ICL TECHNICAL JOURNAL

Volume 4 Issue 3
May 1985

ICL TECHNICAL JOURNAL

The ICL Technical Journal is published twice a year by Peter Peregrinus Limited on behalf of International Computers Limited

All correspondence and papers to be considered for publication should be addressed to the Editor

1985 subscription rates: annual subscription £15.00 UK, £17.00 overseas, airmail supplement £7.50, single copy £9.00. Cheques should be made out to 'Peter Peregrinus Ltd.', and sent to Peter Peregrinus Ltd., Station House, Nightingale Road, Hitchin, Herts. SG5 1SA, England, Telephone: Hitchin 53331 (s.t.d. 0462 53331).

The views expressed in the papers are those of the authors and do not necessarily represent ICL policy

Publisher
Peter Peregrinus Limited
PO Box 8, Southgate House, Stevenage, Herts SG1 1HQ, England

Printed by A.McLay & Co. Ltd., London and Cardiff ISSN 0142–1557

ICL TECHNICAL JOURNAL

Contents

Volume 4 Issue 3

Foreword

In late 1981 ICL Mainframe Systems set out to develop a new range of mainframe computers in 3½ years. This timescale represented an enormous challenge. No other comparable system had ever been designed and validated in such a short period, and we were attempting not one but two machines in radically different technologies.

Timescale was not the only challenge to be faced. We were determined to produce a quality product. This meant not just a reliable system, but one which could be manufactured and serviced efficiently. We had tight cost targets to ensure a competitive product in the marketplace. To this end, we set out to intercept the technologies that we anticipated would be available 3 years on.

This issue of the *ICL Technical Journal* deals with the story of one of the products in our new series, that which became known in development as DM1. It tells the technical story. The human stories of long hours, untaken holidays, frustrations and triumphs will probably never be told.

My sincere thanks go to those many people in ICL and our suppliers who have lived with the developments over the past few years, for their contribution and total dedication. Above all, my thanks go to the innovative hardware and software engineers who created the system.

D.J. Dace
Director, ICL Mainframe Systems Division

Queen's Award

On the 21st April 1985 it was announced that Mainframe Systems Division had won the Queen's Award for Technological Achievement, in recognition of the successful innovation represented by the development of CAFS-ISP.

This signal honour marks the culmination of some 15 years of work from primary research through the construction and evaluation of prototypes, the introduction of CAFS 800 as the first product, and finally the integration of CAFS-ISP into the mainstream of VME, data management, Quickbuild, applications software and the 2900/Series 39 hardware. CAFS had previously won the 1980 Technology Award of the British Computer Society and was voted 'Product of the Decade' by *Computing* in 1983. During 1984 and the first half of 1985 over 450 orders were registered for CAFS on 2900 systems; from now on it is an automatic constituent of every Series 39 system. This success must be attributed, not only to the excellence of the product, but also to the quality of the collaboration between many people in many parts of ICL.

Overview of the ICL Series 39 Level 30 system

C.J. Skelton
ICL Mainframe Systems Division, West Gorton, Manchester

Abstract

The paper summarises the essential features of the Level 30 system, a member of the new ICL Series 39 of distributed mainframes, the production of which was the culmination of a four-year exacting design and development project involving some 300 engineers and programmers. The dependence of the successful implementation of a number of different state-of-the art technologies is emphasised, in particular on the collaboration between the ICL and Fujitsu teams in the design (in England) and manufacture (in Japan) of the 8000-gate LSI CMOS chip embodying the logic. The paper provides an introduction to the more detailed accounts of particular features of the system given in the papers that follow in this issue.

1 Introduction

When the chip collaboration contract was signed with Fujitsu in November 1981 the project team was conscious of the importance of the new Series 39 of distributed mainframe systems to the future of ICL. This was to be no easy development, no simple evolution from a previous product. Not just one but a significant number of different advanced technologies were to be 'intercepted': powerful mainframes designed to exploit the latest developments in VME, with 256 kbyte memory chips and new fast RAMs for local stores, a commitment to local area networks for all input/output including a fibre-optic high-speed LAN for fast peripherals and OSLAN for lower-speed devices; a new range of peripherals; and remote telediagnostics. For the Level 30 system, the subject of this and the following papers, even the structural foam mouldings for the cabinet were to become the most complex in Europe. The team knew that the failure of any one technology could wreck the entire project: each had to work. This was to be the original 'Hole in One' project.

2 Market objectives

The new machine was required to broaden the power range of the VME family from around 10:1 to beyond 20:1 with marked gains in inherent reliability and resilience. Implicit in this was a multiprocessing capability: it would exploit the growing opportunities for distributed processing, be capable of entering nontrad-

itional data processing environments, be easy to use, give fast response times and easy access to local CAFS-based and large central corporate databases. Level 30 was required to provide a natural path for ME29 users to migrate to VME by supporting CME, which has TME hosted by VME.

Costs had to be kept to a minimum and a reduction in support costs was planned by exploiting telesupport.

3 Key characteristics

The main characteristics of Level 30 to meet the marketing requirements are as follows:

- performance comparable with the 2957 mainframe
- a compact design, styled and engineered for the office environment (Fig. 1): quiet, low power dissipation, no clean air requirement, a wide range of temperature and humidity operating conditions, no air conditioning or false floor needed

Fig. 1 The ICL Level 30 in use

- multiprocessing capability to exploit the evolving nodal architecture of VME
- LSI CMOS: 42 Fujitsu C8000 gate arrays giving dramatic reduction in size and power dissipation, and increase in reliability
- 4-16 Mbytes of main store, based on 256 kbit RAMs, on a single PCB

- high-speed 50 Mbit/s MACROLAN local area network (LAN) for internode, disc and tape connection, using a fibre-optic system which includes VLSI transmitter and receiver circuits
- medium-speed 10 Mbit/s OSLAN, based on the industry standard Ethernet, for printers, communications, workstations etc. MACROLAN and OSLAN give an impressive reduction in cabling and hence improved reliability and interconnect distances measured in hundreds of metres
- new high-density sealed 300 Mbyte and 625 Mbyte fixed discs (FDS 300 and 2500, respectively), factory primed with the system software
- ICL's Content Addressable File System (CAFS) integrated into each disc controller, as standard
- full remote telesupport and telediagnostics using a microprocessor node support computer.

A schematic diagram of a typical system can be seen in Fig. 2.

4 Fujitsu collaboration

A world-wide search was made in mid-1981 for a technology to supersede the 2966 MSI machine. The team had no hesitation in choosing Fujitsu in Japan, with its new 8000-cell CMOS gate array for Level 30. Agreements were signed in November 1981.

Not only had a formal interface for the transfer of data between the two companies to be worked out and validated, but a fusing of two subtly different cultures was required; a process which has led to personal friendships and the build up of a very deep respect between the two teams.

Fig. 3 shows the interface between ICL and Fujitsu. After careful design automation validation, three tapes for each chip type are sent to Japan. One, FLDL, represents the logic and enables Fujitsu to set up a simulation model of the chip. Another, FPDL, defines the physical cell placement and interconnect tracking, and the third, CTPF, provides the test data for the initial checking on the model and for the final testing of the completed chips.

The C8K gate array (Fig. 4) is a 2·3 μm CMOS technology, with two layers of interconnect metalisation of 6 and 8 μm pitch, respectively. ICL and Fujitsu together designed 53 cell types and up to 8000 can be placed and tracked. The chip size is 1 cm^2 and it is housed in a 2·5 cm^2 ceramic package with 160 signal pins (179 total). At the time of writing, prolonged validation tests on over 30 machines have shown no failures of C8K chips.

5 Top-down design

The Level 30 required 42 C8K chip types, with approximately 60 in total depending on the configuration. A major programme risk was the possible need to reiterate chips for modification, each cycle taking about three months. It was

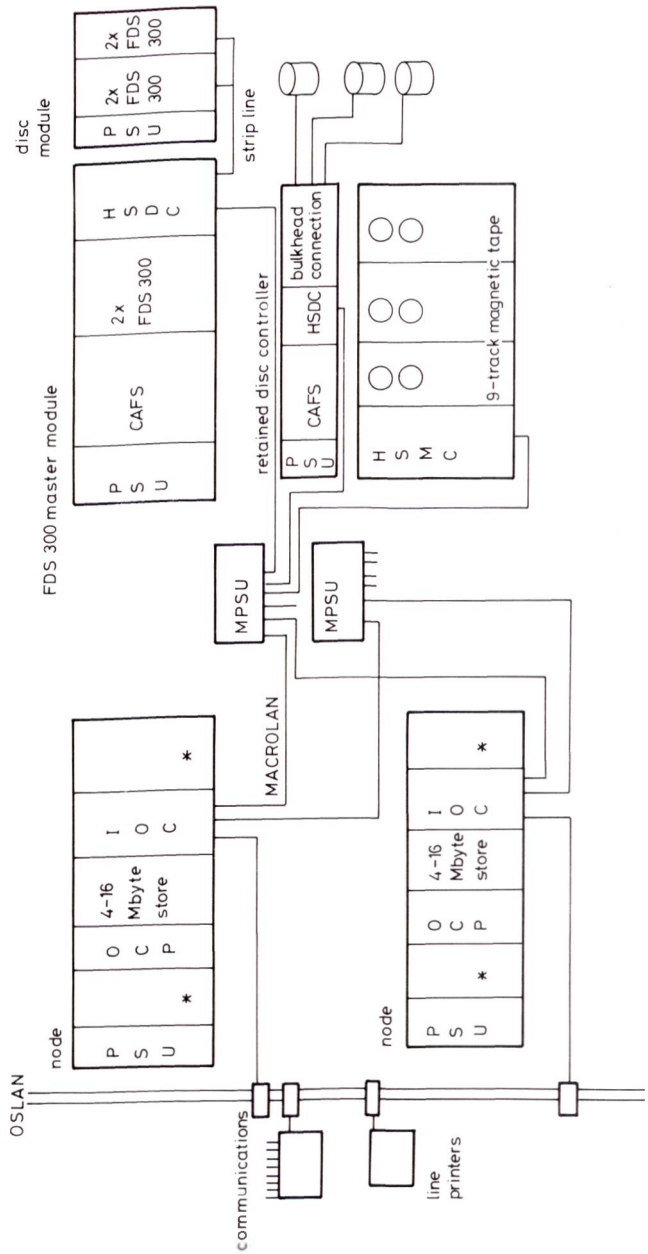

Fig. 2 System block diagram (two single nodes)

OCP order code processor
IOC input/output controller
PSU power supply unit
* spare slot for OCP, IOC

CAFS Content Addressable File System
HSDC high-speed disc controller
HSMC high-speed magnetic tape controller
MPSU MACROLAN port switch unit

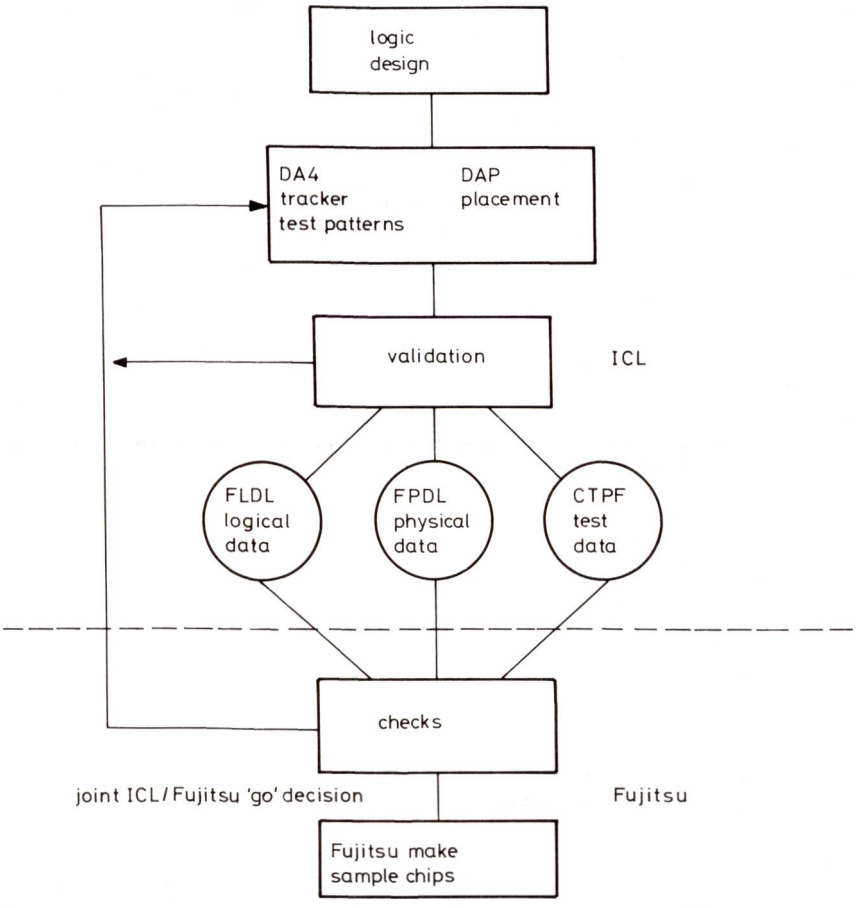

Fig. 3 ICL/Fujitsu interface

essential, therefore, to validate the design of the complete machine down to chip level before releasing any of the chip data. Analysis of previous developments had shown (Fig. 5) that 85% of the probable machine design errors could be removed by running ICL test programs on a completed design. Running higher level software would have been beyond the available computer resource.

Three parallel simulation streams were established (Fig. 6). From the top level system design SAUL (simulation at unit level) models were written by the engineers and test programs and target microcode were run on these. In parallel, unit overall diagrams (ODs) were derived from the system design, and SAUL models were written defining the chip functionality. The third stream automatically generated a gate-level model from the chip logic engineering drawing; test patterns from the two previous DA streams were used to check the gate-level simulation. The validated chip design could now enter the physical placing of cells and interconnect tracking, to generate the three tapes for Fujitsu.

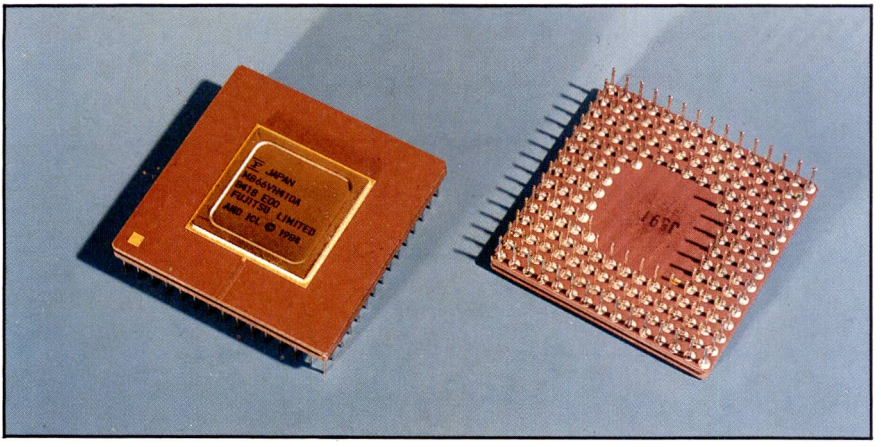

Fig. 4 C8000 CMOS chip

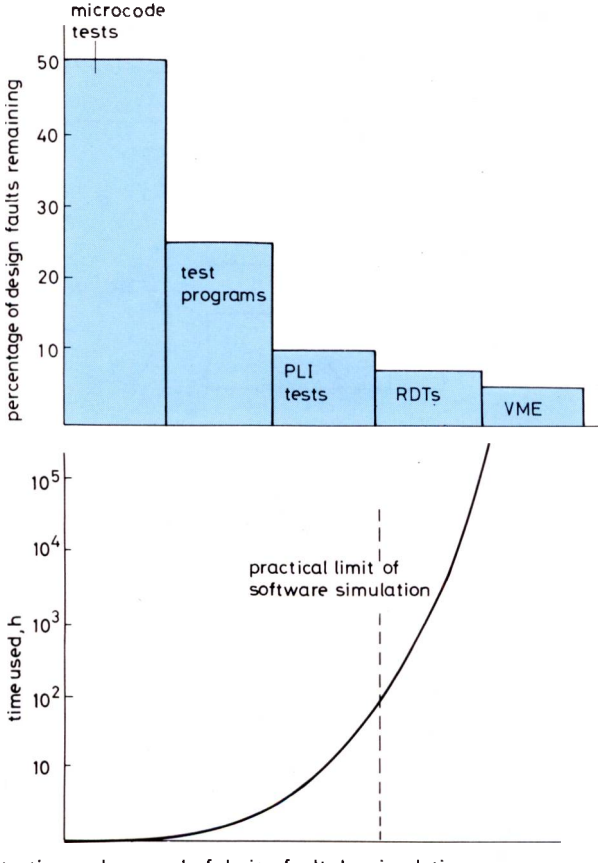

Fig. 5 Detection and removal of design faults by simulation

PLI primitive level instructions RDT range definition test

The simulation and tracking of 42 chip types was very demanding on machine time: a tracking run alone could take 35 h on a 2966 or 5 h on an ICL DAP (Distributed Array Processor), and most chip designs went through several design or tracking iterations before release to Fujitsu. The equivalent of seven 2966 processors were engaged round the clock, seven days per week, for over six months to complete the chip programme.

The success of this activity can be judged by the fact that, although a small number of types were subsequently recycled to maintain the strict worst-case tolerancing timing rules set by the project, no chip design fault stopped development and the processing node ran VME to within 5% of expected performance without any chip iteration.

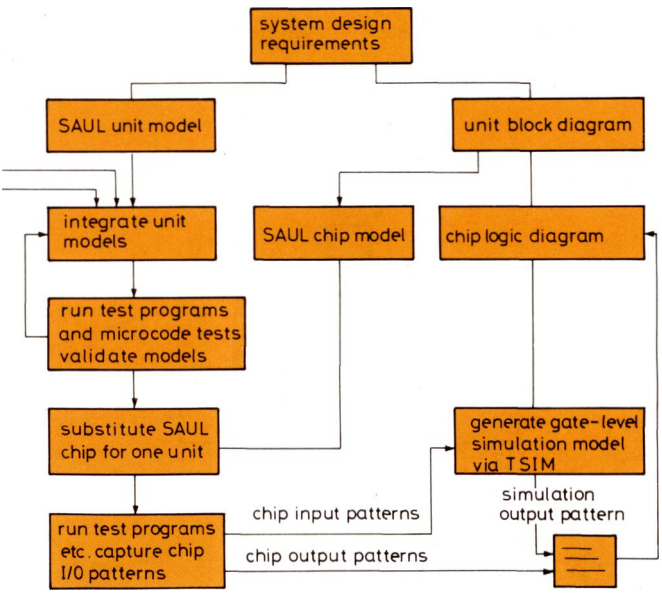

Fig. 6 Top-down design simulation

6 Local area networks on Level 30: MACROLAN

Local Area Networks have been introduced to dramatically reduce cabling, and hence increase reliability, cut costs and improve system configuration flexibility. MACROLAN[1], an ICL proprietary LAN, has a wide bandwidth, running at 50 Mbit/s. It is the fastest LAN in production in the world. The network is configured as a star and the token-passing protocol enables it to pass data at up to 5½ Mbit/s. A central MACROLAN Port Switch Unit (MPSU) contains the MACROLAN clock and automatically includes a new unit as it is plugged in. The transmission medium is a twin 200 μm glass-fibre cable with a 130 μm core. It is terminated with twin optical connectors specially developed by Thomas & Betts. The 0·85 μm near-infra-red light is generated by a low-power light-emitting

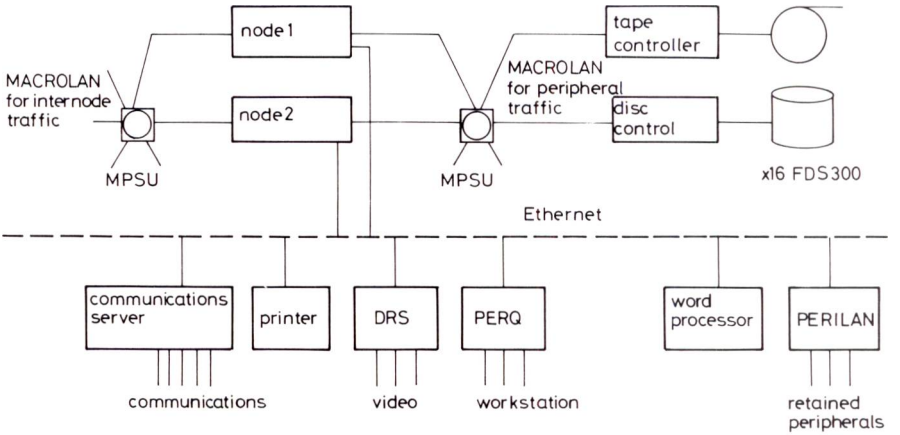

Fig. 7 Local area networks on Level 30

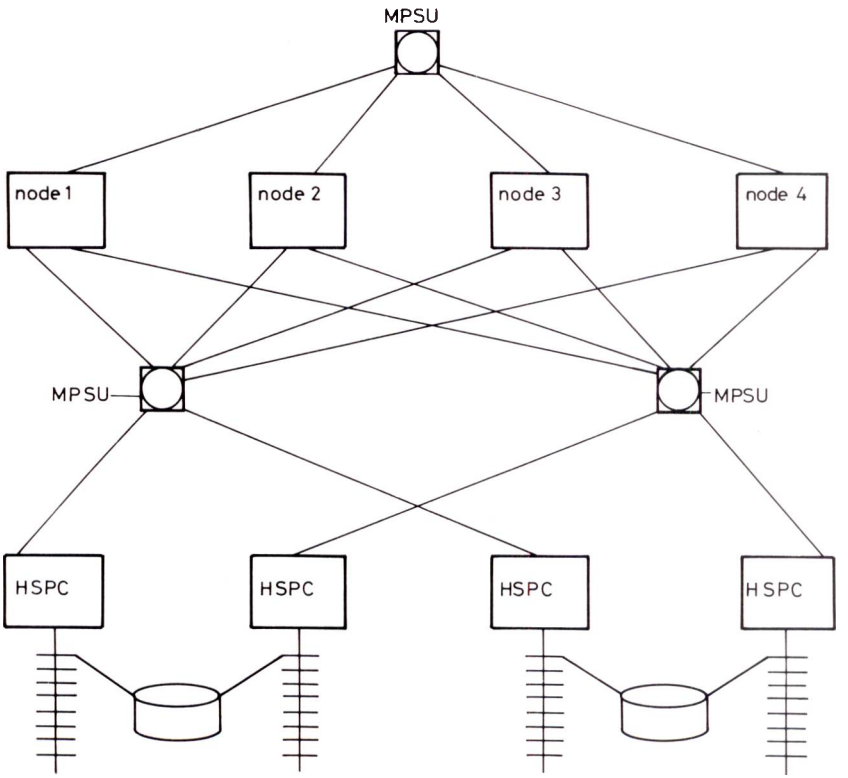

Fig. 8 Multinode resilience

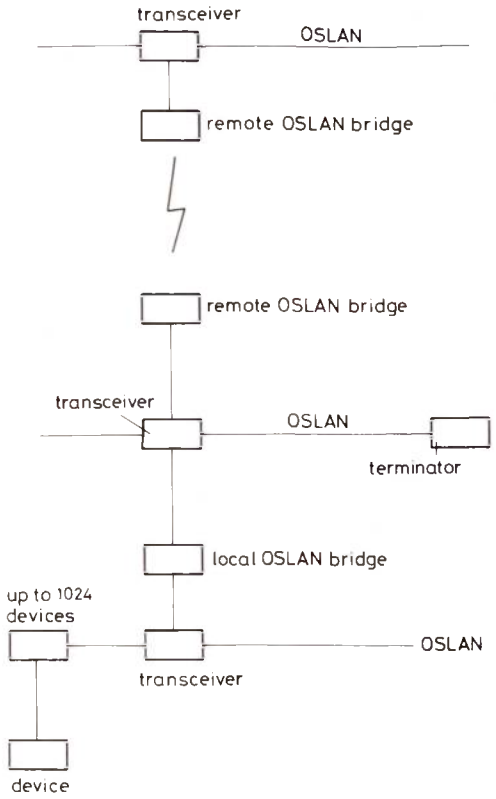

Fig. 9 Open systems local area network

diode (LED) and received by a PIN photodiode backed by a VLSI receiver developed by Plessey. MACROLAN is used in Level 30 to interconnect processing nodes and to connect discs and tapes (Figs. 2 and 7). Each MPSU leg can be up to 500 m giving very flexible configurability. Fibre-optic MACROLAN is free from electromagnetic interference, making it very interesting for electrically noisy or military applications.

The potential use of MACROLAN in a large, future, fully resilient multinode configuration is illustrated in Fig. 8.

7 Local area networks on Level 30: OSLAN

All medium and low-speed peripherals are connected via ICL OSLAN, which is based on the Ethernet-like industry standard IEE 802.3. This is a CSMA-CD (carrier sense, multi-access, collision detect) LAN which uses a copper coaxial cable and can support up to 1024 devices. Connection to this is by a 15-way cable to a transmitter/receiver box: these latter cables are made by several

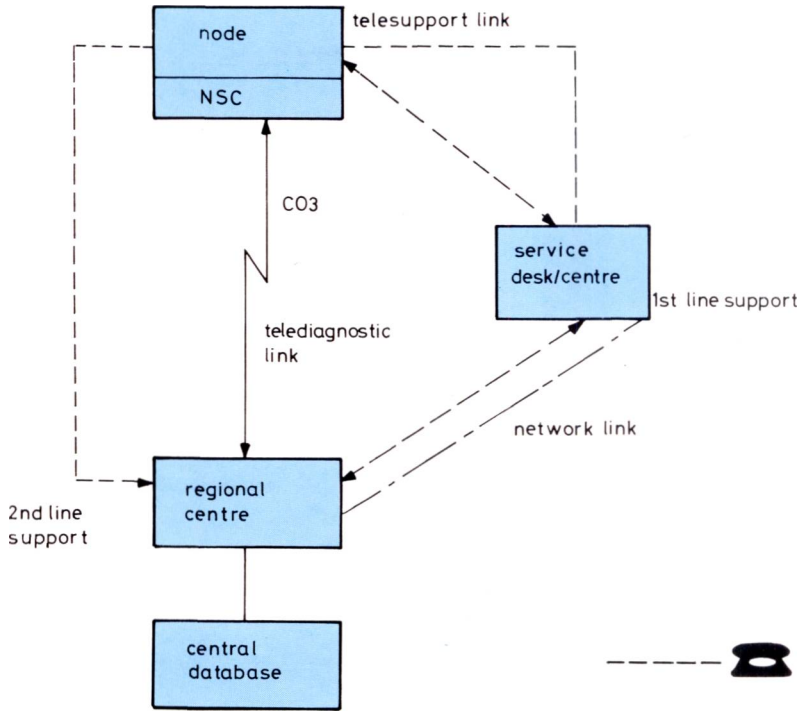

Fig. 10 Field maintenance

manufacturers (BICC and Olivetti among others) and can be inserted into the OSLAN cable or in some cases can be connected by a 'bee sting' electrode which can be screwed through the outer screen on to the inner core.

