 1             initialise the index, held in the B accumulator
L1:     B => i            start of the main loop
        ACC = x[B]        load main flpt acc. using named descriptor x
        ACC * y[B]        multiply, using named descriptor y
        ACC + sum         add in the current sum
        B CINC n          compare-and-increment B, setting a status-bit
        IF =/,  -> L1     jump if status-bit indicates not equal to zero.
```

Each of the nine MU5 object-code instructions is a short (16-bit) order. This may be compared with a CYBER 205 FORTRAN object-code sequence for performing the same task, which consists of 18 32-bit instructions and one 64-bit DOTV macro instruction. The MU5 jump-prediction unit and 128-bit wide instruction highway (see Figure 6) helped to reduce instruction-fetching time; likewise, the associatively-accessed primary and secondary operand caches helped to minimise operand-fetch times.

An MU5 descriptor was 64 bits long, including a 32-bit origin address, a 24-bit bound, and option-indicators for element-sizes of 1,4,8,16,32 and 64 bits. Automatic bound-checking was provided. Further details will be found in [20].

Optimising the pipeline throughput during a scalar product loop was taken as an important design-objective for both Atlas and MU5. In one Atlas paper [8], the time for a five-instruction Atlas scalar product loop is quoted as 12.2 microseconds, though it was measured to be 10.95 microseconds at the Manchester Atlas inauguration ceremony on 7th December 1962. A hand-coded 5-instruction version of the six-instruction MU5 scalar product loop given above executed in just less than one microsecond, once at least one cycle of the loop had been executed and hence the two vector descriptors and the scalar control-variable had all migrated to the associatively-accessed cache. The MU5 floating-point multiply time, which was considered relatively slow compared with contemporary supercomputers such as the CDC 7600, accounted for about half of the total loop-time [20]. Note that, unlike some present-day designs, near-optimum MU5 scalar product performance could be obtained without the use of a vectorising compiler and without significant vector start-up overheads.

A final point about both Atlas and MU5 instruction formats: neither machine permitted register-to-register arithmetic in the normal sense. On the (rare) occasions when the result of a computation in (say) the main accumulator was then required in the B-accumulator, pipeline continuity was considerably disrupted.

3.3 Context-switching and procedure calling

It can be imagined that the large number of index registers ('B registers') on the Atlas computer contributed to long delays when program-swapping. However, the switching from user-program to the operating system software was speeded up by two features. Firstly, Atlas had *three* program counters: one for users, one for operating system services, and the third for the 'extracodes' which were library sub-routines in fast ROM (see Appendix B). Secondly, 37 of the 128 B-registers had dedicated system usage and were not swapped on a program-change. Nevertheless, the 90 B-registers that were available to user-programs had to be preserved on major context-switches; this fact biassed the MU5 design team towards the address-generation strategy described in Section 3.1.

In addition, the MU5 design took block-structured languages and recursive procedure-calling more seriously - influenced by the Burroughs B5000. Stack-frame manipulation upon procedure entry/exit was carried out automatically with reference to dedicated Name base and Stack Front registers - actions which today are commonplace. In the mid-70s, however, these features helped MU5 to out-perform rival machines such as the CDC 7600 on Algol benchmarks [17], even though MU5 was slower in terms of raw technology. It should also be remarked that such Algol successes were hollow victories so long as the majority of scientific users continued to program in Fortran!

3.4 Memory management.

The early exposure to difficulties of integrating a small but fast random-access primary memory with a larger but slower spinning memory, engendered an interest that persisted at Manchester throughout the period under review. Memory management was certainly a research issue in the autumn of 1948. From informal conversations with Kilburn, Newman, Tootill and Williams, the four inventors whose names appear on the B-line (index register) patent, it seems possible that the idea of a page- relocation register preceded the discussion of B-lines as a more general vehicle for address-generation purposes.

The challenge of removing from users the burden of organising their own overlays ('drum transfers') was a design-objective in the Mark I Autocode - a very early high-level language that was in use by March 1954 [6]. Atlas took this challenge down to the hardware level, by providing a set of associatively-accessed (content-addressable) page- address registers (PARs, see Figure 5) - which today would be recognised as an address-translation look-aside buffer. (The actual associative look- up was implemented using exclusive-or ('not equivalence') logic gates). If the PARs signalled a page-fault, page-replacement was organised by a 'drum learning program' which resided in fast ROM [8].

The Atlas virtual address space of 6Mbytes was segmented both formally (by hardware) and informally (by compiler-writers). Hardware segmentation distinguished four areas: the user-program space; three operating- system areas, namely the fast ROM, a subsidiary RAM Working Memory, and the so-called V store. This last contained the memory-mapped addresses of all I/O data and control registers and other 'system' registers. In MU5 the notion of a paged, segmented Virtual Memory was taken further, with the address containing process-number as well as segment-number; variable-sized pages were also accomodated by

the associative address-translation hardware - (the Current Page Registers, CPRs, of Figure 6).

Although invisible to the programmer, the MU5 memory hierarchy actually consisted of four physical storage 'units': closest to the ALU came the associatively-accessed special caches called Name Stores - (see also Section 3.2 above); then came the primary RAM, then a larger (but slower) RAM, and finally the fixed-head discs (or drums). A general-purpose autonomous block-transfer unit (STU in Figure 6) was used to move blocks between the two RAMs, via a 16-port time-multiplexed cross-bar switch (the Exchange, in Figure 6). MU5 had three main data-highways between CPU and memory (see Figure 6) - in what might today be called a 'super-Harvard architecture'. The Instruction-Buffering Unit (IBU) had an associatively- accessed jump-prediction memory which helped to ensure profitable pre-fetching of instructions, hence reducing pipeline discontinuities.

