University of Edinburgh

Department of Computer Science

The Evolution of the Fred Machine

by

Gordon Brebner and Fred King

Internal Report

CSR-246-87

September, 1987

James Clerk Maxwell Building,
The King's Buildings,
Mayfield Road,
Edinburgh,
EH9 3JZ.

The Evolution of the Fred Machine

Gordon Brebner and **Fred King**
Department of Computer Science
University of Edinburgh
The King's Buildings
Edinburgh EH9 3JZ
Britain

Abstract

The history of the Fred Machine (FMAC), a flexible computer designed in the Department of Computer Science at the University of Edinburgh, is reviewed. A main intention is to illustrate the progress of an advanced computer system design under the influence of the constraints and demands of an academic computing environment; therefore, technical detail is minimised.

The report is organised so that the reader can follow the evolution of the FMAC over the last seven years. Section 2 briefly outlines the state of relevant work which had been pursued in the Department in the period immediately prior to the FMAC project. Section 3 describes the central feature of the FMAC, the system bus, which was designed to allow the interconnection of a wide variety of hardware modules. Section 4 discusses the range of hardware and software which has been provided as part of the project work; in the main, this has been directed towards the development of a powerful single-user workstation called "the Advanced Personal Machine" (APM). Section 5 discusses how the resulting systems have been used for research and teaching, and mentions some of the additional hardware and software which has emerged from such activities. Finally, Section 6 draws some conclusions from the way in which the project has progressed, and comments on the future prospects of the FMAC.

The Evolution of the Fred Machine

Gordon Brebner and **Fred King**
Department of Computer Science
University of Edinburgh
The King's Buildings
Edinburgh EH9 3JZ
Britain

1. Introduction

This paper reviews the history of the Fred Machine (henceforth abbreviated to FMAC), a flexible computer designed in the Department of Computer Science at the University of Edinburgh. A main intention is to illustrate the progress of an advanced computer system design under the influence of the constraints and demands of an academic computing environment. Accordingly, the paper is not rich in technical detail, which may be found elsewhere [1].

The FMAC project has been active since 1981, and is now in a position in which further major development is unlikely to take place. In common with many pieces of academic work involving computer systems, it has occupied an uneasy niche between the areas of research and development. At the outset, the aim was to produce a flexible architecture to facilitate experimentation with sophisticated computer modules. However, the subsequent evolution of the FMAC has been such that much work has been done with the dual aims of providing a better computing service to the Department, and of enhancing practical educational facilities. Because of this, no research funding has ever been sought for the project (although other research has made use of its products).

The FMAC project involved an extension of the traditional areas of expertise in the Department, namely software and theory, into hardware and firmware. It is reassuring that the products of the project have, in general, achieved goals such as modularity and flexibility which are associated with the former "higher level" activities. In this respect, the project has achieved an objective worthy of a research project, namely the design of a computer system using structured principles from the lowest level upwards. The disappointing shortcoming of the project has been a marked lack of provision of sophisticated system software to complement the hardware facilities. Innovation in this area has largely been delegated to the authors of application programs.

The remainder of the paper is organised so that the reader can follow the evolution of the FMAC over the last seven years. Section 2 briefly outlines the state of relevant work which had been pursued in the Department in the period immediately prior to the FMAC project. Section 3 describes the central feature of the FMAC, the system bus, which was designed to allow the interconnection of a wide variety of hardware modules. Section 4 discusses the range of hardware and software which has been provided as part of the project work; in the main, this has been directed towards the development of a powerful single-user workstation. Section 5 discusses how the resulting system has been used for research and teaching, and mentions some of the additional hardware and software which has emerged from such activities. Finally, Section 6 draws some conclusions from the way in which the project has progressed, and comments on the future prospects of the FMAC.

2. Background

In the late 1970s, a significant amount of work had been carried out in the Department of Computer Science at Edinburgh on the development of various computer systems. A number of projects had come to fruition around 1980, and so it was not surprising that a collection of people focussed upon the FMAC project in order to apply their skills further. Thus, past work had a significant impact on the course of the project; the main strands are described in the next four subsections.