Again, special complex VLSI Ethernet coupler chips were required in the same timescale: the availability of these chips at low price will encourage the making available of a wide range of Ethernet peripherals. 2000, 1440, 800 and 400 line/min printers will be available on OSLAN, also the ICL PERQ workstation, DRS terminals and an Open Systems Gateway Modular Communications Unit (OSG-MCU 0). This last can have up to eight 4·8 kbaud or four 9·6 kbaud or two 19·2 kbaud lines, or an equivalent mix of line speeds. Interconnect distances using OSLAN are virtually unlimited. This is illustrated in Fig. 9.

8 Remote maintenance

All Level 30 systems are backed by Regional Support Centres (Fig. 10). These comprise a VME service supporting a maintenance database, providing software and hardware Known Error Log matching, repair and modification control. Support and maintenance (SAM) software can be run in the machine and there is a dedicated microprocessor node support computer (NSC) with a separate commu-

nications line direct to the regional centre. Software in this centre known as VISA (VME Inoperable System Access) and software in the NSC can resolve node faults down to replacement part level without any need for expensive site visits by skilled support engineers.

9 Conclusion

Level 30 is the culmination of the dedicated work of up to 300 engineers and programmers, each of whom has considered it a privilege to work on such an exciting project. The product needed design automation tools, VLSI developments in several companies round the world and a number of new technologies. It is very gratifying to see all of these coming together, all of the technology intercept not only working but still state-of-the art: a well balanced, exciting new mainframe system. Level 30 has arrived. The first Hole in One venture has been a success.

Reference

1 STEVENS, R.W.: 'MACROLAN: A high-performance network', *ICL Tech. J.*, 1983, 3 (3), 289-296.

VME nodal architecture: a model for the realisation of a distributed system concept

B.C. Warboys

ICL Office Systems Division, Bracknell, Berkshire

Abstract

The paper informally describes the generalisations which have been made to the VME architectural model to allow the construction of flexible distributed processing environments. It then identifies how these generalisations have been exploited in the construction of the latest generation of VME systems – the nodal generation. It seeks to demonstrate that the earliest principles of the new range architecture which begat the VME model are still valid today; that, indeed, with the imminent emergence of new non-von Neumann styles of processing, these principles are crucial to the rapid assimilation of the new techniques into existing operational systems.

It always takes longer than you expect, even when you take into account Hofstadter's Law[1] Douglas R. Hofstadter[1]

1 Background

A further half of a decade has elapsed since my last VME architectural paper for the *ICL Technical Journal*. Given the fact that this issue is dedicated to the development of the VME nodal systems it seemed appropriate to provide a short update to that paper and hence also to structure this paper in the same generic style. It reflects the extent to which the architecture has achieved its objective of acting, as all good architectures should, as a mechanism for steering product design decisions over a relatively long time period. The degree to which the new generation of VME nodal systems has usefully exploited the earliest principles of the new range architecture has, I believe, demonstrated the open-endedness of the simple set of system concepts which are the basis for that architecture. This property is, of course, fundamental to system infrastructures which wish to cope with the new techniques which will mature over the next decade. The paper assumes the definition of VME entitites given in the previous paper. A more comprehensive description of the details, as distinct from the motivations, of the VME architecture are to be found in Parker's paper[2].

The paper restricts itself to informally outlining the generic model, evolved from the previous architectural base, which forms the basis for the development of the new generation of VME nodal systems. Other papers in this issue will describe

the detailed structure of the component subsystems and how they interact in forming an operational system.

The paper is written as an update to the previous model paper[3]. As such it will, I fear, give some readers a sense of aimlessness. I therefore thought it worthwhile to include a road-map through the paper. The numbers refer to sections of the paper.

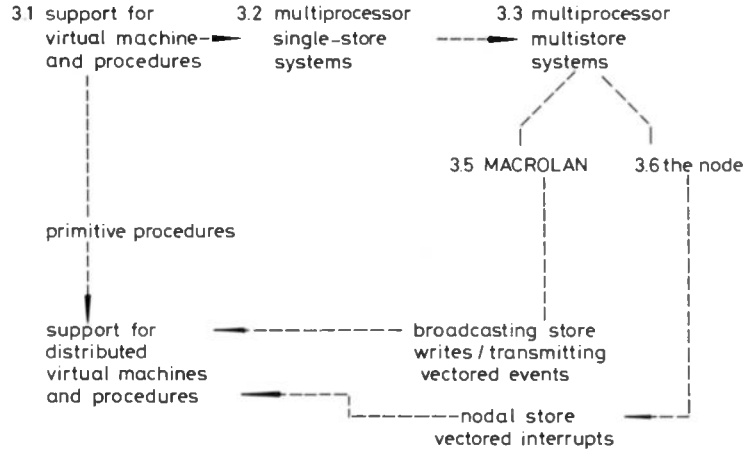

Fig. 1 Road-map through the paper

2 Introduction

The previous architectural paper[3] showed that, historically, the monolithic and therefore expensive development of applications products had been the consequence of two main circumstances:

- the widespread classification of computer systems into two worlds – operating system and application program
- the lack of genuine support for modular program development systems.

The basic elements of the VME building style, VME virtual machines and VME procedures, were introduced to provide both ongoing strategic design focus and implementation control over the long-term possibility of system decay with respect to these two vital restrictive system practices. It will therefore come as no surprise to find that when opportunities presented themselves for closer integration of software and hardware virtual machine and procedure support should turn out to be the most cost-effective areas for implementation.

Basically the technological advances, in the integration of hardware components and distribution agents, which had made better component integration feasible, will be the dominant factors in system design for the foreseeable future. Thus

the architectural advantages exploited in the latest product structures will continue to benefit VME-based systems for many more product iterations. To that extent this merely confirms what all system designers have always known, that initial economy of concepts at a generic architectural level is the key to flexible design of subsystem components in subsequent products.

So what integration of software/hardware has been possible in the new generation machines?

Basically, integration that takes us one step nearer to realising the 'paper' architectural model as an operational system infrastructure; and thus to extending the role to component control at an operational as well as an implementational level. Such an all-embracing infrastructure becomes increasingly attractive as, in parallel with the increasing variety of types of subsystem components, the demand for simple coherent system behaviour becomes irresistible with computer systems becoming endemic in our society.

3 The nodal model

The 1970s had generated and, to some extent, rediscovered a number of implementation possibilities:

- the large scale integration (LSI) of semiconductor components had offered the opportunity for very high functionality at individual chip levels
- fibre-optic technology had offered the opportunity for very high-capacity interconnect structures
- functional programming styles – with related technologies (precedents or antecedents depending on religious viewpoints) such as object-oriented languages, logic languages, applicative languages. These all reinforced the need for a clean definition of the Virtual Resource Paradigm incorporating notions of encapsulation, instantiation and communication.

3.1 Support for virtual machines and procedures

These possibilities were channelled into the two features of greatest significance to the VME system concept – the VME virtual machine and the VME procedure. The VME virtual machine acts as the operational context for each user/task or their agent to instantiate their personal virtual copy of a real system resource. The VME procedure is *the* atomic operational resource which provides the means of affecting the state of a virtual machine.

In particular, architectural support for the procedure had always recognised the need to support many procedure types and yet to retain a single call interface. The architecture allowed that since the call instruction (of one procedure from the body of another) was independent of the privilege and priority of the concerned procedures, a multilevel, in a privilege and priority sense, system of procedures could be constructed. Further, since the binding of one procedure to another was accomplished by the insertion of a pointer (a call descriptor)

between the call and the target procedure, this binding could be delayed until call time.

This flexibility of context construction had been exploited through a naming and loading system to produce a general mechanism for the development of a system constructed from a set of synchronous, reusable, procedure bodies. These bodies, when instantiated on being 'loaded' into the system, give rise to two types of variables: local variables relevant to the subsequent execution of this procedure in its local process context, shared variables relevant to the execution of the local process in the current system context. Clearly the access to shared variables needs to be controlled by appropriate critical region control mechanisms and indeed, as might be anticipated, these eventually lead to hardware-supported processor semaphores, system-wide data items with a binary state to indicate True/False conditions.

3.2 Multiprocessor/single-store systems

If the system is physically distributed among a set of processors then any one processor accessing these hardware semaphores *must* broadcast to *all* like processors before reserving/releasing any semaphores; that is before resetting or testing the True/False condition. There can clearly be only one instance of each system semaphore for the system to function. This instance in all VME systems to date has been a location in a single physical store.

3.3 Multiprocessor/multistore systems

This restriction, although unnatural, has not been, in the past, a severe handicap in system design. The technology available has not, until now, allowed for further cost-effective distribution beyond the multiprocessor, single-store style of existing VME systems. However, the current and future integrated and distributed technologies not only allow for greater functional distribution but actually encourage it, since the more integrated the component or subsystem the greater are the benefits of replication as a means of system growth – of course always assuming a decent distributed architectural base.

The restrictions due to the multiple-processor/single-store approach need to be lifted for a number of reasons:

- the increasing integration of components made possible by technological advances argues against rigid definitions of processors and stores
- queueing, with consequential limitations on replication, will occur in a system model based on the restriction of at least one component to single-server behaviour – in this case it is the store
- better system reliablity characteristics result from the ability to replicate all components
- greater power range can be covered with one set of subsystem components.

What then must be done to lift these restrictions? Very simply, generically

speaking, to avoid any single-server subcomponents in the system and to create a multimaster distribution system such that the only synchronisation which is needed in the system either occurs 'naturally' or by explicit request by VME procedures to exclusively own real/virtual resources for a period of time. This exclusive ownership is a necessary property of the type of resource and access needed rather than a consequence of the system's construction in terms of component interconnect.

3.4 Generic physical components

To achieve this generic architectural structure the system is defined to comprise two general types of physical components.

- a set of 'nodes', each consisting of a code processor and storage which provide the execution means for the operational resources (VME procedures) of the system
- a 'ring' system to which *all* 'nodes' are attached in a multimaster control structure.

'Ring' is put in quotes because the ICL MACROLAN, while it behaves functionally as a ring, is implemented as a centrally clocked and polled star system.

There is no *a priori* restriction on the order code which any processor supports, but the processor interface to the ring system must conform to a set of system standards. These standards will in practice, therefore, define the limits of the operational architecture definition and represent the generalisation of the architecture needed to achieve the removal of the restriction to single store system.

These standards, as could be foreseen, cover the operation of the two fundamental building blocks – the system building block (the VME virtual machine) and the VME virtual machine building block (the VME procedure).

The virtual machine *and* its process state are effectively defined by the virtual storage 'owned' by that virtual machine at any point in time. This virtual storage comprises a set of segments with access property rights which define the rules covering their access by appropriate subsystem collections of VME procedures. These access rights cover both privilege access rights and the type of access allowed. This access type can be execute, read or write (and combinations of the three) and either local (to a given virtual machine) or shared (by other virtual machines).

The generic distribution model has, as its objective, two types of functional distribution. The first allows for the distribution of the system processing elements (the virtual machines), the second for the distribution of operational resources (the procedures) within a processing element across a variety of support processors.

3.5 MACROLAN support

To support this distribution model it is necessary to allow for storage to be distributed among the nodes in a general fashion. Therefore it was clearly a requirement to introduce a means of updating the contents of shared segments on whichever nodes they had been instantiated. Thus the ring has the ability to broadcast a store write to a nominated segment in all connected nodes. Since the only system-wide storage address is the virtual address within a virtual machine the broadcast address is always such a virtual address.

To enable procedures to be similarly distributed among the nodes in a general fashion there is clearly a requirement to introduce a method for invoking, from the calling procedure's node, a procedure within the same virtual machine but in a different node. Thus the ring has the ability to transmit a 'vectored event' to a nominated virtual machine in all connected nodes.

3.6 Node support

To complement these architectural facilities of the ring there need to be equivalent support facilities in every node.

Generally, as far as mapping is concerned, the architectural model already contains adequate facilities for defining the distribution of storage as far as any one node is concerned, i.e. local or shared, to indicate accessible state and present or not present to indicate processable state with respect ot a particular node.

Clearly, however, each virtual machine has to maintain a virtual-to-real mapping for the instantiation of that virtual machine in any node. Since nodes are not physically identical, and indeed may obey different order codes, this mapping is on a per-node basis. This mapping, as is to be expected since the operational resources maintaining the mapping are public, is held in segments which are 'public' to the system. In other words these are *identically* named locations in every virtual machine. However, the contents are different for each node and hence a new type of public store is introduced called nodal. This store is addressed as a normal public store but there will be a separate instance of each such segment on each node. Thus, clearly the contents are not of system-wide interest and therefore updates to these segments will *not* be broadcast around the 'ring' and hence the contents for any node will not be 'visible' outside of the owning node. In this sense, in the same way as virtual machines have segments some of which are local, the set of all virtual machines have public segments some of which are nodal. These distribution mechanisms are shown in Fig. 2.

	single virtual machine	multiple virtual machines
single node	local	nodal
multiple nodes	public	public

Fig. 2 Distribution of storage mechanisms

As far as procedure calling and exiting are concerned the architectural model already contains facilities for passing of control within a virtual machine:

– without privilege increase – level call
– with privilege increase – inward system call
– with privilege loss -– outward system call
– by interrupt – interrupt system call

To allow for procedure distribution, support for the vectored event needed to be added. This was achieved by two generalisations reflecting the source and destination attributes of a procedure call and exit. The first was to define the action taken by a node on the receipt of a vectored event to be the invocation of an interrupt procedure within a nominated virtual machine. The second was to allow the invocation of a microcode or privilege code function by a code procedure through a primitive procedure call. Of course, as in the existing model, these types of call are distinguished by the call descriptor being implanted by the 'loader' and not by the calling procedure. The vital importance of this delayed binding capability was discussed in the previous paper[3]. Thus the existing interrupt action is replaced by the ability to call and exit from a procedure in the caller's virtual machine running in another node. The introduction of primitive procedures allows processors to deal with the support of a full VME procedure interface while not supporting other VME characteristics.

There are two motivations for this; the first to allow 'subarchitectural' functions in dealing with intimate hardware interrupt concerns (while still retaining normal procedure call conventions;) the second to allow alternative order codes in procedure bodies, and thus to develop a set of special-purpose processing engines. These distribution mechanisms are shown in Fig. 3.

	single virtual machine	multiple virtual machines
single node	local call	vectored event
multiple nodes	vectored event	vectored event

Fig. 3 Distribution of procedures mechanisms

Lastly to enable a set of distributed virtual machines to co-operate so as to present when necessary a coherent single image, the ring and attached nodes must also be able to support the distribution and actioning of a system-wide internodal semaphore by means of which system-wide state changes can be synchronised. An example is the updating of a variable that is stored not nodally but publicly.

Fig. 4 summarises the distribution mechanisms which the two generic physical components support to implement the nodal model.

4 Model exploitation

Now, given this architectural model, which allows for the flexible distribution of

both virtual machines and procedures throughout a 'ring'-linked system, we can construct some useful instances, i.e. products, of this model.

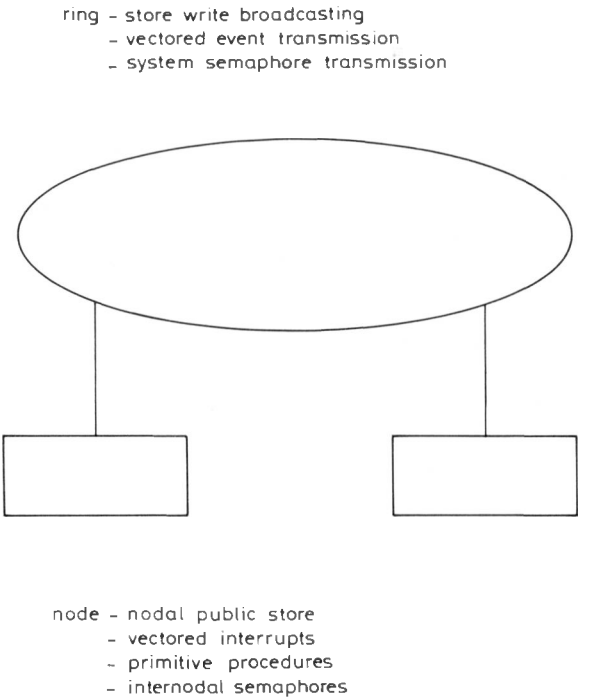

Fig. 4 VME nodal architecture support

To do this, however, some additional implementation rules need to be added, covering, for example:

- real-time clock behaviour
- node connection/disconnection
- node error management
- node failure
- system load
- virtual machine node scheduling.

Discussion of the details of these decisions for the first products of this generalised VME architecture, those systems based on nodes, is beyond the scope of this paper. What does concern this paper is the way in which the architectural features have been exploited in constructing these systems.

Basically, the facilities have been used to generalise the way in which the real resources of mainstore and input/output streams are managed in a multimaster,

multistore, multiprocessor computing system. The mapping and use of these (real) resources are the responsibility of the most privileged set of VME procedures, the kernel set. Hence the software changes necessary to achieve this generalisation are also contained in this kernel set.

4.1 On local routes

The key to effective multiprocessor, multistore operation is clearly the minimisation of those cross-system messages which cause single-server effects on the system: i.e. messages requiring the co-ordination of two or more nodes before they can be processed. Hence, in general, the strategy is to utilise local storage updates, or at least nodal storage, and local semaphore (interlock) mechanisms requiring no system-wide scheduling and hence no co-ordination.

The cause of such system-wide bottlenecks is basically the introduction of nonlocal routes between processing elements. These routes exist for a variety of purposes, reflecting the different mechanisms used to communicate/co-operate between processing elements. Thus examples might be procedure calls, interrupts and store updates. In each case, if the route between two processing elements is nonlocal, then a third agent is involved in the communication process (the global scheduler for this route) and hence some form of non-multimaster processing is introduced.

The routes may clearly be physical or logical, and, for example, the Kernel procedures supporting the existing multiprocessor single-store systems are faced with global management of the storage system. This had effects on both input/output management and virtual storage management procedures/global scheduling implementations. This approach, forced by the physical hardware implementation, had many undesirable effects – scheduling bottlenecks, resilience properties and complexity of software. Clearly, with the introduction of the mechanisms for allowing local routes to be established in all these cases, the software structure can be both simplified and made more efficient – a normal consequence of simplification.

Basically the system approach can be illustrated by looking at the coupling of two types of processing elements.

First, the virtual processing element, the virtual machine: the objective in this case is to allow a coupling of virtual machines so as to achieve a minimisation of system-wide scheduling events and a maximisation of nodal (restricted to one node) scheduling events. This is achieved by connecting virtual machines together when necessary, via a set of explicit message routes rather than implicit shared store mechanisms. The scheduling of the various real and virtual resources of a given virtual machine can then be achieved by procedures within that virtual machine with local 'nodal' access to the various resource management controls; each node locally defining its (the node's) view of the system.

Secondly, the real processing element, the node: the objective in this case is to allow the coupling of nodes whether they are general processors or specialised controllers, for example the ICL CAFS[4], so as to provide local, from any node, virtual paths to all other nodes. This ensures that only synchronisation between the path-end nodes on the use of a specific path is required and no system-wide synchronisation with all the other nodes is necessary.

The previous paper[3] had established

'what the VME/B design identified as the four main system concepts necessary for the support of both distributed and modular system developments. That is, system description, user entry, object naming and object manipulation.'

These provided the mechanisms for users to select objects, from a named store, into a tailored personalised processing environment (of VME procedures within a VME virtual machine). In the generalised model, at a naming level, there is no real change to this system to accommodate the more generalised nodal architecture. What is affected is the possibility of providing a more general set of processing capabilities. The existing system had provided, through a system of VME currencies, the mechanisms for describing the 'correct' behaviour of any entity processing a selected object.

4.2 Virtual streams

The nodal architecture extends this concept into the virtual streams which interconnect the differing node types in the node/ring structure. Thus, given the general intention to construct a system by the flexible, including runtime, distribution of resources across a set of such interconnected nodes, any given vitual machine may contain resources instantiated on a number of processing nodes. Each such node establishes a local (to the node) communication mechanism (a currency) between the nodes. This currency, in this case represented as a stream, defines the set of operational resources (procedures) available to the given virtual machine on a given node. This is analogous to the process, described in the previous paper[3], used to define the set of manipulation features made available by a given object selection from the VME object catalogue.

Thus take the case of an input/output stream linking a VME order code node to an ICL CAFS-ISP (content addressable file store) search processor. This stream would define the set of procedures to be remotely executed in the latter to service a request (procedure call) from that part of the virtual machine instantiated in the VME order code node.

The whole process of call and exit from a remote procedure is enabled by the vectored event mechanism outlined earlier; and it in turn, in concert with the multimaster ring structure, enables a true distributed-procedure multimaster system. However, the single-system image presented by a single-server store system can be retained if needed by use of the distributed store-write mechanisms.

5 Levels 30 and 80 and the nodal architecture

Nodal VME systems based on Levels 30 and 80 nodes have been developed with the following components:

- *Architectural 'ring' instantiations*
  - a fast (50 Mbytes/s) token ring – MACROLAN
  - an open systems (10 Mbytes/s) ring – OSLAN

- *Architectural 'node' instantiations*
  - VME order code processor – Level 30, Level 80
  - high-speed peripheral processor – HSPC
  - CAFS-ISP search processor
  - OSLAN-based peripherals and processors.

The VME order code processor is capable of executing both 2900 and alien order code support, the latter currently restricted to ICL 1900; it is connected to both MACROLAN and OSLAN.

MACROLAN is used for

- distributed store updates
- distributed semaphore handling
- connection of HSPC nodes

OSLAN is used for

- connection of communication couplers
- connection of slow peripherals
- connection of distributed workstations and workstation clusters

These systems are currently packaged as single 'operating system' offerings, thus allowing existing VME users to migrate to nodal systems with no change in system administration practices, application packages or databases. Yet these users will gain the full benefits of distributed nodal systems with respect to the modularity of system composition and error management/containment practices.

6 Conclusion

It is, I hope, evident that architecturally there are no obstacles either to greater system distribution or to the insertion of more varied order code and storage nodes into VME nodal infrastructurees. The need for this variety of component will become an increasingly important consideration as the use of VLSI components and the so-called fifth generation and applicative programming styles become dominant.

However, it is vital that these new subsystems can be easily inserted into existing and functioning systems to allow the rapid exploitation of new techniques, both

hardware and software, for specific application building and yet not implying the reimplementation of expensive system infrastructures at great cost in both time and manpower.

It is, I hope, also evident that the distributed support generalisations of the architecture, although allowing much greater flexibility in system construction, are limited to detailed real-to-virtual mappings and hence have only affected the implementation (but not the interface to) the kernel portion of VME procedures. No change has been necessary to the rest of the system although clearly there are great opportunities for functional distribution of VME procedures in the future.

Acknowledgments

A repeat of the previous papers acknowledgements is called for; however, this time my real debt is to my wife for both persuading me that 'whooping-cough' was a stay-at-home disease and then selflessly providing the environment conducive to the production of this paper.

Special thanks are due to Di Turner and Jack Howlett for, in their individual ways, turning my scribbles into a readable paper.

References

1 HOFSTADTER, D.: '*Godel, Escher, Bach: an eternal braid*', Penguin Books.

2 PARKER, T.A.: 'Security in a large general-purpose operating system: ICL's approach in VME/2900', *ICL Tech. J.*, 1982, **3** (1), 29-42.

3 WARBOYS, B.C.: 'VME/B a model for the realisation of a total system concept', *ICL Tech. J.*, 1980, **2** (2), 132-146.

4 MALLER, V.A.J.: 'The content addressable file store – CAFS', *ICL Tech. J.*, 1979, **1** (3), 265-279.

Processing node of the ICL Series 39 Level 30 system

D.W. Ashcroft
ICL Mainframe Systems Development Division, West Gorton, Manchester

Abstract

The principal constituents of a node in the Level 30 system are the main store (4-16 Mbytes), the processor (one or two) and the input/output controller. The processor itself includes special units for the processing activities, including numerical computation, for store access and for scheduling. The paper describes these units and their functioning, and also gives the general physical details of the node.

1 Background

September 1981 is when it all began. Prior to this we had been working on the design of a cheaper 2966, using a 750-gate TTL array made by Fairchild (FAST). In fact, we had had a couple of chips made and were processing the results when the decision was taken to use the 8000-gate CMOS array made by Fujitsu, the

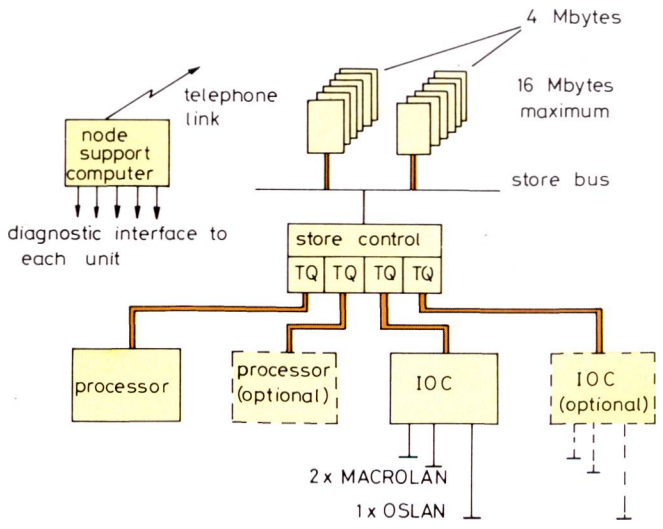

Fig. 1 Schematic diagram of Level 30 node

C8K; and at the same time fibre-optics was chosen as the fast peripheral connection (i.e. for discs and magnetic tapes) and an Ethernet type of connection for slow devices such as printers and communications links. The processor, store and connections to the fibre optics (MACROLAN) and Ethernet (OSLAN) were to be grouped together as the processing node and housed in a single cabinet, the volume of which was later determined as under half a cubic metre (Fig. 1).

The building bricks within the cabinet were chosen as large 'motherboards', 42 x 52 cm, having 14 layers, including three horizontal and three vertical for logic tracking and a separate layer for clock distribution. Because of the intrinsic reliability of the C8K chips they are soldered directly on to the motherboards; RAMs and other different components are housed on 'daughterboards 10 x 19 cm in size which plug into the motherboards via 96-way Euro connectors (see Fig. 2). The majority of the daughterboards are surface tracked with buried power. Connections between the motherboards are via 20-way or 50-way stripline connectors.