3.5 Other points of architectural interest.

(a) The commercial version of the Mark I computer had a hardware random number generator, a requirement specified by Alan Turing, and a 'sideways add' instruction which summed the number of logical 1s in a word. One can imagine how these two instructions could have been of interest to programmers whose field was AI or cryptanalysis.

(b) Both Atlas and MU5 had a hardware instruction-counter, which interrupted (eg) every 2048 obeyed instructions; (the counter's limit was variable up to 2^{16} on MU5. This was used as the basis for charging for computing time; it was also very useful for performance- monitoring purposes.

(c) Both Atlas and MU5 were asynchronous designs. This made hardware fault-finding particularly difficult in cases where the machine simply 'stopped dead', since it was not possible to get a repetitive trace on an oscilloscope. Both computers had a diagnostic facility which was able to generate, at a selectable frequency, sequences of Reset/ Interrupt/ Go pulses which caused the CPU to be forced back and start running from a known initial hardware/software state at periodic intervals. During the MU5 design, the problem of arbitration and the synchronising of asynchronous events was first properly analysed [15].

4 Significance

In studying the history of the design of cars or planes, particular machines are seen to have served particular needs, according to the technological limitations of the age. This is the same with computers. Disentangling generic design-ideas from those ideas which have had only limited time-significance is not easy - especially since computer architecture is still (regrettably) a somewhat ad hoc subject.

If there is one area where the Manchester ideas may have had a lasting influence, it is probably the recurring topic of hardware support for memory-management. By this we mean memory-management in its widest sense - ie the transparent organisation of efficient and safe access to

all the data and instructions required to solve a user's high-level problem. This is a topic on which research is still active today, though the requirements demanded by applications and languages are now rather different from those of 30 years ago. Thus, topics such as persistent stores, object stores, structure stores ('active memory'), etc. are all under investigation in various centres world-wide. Unlike the period reviewed in this paper, however, current research embraces the added challenges of parallel and distributed computing platforms. It has to be said that it was not until the late 1970s and the Dataflow project that computer design at Manchester started to address parallelism in the modern sense of the word.

References.

1 S H Lavington, "A history of Manchester computers". NCC Publications, (National Computing Centre, Manchester M1 7ED, UK), 1975.

2 M Campbell-Kelly, "ICL: a business and technical history". Oxford University Press, 1989.

3 F C Williams and T Kilburn, "A storage system for use with binary digital computing machines". Proc. IEE, Vol. 96, part 2, No. 30,1949, pages 183 ff.

4 T Kilburn, G C Tootill, D B G Edwards and B W Pollard, "Digital computers at Manchesters University". Proc. IEE, Vol. 100, Part 2, 1953, pages 487 - 500.

5 B W Pollard and K Lonsdale, "The construction and operation of the Manchester University computer". Proc. IEE, Vol. 100, part 2, 1953, pages 501 - 512.

6 R A Brooker, "The programming strategy used with the Manchester University Mark I computer". Proc. IEE, Vol. 103, part B, supp. 1-3, 1956, pages 151 - 157.

7 T Kilburn, "Muse". Paper presented at the first International Conference on Information Processing, UNESCO, Paris, June 1959. Proceedings published by Butterworths, London, 1960, pages 433 ff.

8 T Kilburn, D B G Edwards, M K Lanigan and F H Sumner, "One level storage system". IRE Trans on Electronic Computers, Vol. EC-11, No. 2, April 1962, pages 223 - 235.

9 T Kilburn and R L Grimsdale, "A digital computer store with very short read time". Proc. IEE, Vol. 107, Part B, No. 36, Nov. 1960, pages 567 - 572.

10 T Kilburn, D B G Edwards, D Aspinall, "A parallel arithmetic unit using a saturated transistor fast-carry circuit". Proc. IEE, Vol. 107, Part B, No. 36, Nov. 1960, pages 573 - 584.

11 T Kilburn, D J Howarth, R B Payne and F H Sumner, "The Manchester University Atlas operating system: part 1, Internal organisation. Computer Journal, Vol. 4, No. 3, 1961, pages 222 - 225. Part 2, Users' description, Computer Journal, Vol. 4, No. 3, 1961, pages 226 - 229.

12 T Kilburn, D Morris, J S Rohl and F H Sumner, "A system design proposal". Proc. IFIP Congress 1968, Section D, pages 76 - 80.

13 D Aspinall, D J Kinniment and D B G Edwards, "An integrated associative memory matrix". Proc. IFIP Congress, 1968, Section D, pages 86 - 90.

14 R N Ibbett, "The MU5 instruction pipeline". Computer Journal, Vol. 15, No. 1, Feb. 1972, pages 43 - 50.

15 D J Kinniment and J V Woods, "Synchronisation and arbitration circuits in digital systems". Proc. IEE, Vol. 123, 1976, pages 961 - 966.

16 D Morris, G D Detlefsen, G R Frank and T J Sweeney, "The structure of the MU5 operating system". Computer Journal, Vol. 15, No. 2, 1972, pages 113 - 116.