2.1 The "Micro Kits"

The micro kits [2] had been produced mainly as hardware teaching aids, and provided a breadboard upon which an assembly of chips could be placed, interconnected, and used experimentally. In particular, various microprocessor chips (Motorola M6802 and M6809, and Zilog Z80) could be incorporated; programs for these were stored in EPROMs, which could be written and erased as required. The micro kit can be seen as the hardware progenitor of the FMAC; while it allowed experimentation with the contents of one circuit board, the FMAC was designed to allow easy experimentation with a collection of circuit boards, via a flexible system bus under the overall direction of a control board. Therefore, the main intended use of the FMAC was to assist design work at a higher level of hardware functionality. (It is interesting to note that the micro kits are now mainly used for testing VLSI chips designed in the Department - technology now allows rich functionality at the chip level.)

2.2 ELAN

The immediate interest aroused by the development of the Ethernet Local Area Network (LAN) at Xerox PARC [3] had led to the proposal of two alternative designs for a contention bus network to be used within the Department. By 1980, one design had been preferred, and preliminary work was being done on cabling the whole Department, and producing interfaces to allow various computers to use it. In general, the Edinburgh LAN (ELAN) [4] had similar properties to Ethernet, the main operational difference being that it ran at 2 MBits/sec, rather than 2.94 MBits/sec. Given this communications infrastructure, it was an important goal to interface the FMAC to it, with a view to subsequent resource sharing.

2.3 ISYS, LEGOS and the Filestore

Within the Department, minicomputer provision was mainly concentrated upon Interdata series 70 minicomputers. The Interdata System (ISYS) [5] had been written to enable the use of clusters of disc-less Interdata 74s as limited workstations, backed up by more powerful Interdata 70s with disc storage. With the advent of further disc-less minicomputers of various kinds, the Filestore [6], a file server based on an Interdata 70, had been produced. It communicated (via point-to-point links) with minicomputers using a protocol which allowed the reading and writing of files, either as a whole or block-wise, together with elementary directory operations. The LEGO System (LEGOS) [7] had been written; based on ISYS, it enabled a disc-less Interdata 70 to be used as a workstation, with the Filestore as a storage facility. Importantly (as the name may suggest to a reader familiar with children's toys), it had been designed to allow users to tailor the system to their own needs by using appropriate software building blocks. This philosophy was important in determining both hardware and software goals of the FMAC project.

2.4 EMAS and IMP

The Edinburgh Multi-Access System (EMAS) [8] had been developed at the University of Edinburgh during the 1970s. It is a time-sharing operating system for large mainframes (then, the ICL 4-75 - an IBM 360 simulacrum - and currently, the ICL 2900 and Amdahl 470). While many features of EMAS were not applicable to smaller-scale systems based on the FMAC, its guiding principle of well-structured system software organisation was applicable, in particular the policy of writing all software in a block-structured high-level language. The Implementation Language (IMP) [9], an Edinburgh-designed derivative of Atlas Autocode, had been used to implement EMAS. Since languages such as C and Pascal were not widespread at that time, IMP had become the usual language for all programming in the Department. Although doubts about the wisdom of concentrating on a parochial programming language were beginning to surface when the FMAC project started, the main initial software goal was a reliable IMP compiler and run-time system.

3. The Fred Bus

The central feature (and, indeed, the only characteristic feature) of an FMAC is the common system bus, the Fred Bus. An FMAC consists of a Fred Bus with slots for up to nineteen double-height extended-depth Eurocards and a power supply (which also contributes a mains-derived clock signal and a power failure signal to the bus), contained within a tinted-perspex fronted cabinet which lends it the appearance of a microwave oven. The Fred Bus was designed to allow experimentation involving the interconnection of sophisticated modules built on circuit boards, as mentioned in Section 2.1. To achieve this aim, it was intended to have sufficient capacity and flexibility to handle not only current modules, but also those which were likely to be available in the future. Further, the specification was deliberately kept simple, in order to allow new modules to be introduced easily and quickly.