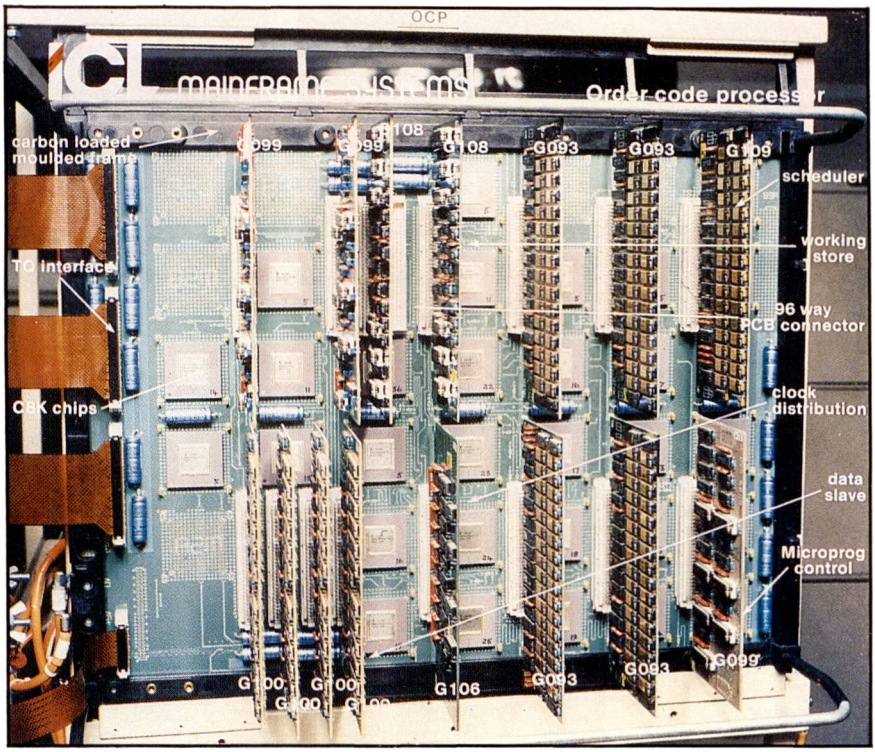

Fig. 2 Printed circuit board assembly

There is space for five motherboards in the processing node and these can be occupied by up to three different types of unit: processor, store and input/output controller (IOC), of which the processor and the IOC can be duplicated for purposes of increased power or connectivity. Thus the node can contain

a minimum of three or a maximum of five motherboards. The store motherboard contains, as well as the store (which can be from 4 to 16 Mbytes), the clock and the clock distribution (all the components of the node run on the same 190 ns clock) and the node support computer (NSC), a 8088-based diagnostic processor used as the engineering interface to the node.

Connections between the motherboards are restricted to the main store interface, known as the TQ interface, the clock and the diagnostic interface from the NSC. All connections are distributed from the store motherboard and there are no connections between any of the other four motherboards.

The input/output controller (IOC) provides the connection from the I/O to the store. The input to the store (TQ) is 32-bit multiplexed address and data; connected to the I/O are two high-speed couplers (HSC), each driving a fibre-optic MACROLAN connection, and a low-speed coupler (LSC) driving a coaxial-style OSLAN connection.

Table 1 Basic physical data for the Level 30 node

Cabinet (including five motherboards)		
size		60 x 60 x 90 cm
	approximately	2 x 2 x 3 ft (wide)
power		650 W
weight	approximately	300 lbs 66 kg
Processor (per motherboard)	C8K chips	30
	daughterboards	16
	engine microprogram	64 bytes x 32 k
	slave store size	
	data	32 kbytes
	instructions	16 kbytes
	CPRs	4 kbytes
IOC (per motherboard)	C8K chips	18
	daughterboards	12
	I/O connections	
	2 x MACROLAN (fibre-optics)	50 Mbits/s each
	1 x OSLAN	5 Mbits/s
Store (including clock and NSC)	C8K chips	11
	daughterboards (4Mbytes)	14
	additional per 4 Mbytes	6
	access time	6 beats
	cycle time	5 beats
Order codes	2900	
	1900 (via Concurrent Microcode Emulation)	
Beat	190 ns	

The node support computer (NSC) has two main purposes: initial program load (IPL) and diagnostics. At switch-on the NSC, after self test, loads the microcode for one of the HSCs and then, through this coupler, establishes communication

via a high-speed disc controller (HSDC) with a disc containing the remainder of the node microcode. After loading the microcodes into the various units the NSC loads a 2900 IPL program into the main store, which when activated loads the full VME system. All the node registers incorporate loop-spinning registers which enable them to be read or written to by the NSC for maintenance and fault finding. A telephone line connection capability is provided for diagnosis by a remote VME system in a diagnostic centre, running the diagnostic interface software VISA (VME System Inoperable Access).

The basic physical facts about the node are given in Table 1 and a summary of the essential components in Table 2.

Table 2 Details of Level 30 node processor

Chips		
scheduler	4	
engine	12	
store access unit	8	
computation unit	5	
diagnostic control	1	Total 30
Daughterboards		
microprogram store	4	
slave store	4	
working store	2	
slave control store	2	
computational unit control store	1	
clock distribution	1	
microprogram address monitoring plus	1	
scheduler RAM store	1	Total 16
RAMs		
microprogram store	64 bits x 32 k	
working store	36 bits x 2 k (duplicated)	
store access unit control store	112 bits x 2 k	
computational unit control store	56 bits x 2 k	
scheduler RAM store	30 bits x 16 k	
slave stores (total)	96 bits x 16 k	

2 The processor

Like the ICL 2966 and earlier 2960 the Level 30 processor comprises a scheduling unit, a processing engine and a store access unit; it has also a computational unit to assist in fixed-and floating-point arithmetic. The sequencing of an instruction through the processing unit starts with this being decoded in the scheduling unit and divided into one, two or three tasks which when obeyed sequentially perform the function of the instruction. These tasks are passed to the microprogram-controlled processing engine in the form of the first microprogram address of the sequence to be obeyed, together with operand information. During the execution of the tasks the microcode requests the store access unit for the data required from the main store, this unit acting as interface between the processor and the main store. To minimise data access time

and store utilisation the store access unit includes slaves for data, instruction and current page registers. Any multiplication, division or floating-point add or subtract operations required within the tasks are passed by the engine to the computational unit. Fig. 3 gives a block diagram of the processor.

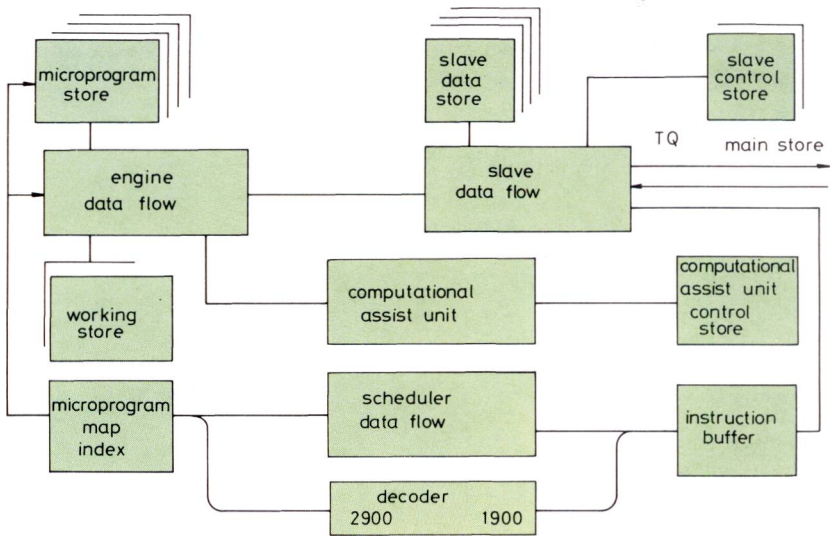

Fig. 3 Block diagram of the processor

2.1 Scheduler

This unit has the job of decoding the primitive level instruction (PLI) and breaking it into a number of tasks to be performed by the microcoded engine. The instructions are obtained from the store access unit and the scheduler contains a 16-byte buffer to reduce store latency; a request for a further 4 bytes is made as soon as there is sufficient space in the buffer. When a program jump is encountered the buffer is invalidated and a store request for the new instruction data is initiated by the engine. Copies of the relevant visible registers that contribute to task variations, for example the program status register (PSR) and the descriptor register (DRO), are kept by the scheduler so that the tasks can be more specific.

The decoding of the instruction is contained within a C8K gate array, so the task start addresses are hardwired. To allow flexible management of the microprogram store of the processing engine each task selected addresses a 16 k deep RAM, the output of which provides a changeable task address, and parameters to the engine.

Fig. 4 shows the scheduler data flow split across three C8K chips; one contains the instruction buffer and sequencer, another the manipulation of the program status register (PSR) and the parameters required by the engine with each task,

and the third the logic for decoding the 2900 instruction set. In parallel with the 2900 decoder is a 1900 decoder for use when the 1900 instruction set is being obeyed.

The construction of the scheduler is such that the number of steps (beats) per task is not less than two and the number of steps per instruction is not less than three.

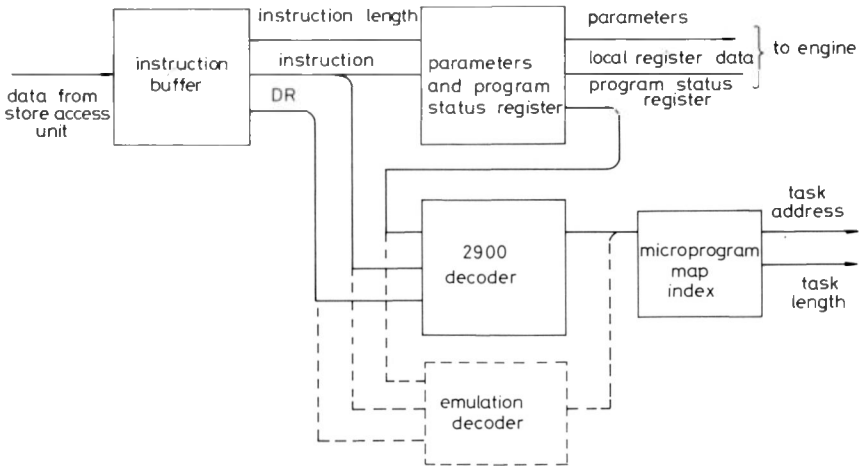

Fig. 4 The scheduler

2.2 Processing engine and computational unit

The engine consists of a 32-bit mill, full cyclic shifter, large register file, microprogram store and working store, all connected together as shown in Fig. 5.

The mill is spread across four gate arrays, each 1 byte wide, and can operate either in 2's complement binary or in binary coded decimal mode for 32-bit add or subtract. Various checks on the output provide conditional information to the microprogram for jump conditions. The shifter, on one chip, is connected in parallel with the mill since timing conditions do not permit serial connection and is capable of up to 31 bits of shift in single length, multilength or cyclic logical mode. The multilength facility is provided by a 32-bit shift end register which can be loaded with the unshifted operand. Masking is provided by passing the shifted data through a logical mill contained on the mill gate arrays. The shifter also handles Compress and Expand for 1900 emulation.

Also in parallel with the mill is the computational unit, which obtains its operating data directly from the output of the register file. This is built from five gate arrays, of four types, and performs operand alignment, add or subtract functions, multiplication via an 8 x 24 bit multiplexer register, a divide function performed simply by subtract and shift, and addressing and sequencing of a

RAM control store 56 x 2 k deep. The divide function is speeded up by forward searching three places for zeros, which when found allow skipping of the subtract step.

Single-length (32 bits) performance in the computation unit is:

floating point add/subtract	2 beats
multiply	3 beats
divide	16-17 beats, average

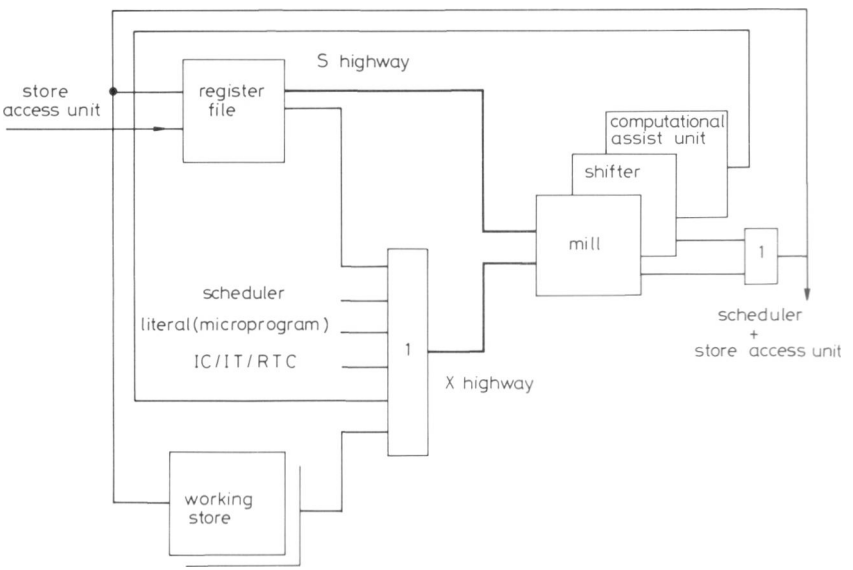

Fig. 5 Engine data flow

2.2.1 S and X inputs: The S input to the mill comes from a register file 32 bits wide by 52 words deep, spread across four gate arrays. The structure of this file allows two reads and one write within one clock beat, in addition to a write to one of three defined registers with data returning from the store access unit.

The 52 locations within the register file are split into three blocks. The first, of 18 locations, contains the range-defined registers for the primitive level instructions being obeyed. This block is copied into the second block, also of 18 locations, at the end of each instruction or at a checkpoint defined by the microprogram; this second block is used for restart purposes in the event of an intermittant fail or virtual store interrupt. The remaining 16 locations are used for temporary data storage within a primitive level instruction and are not valid between instructions; the three highest-numbered locations within this block are reserved for data returning from the store access unit.

The X input comes from a multiplexed assortment which includes a microprogram literal, a scheduler literal, the instruction timer, instruction counter, real-time clock and working store. The last, a 32 bit wide, 2 k deep RAM store, is used mainly by the microcode that interfaces between the software and the I/O and is duplicated for node retry purposes; the duplicate also keeps a copy of all data written to the register files, again for retry purposes.

2.2.2 Further features of the processing engine: There are three more chips within the engine. One supplies the working store and its backup address, another the microprogram store address and sequencer and the third the auxiliary functions such as interfaces to the store access unit (described below) and to the scheduler, and conditional jump manipulation and flag setting.

The microprogram that controls the engine is 64 bits wide, 10 of which are used for error detection and correction. The remaining 54 bits are split into five fields as shown in Fig. 6. The first 12 bits are used to control the operation of the mill, the shifter or the computational unit, these being mutually exclusive; the two most significant bits define which facility the remaining 10 bits are to control. The next two fields, of 8 and 12 bits, respectively, dictate the operand to be applied to the mill, shifter or computational unit; the S operand can be used also as a 5-bit literal entry to the mill. The 12-bit auxiliary function field includes the store access unit function passed to the store when the engine makes a store request and deals with microprogram jump conditions and flagsetting facilities. The final 10-bit field, the Y operand, controls the destination of the output from the mill or shifter; it can be used also as an 8-bit extension to the X operand literal, giving a 16-bit literal.

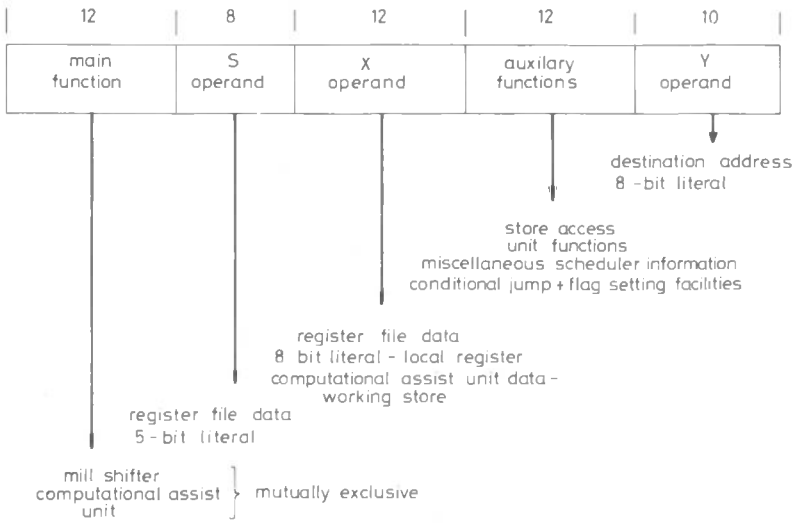

Fig. 6 Structure of microprogram word

Output from the engine to the store access unit and scheduler is via the output of the mill.

2.2.3 Store access unit (SAU): This acts as an interface between the processor and the store. To minimise data access time it contains 32 kbytes of data slave, 16 kbytes of code slave and 4 kbytes current page registers; up to a maximum of four requests from the engine or scheduler can be queued and there can be up to a maximum of two requests outstanding from the main store, accessed via the TQ interface. Incoming data from TQ can be either data previously requested from the main store or I/O messages from the MACROLAN or OSLAN couplers within the node; this incoming data is loaded into one of three input buffers, one being reserved for I/O messages, another for 'move' data and the third for up to four data words, representing two requests. The special buffer for 'move' data acts as a pre-fetch buffer, the aim being to have data from a read string available to the engine on demand.

Fig. 7 gives a simplified block diagram of the SAU. To minimise data highways between the unit and the scheduler a copy of the program counter is kept in the SAU, which transfers this address to the queue when directed by the scheduler and automatically increments the copy to point to the next word of store. The engine updates the copy when a jump is encountered.

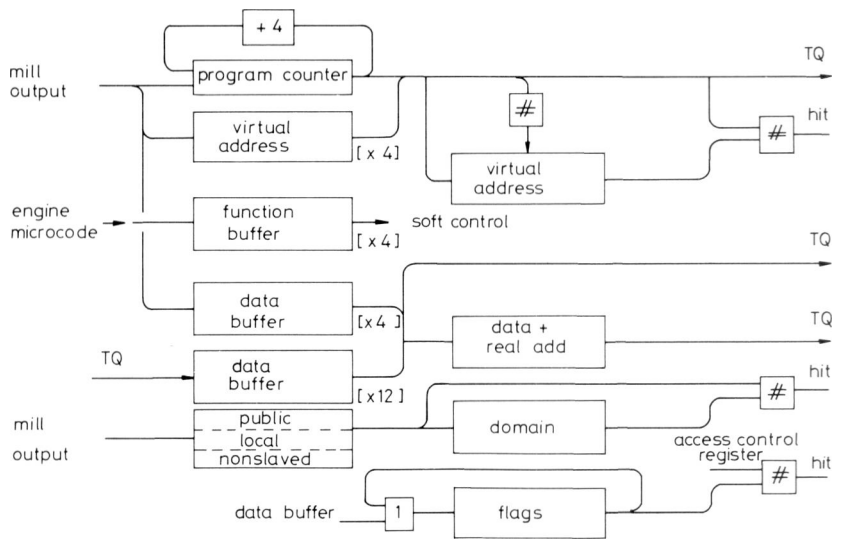

Fig. 7 Store access unit (SAU)

To allow preservation of slave data across interrupts, and to minimise slave store invalidation time, each slave entry includes a 16-bit domain tag which is compared with the current domain register at each request and must correspond in order to obtain a data hit. To invalidate the slaves the current domain register

is incremented by unity. The current domain register is stored, along with the visible registers, when the current process is interrupted and is retrieved when that process is restarted.

The logic for the SAU is spread across eight gate arrays. One concerns the queue, the data and the current page register; the second, the data hit-or-miss comparison; four identical arrays deal with the slave data flow, which is controlled by a RAM control store addressed and sequenced by the seventh; and the eighth controls the interface between the unit and its neighbours, the engine, the scheduler and the TQ highway.

2.2.4 Time sequencing: Fig. 8 shows how the various units combine to obey a primitive level instruction, starting in the scheduler instruction buffer; each horizontal division represents one machine beat of 190 ns.

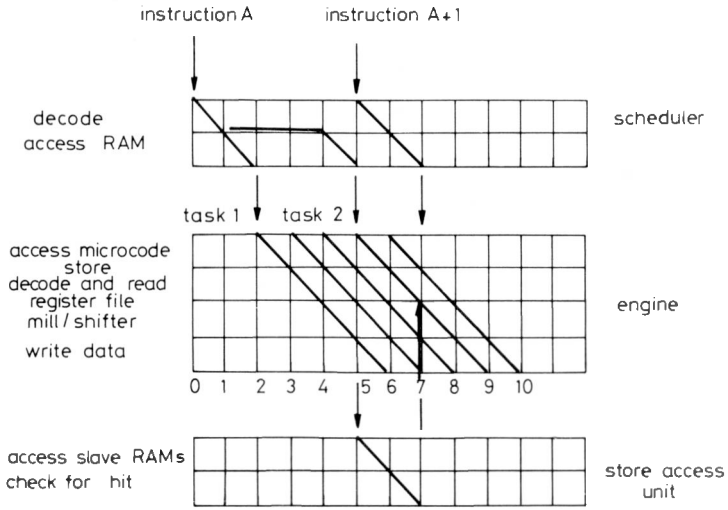

Fig. 8 Level 30 processor pipeline

At the end of the first beat after the start the decoder splits the instruction into two tasks, T1 and T2, say. In the next beat T1 accesses the scheduler's 16 kbytes RAM and passes the microcode start address to the engine. The engine requires three microprogram steps to perform T1, each step involving a sequence of four actions: access the microprogram store, decode this and access the register file, perform the mill or shifter operation, write the output of the mill or shifter to the register file or working store. Each action takes one beat and the three steps are started at intervals of one beat, so T1 is completed at beat 8. T2 involves the same actions but needs only two steps; it is started at beat 5, so the whole process of obeying the instruction is completed at beat 10. If a store request is required T1's mill output at beat 5 is fed to the store access unit (SAU), which requires two beats to access and check the data before returning hit data to the engine or making a store request; data returned from the SAU can be used by the engine in the next beat. The next instruction enters the scheduler at beat 5,

at the same time as the latter starts to perform T2, so that instructions are obeyed at intervals of 5 beats.

3 How it all worked out

What made this development so different from that of the 2966? Certainly in the early days the introduction of LSI design in the form of the C8K gate arrays occupied all our thoughts. Each of these contains the equivalent of 200-300 MSI chips, and with the turn-round of a design modification in Japan taking about 12 weeks we clearly needed a mechanism for getting the design right before turning it into silicon, and a way for overcoming design problems quickly before a chip turn-round. These problems were tackled by using simulation at unit and gate level before committing the design to silicon and by using microprogramming techniques to overcome any design problems.

The simulation was in fact performed at the three levels of unit, chip and gate, a unit typically comprising ten chips. That at unit and chip level was supported by an environment called SAUL – simulation at unit level – in which the models, written in the ICL S3 language, defined all registers and inputs and expressed all outputs in terms of these. The unit level model was used to validate microcodes and, combined with other unit models, to test the system. When the unit and chip models had been validated the design was captured by inputting to the DA4 design automation system. A gate level model was then generated by running MSIM (multilevel simulation) on the captured design; patterns to run on this model were generated by running test software on the unit model at the SAUL level and capturing both the input and output patterns produced. Finally, the chip design was validated by applying these input patterns to the MSIM model and comparing the outputs of the unit and the chip models.

The basic gate of the C8K chip is a two-input NAND; these gates are combined to produce a cell library in terms of multiplexers, register, parity gates etc. The number of basic gates on the chip ranges from 5000 to 7500, which translates into about 500-700 cells. To reduce simulation time the MSIM model was produced at the cell rather than the gate level.

The amount of simulation performed, and the number of patterns applied to the chip model, were controlled by two factors. The minimum was set by what was needed to give us confidence that we could microcode our way round any remaining problems, and the maximum by the amount of machine time we could afford. During the simulation phase we had seven 2966 machines dedicated to design automation capture, simulation and production of manufacturing data for Fujitsu, of which a minimum of four were used for simulation.

At the unit level the simulation test programs consisted of functional tests written in the unit microcode and a number of 2900 test programs. Altogether approximately 250 000 beats were run through each of the SAUL models and between 30 000 and 150 000 patterns were applied to the chip MSIM models,

averaging about 80 000. Run times for the unit level model on a 2966 were about 6 beats/s, with an additional pattern extract time of about 9 beats/s; the MSIM model ran at about 2 beats/s.

The prototype of the Level 30 was switched on in August 1983 and we quickly proved the success of the simulation by running all of the test material used for the validation on the real machine without incident. Some time was then spent on validating the microcodes by running the 2900 Range Definition Test suite and driving real I/O, culminating in January 1984 in running a simple batch job under VME on the set of chips received in August 1983.

Input/output controller and local area networks of the ICL Series 39 Level 30 system

P. Broughton
ICL Mainframe Systems Division, West Gorton, Manchester

Abstract

The input/output controller (IOC) provides all the normal connections of a Level 30 node to input/ouput devices, which are connected via MACROLAN or OSLAN into couplers, purpose built for each type of local area network (LAN). The paper outlines the operation of each coupler and its LAN and highlights important or novel features of the designs.

1 Introduction

An important part of ICL's networking policy is the provision of distributed mainframes such as the Levels 30 and 80 of the Series 39. To design a mainframe with all its system and I/O interconnections made by local area networks was to break new ground. During the design phase of the input/output controller the networks to which it was to connect (MACROLAN and OSLAN) were themselves being developed, and to a small extent they continue to change even now. When to these unknowns are added the parallel developments of the Fujitsu chip (the C8K gate array) and the ICL design automation system, plus the commitment of the design to unmodifiable LSI, it will be seen that the IOC has been a very significant and successful design achievement.

2 IOC overview

The system described below comprises five major parts: the coupler interface concentrator (CIC), the high-speed coupler (HSC), the low-speed coupler (LSC), the MACROLAN and the OSLAN. These are shown in Fig. 1.

Connections to two very different forms of LAN are provided. MACROLAN is a high-speed LAN running as a 50 Mbit serial interface and is used to connect fast I/O streams, such as the main filestore on the FDS 300 discs, via the high-speed disc controller (HSDC). It is also used to interconnect the nodes of a multinode system since data passing between nodes to be carefully synchronised and fast. OSLAN is an international standard LAN based on Ethernet; it is slower than MACROLAN and running as a 10 Mbit serial interface, is used to connect to slower I/O streams such as printers and VDUs. OSLAN will frequently be

found as an existing installed network with connections throughout a building. Two IOCs may be fitted, giving each node the capability of connecting up to four MACROLANS and either one or two OSLANs.

A brief description of the main parts of the IOC follows. Two HSCs are provided on a single IOC to permit a Level 30 node to have independent MACROLANs for its I/O and its internode traffic, or to allow resilient connections to the I/O of a single node without the need for a second IOC. The importance of the HSC in multinode working has meant that its design has to have both high integrity to data and resilience to failure, whether of node or network. A large amount of thought and of hardware has gone into these areas and early Level 30 commissioning has shown the success of this in detecting faults and in recovery from network faults.