17 S H Lavington and A E Knowles, "Assessing the power of an order code". Proc. IFIP Congress 1977. Published in Information Processing 77, North Holland Publishing Co., 1977, pages 477 - 480.

18 S H Lavington, "The Manchester Mark I and Atlas: a historical perspective". Comm. ACM, Vol. 21, No. 1, January 1978, pages 4 - 12.

19 C G Bell and A Newell, "Computer structures: readings and examples". McGraw-Hill, 1971.

20 D Morris and R N Ibbett, "The MU5 computer system". Macmillan, 1979.

21 R N Ibbett and P C Capon, "The development of the MU5 computer system". CACM, Vol. 21 No.1, Jan. 1978, pages 13 - 24.

22 J K Buckle, "The ICL 2900 series." Macmillan, 1978.

23 S H Lavington, "Early British computers". Manchester University Press, 1980. (Published in the USA by Digital Press).

24 T Kilburn, D B G Edwards and G E Thomas, "The Manchester University Mark 2 digital computing machine". Proc. IEE, Vol. 103, Part B, supp. 1-3, 1956, pages 247 - 268.

25 Prof. T Kilburn - private communication, 1974. For an earlier view of the penalties of floating-point in respect of memory capacity and ALU complexity, see the famous 1946 Burks/Goldstine/von Neumann report reproduced as Chapter 4 of reference [19] given above).

26 K Lonsdale and E T Warburton, "Mercury: a high-speed digital computer". Proc. IEE, Vol.103, Part B, Supp. 1-3, 1956, pages 483 -490.

27 T Kilburn, R L Grimsdale and D C Webb, "A transistor digital computer with a magnetic drum store". Proc. IEE, Vol. 103, Part B, supp. 1-3, 1956, pages 390 - 406.

Appendix A: Brief description of the second and third Manchester computer projects - (see Table 1).

The Manchester Mark II, or Meg, project spanned the period approximately 1951 to 1955 [1]. Meg had the same 40-bit word length as the Mark I, with the same one- address instruction format and the same arrangement of two 20- bit instructions per word. Meg (the 'megacycle machine') was planned as a faster, more compact, easier-to-maintain version of the Mark I. Meg's serial arithmetic unit had a one microsecond digit period, but *parallel* access was made to its random-access main memory via a 10-bit wide highway. (The Mark I was entirely serial). From today's viewpoint, an interesting architectural feature is Meg's incorporation of a hardware floating- point arithmetic unit - one of the first electronic machines to have this

facility. Meg used a 30-bit mantissa, 10-bit exponent, and base of two [24]. It has been said [25] that the Mark I programmers were at first hostile to the suggestion that the new machine should incorporate hardware support for floating-point arithmetic because, as mathematicians, they did not care to be limited to a fixed-format arrangement regardless of their application. Although floating-point accuracy is still an issue today, most users now take hardware support for granted.

The Ferranti Mercury [26], the commercial derivative of the Manchester University Meg, would be rated according to present-day standards as a 0.017 MIP machine. Its market rival, the more expensive IBM 704, was faster at about 0.04 MIPs [1]. At least four Mercury computers were still working in 1970.

The third entry in Table 1, the experimental transistor computer project, spanned the period approximately 1952 to 1955 [1]. This was a relatively slow and small-scale machine compared with Meg. It used a magnetic drum ('fixed-head disk') for all storage purposes, including the main computational registers [27]. The principal impact of this computer project at Manchester was that it gave invaluable experience in the design of logic circuits which used the new 'crystal triodes', better known as transistors. In 1953 the available transistors were germanium point-contact devices, of modest reliability. In the commercial derivative of the University project, the Metropolitan- Vickers MV950 computer, the logic circuits were converted to use germanium junction transistors which had by then begun to replace point-contact devices. The MV950, of which six were manufactured, was one of the first commercially-available transistor computers. Contemporary transistor- based digital computer design took place at the UK Atomic Energy Authority's Harwell Lab. (CADET), at the MIT Lincoln Lab. (TX-0), and at the Philco Corporation (TRANSAC S-1000) - see [23].

Appendix B: additional hardware details of the Mark I, Atlas and MU5.

The Mark I used thermionic vacuum tubes (mostly EF 50 pentodes) and thermionic diodes in its serial ALU. The digit period was 8.5 microseconds. Both the RAM primary memory and all central registers (ie double-length accumulator, multiplier register, instruction register, program counter, and the two index registers) were implemented as CRT electrostatic storage ('Williams Tubes'). In a survey of six British and nine American pioneering computers which came into operation during the period 1948 to 1953, five used Williams Tubes for main memory [23]. The Mark I's serial addition time of 1.8 milliseconds was one of the slowest; in comparison, the MIT Whirlwind computer was probably the fastest machine with a 49 microsecond parallel add time.

The Atlas used germanium junction transistors (mostly OC107) and germanium semiconductor diodes for the main logic circuits in its pipelined ALU, with a special symmetrical surface-barrier transistor (the SB240) being employed in the parallel adder to give an exceptionally-short carry- propagation delay (for those days) of 200 nanoseconds. Most of the 128 general- purpose registers were held in a fast, 0.7 microsecond cycle-time, magnetic core store. Operating system subroutines and some frequently-used library subroutines were held in a special 48Kbyte read-only memory made from a woven wire mesh into which were inserted very small rods of copper or ferrite; the access time was 0.3

microseconds. The main memory (core) had a cycle-time of 2 microseconds and was four-way interleaved. The memory hierarchy is illustrated schematically in Figure 5.