Therefore, the general approach in designing the Fred Bus was to give a minimal electrical specification which allowed modules to communicate reliably, leaving other issues to be dealt with by software convention. The specification supports transfers of (up to) 32-bit wide data for a collection of heterogeneous processors in a byte-oriented 32-bit address space. At the time the FMAC project started, the normal microcomputer data bus width was 8-bit or 16-bit, and so the Fred Bus was anticipating future microprocessor developments; standards for 32-bit buses, such as the IEEE Futurebus project, were in their infancy [10]. The multi-processor capability was specified, not because of a desire for high performance parallelism, but because of a desire for flexibility, since a variety of processors would enable a wider selection of software to be used. This feature is reflected in the current trend towards producing add-on processor modules for standard microcomputers, such as the IBM PC. The fact that processors are closely coupled by the Fred Bus means that the use of shared modules, such as memories, is possible; further, processors can act as servers to other processors on the bus, allowing such things as terminal and network access, as well as supervisory control functions. Therefore, a new processor can be incorporated by producing a simple interface to the Fred Bus, in contrast to the complexity of producing the complete subsystem which would be needed to add a processor to a more loosely coupled standard microcomputer system.

The only operation performed by the Fred Bus is the transfer of data to or from an addressed "slave" under the control of a "master". A master (such as a processor) supplies address and direction of transfer signals before asserting a transfer request signal. Once the addressed slave (such as a memory) has completed the transfer, it responds with a transfer acknowledge signal. This asynchronous handshake protocol accommodates different operating speeds of both masters and slaves. The theoretical limit of operation is around five million transfers per second but, in practice, processor and memory speeds limit this to about three million. There are separate byte strobes so that the master can request 8-bit, 16-bit or 32-bit transfers. The polarity of signals on the data bus is not specified; the only requirement is that memories should present data in the same polarity as originally stored. A slave can signal an unsuccessful transfer, such as when a memory parity fault occurs; this mechanism is also used by a bus watchdog to fail accesses to slave addresses which do not respond within a fixed time-out period.

Before initiating a transfer, a master must acquire control of the Fred Bus. It does this using an asynchronous handshake protocol which is implemented using direct request and acknowledge lines to an arbitration module. A signal on the bus indicates when a master is requesting the bus, and obliges the current master to relinquish control as soon as is safe. This is sufficient to implement atomic multi-transfer operations, such as test-and-set on a location in a simple memory. A further bus signal provides the necessary lock for dual-ported memories. It is important to note that only the bus acquisition protocol is specified; no arbitration mechanism is specified. An arbitration module can use appropriate policies to suit particular configurations of potential masters. For example, a collection of processors, all of which can use every available bus cycle, might have a round robin allocation scheme, while a DMA device controller might use traditional prioritised cycle stealing.

The Fred Bus incorporates no additional mechanism to support interrupts. The approach used for interrupts was inspired by the stated aims of the DEC PDP-11 Unibus interrupt mechanism, but

incorporated substantial rationalisations. Processor modules are designed so that they can also act as slaves on the bus, and have addresses associated with special registers which, when accessed, cause interrupt requests to be generated. Conversely, interrupting devices act as masters when performing a transfer to interrupt a processor; the complexity of implementing this is intended to be small, by virtue of the fact that most devices should have a local control microprocessor. This scheme allows interrupts to be targeted at specific processors; also, either the address or data transferred may contain vectoring information for the interrupted processor. The same approach can also be used by processors to perform control operations, such as halting or resetting, on other processors (a bus-wide reset signal is provided for general initialisation). It is surprising that many modern computer systems use separate serial buses for inter-processor communication and control, rather than using the more efficient path already provided by the parallel bus.

The reader who is familiar with the operation of the Motorola 68000 microprocessor family may have noted that the Fred Bus is well suited to the production of modules incorporating these processors. This is no surprise because, at the outset of the FMAC project, the MC68000 was seen as the most exciting new microprocessor available. However, it was not allowed to influence the design to the point where major difficulties would have resulted when interfacing other microprocessors, the obvious processor-specific features omitted being arbitration and interrupt mechanisms. The generality of the Fred Bus has its price, in that it is not always possible for modules to operate at peak efficiency. For example, while an MC68000 module can operate at almost full speed, an early module consisting of a Zilog Z8001 processor with a Z8010 memory management unit was slowed by the need to have bus addresses completely specified before a bus transfer request signal could be issued. However, as the aim of the exercise was to get speedy experience of a new microprocessor, loss of performance was not critical. This phenomenon is reminiscent of portable compilers, which allow software to be rapidly installed on several computers, but sacrifice the optimisations made by hand-crafted compilers. In summary, it can be said that the Fred Bus has been more than adequate to allow widespread experimentation over the years and, indeed, well beyond its expected lifetime.