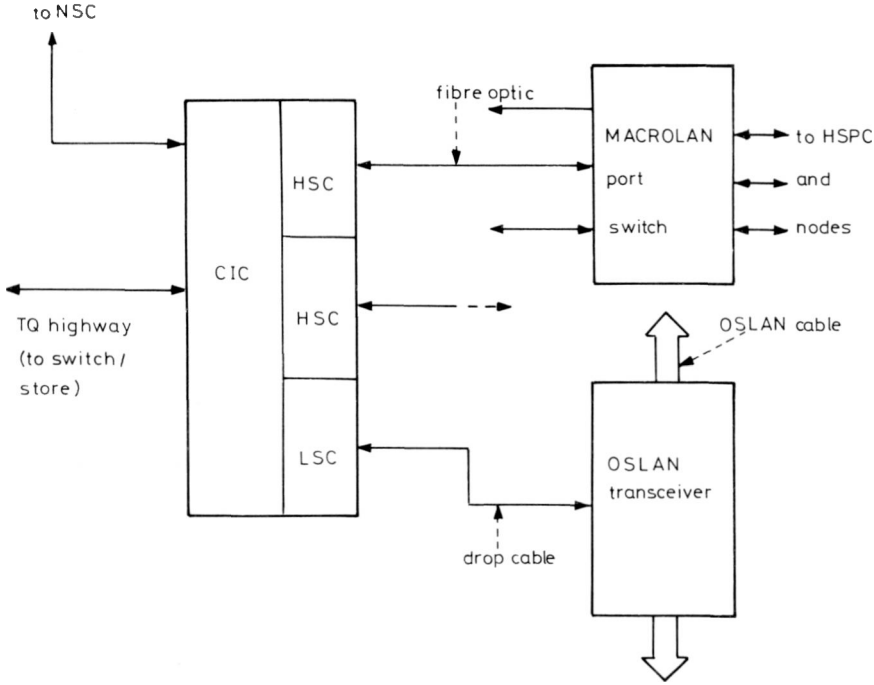

Fig. 1 Overview of the input/output controller for the Level 30 and Level 80 systems

3 Coupler interface concentrator (CIC)

The function of this unit is to sort out contentions for what is called the TQ highway arising from the three couplers of the IOC. TQ provides the only interconnection between the units of the node; details are given in the paper on the Level 30 store elsewhere in this issue[1]. The CIC can support the maximum peak data rate of 13 Mbytes/s that can be generated by the three couplers.

The concentrator also provides a secondary interface to the node support computer (NSC) so as to give this access to MACROLAN. This interface mimics

the TQ interface but because the NSC is a fairly slow computer it is not used during normal operation, as this would severely hinder messages destined for TQ. It is used during bootstrap operations, giving the NSC access to machine micro-codes and other material held on the system disc.

4 MACROLAN

MACROLAN is a novel interconnect scheme specially designed for the new systems. The heavy trunk cables of previous systems have been replaced by a fine fibre-optic 'cable' little larger than a telephone cable. The use of fibre optics brings with it the ultimate in radiofrequency interference tolerance and security against interception. A more detailed paper on MACROLAN has already been published in this Journal[2], so this paper will aim to give an understanding of its main features so as to allow a fuller understanding of the system.

MACROLAN is similar in function to the now familiar token-passing rings, but has been designed instead as a polled network (Fig. 2).

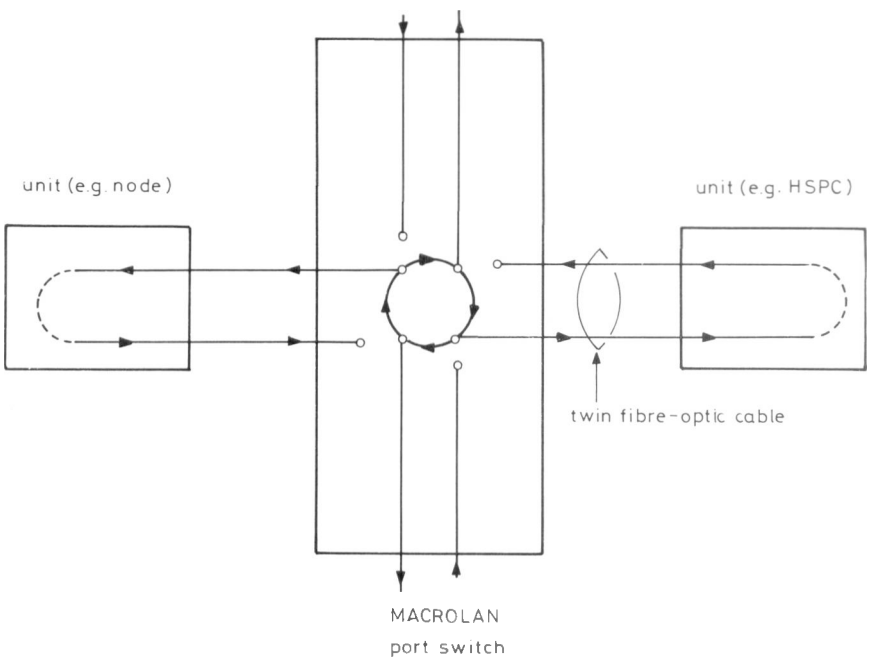

Fig. 2 MACROLAN configuration

All units required to be connected to the network are connected to a central box, the MACROLAN port switch (MPS), by up to 500 m of fibre-optic cable. The MPS polls each of its ports by transmitting a token to each in turn. The end units will either return the token if they do not require the network, or, if they do, send their message, which is then broadcast simultaneously to all ports by the MPS. Each end unit determines from an address field whether or not the

message is intended for it. The network can be extended by cascading port switch units. Operation of MACROLAN is resilient to end unit failure since any port can be configured out. This is not true of normal token-passing rings.

All end units contain one standard 18 x 10 cm printed circuit board which is a part of the MACROLAN system. This board contains the optical receiver/transmitter plus a specially designed TTL LSI chip which SERialises data leaving the end unit for the network and DESerialises incoming data (the SERDES chip).

The MACROLAN, therefore, presents a parallel interface to the HSC together with clock signals synchronised to that interface. Since these clocks are determined by the bit rate of MACROLAN, as strobed by the MPS, they are asynchonous to the node clocks.

4.1 MACROLAN protocols

The serial stream of bits on the network contains certain special patterns which are used by the receiving units. These patterns are distinguished by having greater than five consecutive bits at logical 1. To ensure that normal data bits never form any of these patterns the SERDES chip inserts an additional 0 bit whenever it transmits five 1s, and deletes this whenever it receives five, but not six, 1s. Examples of these special patterns are the token (011111110), the start-of-frame flag (0111111001111110) and acknowledge.

Flag = 0111 1110
FDA = frame destination address
FDF = frame destination function [protocol ID]
FSA = frame source address
data = up to 540 bytes
FCS - frame check sum
Acknowledge ['AK'] 0000 0011 1111 1110
Bit stuffing insert zero after five 1s

Fig. 3 MACROLAN data representation

token ('go ahead')
medium access frame format

Fig. 3 shows the fields of the MACROLAN message; these are largely self-explanatory but it is worth noting that MACROLANs are used here in two distinct ways that have different effects on the protocol. In one they connect nodes to peripherals, when the traffic is all input/output messages and data; a corrupted message can be discarded when it is detected and the operating system will repeat the request. In the other a single MACROLAN, the Public Write MACROLAN, interconnects nodes operating together as a multinode. The sequential polling of this MACROLAN is an important system characteristic. Data shared by all the nodes (public data) are held as separate copies local to each node and all copies are maintained in step. This is achieved by a write to

Public data being broadcast on the Public Write MACROLAN: since only one unit may broadcast at any one time, confusions by more than one unit writing to the word simultaneously in different local stores are avoided. Since it is crucially important to maintain the chronology of Public Writes (of which system semaphores are a special case) all units must be guaranteed to have received every message; to achieve this the Public Write MACROLAN (the MPS) is initialised to check that a link-level Acknowledge has been received from all the configured nodes before polling is allowed to continue. Such Acknowledges are not required on an I/O MACROLAN, where polling continues at the end of each message.

5 High-speed coupler (HSC)

The design of this had to cater for a number of special needs. The team was aware of the fact that, as design progressed, experience of networks as an intimate part of a mainframe system would grow and this would inevitably result in the need for changes. This awareness led to the design taking the form of a fairly powerful although specialised microcoded engine allowing major changes because of protocol changes and because of design revisions.

Since the SERDES logic and interface are synchronised to network clocks, parts of the CMOS HSC have to run in synchronism with them. To make these parts accessible to the diagnostic system and to the NSC a special clock distribution was designed which is switchable between SERDES and node clocks without loss of data.

The HSC is used during initial program load to fetch microdes and initialisation data for itself and for the rest of the node. Microcode and initialisation data capable of supporting this limited function had to be committed to programmable read-only memory (PROM) within the NSC.

Special routines to handle Public Writes were required to prevent data from being lost in the HSC if the node stopped, as for example on a retry by the order code processor (OCP), and so to avoid consequential system crash. The coupler also handles Public Write data in a system having two OCPs in one node, synchronising the data between the OCPs as though this had passed on to MACROLAN and also connecting them to a Public Write MACROLAN if the node is part of a multinode configuration.

5.1 HSC operation

The structure of an HSC is shown in Fig. 4.

At the heart of the unit is the message buffer, through which all data to and from both TQ and SERDES pass. Access to the message buffer bus is requested by the four interface controllers; TQ receiver, SERDES receiver, TQ transmit and SERDES transmit. Each controller contains a small amount of temporary buffering to overcome contention delays and operates as a Direct Memory Access

Controller (DMA), meaning that each has independent hard control to transfer a specified amount of data between its interface and the message buffer.

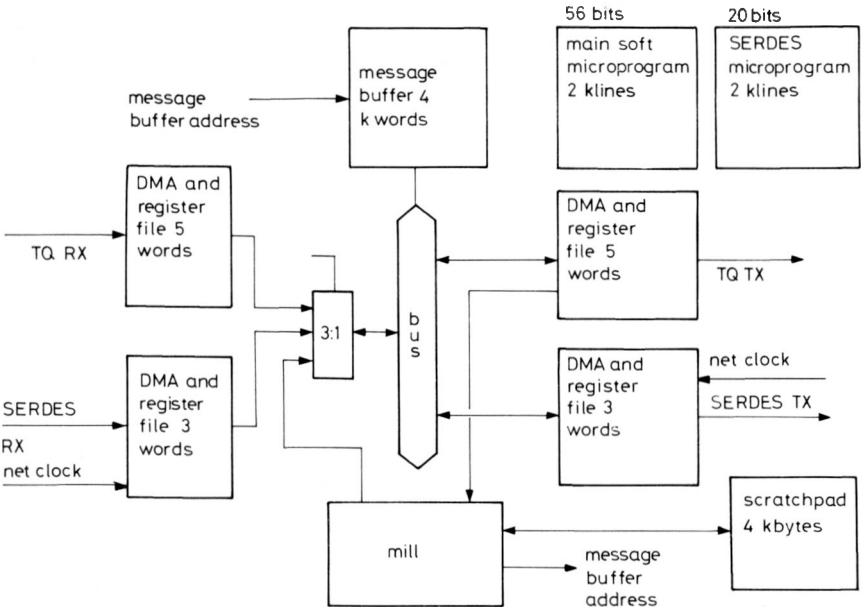

Fig. 4 HSC detail

Overall control of the HSC, such as scheduling these controllers, decoding messages and organising mainstore access, is by the soft controlled main microprogram; this is supported by a mill and a scratchpad for manipulating and holding pointers into areas of the message buffer. An additional soft controlled microprogram supervises the activities of the SERDES chip, which permits the coupler to be fairly soft towards changes to the transport-level protocol. Because the MACROLAN is not synchronised to the node, this soft control and the two SERDES DMAs are clocked synchronously to the LAN but asynchronously to the node.

As originally designed the HSC performed I/O operations by being asked by the order code processor to transmit individual frames of data to the high speed disc controller (see Fig. 3). The returning frame headers, containing either Acknowledges or data, were also passed individually to the OCP, which would then decide whether to ask the HSC to write the rest of the frame to the store or whether to discard it. At a late stage in the design it became evident that even with special microcoded functions to support this the amount of OCP mill power expended would be too great, and the HSC was therefore remicrocoded to perform the 'Dataphase' of the I/O transfers: put simply, this means that the HSC is given a set of mainstore addresses for a complete transfer of possibly many frames and controls the complete transfer, limiting the involve-

ment of the OCP to starting a transfer and handling its termination. The original mechanisms have been retained to handle exceptions such as disc error.

Incorporating this new protocol involved saving microcode space by separating out the operations that handle Public Writes and building a new microcode containing these; they are fetched by the NSC, using its PROM-loaded original microcode to make the HSC operational, and the latter is then reloaded with whichever microcode is appropriate, Public Write or Dataphase.

All this has provided a dramatic demonstration of the power of soft design, since chip redesign to achieve this result would have taken an impossibly long time.

6 OSLAN

OSLAN is based on an international network standard. It is expected to be adopted widely, possibly having been installed by the prospective customer before buying Level 30.

The OSLAN cable itself is a fairly thick coaxial cable of approximately 2 cm diameter, intended to be installed in the building, possibly even in the ceiling or under the floors; it does not enter the Level 30 cabinet. Connection to the cabinet is made via a thinner, more flexible coaxial cable, the drop cable, which taps onto the OSLAN cable via an active box, the transceiver. One version of the OSLAN transceiver allows additional transceivers to be added without breaking communication on the OSLAN. This uses a 'bee sting' connection through the OSLAN coaxial cable. All these OSLAN parts are specified by international standard and can be bought externally to ICL.

6.1 OSLAN protocol

Unlike MACROLAN OSLAN is a carrier sense multiple access – collision detect (CSMA-CD) network. This means that there is no master unit on the network nor a central unit such as the MPS to provide clocks. Instead a unit wishing to send a message across the network must first send a carrier frequency (or preamble) to synchronise the receiving units. Since there is no central unit or circulating token, there is no scheduling of the accesses to the network; so it is possible for

preamble = 1010 1010 repeated eight times
start of information = 1010 1011
destination address = 48 bits
source address = 48 bits
length field = 16 bits
destination link service access point = 8 bits
source link service access point = 8 bits
control = 8 bits
data = 0 to 1497 bytes
pad = padding of data up to 43 bytes minimum
frame check sequence = 32 bits

Fig. 5 Structure of an Ethernet data message

more than one unit to transmit simultaneously. This causes a collision on the network and consequently garbling of the messages. The transmitting units listen for this and, after jamming the network to guarantee that all other units are aware of the collision, stop transmitting and back off for a random time interval before trying again. Subsequent collisions simply increase the back-off time. Fig. 5 gives the structure of an Ethernet data message.

7 Low-speed coupler (LSC)

The design of the low-speed coupler had similar problems to that of the HSC.

Being designed to couple to an international standard interface it was inevitably dependent on specifications outside ICL's control. When design started, the only available specification was that of Ethernet (on which OSLAN is based) and LSC design was complete before this had been evolved by an international committee to the current IEEE 8023 specification. To handle this a very soft design was required.

The fairly complex mechanisms involved in driving the interface, such as handling collisions and synchronising to incoming messages, are all provided by a two-chip set. A number of manufacturers were designing these chips concurrently with the LSC design but even partly working chips were not available until after th LSC design was completed. Consequently, it was important not to commit to LSI any logic which could be affected by these chips. This was helped by the Intel chip set being designed to fit a microprocessor bus.

Once it had been decided that by using a microprocessor both the above requirements would be met, the low level of integrity checking of microprocessor parts became an issue. In particular, it was important that mainstore addresses should have high integrity, since an error here could undetectably corrupt any area of mainstore. This was resolved by containing all address manipulation within the well checked C8K logic, allowing the microprocessor to control only the reading and writing of these addresses.

7.1 LSC operation

The structure of an LSC is shown in Fig. 6. To the left of the dotted line the logic is contained within C8K LSI, whereas the remainder is bought-in microprocessor parts, including the chip set. The microprocessor and C8K chips are run on separate clocks (the normal node clock for the C8K), the interface between them being via the FIFO and by interrupt. Asynchronous microprocessor clocks are avoided in the C8K.

Once again the message store forms the data routing centre of the unit. Messages from the OSLAN are written autonomously by the chip set into message store and are eventually processed by the DMA and sent out to the FIFO and written to store via TQ by the C8K chip control and dataflow. The converse is true for messages to OSLAN.

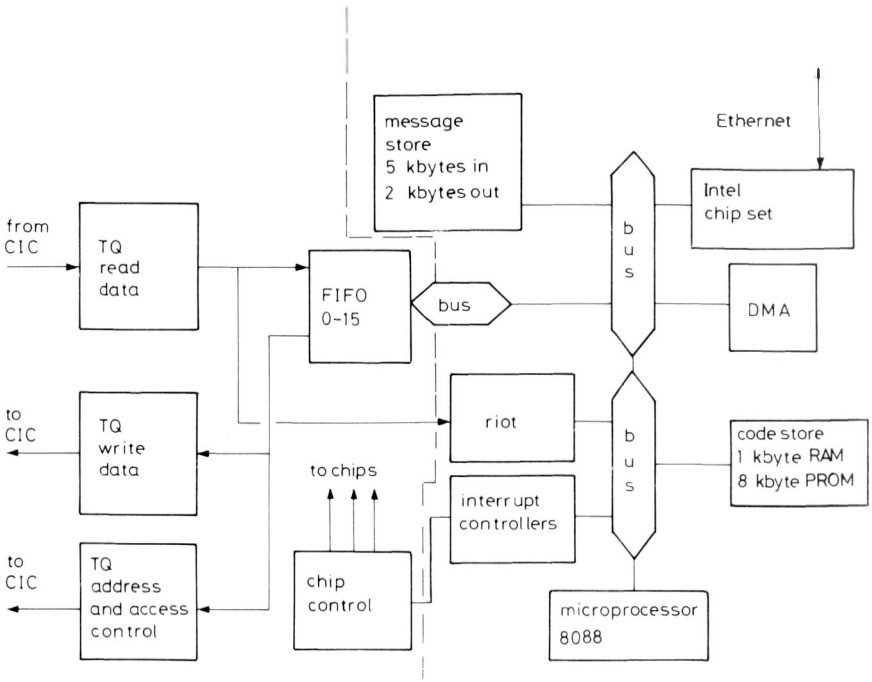

Fig. 6 LSC detail

To achieve this the microprocessor communicates with the OCP to find out the addresses to which the message must be written. The descriptors in which these are contained are written by the OCP to an area of mainstore known to the LSC. These addresses must then be loaded into registers in the C8K containing the base address of an area and the length to be written for the current block and next block. The next block information is prefetched as the current block is being processed.

The fairly complex control requirements of the C8K logic led to considerable design effort to achieve a low-risk design. To provide a separate soft controlled microprogram for these chips would not be cost-effective, especially with the large microcode store already provided for the microprocessor. The method adopted was to divide the C8K control sequences into fairly short simple steps which could be reliably checked with simulation. The overall control of these sequences (which to run and when) was by a microprogram word; this was set up by the microprocessor, giving it control of the chip sequences and allowing faults to be avoided by microcode change.

This method proved very successful, the LSC C8K chips having had no design iterations at all, even in the face of many changes due to OSLAN instability, chip set errors and medium-speed coupler (MSC) design.

8 Testing of LSC and HSC

Very few operational problems have been encountered within HSC and LSC; this has been largely due to the effective and extensive testing method which has been evolved. The principle of this testing is first to prove that the individual mechanisms of the couplers operate to specification. On its own this form of testing is insufficient because couplers of this nature can perform many of their operations simultaneously or can have memory of earlier operations due to buffering. Faults designed into these mechanisms are intermittent, timing sensitive or pattern sensitive: very difficult to isolate in a running system. To flush these faults out of the designs a type of test known as GENTEST (generate test) has been developed.

The principle of GENTEST is to generate a test sequence from randomly selected operations (with random data and data lengths), run the test sequence in the machine, check the results and then generate a new sequence. The program can be used to soak test the machine for as long as required, but any failing sequence can be simply repeated. It has been found possible to direct the tests towards particular fault types, e.g. by increasing the occurrence of certain events or by tuning the system to have high or low throughput. Very high throughputs have been achieved by running concurrent GENTESTs in two OCPs of a node to one or more HSCs and/or LSCs.

An important aspect of this test was that it was written by a member of the design team who was concerned solely with finding ways to validate the design but who had a detailed knowledge of its operation. This has brought benefits in that while the testing can be directed very accurately at difficult problems, the test method itself remains sufficiently random as not to depend on the writer predicting the problems for which to test.

9 Conclusions

1 The use of fairly powerful microcoded engines for the design of I/O controllers of the type described here has been shown to be very important.

2 Testing of the I/O controllers before system use, by GENTEST, has been very effective.

3 The IOC design has given the Level 30 high-technology interfaces to its input/output system. These are world leaders in a world growing to understand the importance of networks.

Acknowledgments

This paper would not have been possible without the ideas, enthusiasm and very hard work of the members of the Level 30 IOC design team whose work it describes. My thanks also to those members who helped by checking this paper.

References

1 BROUGHTON, P.: 'The store of the ICL Series 39 Level 30 system', *ICL Tech. J.*, 1985, **4** (3), 270-278.

2 STEVENS, R.W.: 'MACROLAN: A high-performance network', *ICL Tech. J.*, 1983, **3** (3), 289-296.

The store of the ICL Series 39 Level 30 system

P. Broughton
ICL Mainframe Systems Division, West Gorton, Manchester

Abstract

The store is the physical and logical centre of a Level 30 system. The storage system taken as a whole not only supplies and gives access to main store for order code processors (OCPs) and input/output controllers (IOCs) but also provides the communication routes between these units, supplies their system clocks and houses the node support computer (NSC). The paper outlines these functions, except those of the NSC, and highlights some novel features of the design.

1 Introduction

Design of main stores containing large numbers of dynamic RAMs has traditionally led to an essentially asynchronous design with many tuned delays to provide the waveforms demanded by RAM chips. This often means that main store returns data to the requesting processor after a time related only to these delays and not related to the clock beat of the processor. Attempts to relate the access time and clock beat are made difficult by inevitable changes of both clock beat and access time as design progresses. Also, within a VLSI design, tuned delays need to be avoided.

The Level 30 store has been designed to be internally and externally synchronous to the node clock and to provide soft control of the RAM timing waveforms and other control signals. This has enabled retiming of the store as clock beat and RAM specification changed; and has provided a very important insurance against design error.

To understand the significant features of the Level 30 store it is important to grasp the fundamentals of the node clocking system. A very large amount of design effort has gone into the design of this system which has succeeded in its aims: to eliminate the slugging effect of skew on node clock and to eliminate the risk of race hazards in the LSI logic.

2 Single and two-phase clocking systems

Level 30 has a two-phase system. The more common single-phase clocking regimes work on the principle of all latches being edge triggered and the paths

between any two latches being sufficiently long to prevent data skipping through a latch into the next latch because of skew in the clocks distributed to the two latches (a race hazard) (Fig. 1).

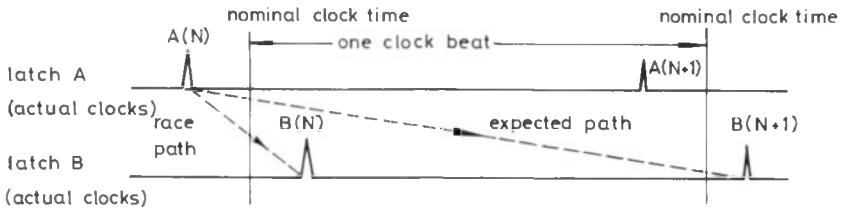

Fig. 1 Effects of skew in a single-phase distribution (time on a horizontal scale)

This race hazard occurs when data clocked into latch A at clock time $A(N)$ which should clock into latch B at clock time $B(N+1)$ skips through to appear in latch B at clock time $B(N)$. Additionally, if instead data were to travel from latch B to latch A then the time allowed for that logic path will be less than one beat, from $B(N)$ to $A(N+1)$. Thus, to allow for an unpredictable amount and direction of skew caused by electrical tolerances, the clock beat must be increased by twice the maximum theoretical skew.

2.1 The Level 30 two-phase system

In this system edge triggered latches are not used. Instead an intermediate latch is positioned approximately mid beat in every logic path. This latch is termed the phase 2 latch. Those latches which are normally a part of the design (the beat ends) are termed phase 1 latches. By designing the clock system such that the phase 2 clocks open the mid beat latches only after the last phase 1 latch has closed and that the phase 2 clocks close the mid beat latches before any phase 1 latch is opened all race hazards are eliminated.

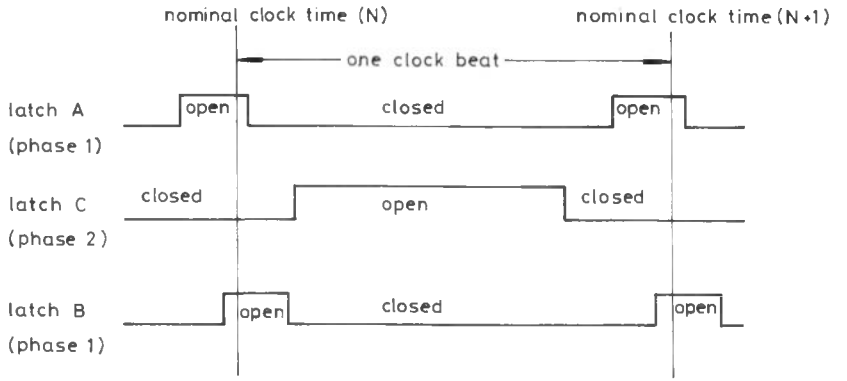

Fig. 2 Two-phase clocking system (time on horizontal scale)

It can be seen from Fig. 2 that data entering latch A as it opens at clock N cannot reach latch B until latch C has opened and closed, one beat later. Additionally, data reaching latch A after it has opened but in time for the nominal beat time (N) can start to set up the logic path from A to B. Provided that the path is set up at latch C before it closes then the time allowed for the logic path A to B is a full beat from N to (N+1). The impact of skew on beat time has been removed.

Hence this two-phase system has contained all the problems of minimum paths and race hazards within the design of the clock distribution, leaving all other engineers concerned only to meet the beat time. This has in turn allowed the design of an automatic path timing program to be simplified to the simulation of maximum delays.

The system has unfortunately imposed additional (though easily checked) paths on the engineer who has used the following three equations.

phase 1 to phase 1 logic path - $SA + SB \leqslant$ clock beat (1)

phase 1 to phase 2 logic path - $SA + SC \leqslant$ clock beat - $PW1 - SG1F - SG2B$ (2)

phase 2 to phase 1 logic path - $SB \leqslant$ clock beat - $SG1B - SG2F$ (3)

The left hand side of these equations can be derived automatically by simple addition of element delays (including latch delays). The right hand side is provided as a number by the designer of the clock distribution, who must strive to minimise skew to reduce the design constraints of eqns. 2 and 3.