Atlas was a pipelined machine. Its system of autonomous sub-units, interleaved memory-banks, vectored interrupts and memory-mapped I/O would be readily recognised by the designers of today's microprocessors.

Both Atlas and its successor, MU5, were asynchronous machines - (ie no system clocks). MU5 employed ECL SSI and MSI integrated circuits. Special tunnel diode flip-flops with high gain-bandwidth product were employed as arbitration circuits in the interfaces between asynchronous sub-units [15]. MU5 was highly pipelined - (see also Section 3.1). For example, the Primary Operand Unit of Figure 6 had a five-stage, 50 nanosecond per stage, pipeline.

The basic gate-delay of MU5's Emitter-Coupled Logic family was about 2 nanoseconds, to which should be added perhaps another two nanoseconds to account for typical interconnection and loading effects. The overall working figure of 4 nanoseconds per gate was reckoned to be slightly longer than the equivalent figure for the contemporary CDC7600 [20], which used bipolar transistor logic. A unique feature (at the time) for MU5 was its use of integrated-circuit Content-Addressable (ie associative) memory. The CAM used was a Manchester design, manufactured by Ferranti's silicon foundry. Each CAM chip contained four lines of two bits each, with an association ('read') time of five nanoseconds [13]. These CAM chips were used in the MU5 address-translation unit (the CPRs of Figure 6), and in each of the fully-associative caches within the IBU, PROP and SEOP sub-units of Figure 6. For example the PROP cache, called the Name Store for reasons described in Section 3.2, had 32 lines or entries. Each line contained a named variable or descriptor having a 19-bit address-part made up of 4 process bits and 15 offset bits, and a 64-bit value part. The address-part was held in CAM chips and the value part in RAM chips.

Photograph Captions

Figure 1: *The Manchester University Mark I computer in 1949.*

The control panel is situated in the fourth rack from the left. Despite its primitive appearance, the prototype Mark I performed useful computation on number theory and ray tracing before being closed down in the summer of 1950 to make way for the installation of the production Mark I.

Figure 2: *The Manchester University Atlas computer in 1965.*

The CPU and primary memory were housed in five cabinets, of which two are shown in the photograph. The cabinet in the centre, partly obscured by the operator's head, held 48 K bytes of fast ROM containing the 'Extracodes'. The one-inch magnetic tape decks in the background used fixed-block transfers which allowed reading/writing in the forwards or backwards direction.

Figure 2: *The Manchester University MU5 computer in 1974.*

One logic 'door' in the central block of cabinets has been swung open to reveal a set of six 12-layer printed-circuit 'platters' or mother-boards. Each mother-board had up to 200 small 'daughter-board' plug-in modules, on which were mounted the ECL chips. The MU5 CPU occupied about 66 mother-boards.

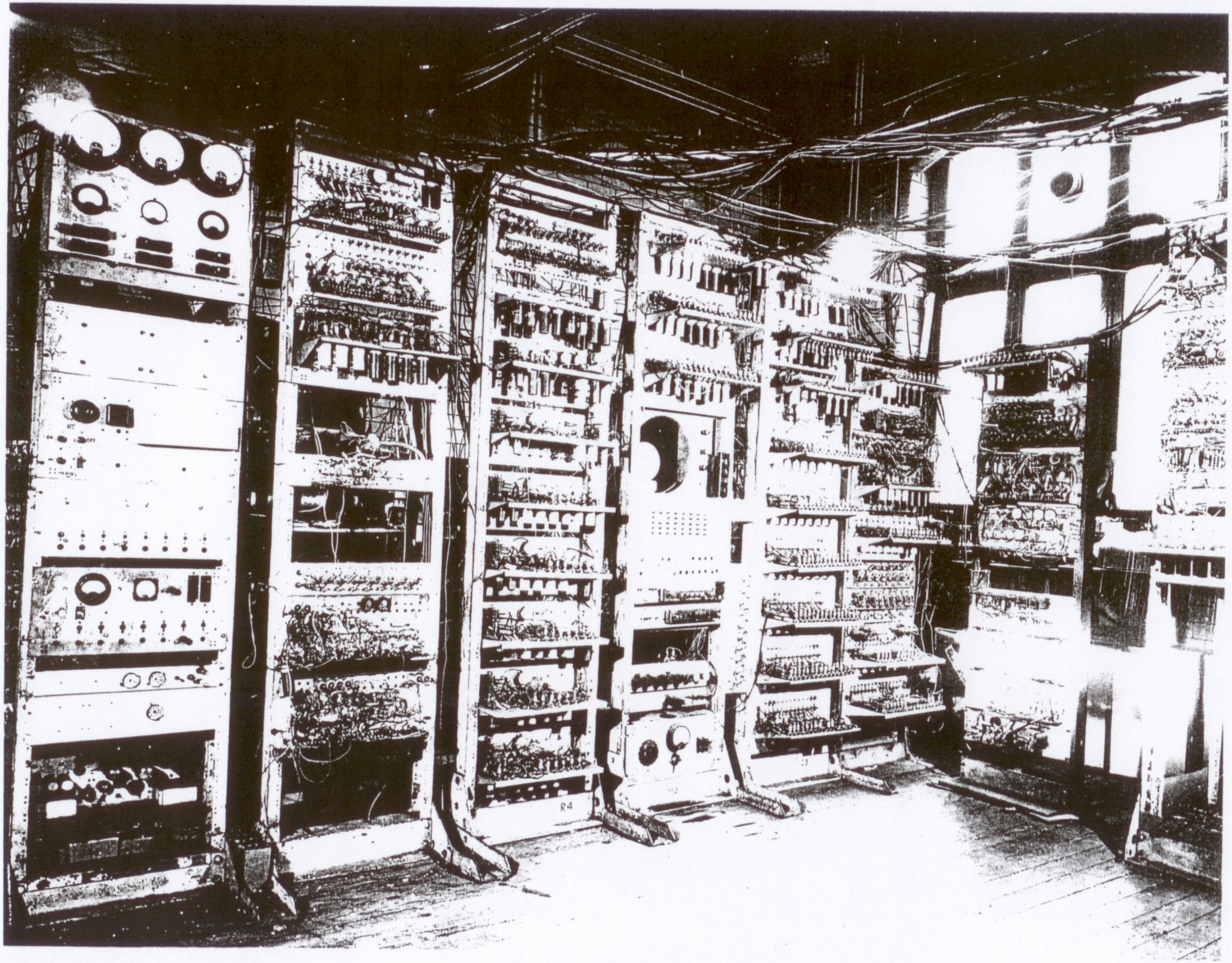

FIGURE 1.