4. Harnessing the Fred Machine

In this section, the development of "official" FMAC hardware and system software is discussed. The first subsection is devoted to the prototype FMACs, and the second subsection introduces the workstation derived therefrom. The third and fourth subsections outline the subsequent evolution of the hardware and system software, in a logical, rather than chronological, order.

4.1 The Prototypes

During the academic year 1981-82, three prototype FMACs were built. As a first step, a system control module was designed, in order to allow software control to be exercised over the system. Then, as a useful experimental exercise, this module was combined with further hardware and software to implement a workstation attached to ELAN, the local area network mentioned in Section 2.2.

The system control module was based on a 8 MHz MC68000 microprocessor, with the address space split into local and system-wide areas. The local address space contained 16K of either RAM or ROM, together with an RS232 port and a programmable timer. The system-wide address space was provided by using simple mapping registers to convert 24-bit microprocessor addresses into 32-bit Fred Bus addresses. The local on-board bus was made available on a separate edge connector to allow off-board expansion of the local resources, independently of the Fred Bus; this was to simplify the incorporation of less sophisticated system modules.

While the system control module was capable of stand-alone operation in conjunction with a terminal, the next two obvious hardware additions were a memory module to increase the storage capacity, and a networking module to enable the bulk transfer of data to and from the outside world. The memory module interfaced 0.5 MByte of RAM, with parity checking and an access

time of 225 ns, to the Fred Bus; its address range within the system-wide address space was configurable by on-board switches. The network module interfaced ELAN to the control module local bus, and was of a rather more complex nature, in that it incorporated a Zilog Z80 processor, whose operation was controlled via special registers made available on the local bus. The Z80 was responsible for extracting data packets transmitted along the ELAN channel to the FMAC, and holding them on an on-board queue until the control module copied them into main memory. Similarly, when the control module copied packets from main memory to the network module, they were held in an on-board queue until it was able to transmit them along the ELAN channel. Nowadays, given a standard type of LAN, such functions could be obtained from a single-chip controller but, at that time, LANs were still very much in their infancy.

In parallel with this hardware development, work was done to produce basic system software. As noted in Section 2.4, the priorities were an IMP compiler and an IMP run-time system. The compiler was written in IMP from scratch, and it was compiled on LEGOS workstations (mentioned in Section 2.3) until it was reliable enough to compile itself into MC68000 machine code. Given that IMP can express reads and writes to arbitrary addresses, no special language extensions were necessary to enable full exercising of the FMAC. The IMP run-time system was written in MC68000 assembly language and functioned as a primitive operating system interacting directly with the bare MC68000 processor. It contained features to support IMP input/output and event signalling facilities, to handle a video terminal and the clock, to access ELAN, and to provide a simple software "front panel".

Once this work had been completed, a usable workstation had been created and, unanticipated by anyone then, it was going to exist for a long time. Given the availability of the Filestore (mentioned in Section 2.3), which acted as both a file and print server on ELAN by then, and support for the IMP language, it was not long before the FMAC workstation had an equivalent functional capability to the existing LEGOS workstations. The additional attractive feature was the availability of a faster processor, and eight times as much memory. This fact encouraged various people to move applications to the FMAC workstation; one particular example was ESDL [11], a circuit board design suite, and so an FMAC could soon be used to design extensions to itself. Unfortunately, this quick harnessing of the capabilities of the FMAC was the harbinger of a certain amount of system stagnation.

4.2 The Advanced Personal Machine

In most academic environments, there is a conflict of interests between the research and development of computer systems, and the demands of the user community for a sophisticated computing service. Satisfying the latter need can, of course, supply essential feedback to the former activity, but inevitably requires that systems have firm specifications and stable behaviour. In the case of the FMAC, the experimental workstation effectively became public property as soon as it was functional, and before there had been any detailed thought about what might be the best way to offer a public service. In the summer of 1982, the statement "The Edinburgh Advanced Personal Machine (APM) has been developed in the Department of Computer Science to fulfil its requirements for general purpose computing over the next five or so years" appeared in publicity accompanying a demonstration of the workstation at a University Open Day.