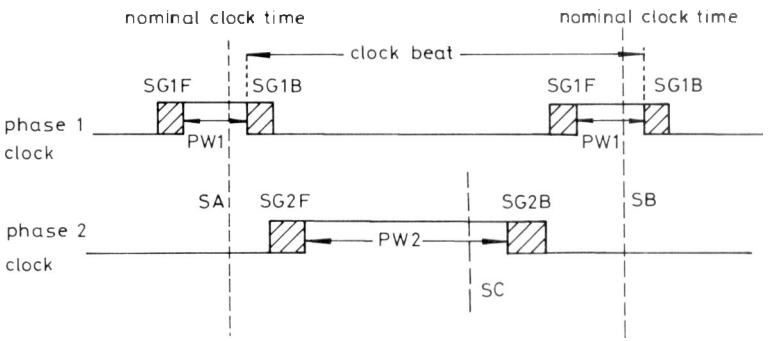

Fig. 3 Effect of clock skew on two-phase distribution

SG1/2F/B: front/back edge skew phase 1/2
PW1/2: period during which all phase 1/2 latches are open
SA/B/C: latch set-up time

By designing the Level 30 distribution to provide phase 1 and phase 2 clocks as separately timed signals to the VLSI chips all skew risks have been effectively removed from chip design throughout the machine. For example, where chip timing specifications have altered causing clocks to overlap, the clock distribution has been able to change rather than all the LSI chips. In addition the phased clocks have been generated from edges, derived from a crystal running much faster than the clock beat, eliminating the need for delay lines.

3 Store dataflow

The dataflow can be broken down into three main parts:

- the switch dataflow
- the store access dataflow
- the main store RAMs themselves

The general arrangements are shown in Fig. 4 which is Fig. 1 of the paper by D.W. Ashcroft on the processing node[1], in this issue.

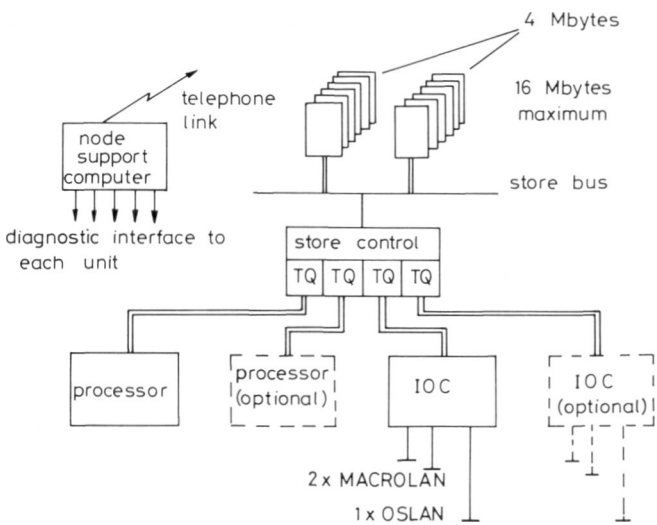

Fig. 4 Schematic diagram of Level 30 node

3.1 Switch dataflow and the TQ interface

The switch takes in four TQ interfaces and multiplexes them on a priority basis to permit access via the store access dataflow to and from the mainstore RAMs. Direct OCP-with-IOC and OCP-with-OCP communication is provided within the switch dataflow.

TQ is a Series 39 standard interface which except for clock and diagnostics forms the only interconnect between node units. It supports the connection of both OCPs and IOCs to Level 30 store and also of IOCs into Level 80 node. TQ is a serial-parallel interface; any one message requires a number of stripes of information, sent serially at the rate of one stripe per beat. Stripes may be either envelope or data stripes. For the example shown in Fig. 5 the first stripe is an envelope stripe. This stripe contains the address and control fields to define the type of request (read, write, block read etc.). A unique feature of the interface is that requesting units may raise their request flag on TQ one beat before the envelope stripe. This gives the requestor one beat to translate its address while the store is entering the request into its interface priority selection and returning Accept. A valuable beat is thus saved on store access. An essential part of this mechanism is the abort flag, which can be sent on TQ with the envelope stripe on the second beat of request. This is used when the requestor has been unable to translate an address but has already sent a request flag.

	byte 0	byte 1	byte 2	byte 3	byte 4	control flags
stripe 1	command	address	address	address	steering	store request
stripe 2		word 0 data			steering	direct request abort
stripe 3		word 1 data			steering	accept, data avail busy

Fig. 5 Example of TQ message format — main store request

In addition to a main store request flag, each unit may set direct request flags. Each TQ interface has these flags corresponding to each of the other TQ interfaces at the switch. When a direct request flag is set the requestor will have his message routed onto the corresponding TQ together with an unsolicited Data Available. This is the interunit communication route. It is possible to get more than one direct request simultaneously; this results in a simultaneous broadcast of the message. Direct requests concurrent with store request result in the data also being routed to and written to main store.

Fig. 6 shows the switch to be a simple multiplex of the incoming TQ with fanout back to any or a set of the outgoing TQ and also toward main store. This dataflow does not allow more than one TQ message to be routed at once.

3.2 Store access dataflow

Although TQ is only one word wide per stripe, main store is two words wide with independent access to each word. To support this the store dataflow provides independent interfaces to each word (address and data and controls). Because word 1 is both received and transmitted on TQ one beat later than word 0, word 1 of the store is usually cycled one beat later than word 0. Fig. 6 is a schematic of the store dataflow and main store RAMs. The scope of this paper precludes a dataflow operational description, although a couple of novel features are worthy of note.

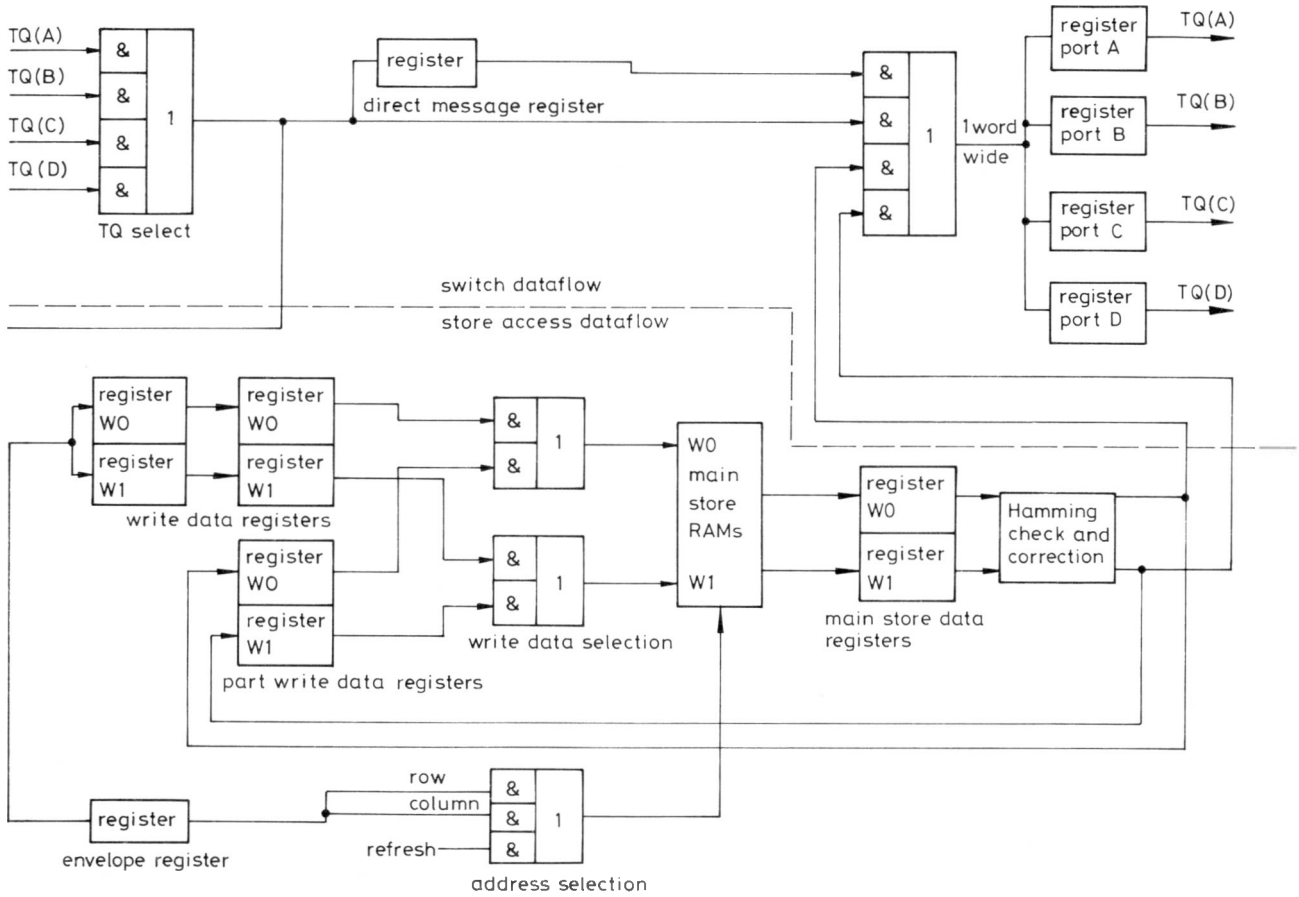

Fig. 6 Switch and store access dataflow

As the address and/or data from TQ passes towards the main store RAMs it is clocked synchronously through registers. Similarly, data returning from store is clocked synchronously into a register before passing into the synchronous Hamming correction subunit. By this means the store cycle is kept in time with machine clocks, removing the need to wait for the period of logical uncertainty (the 'boggle') that is inherent in an asynchronous interface. This has significant impact on the design of control, which is discussed later.

The Hamming correction subunit is treated as a free standing device. Once data has left the main store RAMs a following incoming TQ request cycle starts concurrently with Hamming correction and TQ transmit of this previous cycle. Thus store cycle time is reduced.

3.3 Main store RAMs

These are high-density dynamic RAMs. Their 'nibble' (block access) mode read/write facility allows the store to offer a block read/write feature fairly simply. The mainstore was developed using 64 kbit RAMs, launched with 256 kbit RAMs and is expected to operate later with double and four times these densities. Each variant requires a different timing profile for its control signals, forcing the design to have an innovative soft controlled timing system. This is described later.

One of the chief areas of difficulty in the store design has been the provision of refresh to these dynamic RAMs. Within a synchronous design it is clearly desirable that refresh cycles should also be synchronised to the machine clock. However, since refresh must not be stopped when, for example, the clock is stopped on an OCP fail or to single shot, then all logic involved in refresh must receive special clocks which do not stop. Identification of logic which may at any time interfere with refresh cycles (for example when stopped part way through another cycle with control signals active) has been a major headache. Simulation with extensive testing of this logic will be vital in future designs. Here, although the problems were first encountered after switch-on, they were successfully overcome with the help of the soft control.

4 Store control

Paths through the dynamic RAMs, and many of the control signals to these, are greater than one machine beat or must be maintained over a clock beat and do not necessarily fit beat boundaries. To support this the clock distribution provides three additional midbeat clocks corresponding to quarter beats. These are fairly simply derived since the clock is divided down from a frequency much faster than the required beat. All critical RAM timing signals may be retimed by loading a new microcode and all beats of the cycle after the first beat are soft controlled.

Fig. 7 gives an overview of this innovative soft control.

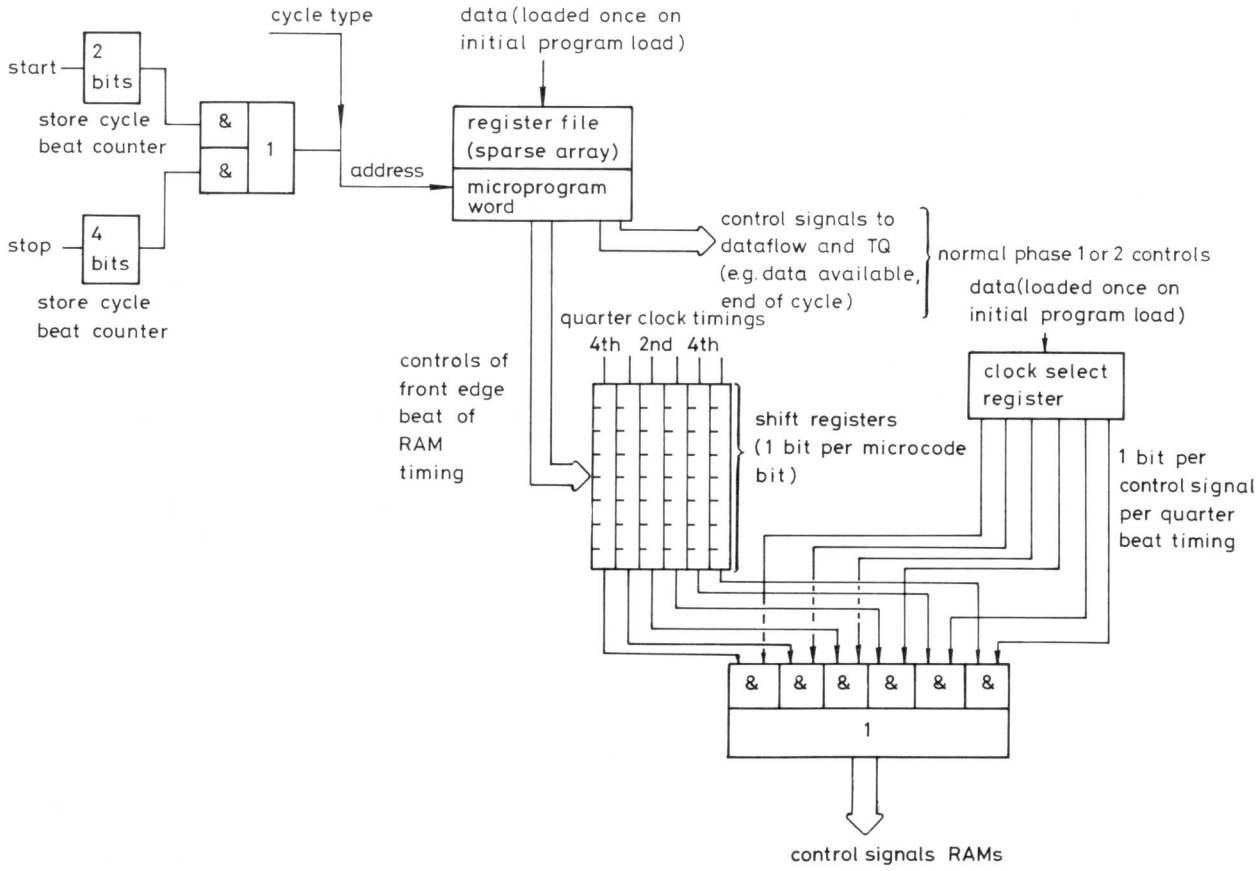

Fig. 7 Store soft control

The register file provides a fairly normal small microprogram store which is read out beat by beat by being addressed by a counter. The resulting microprogram words have two main fields. One field is used conventionally in the control of data flow actions, the other defines at which beat of the cycle the front edge of each dynamic RAM control signal will occur. Each bit of this field is passed through a shift register clocked by quarter clocks.

The required set of quarter clock periods are preset at initialisation time in the clock select register so that this now gates the appropriately timed controls to the RAM.

This configuration allows every signal to be retimed to any beat, starting at any quarter clock of that beat. Both the shift register and register file are, in pactice, sparsely populated to keep the control cost down at the expense of some flexibility.

Two counters are able to address the microcode register file. The smaller two-bit counter is used at the start of cycle and is permitted to overlap with the larger four-bit counter at the end of its previous cycle. This allows Hamming correction to overlap the start of the next cycle.

5 Conclusions

This paper has tried to illustrate the main features of Level 30 store, drawing particular attention to significant new methods. The store design has proved successful in providing a working design from switch-on without the need for chip iteration. In particular, without the soft features this would not have been achieved since they allowed some design error to be microcoded around. The design has also been successful in allowing the dynamic RAMs to be changed to different types with no other hardware modifications.

Lessons learned in refreshing synchronous stores will be carried forward to future designs.

Acknowledgments

This paper was made possible by the ideas and enthusiasm and very hard work of the members of the store design team whose work it describes. My thanks to those Level 30 design team members who helped by checking this paper.

Reference

1 ASHCROFT, D.W.: 'Processing node of the ICL Series 39 Level 30 system', *ICL Tech. J.*, 1985, **4** (3), 248-259.

The high-speed peripheral controller for the Series 39 system

J.A. Maddison
ICL Mainframe Systems Division, West Gorton, Manchester

Abstract

Fast peripherals such as magnetic discs and tapes are connected to Series 39 mainframes by high-speed MACROLAN fibre-optic links. The paper describes the special unit developed for controlling this traffic, noting the requirements, in addition to high speed, of flexibility, enhancement potential to handle new peripherals and very high reliability.

1 Overall objectives

The overall objective of the development was to create an LSI controller (or range of controllers) to connect a variety of high-speed peripherals (discs and magnetic tapes) to several distributed mainframes via MACROLAN.

The controller(s) was required to provide high connectivity (up to 256 independent software streams) while connecting to a single MACROLAN, to permit the attachment of new peripherals, and to be usable on Level 30 and Level 80.

It was also required to support a maximum device transfer rate of 3 MBytes/s, to allow operation in the absence of any guaranteed service rate from MACROLAN (i.e. use large internal data buffers) and to be optimised for maximum total throughput.

A high reliability (target mature MTBF [mean time between faults] of greater than 25 000 h), very good fault resolution and an extremely low undetected error rate (less than 1 undetected data corruption per controller per 100 years) were also required.

Some technology and size constraints were placed on the design. It should employ LSI CMOS using Fujitsu C8k gate arrays mounted on one LPCB (large printed circuit board) with low-cost plug-in daughterboards for RAMs etc. Bipolar B2k (2000 gate) chips were used where necessitated by data-rate requirements.

2 Realisation of the objectives

Two versions of the controller connect a peripheral subsystem of either discs (the HSDC) or magnetic tapes (the HSMC), but contain a large amount of commonality between the two, only the peripheral application module and the control programs being different.

Fig. 1 illustrates how HSPCs (HSDCs and HSMCs) fit into a typical system.

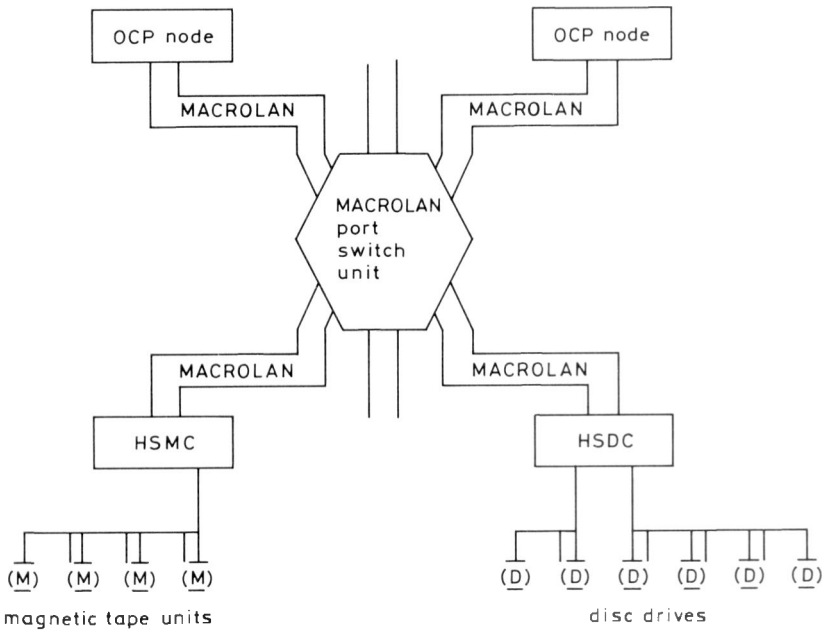

Fig. 1 HSDCs and HSMCs in a typical peripheral control system

3 Design structure

The design was structured into hardware and software modules from which the required controller could be assembled. Maximum flexibility was ensured by putting a large part of the function into the software modules, and using the hardware where speed was essential and where the function was unlikely to change. This flexibility was also designed to ease the attachment of any new peripherals.

The self-contained modules which provide this design flexibility are connected functionally as shown in Fig. 2.

Schematically the disc HSPC can be drawn as in Fig. 3. The design centres around the image store interface. The significance of the box shapes is that

daughterboards are shown as rectangles and C8K and B2K chips are the squarer boxes.

The data flow is represented thus ++++

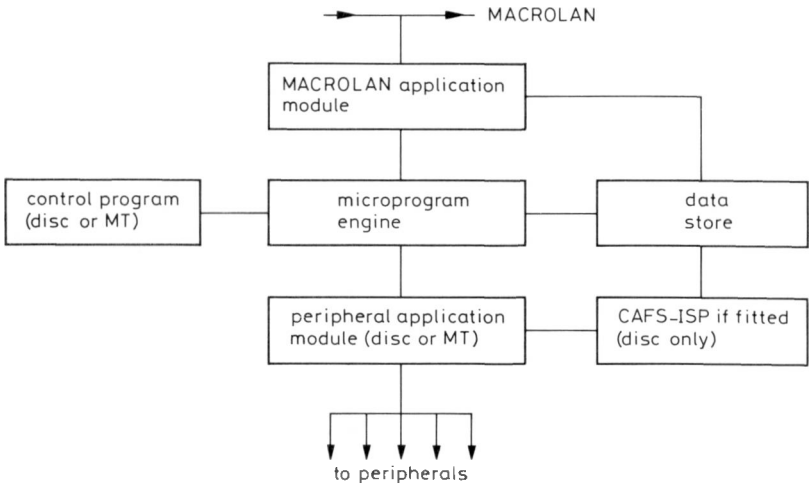

Fig. 2 Modular structure of the controller

4 Elements in the design

The microprogram engine is a specialised microprocessor whose design is optimised for speed at making single logical decisions. Its high speed allows a large part of the control of the overall HSPC to be handled by the control program rather than by the hardware.

The engine reads and interprets a sequence of instructions stored in the program store (ROM). The sequencing of instructions through the engine is such that one instruction is completed every 200 ns.

The program store is 32 bits wide plus 4 parity bits and contains 16 384 locations (HSMC) or 24 576 locations (HSDC). The instructions have a general format as shown here.

8 bits	8 bits	8 bits	8 bits
function code	source operand address 1	source operand address 2 or literal	destination operand address

Microprogram engine instruction format

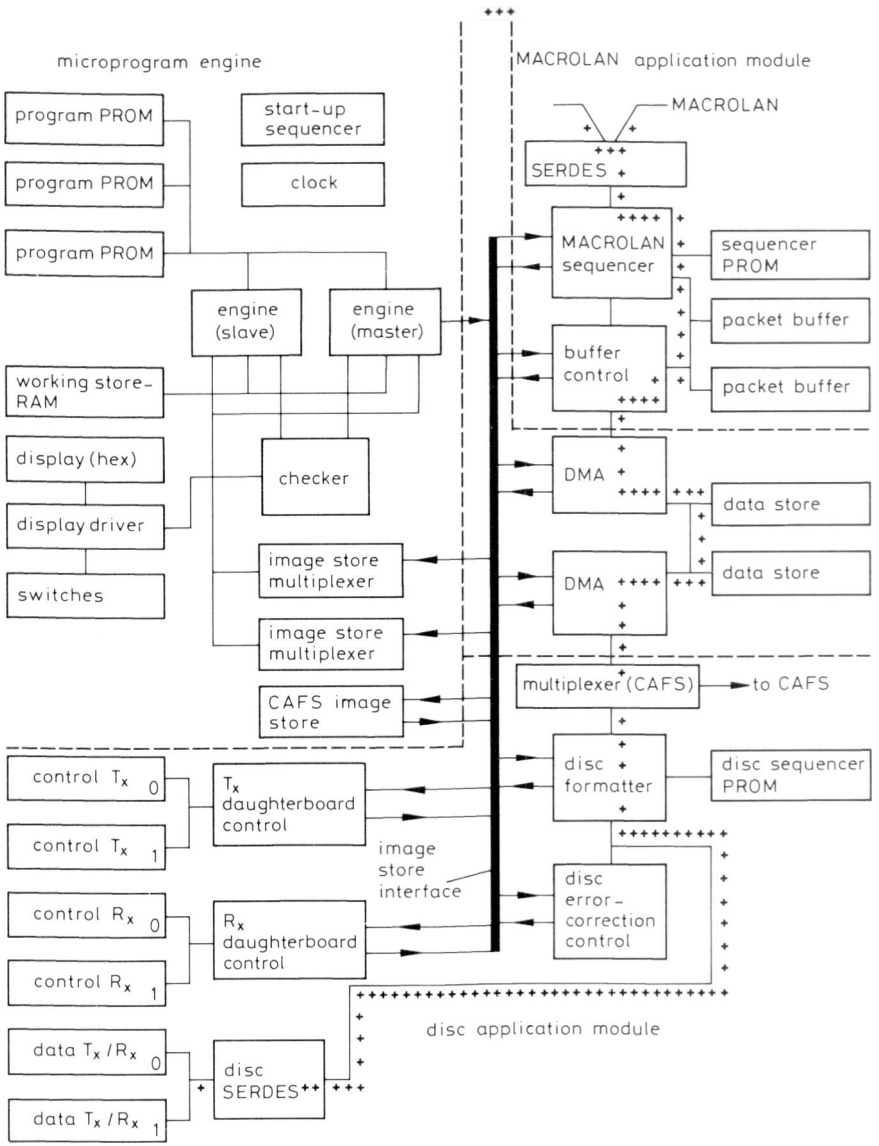

Fig. 3 The HSPC. Data flow is represented ++++

The operand fields in an instruction can define one of a number of different sources and destinations:

- working store (16 384 single byte locations + some expansion possibility)
- register file (a register within the engine containing 16 single-byte locations)

- image store – external (the interface between the engine and other HSPC hardware)
- image store – internal (directly addressable registers within the engine)

The engine drives the other hardware modules in the HSPC by defined procedures on the external image store interface. These procedures are, briefly:

- the engine sets an address on the image store address lines which

  (i) addresses the required module
  (ii) addresses the image store register within that module
  (iii) sets either a write to the module or a read from the module

- performs the appropriate function
- on each successive clock cycle the address and the data may be changed.

The data store is a 256 kbyte store arranged as 65 536 words; each word having 32 bits and 4 parity bits. 4-byte wide words are used to achieve the required throughput. The data store has two high performance DMA channels which handle the data transfers other than at the start and end of a transfer when the micro-program engine is in direct control.