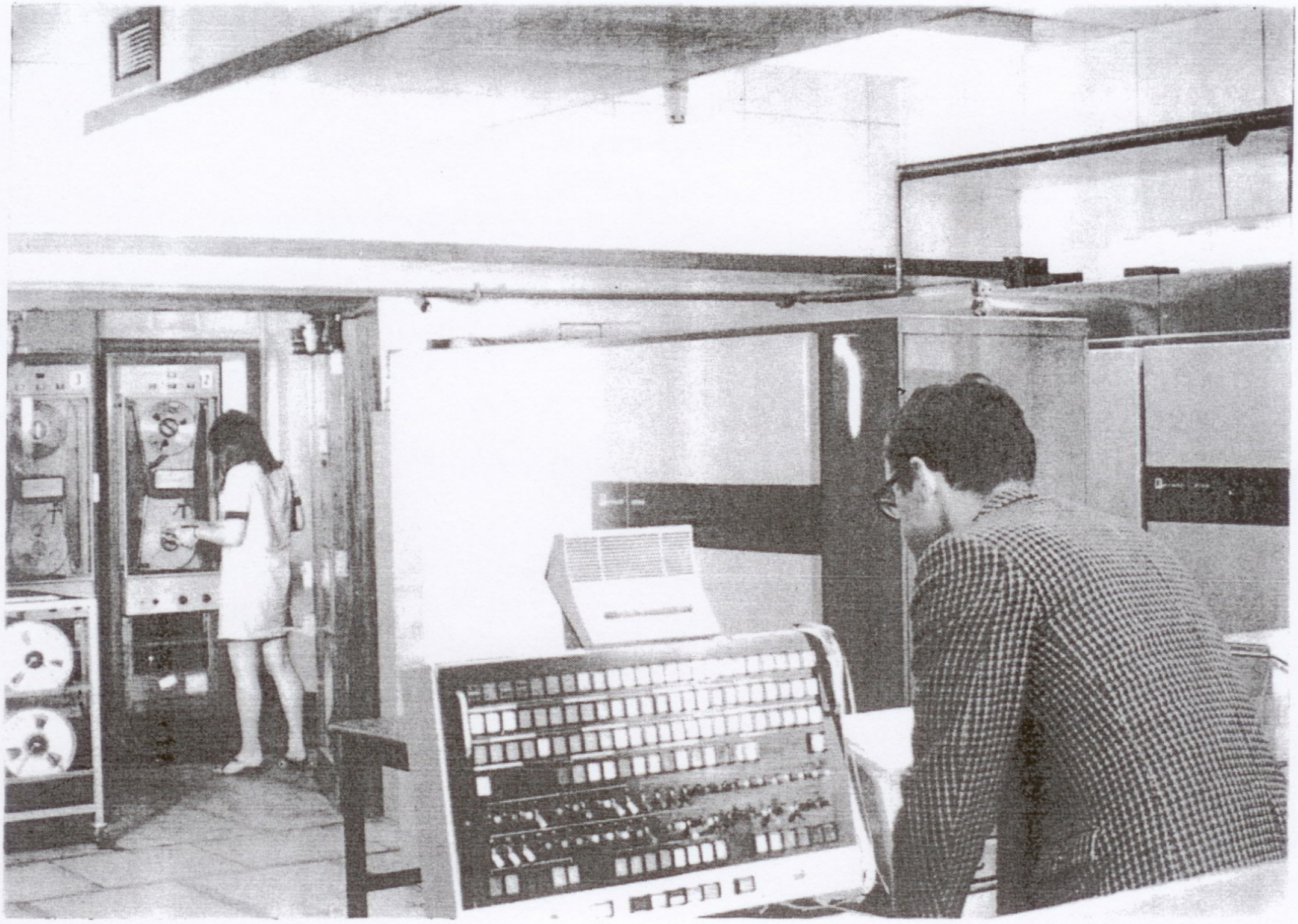

FIGURE 2.