This marked the transition of the FMAC from experimental research and development to official computing resource. The name APM (chosen in haste for publicity purposes) has become an accepted mode of address for the FMAC although, strictly speaking, it only refers to a particular application of the FMAC as a workstation. It is encouraging to note that the APM has, indeed, served its purpose for the last five years, and continues to do so. Unfortunately, the rise of the APM caused the loss of the pioneering momentum which should have led to the FMAC continuing to justify the "advanced" tag as the years progressed. This was exacerbated by the departure of personnel from the project, and lively discussions of the advantages and disadvantages of having more managerial control over what was a group of self-motivated and competent individuals. It is important to stress that complete inertia did not set in with the emergence of the APM. A substantial amount of further work, particularly on the hardware side, has been done since.

4.3 Hardware Evolution

As far as the hardware described above is concerned, developments have reflected advances in technology. On the control module, the MC68000 processor chip has been replaced by a small "daughter board" carrying an MC68010 processor chip and associated memory management chips; this gives a virtual memory capability to the module. The next likely Motorola candidate for a processor module is the MC68030, which would fully exercise the 32-bit data and address buses, and has an on-chip memory management unit. On the memory module, 64K chips have been replaced by 256K chips, leading to a four-fold increase in on-board storage capacity to 2 MBytes. FMACs can contain multiple memory modules to further increase the available storage. The network module remains the same, although extensions have been made to the controlling software in the Z80. However, the advent of an IEEE 802.3 standard 10 MBits/sec contention bus LAN has led to an experimental interface being built for the FMAC. As well as updating the original three modules, new modules have been designed to interface various devices to the Fred Bus, so that FMACs can have more powerful capabilities when used as workstations, and so that FMACs can be used as the basis of specialised servers attached to ELAN.

The most important new facility for workstations has been a colour graphics module. An initial version consisted of a single-board frame store which generated a display of 1024 x 1024 4-bit pixels at a rate of 25 MHz, and was accessible as a write-only slave on the Fred Bus. The current version has a second board added, which extends the frame store to have 8-bit pixels, and offers a range of 32768 colours. This module has been substantially responsible for ensuring the popularity of FMAC-based workstations with the user community.

In the context of server implementation, two main modules have been produced. The first is a Winchester disc controller, which has been used in producing new file servers, as well as in supporting stand-alone FMAC systems. The original Interdata 70 Filestore no longer exists, and has been replaced by several FMAC-based file servers equipped with either 160 Mbyte or 300 MByte Fujitsu disc drives. These servers support a large number of disc-less FMAC-based workstations via ELAN. The second module is a laser printer driver which can transfer arrays of pixels from FMAC system memory to a laser printer. Several FMAC-based printer servers using Canon laser printers are now available on ELAN; these print plain text files, output files from the Scribe and TeX text formatters, and input files for the Layout text formatter [12].

Three multi-function modules have been made available. The Disc, Mouse, Arbiter and Keyboard (DMAK) module contains a collection of useful facilities. It has a floppy disc interface, which has made the production of a portable stand-alone FMAC system possible; a mouse interface, which allows a three-button mouse to be used in association with the colour graphics module; a bus master arbiter, for use in multiple master systems; and a serial interface which, amonst other things, has been employed to connect terminal-type keyboards to FMAC systems. The Quadruple Synchronous/Asynchronous Receiver/Transmitter (QSART) module interfaces up to four synchronous or asynchronous links to the Fred Bus. Connected to RS232 links, it has been used in both forward mode, in which four terminals are multiplexed on to an FMAC, and reverse mode, in which an FMAC can access four terminal ports on other computer systems. Finally the Real Time (RT) module contains a serial interface, two 8-bit parallel interfaces, a programmable timer and a Motorola M6809 microprocessor, which can be downloaded and controlled via the Fred Bus. It has been used to interface various real-time equipment to FMACs.

4.4 Software Evolution

Obviously, the hardware is of only limited use without software to facilitate its operation. A prevailing philosophy of the FMAC project was that system software should be a thin filling sandwiched between the hardware and user software. While this approach fitted with the original spirit of the FMAC project, it was less desirable for the APM, where fairly rich support for user software was required. Therefore, it can be said that FMAC system software has not advanced as much as the hardware.

The core software for the FMAC remains the original IMP run-time/operating system. It supports programs written in IMP, and in Pascal (compiled by a compiler derived from the IMP compiler). It also supplies libraries for floating point arithmetic, for handling virtual video terminals, for drawing colour graphics primitives, for using ELAN, and for accessing files on the filestore. User object code libraries may also be created and, irrespective of original language, can be dynamically linked with main programs at run-time. As well as the compilers, many workstation facilities are built upon this basic core, including several screen editors and text formatters, chip and circuit board design suites, file maintenance and printing commands and the ability to communicate with other computer systems on ELAN. The standard user interface is via an interpreter which reads commands from a terminal keyboard, and reports to the terminal screen.