The control programs organise this 256 kbytes store as they require. For disc applications it is partitioned as six data buffers, each of 32 kbytes, whereas for magnetic tape there are three data buffers, each of 64 kbytes. The remaining 64 kbytes in both cases is used as an extension to the working store to provide 192 bytes of stream storage per stream for discs.

The DMA channels transfer data between the MACROLAN application module and the data store, between the peripheral application module and the data store, and between the data store and the image store.

The peak throughput data rate of the data store (i.e. the maximum transfer rate of a DMA channel) is 5 Mbytes/s but 5 Mbytes/s cannot be maintained continuously on both DMA channels because of the effects of store refresh.

The MACROLAN application module provides the hardware connection to the MACROLAN and buffering for MACROLAN packets. The control program can access buffered packets directly via image store, or can use a DMA channel to transfer packets into or out of the data store.

The job of SERDES (serialisation/deserialisation logic) is to turn a stream of MACROLAN bits into data bytes and vice versa. It also handles the beginning and end of packets and the cyclic check bytes. The same SERDES daughterboard is used in the Level 30 and Level 80 nodes.

The job of the sequencer and buffer control is to transfer incoming packets from MACROLAN to the packet buffer and to transfer outgoing packets from the

packet buffer to MACROLAN. The buffer has separate stores for incoming and outgoing packets. Each is arranged as a 4 kbyte FIFO store capable of holding several packets. The sequencer automatically discards any incoming packets that are corrupt. It also discards any that are not addressed to this HSPC. This is the limit to its understanding of packets: the rest is done by the control program.

The disc application module is specific to the disc controller. It provides the hardware connection between the data store and the connected discs. It consists of transmitter/receiver logic and a disc formatter plus associated hardware which:

- generates/checks the field formats written to/received from the disc drives
- converts the serial data received from the disc drives into parallel data and transfers it to the data store or CAFS
- converts the parallel data received from the data store into serial data and transfers it to the disc drives
- generates the error-correction control (ECC) bytes for the data being transferred to the drives
- does the ECC check for the data plus ECC bytes received from the drives.

The formatter has a maximum data throughput of 3 Mbytes/s. It can instantaneously handle one drive at a time.

The transmitter/receiver logic provides the disc formatter with a common interface to the various types of disc drive that may be connected, and converts the common formatter interface to the specific interface(s) for the connected drives.

The following types of disc drive are supported:

- EDS80, FDS160/640
- EDS200
- FDS2500 (IBM3380 compatible) – approximately 2490 Mbyte formatted
- FDS300 – approximately 310 Mbyte formatted

The following mixes of drives are possible:

- 4 x FDS2500 (two strings each containing two modules)
- 16 x EDS80/FDS160/FDS640 (two strings each containing eight drives)
- 16 x FDS300 (two strings each containing eight drives)
- 8 x EDS200, (two strings each containing four drives)
- 8 x EDS80/FDS160/FDS640 (on one string) and 4 x EDS200 (on the second string)

The magnetic tape application module is specific to the magnetic tape controller. It provides the command control route between the magnetic tapes and the controller as well as performing the compress/expand function in the data route.

For MTS (and GTS-2) 310/470/780 and MTS61 devices a cluster of one to four devices can be supported. MT types cannot be mixed within a cluster.

The MT application module has a maximum data throughput of 3 Mbytes/s.

The control programs used to support these two applications have many similarities; therefore the disc version will be considered and only the specific differences will be noted.

The control program acts in conjunction with the controller hardware to execute the PSD 2.5.13 (2900 peripheral device control procedures) level commands sent from the mainframe operating system via MACROLAN and to convert the commands to the low level actions required by the devices and MACROLAN.

The controller hardware handles the data while the control program performs the following major functions:

- queueing of commands – one command per stream can be queued for each of the 256 streams. (magnetic tape – three commands and four streams)
- scheduling of resources:
  up to 256 streams are competing for up to 16 drives (NB there is a limit of 16 streams maximum per drive). The 16 drives are competing for the six data buffers. The six drives with data buffers are competing for the one disc data channel. (Magnetic tape – four mechanisms competing for the data buffers, the formatter, and for the one magnetic tape data channel)

  ensuring a transfer does not commence until buffer conditions are such as to allow completion of at least the next block

  maximising of the overall data channel throughput rather than minimising individual access time
- recognition and execution of commands as defined in the applicable device class specification
- analysis of all headers of the incoming packets (as defined in PSD 2.17.21 and related specifications) and the routing of any data to the data buffer via a DMA
- Loading of the headers into the MACROLAN application module buffers for outgoing packets (as defined in PSD 2.17.21 and related specifications) and connecting the DMA if data is a part of the packet
- actioning timeouts – both on MACROLAN and the peripheral interface
- servicing and controlling the disc formatter at disc block level
- handling disc skip displacement calculations
- monitoring RPS positional indications from disc drives
- attention scanning – periodic monitoring of peripheral interfaces to detect state changes, e.g. auto/manual
- error handling – i.e. detecting errors, monitoring error stats, taking the necessary action and assembling and reporting status.

  The control programs have a large role in the overall error management within the HSPC. They collect error status from the hardware units, act upon it, and report the failure. They also perform many other error checks

– error logging – for classes of failure that would otherwise pass unreported: e.g. MACROLAN packets that had to be repeated
– indicating errors locally when unable to report via MACROLAN. HSPC errors normally cause a crash + dump + display sequence to be entered. Other errors are reported to the mainframe software by dumps or by primary, secondary and tertiary status as defined in the appropriate device class specification, provided that the microprogram engine, MACROLAN application module and MACROLAN are working. Whether or not this route is available on HSPC error is also reported via the HSPC hex display
– self-test and establishment tests – establishment tests are run at power-on time and following a hardware or control program detected error
– performance monitoring:
  a thorough check on incoming packet headers is carried out.

5 CAFS application

The hardware for the CAFS application is functionally introduced between the data store and the disc application module as shown in Fig. 2. CAFS is basically the same unit as that currently used on 2966 etc.

6 Physical implementation

The HSPC exists on one large PCB. Some of the options for fitting this into appropriate cabinets are illustrated in Fig. 4. The HSPC design does not, of itself, preclude its combination with the other system building blocks. See also Fig. 2 of the overview paper by Skelton[1].

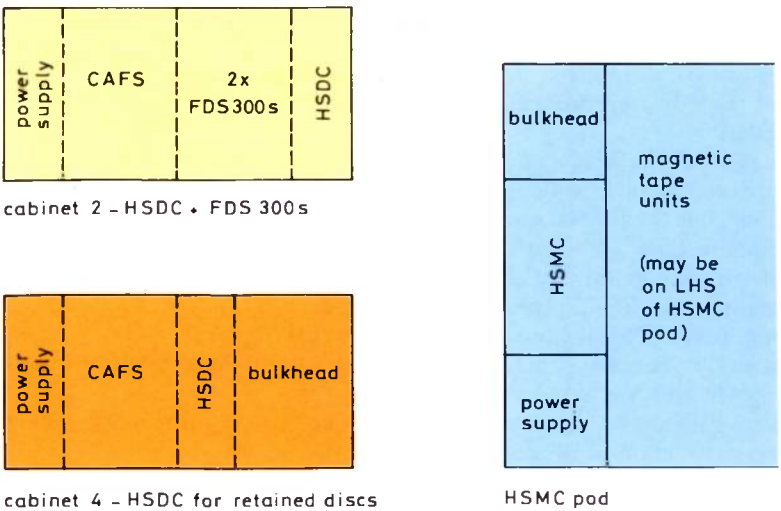

Fig. 4 HSPC placement

7 Development method and validation

In general, the HSPC followed the overall Level 30/Level 80 development route with considerable emphasis on the use of design simulation during both software and hardware development. SAUL (simulation at unit level) models of chips and modules were constructed from information contained in overall drawings and specifications. Test programs were run on the SAUL models of the modules and the patterns at the chip boundaries were extracted from these runs.

A simulation model was derived from the captured logic diagrams for each chip, and the patterns which were extracted from the SAUL tests were run against this model.

Simulators for three types of store were employed during early prototype testing, for the program store, working store and formatter store. These were used in place of the corresponding PROM and RAM daughterboards.

The software development required for this product was different in type from that employed in OCP node development. It related more closely to the development of LAN management and stream control code in VME. It therefore shared many of the development and validation problems, such as:

- Multithreading
- asynchronous operations
- file store control (for system IPL)

and other features.

The approach adopted was to decouple the software from the detail of the machine by using a higher-level language which facilitated concentration on the logic of the problems involved.

8 Manufacturing, maintenance and testing

The HSPC follows overall system practices with regard to manufacturing and maintenance.

The HSPC test strategy (for field use) is based upon initial fault detection during normal use by HSPC establishment checks, the HSPC hardware error status (via error recording etc.) and analysis of HSPC dump information. This resolution will be achieved by the use of the SAM (support and maintenance) system.

For hardware faults reported via MACROLAN to mainframe the maintenance engineer will be informed of the suspect item(s) and so can take the necessary spares to site to effect the repair and return the controller to an operational state.

The same principle holds where the error is reported locally at the HSPC, except here the user must examine the indicators and report the fault to the engineer before the site visit.

9 Enhancement potential

The ability to connect new peripherals is provided by the modular structure.

e.g. cartridge tape (SCSI: small computer system interface)
optical disc (SCSI)
new disc types (IPI: intelligent peripheral interface)

Reference

1 SKELTON, C.J.: 'Overview of the ICL Series 39 Level 30 system', *ICL Tech. J.*, 1985, **4** (3), 225-235.

Development of 8000-gate CMOS gate arrays for the ICL Level 30 system

Y. Suehiro, N. Matsumura, Y. Sugiura, M. Yamamoto and R. Hoshikawa
Fujitsu Ltd., Kawasaki, Japan

Abstract

High-performance 8000-gate CMOS gate arrays applied to the Level 30 project are now in mass production. These LSIs could be developed within a short period utilising a special user/vendor design interface and performing sufficient validation and evaluation on both the user's and the vendor's sides.

1 Introduction

VLSI technology is developing very rapidly to enhance both complexity and operating speed, and it has become the most important factor which determines overall system performance. Thus system designers should implement their designs into customised silicon chip to achieve higher performance with a nominal cost. A CMOS gate array is one of the best technologies to pursue such silicon implementation due to substantial low power consumption and high speed operation.

When the Level 30 system was designed at ICL, they tried to utilise the best CMOS technology at that time and finally, they decided to design 42 types of Fujitsu's 8000-gate CMOS gate array C-8000VH considering performance, fast turnaround time, and cost. Although such CMOS gate array has been a standard product of Fujitsu (product name MB66VH000 series), this project would be the biggest design task which no one could imagine and require special co-operation between user and vendor to realise performance as expected and minimal turnaround time.

This paper will cover Fujitsu's C-8000VH CMOS gate-array technology and describe a key feature of this project – design interface between ICL and Fujitsu, and co-operation in evaluation of characteristics for C-8000VH.

2 Main features of C-8000VH

In this section, general features of the C-8000VH are mentioned. The design interface between ICL and Fujitsu is special and is described in Section 3.

2.1 Chip structure

The structure of the C-8000VH chip is shown in Fig. 1. The chip consists of four blocks of cell arrays and input/output cells in the chip periphery.

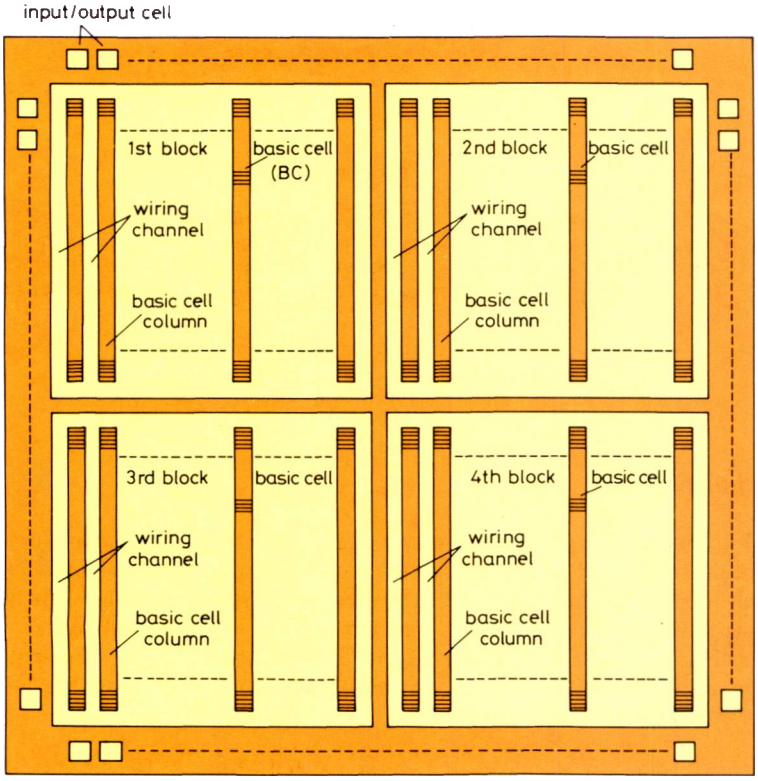

Fig. 1 Chip structure

A fundamental element of an array is two pairs of n-channel and p-channel MOS transistors which we call basic cells[1] (referred to as 'BC'). BCs are arranged in rows in each block. The area between BC columns and the area around each block are both used as wiring channels.

The array structure is as follows:

block : 100 BCs/column x 20 columns = 2000 BCs

chip : 2000 BCs/block x 4 blocks = 8000 BCs
45 I/O cells/side x 4 sides = 180 I/O cells
(40 signal pins/side x 4 sides = 160 signal pins)

For the benefit of the general user the chip is divided into four blocks. The reason for this is:

2.1.1 Prediction of chip performance before layout: To manufacture an LSI with the performance which meets the customer's request within a short period, it is necessary to estimate the performance with simulation at the logic design stage. The simulation is performed using the table of predicted interconnection wiring capacitance. Since the chip has been divided into four blocks, the simulation can be performed using the table of predicted wiring capacitance of interblock connection against various fanout as well as that of interblock connection. Thus the performance of a chip can be predicted more exactly compared with the case that such division is not made.

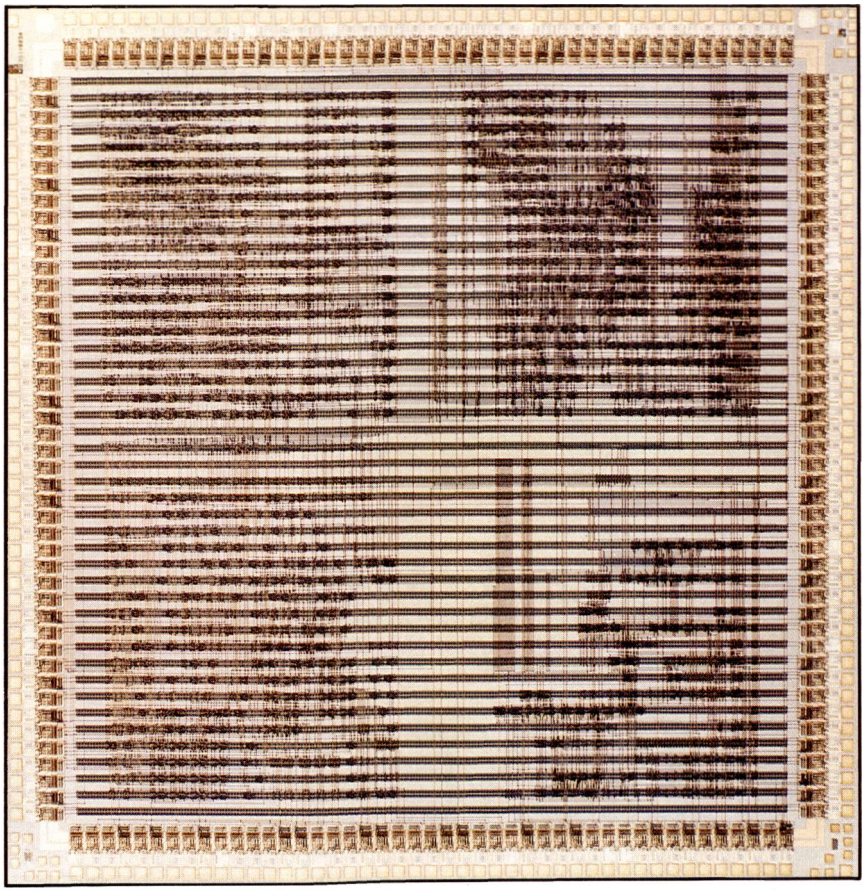

Fig. 2 Photomicrograph of a C-8000VH chip (MB66 VH442)

2.1.2 Increase of efficiency in automatic chip layout: Since such chip structure is considered at the logic design stage, logic circuits which are strongly related to each other are placed in one block and this leads to more effective automatic layout[2].

A logic function cell called the unit cell (referred to as 'UC') consists of one or more BCs with contact holes and wiring patterns fabricated in customised metallisation process. More than 100 kinds of UCs including 4-bit counter, 4-bit full adder and 4-bit magnitude comparator etc. are prepared for customers. After placement and routing of UCs and I/O cells for an entire chip, the chip is fabricated with customised metallisation. Fig. 2 shows a photomicrograph of a C-8000VH chip after customised metallisation.

2.2 Summary of performance

The C-8000VH chip is fabricated using an advanced wafer process technology whose summary is shown in Table 1. It employs double-layer metal CMOS technology with 2·3 μm gate length, which has been evolved from Fujitsu's CMOS gate arrays standard and H versions. The metal wiring pitches of 6 μm for the first metal and 8 μm for the second metal were attained. Such small wiring pitches can improve the propagation delay time due to small load capacitance.

Table 1 Summary of process technology

p-well, silicon gate CMOS	
n-channel transistor gate length	2·3 μm
p-channel transistor gate length	2·8 μm
1st metallisation pattern width	2·0 μm
1st metallisation pattern pitch	6·0 μm
2nd metallisation pattern width	3·0 μm
2nd metallisation pattern pitch	8·0 μm

The chip size of the LSI is 9·7 x 9·5 mm, and is almost the same as that of Fujitsu's 3900-gate standard version CMOS gate array[3]. The performance of the C-8000VH was evaluated using a test chip and the faster switching speed of 2·2 ns per gate was obtained.

Figs. 3 and 4 are examples of the measurement results on switching speed. Fig. 3 shows the propagation delay time obtained from the ring oscillators which are composed of inverters (UC name: V1N), two-input NAND gates (UC name: N2N), and two-input NOR gates (UC name: R2N). Fig. 4 shows a comparison of toggle frequency between J-K flip-flops of Fujitsu's CMOS gate arrays – standard, H and VH versions. The toggle frequency of the VH version for a J-K flip-flop is 76 MHz at 5·0 V power supply voltage, and that of a D-type flip-flop is 95MHz.

Fig. 5 shows typical transfer curves for the cascaded input and output buffers, and Fig. 6 shows typical characteristics of the output buffer. These Figures show that C-8000VH is compatible with TTL.

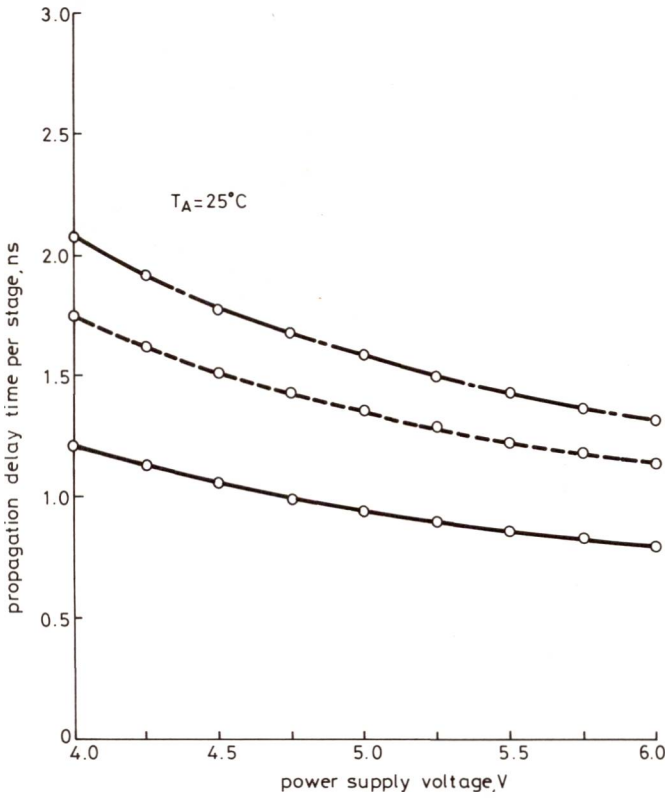

Fig. 3 Propagation delay time of ring oscillators

——— V1N
- - - - - - N2N
— — · — — R2N

3 Design interface for ICL

In this project for ICL, high-performance LSIs had to be developed in a short period. The C-8000VH was chosen for the project because it features high speed, low power consumption, high complexity and fast turnaround time (TAT), and it met ICL's specification. Chip designs for the project amounted to 42, so in order to develop such designs in fast TAT, co-operation between ICL and Fujitsu was indispensable. Thus, logic data, test data and layout data shown in Fig. 7 were decided as a special interface.

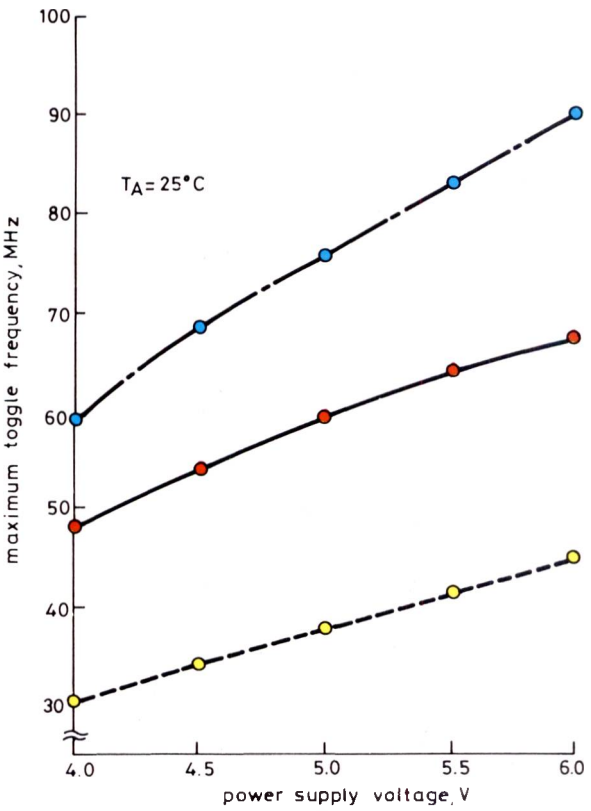

Fig. 4 Toggle frequency of J-k flip-flop

- - - - - standard version
——— H version
——— VH version

A logic data has logic design information about a single chip, and a test data has the test patterns which consist of input/output test vectors and input/output timing information. A layout data contains chip layout information, which is seldom used as user/vendor interface.

Use of layout data as interface was very important in respect that ICL could route critical paths on their own chip designs. ICL could make the layout data and Fujitsu could generate the mask data using it. As information necessary for ICL's routing, Fujitsu offered the metal interconnection routing rules, the information on the chip structure, and the pin location/assignment description on I/O terminals of unit cells and I/O cells. The information on the chip structure covers the location of basic cells, I/O cells, power supply lines and earth lines, and inhibited areas for wiring on the chip. A chip can be laid out automatically by computer dealing with a cell as a black box.

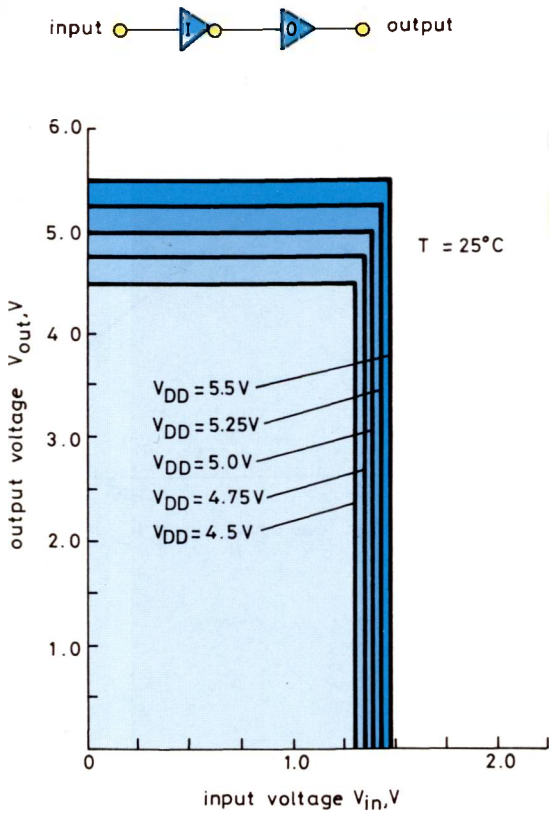

Fig. 5 Typical transfer curves for the cascaded input and output buffers

Three kinds of interface data are fully checked on the ICL side, but are checked again in Fujitsu side as shown in Fig. 7. Fujitsu generate a logic data file in our database from the logic data received, and a logic design rule check (LDRC) is performed on the logic data file to check syntax errors and design rule violation. After that, physical validation is performed using the logic data file and the layout data file generated from the layout data. In this validation, a comparison between logic design data and layout data with respect to connections inside a chip is made, and violations of metal interconnection routing rules are checked. If LDRC and physical validation are successfully completed, then test data validation is performed using the actual metal wiring capacitance obtained from the layout data file. From this validation, the function and the AC performance of a device can be guaranteed prior to silicon implementation. After successful completion of test data validation, Fujitsu start fabrication of the LSI. Such error-resistant interface makes it possible not only to greatly simplify users' work, but also to attain their requirements of implementing as high a performance as possible by applying the C-8000VH to their own specification.