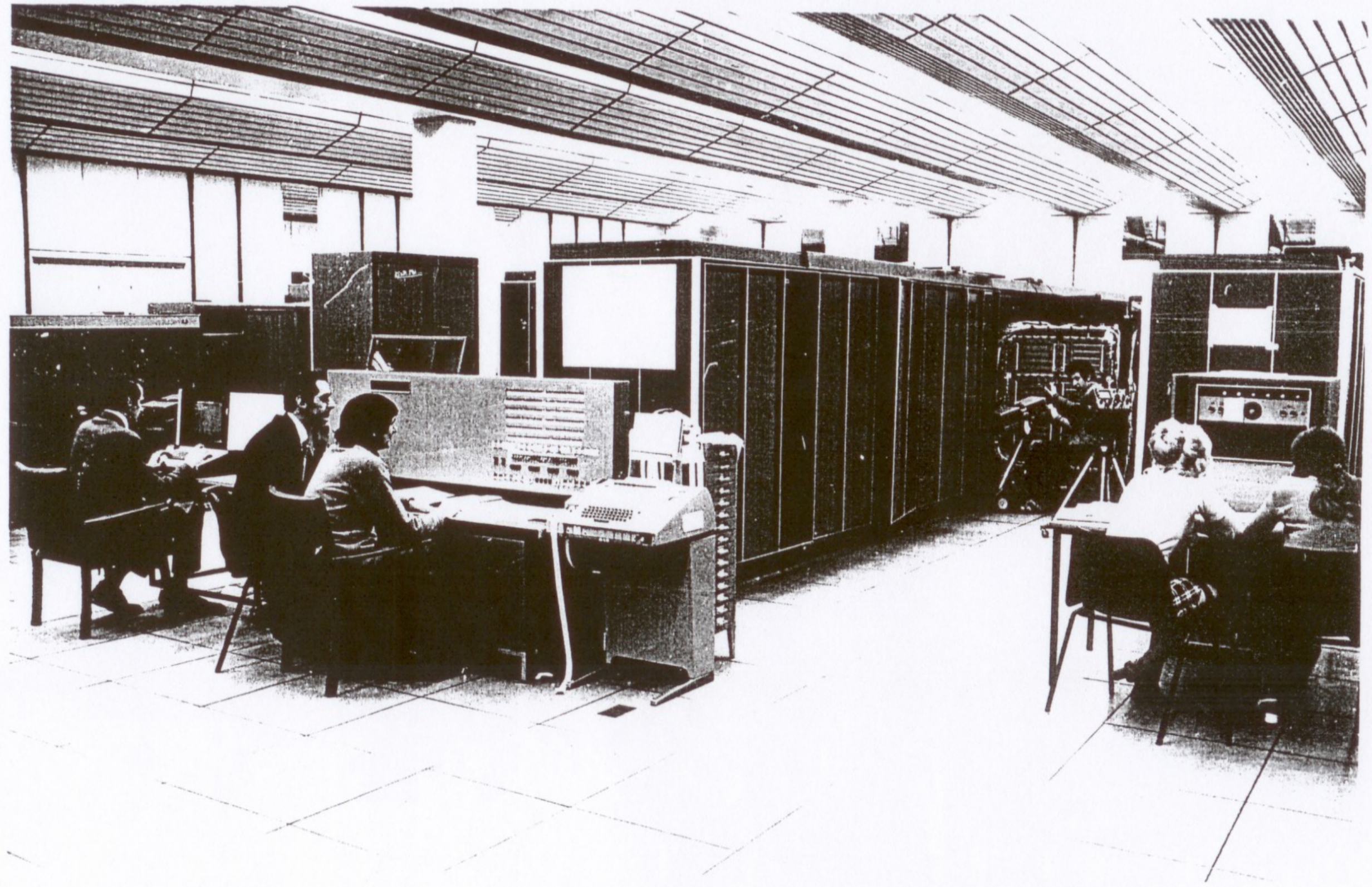

FIGURE 3.

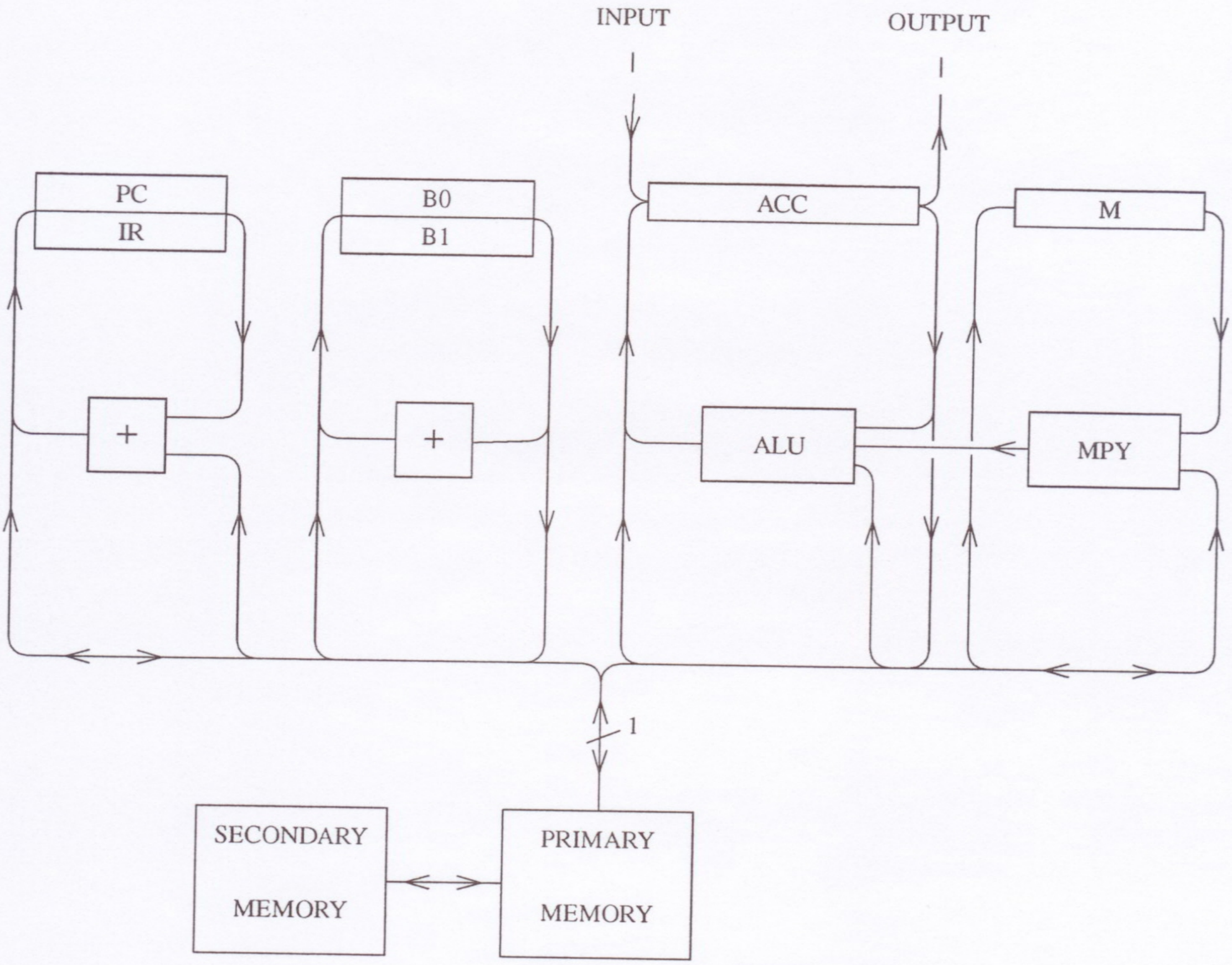

Figure 4: Simplified block-diagram of data-paths within the Manchester Mark I. Key: PC = program counter; IR = instruction register; B0, B1 are B lines (index registers); ACC = double-length accumulator; MPY = multiplier. (The 1949 prototype Mark 1 had two B-lines; the 1951 production Mark I had eight B-lines).

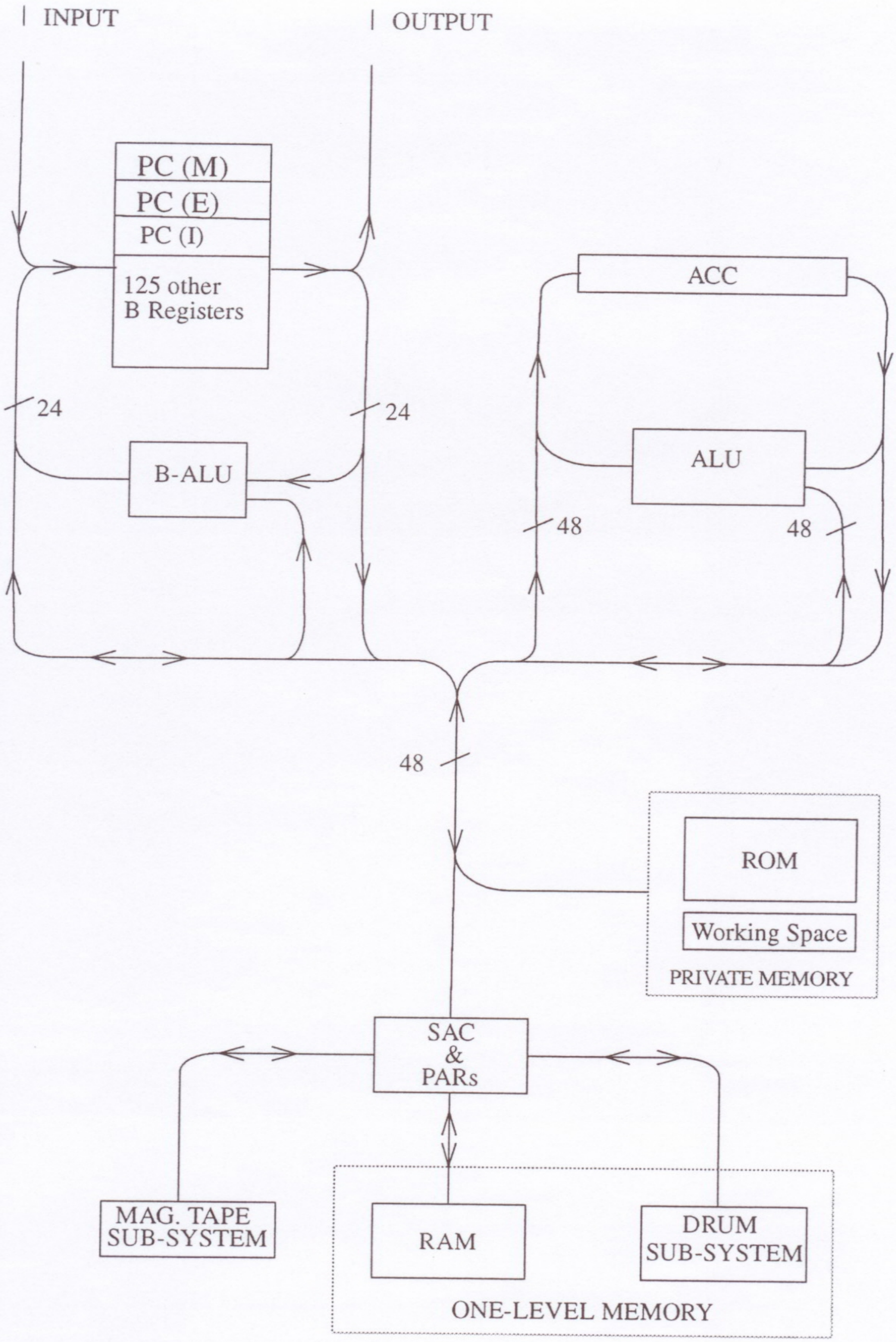

Figure 5: Simplified block-diagram of data-paths within the Manchester ATLAS Computer. Key: SAC = Store Access Control; PARs = Page Address Registers.