In most respects, this collection of system software supplies what academic computer scientists need and, to an extent, it is consistent with the "building block" nature of the hardware in that it provides a solid, if monolithic, basis for users to add new software modules. Two major criticisms have been aimed at the system, first that it is non-standard and so cannot run software written elsewhere, and second (particularly since the rise of Unix) that it does not support multiple processes. A simultaneous solution to both of these problems, namely implementing a standard multi-process operating system on the FMAC, is mentioned in Section 5. The absence of processes has caused the evolution of an event-driven style of program organisation, akin to that recommended for the single-process Apple Macintosh. This is adequate in many circumstances, but user programs cannot be expected to handle all system events and, as a result, one conspicuous failing of the FMAC systems has been the absence of background communications activities, such as the receipt of electronic mail. In the interests of simplicity, interrupt handlers in the run-time system have not been expanded into complex event handlers in order to deal with this problem.

In fact, a lightweight multi-process kernel was produced for the FMAC as long ago as 1983 but, for no good reason, it remained in cold storage until 1986. This kernel was a software analogue of the Fred Bus, being restricted to first-level hardware event handling and process coordination over multiple processors, with a flexible range of process modules used to deal with such things as store management, naming and device/user interfacing. A variant of the kernel is now in use in the FMAC-based file servers and, after some modification to deal with such things as virtual memory, a further variant is scheduled for release into user service shortly [13]. The mistake of bringing new software to the attention of users immediately has not been repeated this time, and it is to be hoped that the design will have had time to mature. Unfortunately, however, the delay in introducing it may mean that it will not be fully exploited, given that the lifetime of the FMAC is limited.

5. Using the Fred Machine

The FMAC, particularly in its APM guise, has found widespread application in the research and teaching activities of the Department of Computer Science at Edinburgh. A large amount of usage has been for the development of application software; inasmuch as this could have been performed on most workstations or mainframes, it is not detailed here. However, given that the FMAC was designed locally, and that ELAN servers do not rely on FMAC clients being trustworthy, there has been considerable scope for using the FMAC in a wide range of contexts which could not have been offered by most proprietary equipment. Projects performed by final year undergraduates and M.Sc. postgraduates have been a particularly fruitful source of innovation (even when they have not yielded completely functional products).

As a result of user efforts, much useful software exists. Several window and menu based user interfaces, using colour graphics and a mouse, have been produced, as well as a more advanced command line interpreter. A compiler for C is available, and has been used to port various applications, such as the TeX text formatter and an interpreter for ML (a functional programming language), to the FMAC. The multi-process MUSS operating system [14], developed at Manchester University, has been ported to an FMAC system with local disc, and this sytem can run Unix on top of itself. Independently, a direct Unix implementation was started but, with the current proliferation of other Unix workstations in the Department, was recently abandoned.

Research has been done on a microprogrammable processor module for the FMAC. It is centred upon a 16-bit wide horizontally microprogrammed data path using AMD AM29203 bit-slices. Its major application has been in the production of an intelligent graphics module, which can compute images from structured descriptions residing in main memory, and display them. Research work in computer graphics, image processing, computer-aided design and VLSI design has made extensive use of the standard colour graphics capability of the FMAC. Research work on distributed computing systems, in particular on network file and name servers, has had the collection of FMACs, connected via ELAN, available as an experimental testbed; a spin-off from this work has been the latest type of FMAC-based file server [15].

The FMAC has been used in conjunction with teaching at all levels. Courses using it in an interesting way have included Microprogramming, where the microprogrammable module mentioned above has been used to emulate various instruction sets; Operating Systems, where simple multi-process operating systems have been written for the FMAC; Digital Communications, where several FMACs have been linked together using standard protocols; Advanced Graphics, where various algorithms have used the colour graphics module to produce sophisticated three-dimensional images; and Real Time Systems, where an assortment of real-time equipment, including robot arms and turntables, has been interfaced to the FMAC.