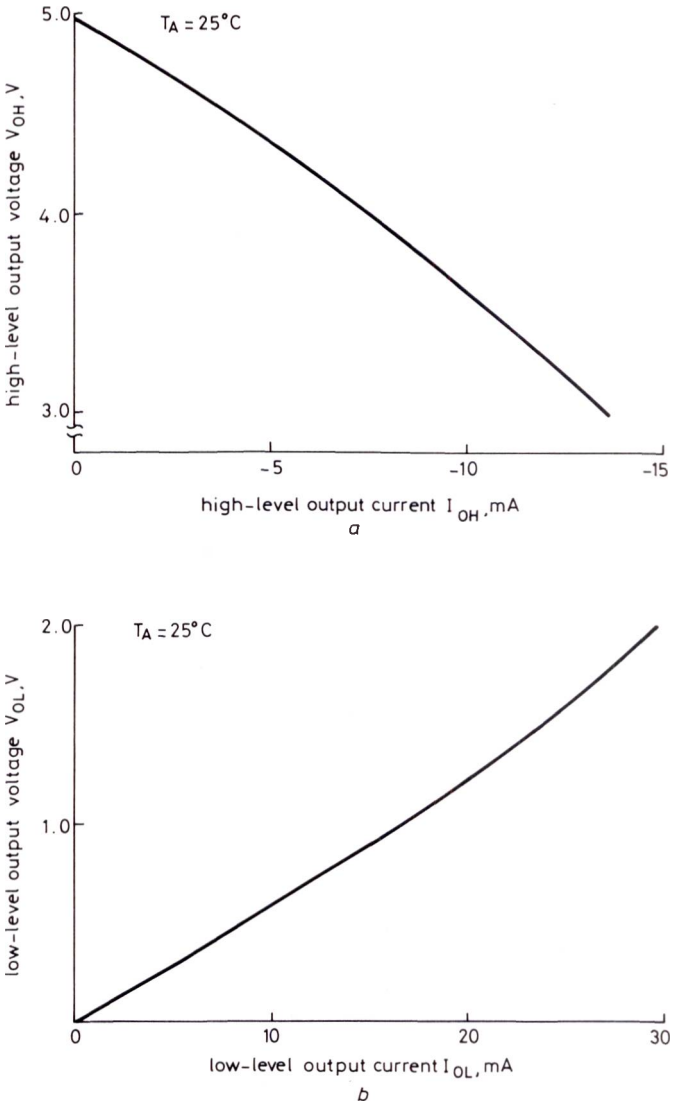

Fig. 6 Typical characteristics of output voltage against output current; (a) V_{OH} against I_{OH}, (b) V_{OL} against I_{OL}

The interface data at the earlier stage of the project had some syntax errors; however, such errors have been avoided. And sufficient validation on both the user's and the vendor's sides made it possible to get satisfactory performance of the chips as expected.

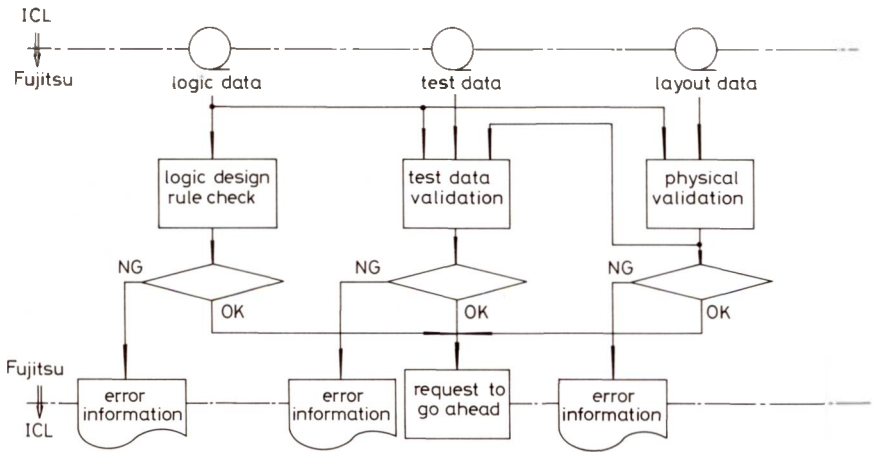

Fig. 7 C-8000VH interface data and file validation route for ICL

4 Evaluation of performance

4.1 Test chips

Before making real chip designs for the Level 30 project, C-8000VH test chips were designed by ICL and manufactured by Fujitsu. These chips covered all unit cells to be used for the project. By making this trial, ICL and Fujitsu could check their CAD interface, and the performance of the C-8000VH including these unit cells.

4.2 AC correlation

The test results of C-8000VH electrical characteristics for three early chip types designed by ICL showed that the C-8000VH was well within the specification. However, to improve the specification for propagation delay time of cells, the correlation work of propagation delay time for various paths on ten chip types was done between ICL and Fujitsu.

A comparison between calculated and measured propagation delay time was made. The calculation was performed using the gate propagation delays which were described on the AC specification. These delays were defined under the conditions of nominal power supply voltage (5·0 V), nominal temperature (operation temperature T_{op} = 25°C), and typical process parameters (gain factor of MOS transistors etc.).

The actual wiring capacitance in a chip was also applied to the calculation. The measured delay time was obtained at 4·7 V/5·3 V power supply voltage at 25°C.

Fig. 8 shows an example of a comparison between calculated and measured delay time for a chip.

The ratio R in the figure refers to $tpd_{M1}/tpd_C \times 100$ (%) or $tpd_{M2}/tpd_C \times 100$(%)

where tpd_{M1} = measured propagation delay time at power supply voltage = 5·3 V

tpd_{M2} = measured propagation delay time at power supply voltage = 4·7 V

tpd_C = calculated propagation delay time.

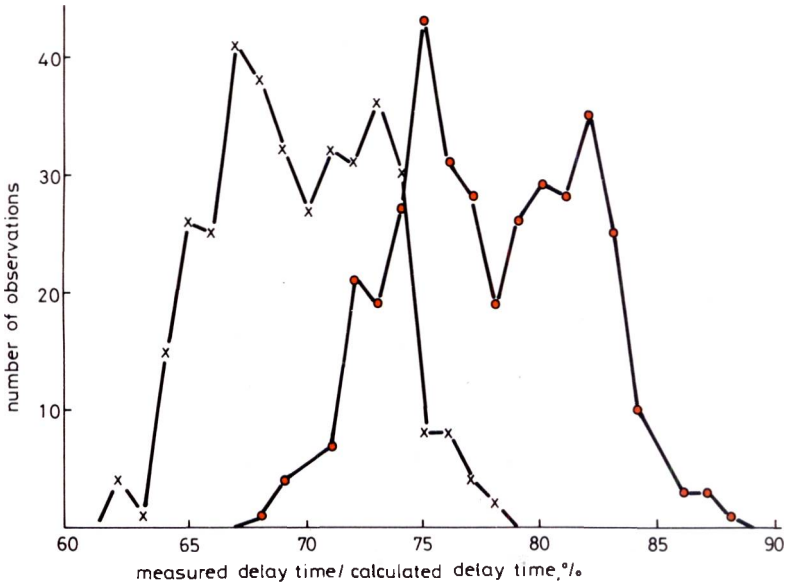

Fig. 8 Comparison between calculated and measured delay time for a chip

O power supply voltage 4·7 V
X power supply voltage 5·3 V

The number of measured samples is 20 and the gain factor of their transistors is typical. 18 various propagation delays per sample were checked, so the number of plots in each Figure is 18 x 20 (360). The calculated values of 18 propagation delays exist between 154·0 ns and 392·8 ns. As shown in the Figure, the centre values of R at 5·3 V and 4·7 V power supply voltage are 69% and 77% respectively. So, the value at nominal power supply voltage (5·0 V) may be 73%, which means that the propagation delays of the devices are smaller than those which are calculated by using the AC specification.

The switching speed of a device is dependent of the gain factor g_m of a transistor. Fig. 9 shows the relationship between the measured gain factor and the ratio R_a on ten chip types. R_a refers to the average value of R_n, where R_n = (1/2) $(tpd_{M1} + tpd_{M2})/tpd_C \times 100$ (%) (for each delay of a sample). In this figure, R_a is about 75% at g_m = 1·0. Since an expected value of R_a is 100% at g_m = 1·0, this shows that propagation delays of devices are faster than those of calculated values.

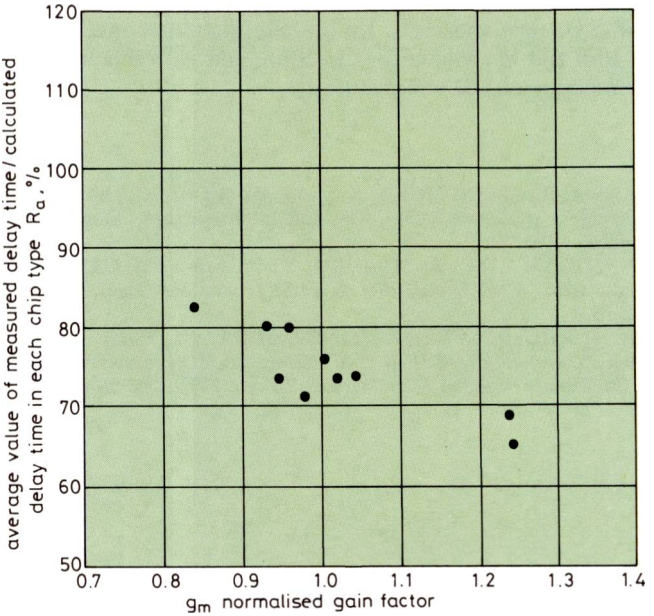

Fig. 9 Relationship between g_m and ratio of measured delay to calculated delay

The measurement results of propagation delay time in both ICL and Fujitsu gave good correlation. In parallel with the correlation work, Fujitsu researched the propagation delay time of I/O buffers and several unit cells. As a result, the specifications for the I/O buffers, several unit cells and delay tolerance were changed to obtain better correspondence between the switching speed of a real device and the switching speed calculated by simulator.

5 Conclusions

The C-8000VH CMOS gate array is an original product of Fujitsu, but was very much assisted by ICL with the early evaluation. In development of 42 chip types for the Level 30 project, logic data, test data and layout data were selected as design interface, and sufficient validation for each chip design was performed in both the user's and the vendor's sides. Moreover, test chips were evaluated prior to real chip designs, and the improvements of the AC specification were made after the correlation work. Due to these co-operations, the

development of 42 chip types has been achieved in fast turnaround time. Those chips are now in mass production.

Acknowledgment

The authors wish to thank M. Eyre and B.J. Proctor for their initial assessment and evaluation work on application of C-8000VH chips to the Level 30 project, and C.J. Skelton and J. Lord for their support in development of the LSIs, and J.A. Vernon, S.G. Hale and H. C. Hu for helpful discussions. We would also like to thank J. Ishii and M. Ashida for direction, and H. Yamaura and many other staff for their assistance and endeavour.

References

1 HOSHIKAWA, R., MATSUMURA, N. and KAWAUCHI, K.: 'Introduction of an ultra fast 8000-gate CMOS gate array', The 2nd International Conference of Semi-custom IC's, London, 1982.

2 ISHII, J., SUGIURA, Y. and SUEHIRO, Y.: 'A gate array CAD system and future tasks in the field', 1983 Symposium on VLSI Technology, Maui, 13th-15th September 1983, 12-15.

3 ASHIDA, M., HOSHIKAWA, R., MATSUMURA, N., ISHII, J., ICHIKAWA, H., SUGIURA, Y and INAYOSHI, K.: 'A 3000-gate CMOS masterslice LSI', Proceedings of 11th Conference on Solid State Devices, Tokyo, 1979, 203-206.

Development route for the C8K 8000-gate CMOS array

R.J. Metcalfe and R.E. Thomas
ICL Mainframe Systems Division, West Gorton, Manchester

Abstract

The complexity of the design of the Level 30 system in a VLSI technology required an order of magnitude improvement in design integrity over any previous project. The project timescales were to be critically geared to the number of design iterations of the C8K gate array chips. The paper describes the route by which this design was achieved and indicates the control and managerial provisions that were made and were essential to the success of the project.

1 Background

As several papers in this issue of the *ICL Technical Journal* have commented, the decision to base the Level 30 system on the Fujitsu C8K CMOS gate array presented Mainframe Systems with an exceptionally demanding task. Not only had the design to be accurately translated into a set of these chips when it was known that several US companies were having difficulty even with 1200-gate arrays, but the design was to be developed in Manchester and the chips made in the Fujitsu factory on the other side of the world. It was clear from the start that success in this project depended on there being an accurate and predictable development process, and this is what the present paper sets out to describe.

The essence of the development process is that detailed logic design is captured and checked for consistency with definitive high-level design, and transmitted to Fujitsu in a form that can be used to drive the fabrication process; test data are derived from the logic and are used to check the finished chip.

2 Audit procedures and design rules

Before the chip program was started a full set of design rules was formulated and documented. The majority of these rules were then turned into automatic checks on the design automation process, and as new rules were found to be required these error checks were enhanced.

Quality audit procedures were also established at the start of the project, to determine that the required level of design integrity was maintained. The chips

were audited at each stage in the development route, and the procedures enhanced as experience was gained and errors found. Where possible additional checks were built into the design automation software to remove as much manual checking as possible.

3 Pilots

The entire chip development route was validated at an early stage in the project with a series of pilot chips, starting with data-only releases and building up to silicon processed in Japan.

4 Development route stages

4.1 High-level simulation

The high-level models of the chips were written in such a way that they could be put together to form functional units. Test programs were then run on these units and data patterns corresponding to the physical chip boundaries extracted for use on the low-level simulation models.

4.2 Hand-drawn logic drawings

Sets of logic drawings corresponding to the functional specification for each chip were drawn by the chip designers; these used the cell library that had been issued and showed every cell and all the interconnections.

4.3 Creation of assembly file database

The hand-drawn logics were coded, by a dedicated team of logic technicians, into the design automation format so as to produce the assembly database. Logic diagrams and error listings were then returned to the designers for correction, and the cycle repeated until the results were error-free. Any subsequent modifications and corrections were handled by the logic technicians, who ran all the DA software and maintained control over the databases.

4.4 MSIM model

When an error-free assembly had been obtained a simulation model, built up from the low-level cell models, was extracted from the logic databases. The high-level patterns were then run on this model and the outputs compared with the expectations; any errors were fed back to the original logic diagrams which were updated and the cycle repeated until error-free.

4.5 Generation and validation of test data (DGEN)

In parallel with Stage 4.4 the logic database was fed into the test generation system to determine its testability. Any areas found not to be testable, and any test rules found to be broken, were fed back for correction by the designer until

a very high level of fault cover was achieved: better than 99% for 'stuck-at' faults. When the test data were satisfactory they were run on the low-level simulation model as a validation exercise and test of confidence.

4.6 Physical placement

Once an error-free logic assembly had been obtained, trial autoplacement runs were carried out. The placement was an automatic process but could be steered by a number of parameters, depending on what effect was required.

4.7 Automatic timing analysis

Because of the nature of CMOS technology and the dramatic effect of physical tracking on signal delays, a separate piece of software was developed which enabled all the paths from one register to another to be timed in three phases. Initial timing was done on the logic diagrams, using very approximate equations. A more accurate estimate was obtained once placement had been completed, using fairly accurate estimates of possible track lengths. A final run when fully tracked gave an accurate check on performance, taking into account the full metal loading of all logic strings. Corrections and improvements from each stage were fed back into the original logic database.

4.8 Autotracking on 2966 mainframes

When simulation, testability and timing were all correct the chip was tracked, initially on an ICL 2966 but later on an ICL Distributed Array Processor (DAP). If there were found to be more than about ten failures or handwires needed, manual intervention was used, involving examination of the types of failure and of the areas that were failing so as to identify areas of high congestion or other problems. Depending on the number and types of failures it was possible to change the placements of a small number of cells, to specify certain critical strings or to rerun the autoplacement software with different parameter values. Once a run had produced a result with an acceptable number of handwires these were added by manual input of the required paths.

4.9 Autotracking on the DAP

Part way through the chip program it became obvious that computer time for chip tracking and simulation was going to be a bottleneck. To reduce the time needed the DAP tracker was introduced; this provided a very much faster autotracking performance, on some occasions allowing up to three runs on one chip or up to six runs on several chips to be made in 24 hours. Its use allowed many more chip designs to be completed in parallel with a reasonable turnaround.

4.10 Cross checks

The results from the two autotrackers were used to update the original logic database with the physical information, and cross checks made to ensure that

the physical and logical information were mutually consistent. This confirmed also that any handwires inserted were correct and that all the physical design rules had been obeyed.

4.11 Data tape generation

When the cross checks had been completed successfully and the final 'freeze' audit had been made, the data tapes for the logical, physical and test data were generated. These three files were then read back and cross-checked again to make sure that everything was readable and internally consistent.

4.12 Timing at board and system level

The timing data produced for each chip were gathered together into a group file representing a board level unit, and several units then combined to make a complete system. Timings could then be accurately validated between pairs of chips and between chips and other printed circuited boards, for nominal and for worst-case individual timings; this proved extremely useful and indicated some significant problems in certain cases. Any changes made to the chips to improve performance were input to the original chip database and the validation repeated.

The complete process is represented diagramatically in Fig. 1, in which the audit points are indicated by Q.

5 Important managerial provisions

- The design rules and the development route were laid down at a very early stage and were firmly enforced.
- Design managers were acquainted with the development route by walkthrough exercises, which also served to verify the route.
- Design engineers were constrained to work within the documented route by means of a combination of manual and automatic checks, monitored by a series of predefined audits throughout the route. These checks were controlled by an independent design integrity manager.
- Checking was made as automatic and as foolproof as possible, by steadily incorporating the checks into the design automation tools. Manual testing proved very suspect, almost all errors in the data being found to be in areas that had been checked manually.
- Enhancements of design automation tools were carefully controlled and monitored, with regression testing of all significant changes.
- The project management framework encouraged and harnessed the use of peer groups across the design, technology and tools teams to enhance the process and solve problems.

We should add that the checks applied by Fujitsu proved an excellent back-stop for errors in the data sent to them; if both our tests and theirs were passed we could be confident that it was safe to go to manufacture and that a good chip would be produced. In fact, every chip returned was found to function precisely

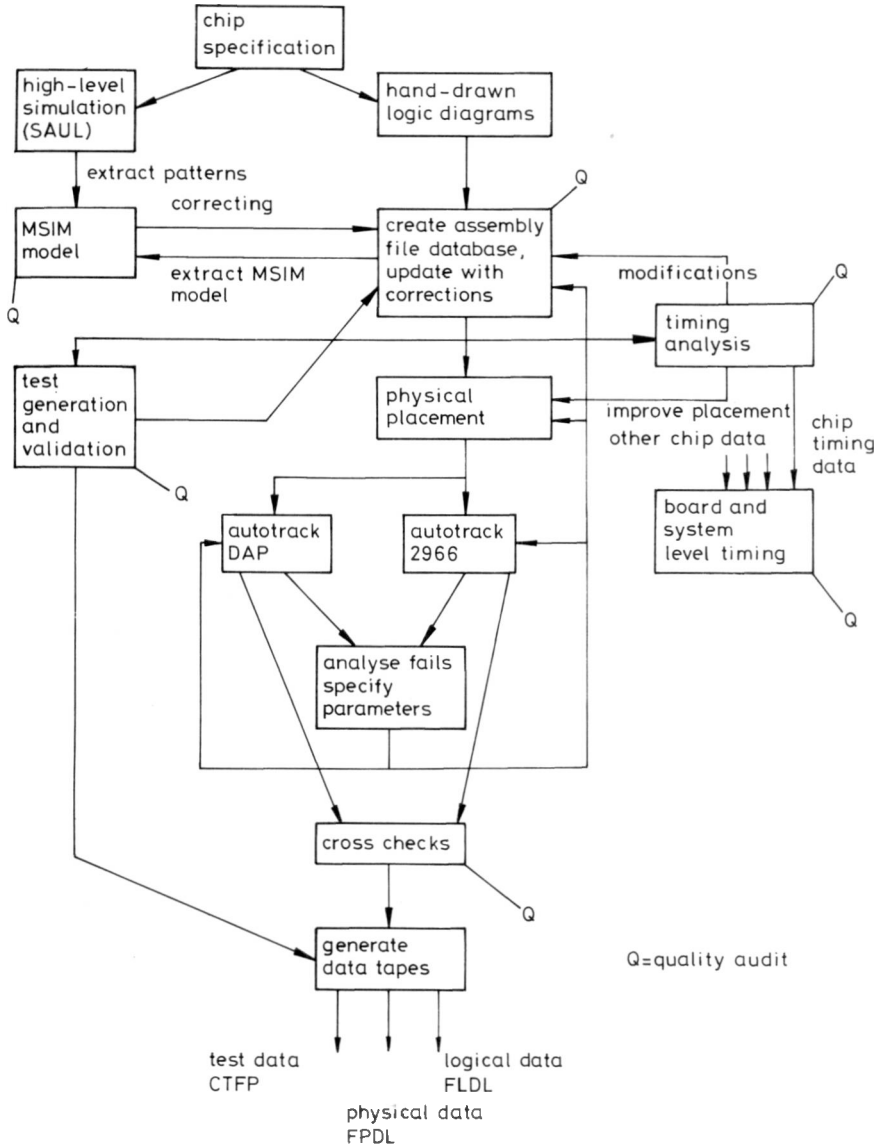

Fig. 1 Development route flow chart

as designed; a few designs were found to need corrections and therefore remaking of the chip, almost all of these when sets of chips were exercised for the first time as they would be grouped in the complete Level 30 system.

6 General notes

- Any changes to the database, including changes to physical placement to aid tracking, involved resimulation and regeneration of the test data, to ensure that integrity was maintained.
- Automatic checks were built in at every stage and improved as the development proceeded. Any errors found in the later stages of the route were investigated, and where possible error checks were introduced into earlier stages to detect these.
- Any changes to the design automation software were very carefully validated and regression tested against proven designs before being issued.
- Audits were established at each major stage as part of the very tight control system imposed in order to ensure very high design integrity.
- A comprehensive DA control system was established to ensure that stages could only be run in a certain order, and to identify to the auditors if any stages had not been completed successfully.

Design automation tools used in the development of the ICL Series 39 Level 30 system

M. Hewson and H.C. Hu
ICL Mainframe Systems Division, West Gorton, Manchester

Abstract

The design automation tools used in developing the Level 30 hardware are an extension to the design automation system, DA4, that has been in use in the Mainframe Systems department since 1976. The inherent technology independence of DA4, coupled with the strong design disciplines in place in Mainframe Systems made the job of extending the application of DA4 to the C8000 gate array and Level 30 PCB technologies possible in the very short timescales demanded. The stability of DA4 enabled a full range of automatic facilities to be brought to bear in the Level 30 product development route from system simulation and system test generation through logic simulation and checking to placement, tracking and production output. Design control within DA4 allowed the development to proceed without losing control of the progress of design or of the large quantities of generated data.

1 Introduction

It has been recognised for a long time that computer hardware cannot be developed or manufactured without the aid of other computers. This aid is provided by a set of applications software collectively called design automation (or DA). DA is the computer hardware designers' 'toolkit' which enables them to make their computer product out of hardware technologies.

This paper describes the DA tools (collectively named DA4) that were used to develop the Level 30 system, constructed from Fujitsu 8000-gate arrays, the C8K gate array (see the paper from Fujitsu in this issue)[1], and MSI and LSI components mounted on a hierarchy of multilayer printed circuit boards.

DA is used for a variety of purposes, the chief ones being:

- to *help* to get the design of computers right as early as possible in the development process, i.e. to establish confidence in the design without having to go through the prohibitively expensive and lengthy process of making a prototype
- to *automate* the detailed, tedious, repetitive and error-prone tasks involved in

computer development, e.g. the definition of the many thousands of individual interconnections between the millions of transistors making up the logic circuits of the computers
- to *control* and *correlate* all the data generated to design, develop, manufacture and maintain the hardware
- to *document* the design so that it can be commissioned, maintained and manufactured.

DA can also be called computer-aided design (CAD) or computer-aided engineering (CAE); in ICL it also encompasses a large amount of computer-aided manufacture (CAM) and computer-aided testing (CAT).

2 Structure of DA4

The development of DA4 began in 1975. It was targeted initially on providing tools for developing products built from large numbers of a similar style of multilayer printed circuit boards (PCBs) holding SSI and MSI components.

It runs on ICL DME machines and utilises many of the features of the George-3 operating system, for example, for presenting a user interface, handling security and data control. Unlike many systems developed at this time, DA4 was designed to a large extent to be technology-independent. The design has been demonstrably successful; by September 1981, when the Level 30 project was first identified, DA4 had been used to develop more than 20 products.

As well as technology independence, DA4 was also designed as an integrated system; design data is captured once only via a single hardware description language, and the system presents a consistent user interface.

DA4 software is modular and largely written in a high-level language (Algol-60), allowing the employment of a large library containing commonly used subroutines for accessing and transforming design data.

Finally, because it has this stable infrastructure, DA4 has accumulated a complete range of design tools, providing everything from logic simulation to numerically controlled output tapes.

3 Defining the DA tools for Level 30

Before Level 30 began, Mainframe Systems had an established development methodology, a design automation system to complement it and development staff and management accustomed to both.

However, the technologies to be used in Level 30, in particular the C8K, brought fresh challenges:

- C8K designs must be fully automatically testable and fast enough to meet the system performance requirements.

- C8Ks must be designed correctly, first time, not least because of the chip manufacturing cost and the turnround time.
- C8Ks contain a large amount of logic (average 7000 gates), all of which must be interconnected accurately, completely (no fail-to-track wires!) and within the tight timing and other design constraints necessary to build a state-of-the-art mainframe computer.
- There is a large number (42) of C8Ks in a Level 30, resulting in an unprecedented volume of data to control.
- The C8K vendor, Fujitsu, and ICL must agree that the manufactured chip has implemented the design intended; a very different magnitude of problem from that of goods receiving checks on MSI parts.
- Level 30 development timescales requirements meant that C8K development tools had to be in place in a very short timeframe, i.e. within the five months September 1981 to January 1982. In particular it was essential to provide tools to track and build chips in order to have a route for validating the design of unit cells.
- Four styles of technology (C8K, B2K gate arrays; large boards and daughterboards) were to be used, resulting in four different development routes. In the case of C8K the development route involved interfacing to Fujitsu fabrication and test processes.
- Additionally, the C8K technology was to be 'intercepted', which meant that these interfaces were not in place, and had to be jointly developed with Fujitsu.

The tools needs were defined by first establishing a development route for each of the base technologies:

- CMOS gate arrays (and some B2K Schottky gate arrays)
- daughterboards for mounting the MSI/LSI partitions
- large boards for containing the major partitions of the product.

The DA tools requirements to support these routes and technologies were then identified and their implementation begun. At the same time work was started on the detailed design of the PCB and chip technologies. The DA tools are described below.

4 System design verification tools

The system design was partitioned into high-level functional blocks. Each block was modelled in the ICL programming language S3. This language was chosen because of its facilities and runtime efficiency on 2900. The modules were then evaluated in a special-purpose unit level simulator, called SAUL, which applied existing test programs that had been developed to test previous families of 2900 machines.

This unit level description was developed to a level of design detail such that

functions could be mapped to a C8K chip. The results of this simulation were:

- confidence in the accuracy of the specification of the functional partitions of Level 30
- definition of the interfaces between chips in terms of validated test inputs and outputs.

5 C8K design tools

5.1 Logic design capture

The input to the C8K design tools is a logical diagram called a 'logic page' which uses 'symbols' describing permitted logic functions. Permitted logic functions (e.g. latches, shift registers, gates) are those for which 'unit cell' designs have been developed and evaluated by the technologists and then stored as fixed data in a DA4 component library, called, in the case of gate arrays, a 'cell library'. Unit cells are predefined arrangements of the C8K primitive (CMOS two-input NAND gate).