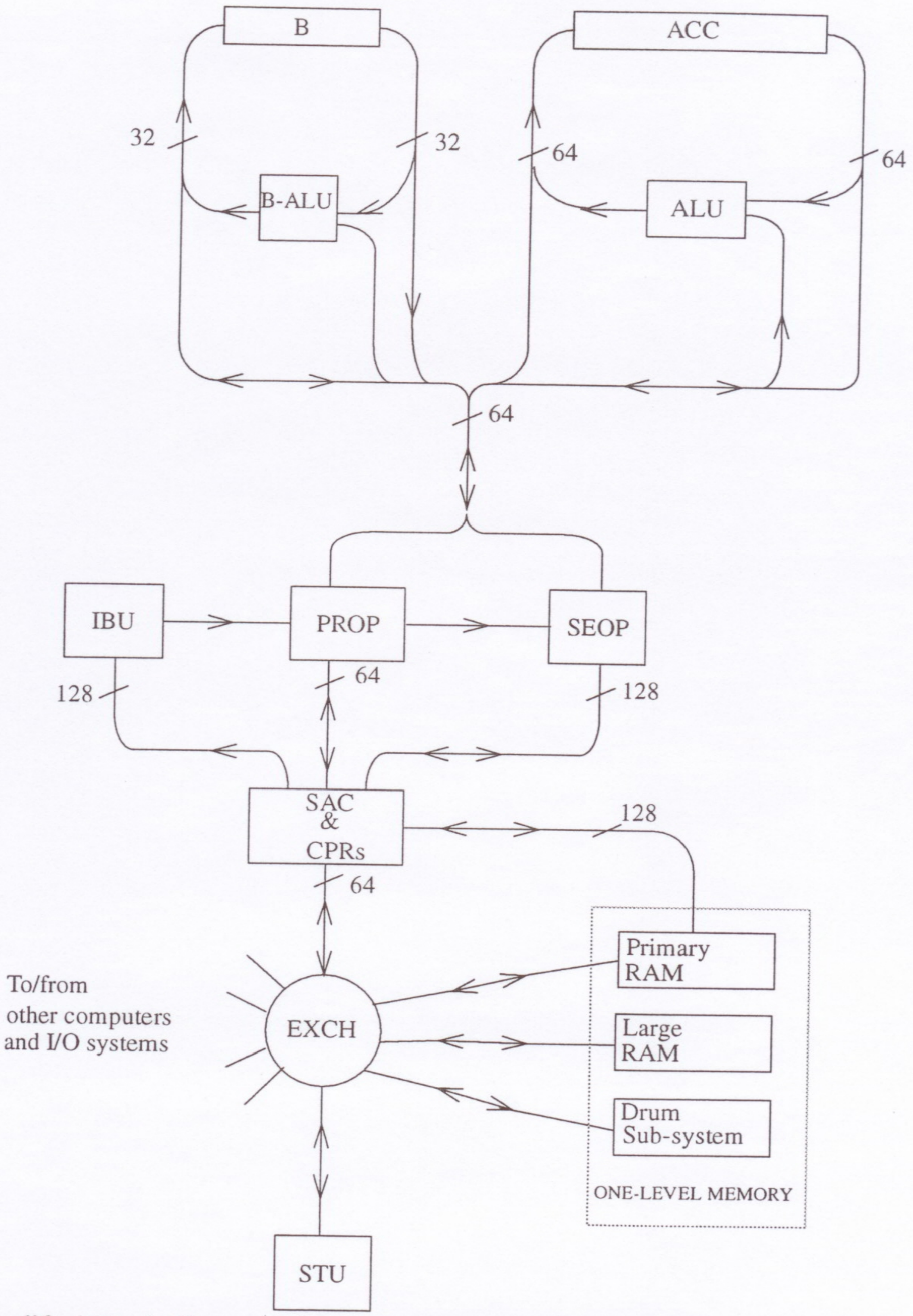

Figure 6: Simplified block-diagram of data-paths within the MU5 computer system. Key: IBU = Instruction Buffering Unit; PROP = Primary Operand Unit; SEOP = Secondary Operand Unit; CPRs = Current Page Registers; EXCH = Exchange (a time-multiplexed Cross-bar Switch); STU = Store-to-store autonomous transfer unit. (Note that there was actually no direct route from PROP to the Main ALU, though there was a route back; the only way into this ALU was via SEOP).