Student projects have resulted in an array of modules for the FMAC, which can merely be listed here. Processor modules have been based upon the Zilog Z8000, Intel 8086 and iAPX286, and National Semiconductor 16032 and 32032. Unfortunately, lack of software, such as compilers, has meant that these modules have only been exercised at a low level. Other modules have included a speech coder-decoder interface, a piano keyboard interface, a digitising camera interface, a Rugby time signal receiver, a Teletext receiver, a satellite picture receiver (in conjunction with the Meteorology Department), an X.25 level 2 communications interface, a floppy disc controller, BITBLT-style graphics and hardware floating point support; further additions to this collection are expected in the future. Student projects have also contributed a Basic interpreter, an Occam compiler and run-time environment, and a Visicalc-style spreadsheet package.

6. Conclusions and the future

The previous section indicates that the FMAC has been a success with computer users in the Department of Computer Science at Edinburgh. Doubts about the success of the FMAC project might arise from two directions. From one point of view, it might be felt that the original potential of the FMAC to encourage work on structured hardware and software design has largely been wasted, due to the premature concentration on the APM; moreover, the commitment to supporting a computing service implied constraints, such as rigidity of specification and an increased degree of managerial interference. From another point of view, it might be regretted that a large amount of effort was spent on providing a parochial computing system, and that further effort is necessary to make imported software available on it. These two points are both valid, and have been reflected in the fact that most FMAC work is now undertaken by computing support staff, and that various proprietary workstations are available in addition to the APM. What cannot be denied is that the FMAC was ahead of its time and that, for the last five years, users in the Department have had access to a collection of disc-less workstations, each with a colour graphical display, backed up by a powerful processor, a large memory and a link to various networked servers.

Two of the three principal members of the FMAC project group have now left the Department, one to found a company selling laser printer controllers partially based on FMAC technology. Therefore, significant further development of the FMAC is unlikely. At the time of writing, around sixty FMACs are in service, about thirty of these being workstations with colour graphics. During term-time, there is often a shortage of public systems for students to use. However, it is unlikely that any more FMACs will be built, since the Department is moving towards proprietary facilities. A large number of Sun workstations, connected by a standard 10 MBit/sec contention bus, are now available, as well as a significant number of Apple Macintoshes, connected by Appletalk. The FMACs will remain in useful service within the Department for several more years, presumably until they are incapacitated by old age.

Acknowledgements

Many people have helped with the Fred Machine project over the years. Hamish Dewar and Rainer Thonnes have made major contributions. Igor Hansen, Jimmy Johnstone, Peter Lindsay, George Ross and Ian Thomson have also made significant contributions.

References

(EUCSD is an abbreviation for the Department of Computer Science at the University of Edinburgh.)

[1] Butler, J. (ed.) "Complete Guide to the Fred Machine" EUCSD Manual 1985.

[2] King, F. "Notes on using the Micro Kits" EUCSD Manual 1978.

[3] Metcalfe, R. and Boggs, D. "Ethernet: Distributed Packet Switching for Local Computer Networks" CACM **19** 1976, pp 395-404.

[4] Enos, W., Hansen, I. and Thonnes, R. "The Edinburgh Local Area Network" EUCSD Manual 1983.

[5] Dewar, H. "ISYS User Reference Manual" EUCSD Manual 1975.

[6] Dewar, H., Eachus, V., Humphry, K. and McLellan, P. "The Filestore" EUCSD Manual 1977.

[7] McLellan, P. "LEGOS User Reference Manual" EUCSD Internal Report CSR-49-79 1979.

[8] Whitfield, H. and Wight, A. "EMAS - the Edinburgh Multi-Access System" Computer Journal **16** 1973, pp 331-346.

[9] Stephens, P. "The IMP Language and Compiler" Computer Journal **18** 1975, pp 131-134.

[10] Borrill, P. "Objective Comparison of 32-bit Buses" Microprocessor and Microsystems March 1986, pp 94-100.

[11] Smith, L. "The Elementary Structural Description Language" EUCSD Internal Report CSR-53-80 1980.

[12] "Layout Formatting Language Reference Manual" Clan Systems Ltd. Available as EUCSD Internal Report CSR-196-85 1985.

[13] Thonnes R. "The Mouse" In preparation.

[14] Frank G. and Theaker C. "The Design of the MUSS Operating System" Software - Practice and Experience **9** 1979, pp 599-620.

[15] Ross, G. "The New^3 Filestore" In preparation.