Two mechanisms for capturing logic diagrams are available in DA4. They can either be coded directly in the DA4 hardware description language RMOD, or they can be captured using an online graphic design station.

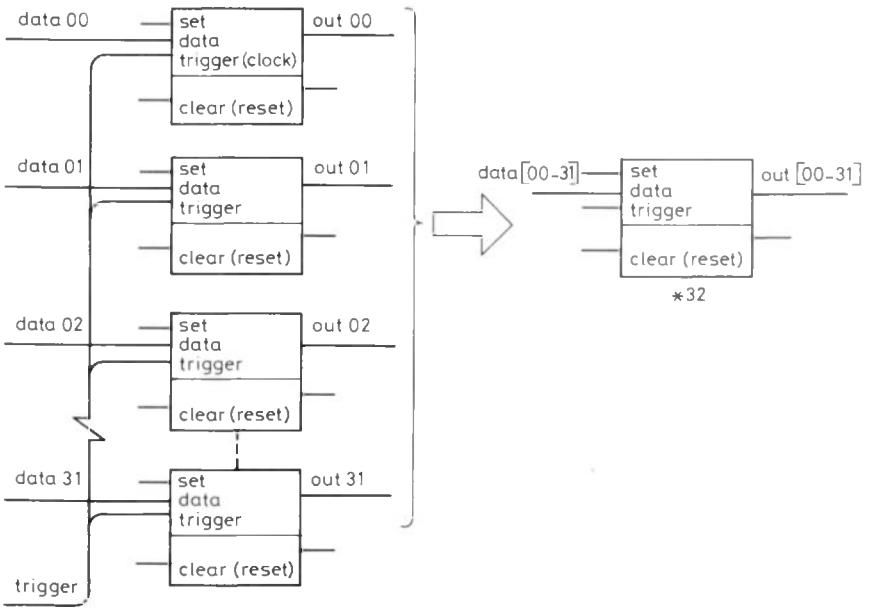

Fig. 1 Sample logic diagram

To condense the amount of code necessary to describe logic and to present repetitive logic in a meaningful manner, RMOD contains several unique features. The principal ones are called multisymbols and multistrings. These and the syntax of RMOD are illustrated in the sample logic diagram in Fig. 1.

5.2 Logical verification

Once the logic pages are captured in RMOD the chip logic can be simulated. The input to the DA simulator (MSIM) is the individual logic pages merged to form a complete description of the chip logic. Simulation was performed at cell level by reference to a library of simulation models which are 'S3' descriptions. The test patterns derived from the SAUL unit level simulations are used to evaluate the logic design. The verification process is summarised in Fig. 2.

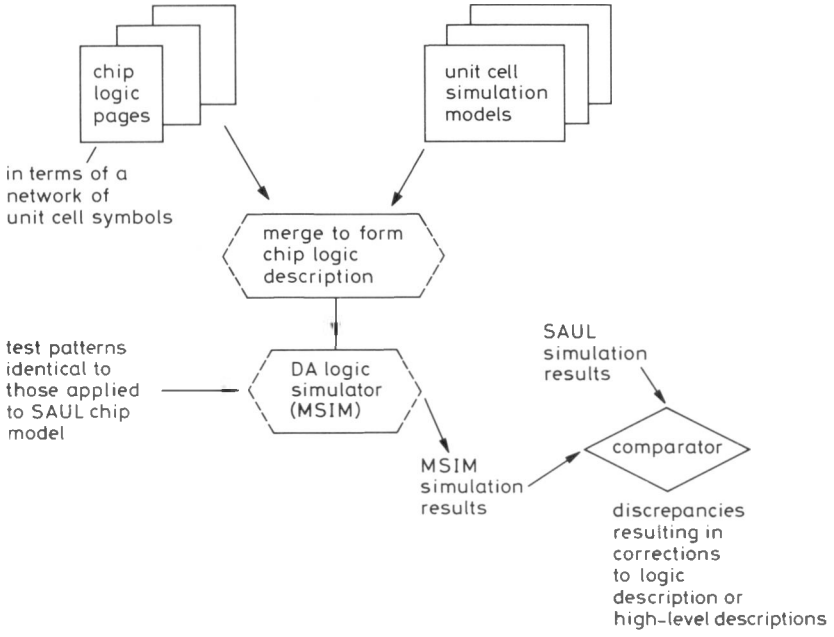

Fig. 2 Verification process

Having established an accurate logical description of the chip in terms of logical function, the RMOD descriptions are merged into a chip assembly file (Fig. 3) which is used as the basis of an expansion to create the full interconnect of the chip in terms of a net list. It is then possible to perform interconnect checks by reference to the interconnect rules coded as part of the DA4 fixed data description of C8K technology.

These error checks will establish, for example, whether there are any violations

of fanout rules (too many loads), unconnected sources or loads, or illegal connections to clock circuitry.

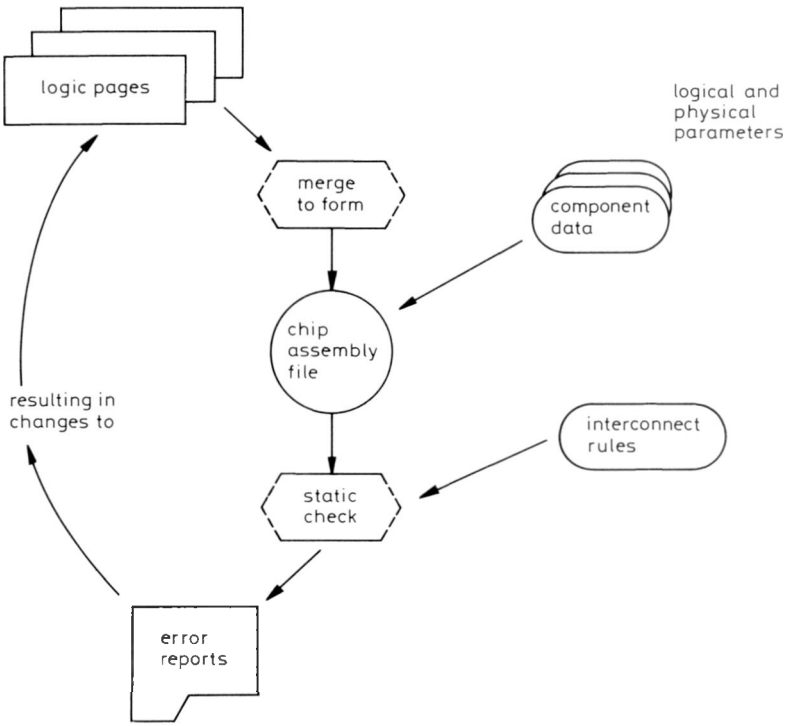

Fig. 3 Chip assembly file

5.3 Testability and timing checks

Having obtained a C8K description that is logically correct and free from interconnect rule errors, it is necessary to establish first that DA4 automatic test generation software can generate a test cover that will guarantee working parts from the chip manufacturer, and secondly that there are no logical paths in the chip that will result in out-of-tolerance timing conditions[2].

The test mechanism implemented in Level 30 is 'level sensitive scan design', or LSSD. The DA4 test generator, DGEN, uses 'path sensitisation techniques' involving the postulation of faults in a software model of the logic design and then computing by forward and backward simulating test paths through this model to ensure that the fault can be detected.

DGEN is a development of the test generator originally designed for deriving system diagnostics tests for complete mainframes. It has therefore been de-

veloped assuming the sort of volumes of test data and performance criteria demanded by C8Ks.

Running the test generator at this stage in the development has other benefits:

- The test cover achieved by the generation process depends critically on the generation models used for the cells and on the logic design of the chip.. It is therefore essential to ensure the testability of the design before commitment to silicon.
  A set of design rules was laid down as guidelines to the logic designers, and the generations were then carried out with a high degree of interaction between logic designers and the DA test generation team, to achieve a high fault cover (typically 99·8% of stuck-at faults) by changes where necessary to the models or the logic design. Manual test patterns were then created to increase the total cover to 100%.
- The test generator simulates the chip logic using a different modelling approach to that adopted by the gate level simulator. A further check on the correctness of the chip logic is therefore obtained simply by running the test generator and then establishing that the chip input and output test sequences are consistent with the results obtained from the gate level simulation. The chip test generation process is summarised in Fig. 4.

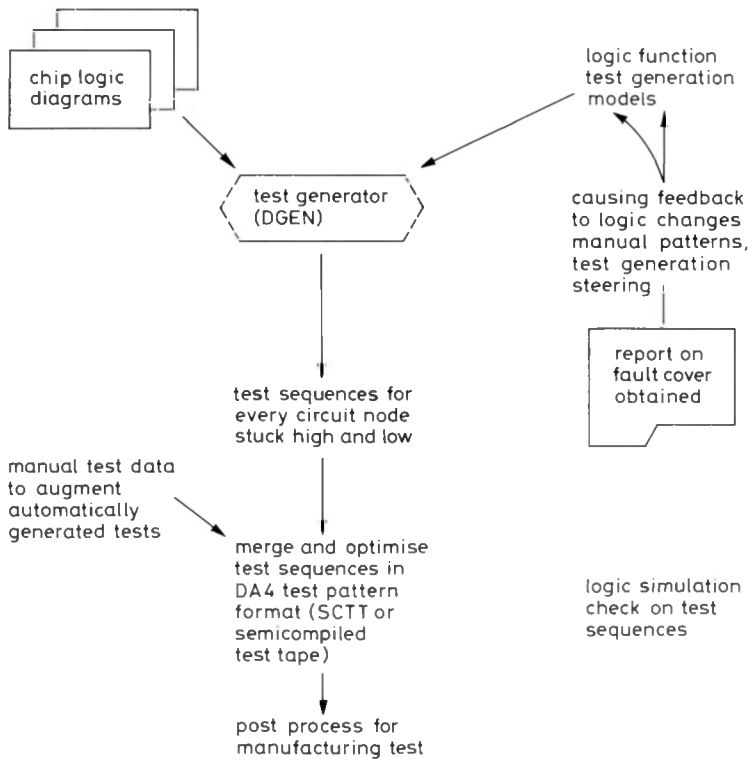

Fig. 4 Chip test generation process

Like the test generation tools, the DA4 timing tools are built around the fact that Level 30 is a synchronous machine divided into areas of combinatorial logic interfaced by storage elements, or staticisers. This approach supports checking of timing by summation of delays between staticisers due to interconnection and due to propagation through the active components in a network. This summation process leads to the identification of logical paths whose transmission times may be critical to the performance of the logic in that the logic may fail to operate within the required clock beat.

Both the timing tools and test tools in DA4 operate at three levels of hierarchy, the chip (and daughterboard), the large board and the system level. The higher levels of hierarchical descriptions are achieved 'bottom-up' by forming appropriate sets of logical interconnect data from logic page files and by using the results obtained by the test and timing tools run at the lower level as input to the next level up. This means, for example, that large board timing can be derived from the chip timing data as illustrated in Fig. 5.

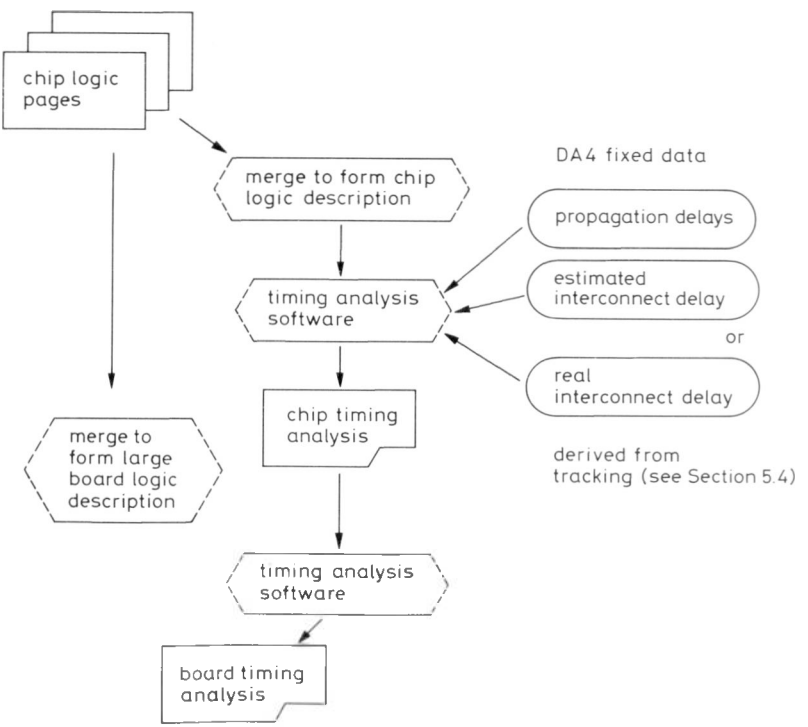

Fig. 5 Large board timing data derived from chip timing data

5.4 Placement and tracking

Given a chip assembly's worth of logic that has been verified logically, that obeys test generation and interconnect rules and has no timing faults, it is now

safe to lay out the logic on the C8K substrate and interconnect it. This process is called 'placement and tracking'.

The placement algorithm used the 'Min Cut' algorithm[3] which seeks to minimise the length of the total interconnect on a chip. This method is chosen because in general it eases the difficulty experienced by the subsequent 'tracking' software. However, the task is difficult and complex and to overcome these problems the C8K placement software offers many opportunities for user-controlled steering e.g. in the manual fixing of the placements of individual cells and in the modification of autoplacement results after the autoplacement software has run. There are also special placement steering features for obtaining an optimally laid out clocking system and giving special considerations for critical logic paths.

Having achieved a mapping of the logic onto the gate array substrate, it is possible to interconnect, or track, the logic. To avoid wasting expensive development time and computer resources, particularly possible in the case of difficult chips (i.e. those with a high utilisation of cells), an analysis of likely tracking difficulty is performed. Two methods have been evolved for the C8K. The first relies on the traditional approach of analysing the demand for tracking in relation to the available tracking routes; the other, developed later in the programme, is based on trial tracking utilising the speed (approximately 8 min to track the chip) with which tracking can be performed by distributed array processing (DAP) techniques (see below). If there is a high suspicion that tracking would prove unsuccessful for any particular chip, the placement is refined until confidence in chip trackability is revived.

Two suites of tracking software were developed for the C8K. They both followed the principle that having established (by all the preceding checking processes) an error-free logical description of the interconnect there will be no manual interaction with the tracking process unless that interaction was first checked out for logical consistency. Manual interaction with the tracking process is necessary, for example, to optimally distribute the clock system or to ensure critical path lengths are minimised. The tracking suite therefore contains extensive cross-checking software which will not allow the master definition of the interconnect to be updated unless there are no inconsistencies between the logical and physical descriptions. That this approach was successful is demonstrated by the fact that none of the chip designs sent for manufacture have evidence of errors introduced during placement and tracking.

Both trackers employed used the Lee algorithm[4,5]. The DAP tracker was approximately 20 times faster than that using the conventional machine.

5.5 Generating manufacturing data

The final stage in the development of a C8K is the generation of manufacturing data. Three categories of data are necessary to define a chip to Fujitsu:

– a logical description (called FLDL) created from the assembly file

- a description of placement and tracking data generated from the tracking database (called FPDL)
- a set of test data generated from the DA4 standard, semicompiled test tape, file (called CTPF).

The manufacturing data is extracted from the relevant files and written to magnetic tape. As well as the tools for generating the above data in the form required by the vendor, it is also necessary to ensure that the data sent is self-consistent, contains no errors and represents the desired design state.

Three mechanisms are used to achieve this:

- an automatic control system which keeps track of the design state of each chip as it progresses through the various stages of the DA system. It also ensures that any alteration to an aspect of the design must be reflected in the whole design
- extensive cross-checking software that processes the FPDL, FLDL and CTPF to ensure, for example, that there is correspondence between the unit cell connection points defined in the logical description and those defined in the physical description.
- rigorous audits by the project teams on the documentation created during the design process, such as logic diagrams and FLDL listings.

6 PCB development tools

Tools support for PCB design and manufacture depends on the characteristics of the PCB in terms of the number of types, physical size, component styles to be supported, interconnect complexity, logical complexity, production cost and production method. Two routes were established to support the two PCB types (large board and daughterboard).

With the very much smaller quantity of large boards and the complexity of the components to be supported on them (179 area pin pad C8K gate array and 96-way connectors) it was obvious that they would be of a complex multilayer construction. The DA4 autotracking software was therefore used in their case.

However, in the case of the daughterboards where a large quantity is involved, it is important that the production cost is kept to a minimum. To this end the daughterboards must have as few layers as possible and must not carry any 'fail to track' wires. These constraints led to a manual tracking approach to achieve a better utilisation of the tracking space.

DA4 was designed assuming automatic tracking and had very few aids to manual tracking. It was therefore decided to use the aids provided by ComputerVision.

The logics were still captured in DA4 and by providing software bridges to and from the ComputerVision it was possible to verify the physical design and also to provide output information to Manufacturing from DA4.

7 System test tools

An important aspect of the Level 30 field and manufacturing test philosophy is the application of the LSSD design style to the OCP, store and IOC large boards as well as to all the chips, (see Section 5.3). This approach allows the automatic generation of manufacturing tests for these hardware units with fault resolution down to a minimum of a chip or a daughterboard.

Using the node support computer it is possible to test and diagnose 'stuck-ats' and some other classes of fault within a chip and in the interconnect between chips, daughterboards and large boards.

The test generation tools used at system level are an extension of those used at chip level. The test software combines the previously generated chip tests with tests automatically generated for a subset of daughterboards and a model of the large board interconnect to build up complete large board tests. This approach avoids the need to generate the system diagnostic from scratch from a cell-level model of the complete system, with its associated requirement for large computer resources.

The test generation system also includes the provision of a test dictionary or interpreter that is used as an aid to fault diagnosis. This software is run in the system on the node support computer, usually from a remote 2900 mainframe connected via communications links to allow manufacturing technicians or field support engineers to remotely diagnose faulty chips, daughterboards or large boards.

8 Conclusions

The tools described in this paper have been applied successfully within the context of the Level 30 development routes for chips and PCBs. This success can be demonstrated by pointing at the successful development of 42 C8K chip types, some 70 daughterboard types and four large board types; all of which were developed within the timescales expected and none of which had to be reprocessed due to errors introduced by DA software.

The reasons for this achievement lie not only in the excellence of the tools provided, but more importantly in the design disciplines employed, especially the attention given to the definition of development routes prior to embarking on the design

Acknowledgments

This paper is the result of the considerable efforts made by the staff in the Design Automation Department in ICL's Mainframe Systems Division. We would also like to take this opportunity of thanking our colleagues in Fujitsu for the full co-operation they have given us in providing a special interface to their C8K process.

References

1 SUEHIRO, Y., MATSUMURA, N., SUGIURA, Y., YAMAMOTO, M. and HOSHIKAWA, R.: 'Development of 8000-gate CMOS gate arrays for the ICL Level 30 system', *ICL Tech. J.,* 1985, **4** (3), 289-300

2 KAY, A.. and OLDLAND, C.: 'Timing evaluation of logic design', IEE Conference on Electronic Design Automation (EDA 84), 1984.

3 HUNT, D.J.: 'Tracking of LSI chips and printed circuit boards using the ICL Distributed Array Processor', Parallel Computing '83, Berlin, 26th-28th September 1983.

4 LEE, C.Y.: 'An algorithm for path connections and its applications', *IRE Trans.*, 1961, **EC-10**, 346-365.

5 BREUER, M.A.: 'Min-Cut placement', *J. Design Automation & Fault Tolerant Computing*, October 1977, 343-362.

Design and manufacture of the cabinet for the ICL Series 39 Level 30 system

S.H. Martin
ICL Physical Design, West Gorton, Manchester

Abstract

The cabinet of ICL's new mainframe is much more than a mere housing. It not only represents a complete break with tradition, but incorporates an impressive range of functions *per se*. The design – as we envisaged from the start – is extremely cost-effective and undoubtedly makes a major contribution towards the favourable economics of the Level 30.

1 Introduction

About three years ago, at an early stage of the Level 30 project, we undertook a comprehensive study of all the options open for producing its cabinet. This study was deliberately freed from any constraints of existing design practice – in particular those imposed by the invariable use of craft-built, heavy steel structures to house mainframe equipment.

Metals have been used since time immemorial and, although still the subject of research and development, are thoroughly familiar to designers. Plastics, on the other hand, have only come into serious competition with metals in the past 30 years. Nevertheless, they can be used with confidence providing their quite different characteristics, such as their tendency to creep under long-term loading, are taken fully into account in a design.

Because plastics can be moulded into complex shapes in a single operation, they give the engineering designer great freedom. Much detail can be incorporated, so that assembly operations are either eliminated or simplified, and unskilled labour can be used. Furthermore, once the tools have been made and the processing parameters established, long production runs are not only possible but normal with plastics moulding. This contrasts strongly with the individual fabrication of each steel housing or cabinet. Finally, plastics are much lighter than metals.

2 Materials and tooling

In designing the cabinet for the Level 30 there were multiple objectives. From the outside, a thoroughly modern appearance was sought, together with favourable ergonomics to make the computer easy and pleasant to operate. The housing material had to provide adequate RFI (radio frequency interference) screening and protection from EMI (electromagnetic interference) and static discharge. Inside, there had to be a closely controlled environment, with a supply of filtered air for cooling and, overall, electrical and mechanical safety needed to be guaranteed. Above all, costs had to be kept down.

Plastics are potentially or actually superior to steel on all the above grounds; indeed, metals can now be regarded as a straightjacket for industrial design. There is no longer any question of whether the housings of smaller computers should be made in plastics or metal: the former are invariably used. The design team wished to extend the same advantages, and have the opportunity to incorporate styling and functional features not available with steel, to a mainframe machine for the first time. The perceived design of the cabinet would have had to have been utterly different in metal, and in practice a 44% unit cost advantage over steel has been realised (without allowing for tool amortisation) in addition to the major reduction in assembly time and the deskilling of the process.

Having made this basic choice, structural foam was identified as the plastics material that could potentially provide the optimum combination of functional opportunity and economy, while at the same time meeting all the electrical and mechanical demands of the application. A structural foam is a thermoplastic which has been expanded by the introduction of an inert gas, evolved by the decomposition of a blowing agent, while it is in the molten stage of an injection moulding process.

Even on the basis of the first sketches, it was evident that the frontiers of structural foam technology would need to be extended to make the cabinet. Mouldings of the same order of size were already being made by a few firms, although in the event the main component, the base, is generally acknowledged to be the largest structural foam part to be produced anywhere in the world to date. However, nothing then existed with anything like the complexity that was already envisaged for the cabinet. Nevertheless, after obtaining very useful advice on structural foam parameters from General Electrical Plastics, the project was considered to be viable.

The initial problem was to interpret a concept drawing in such a way as to provide a practical solution with minimum tooling costs. To this end, and to check the CAD drawings that were now available, before toolmaking a wooden model exactly simulating the parts to be moulded was constructed by Tom Brough & Co. of Brownlees, Stoke-upon-Trent, a pattern maker.

This model not only proved drawings, but gave a clear idea of physical appearance

to all involved in the Level 30 project. It also provided invaluable guidance on toolmaking and, at a later stage, on the production methods to be employed. The ICL Production Department were able to envisage what they would be dealing with at an early stage. In interpreting the model, however, due allowance had to be made for the fact that birch is about five times as stiff as structural foam. In fact, precise stress calculations needed to be worked out to ensure that the integrity of the structure could be absolutely guaranteed, but without over-design which would have had an adverse affect on the economics.

As stated above, a foam plastic material under load is completely different from that of a metal – particularly in the long-term stress at elevated temperatures. The stressing of rigid steel structures such as those employed in commercial computer companies is not usually critical but has to be taken most carefully into account in the initial structural foam design which essentially depends on resilience rather than rigidity. All stress calculations were rechecked empirically as the project proceeded to ensure that the performance criteria had been satisfied.

CI (Polymers) Ltd., Somercotes, Derbyshire (a member of the Cambridge Electronics Industry Group and specialists in structural foam mouldings) was appointed to produce parts, the material selected being Noryl FN215, a foamable grade of modified PPO (polyphenylene oxide). The manufacture of the steel tools was commissioned by CI (Polymers). The project emphasised the need for the closest co-operation between designers, moulders and toolmakers to ensure the viability of the project. The four tools (base, door, top and identical sides) together weigh some 20 tonnes, of which the base (part weight, 8 kg) alone counts for 7 tonnes.

There are important differences between structural foam and conventional injection moulding of a solid plastic material which have a major affect on tooling: because the filling of the tool cavity in the case of foam is assisted by the introduction of gas, the mould or clamping pressures are much less than they need to be when filling is achieved solely by mechanical pressure. This in turn means that a tool for foam moulding, although still made in steel, need not be of the single-piece construction associated with the normal injection process. Rather, the male and female sections of the tool can be assembled from a number of pieces, leading to great flexibility in design and better opportunities for modification.

Although no such modifications in fact proved necessary, this facility was a factor in choosing structural foam. The technique was also selected because it combined the necessity for minimum weight with maximum load-carrying capacity and the ability to cope with the physical size of the project.

Structural foam tools are, comparing like with like, somewhat less expensive than high-pressure injection tools. In view of their complexity, it is notable that only one side-core was needed in the complete set of tools (this being required for the interlocking device at the top of the cabinet).

The single-split cavities are filled in an interrupted moulding cycle from up to four points (gates) simultaneously. There are complex arrangements for venting the trapped air and surplus foaming gas to ensure the homogeneity of the part and to ensure close 'knitting' along the predetermined planes where the different material flow paths meet. It is a feature of structural foam that a solid skin is naturally formed on part surfaces, i.e. where the material comes into contact with the cooler cavity wall.

The extra expense of using 'hot runner' techniques was considered justified in all tools except that for producing the sides – the smaller – to give closer control over moulding. In hot-runner moulding the individual channels conveying the molten plastics material from the main outlet nozzle on the heated injection barrel to each cavity gate are also heated using thermocouple control. It is then also necessary to provide a hydraulically operated shut-off nozzle at each gate as well as at the main barrel nozzle. Much more precise adjustment of the temperature and pressure at which material enters the cavity is thereby effected.

3 Functional considerations

In a traditionally designed mainframe computer the steel cladding and the internal structure are effectively separated. As mentioned above, the use of plastics gives the designer freedom to integrate the design of the cabinet with that of its contents, and thereby to make it highly functional to an unprecedented degree. It is above all this factor which justifies the initial expense of complex tools.

The success of this approach may be gauged from the fact that, apart from its internal systems and their associated fittings, the basic Level 30 node cabinet consists of only eight major parts: the five structural foam mouldings (base, door, top and two sides) with a steel panel at the rear, covered by a further decorative panel, and a steel floor. The monocoque construction of the cabinet achieves its basic stability through the inclusion of this steel back panel, which locates the moulding assembly.

The structural foam amply possesses the robustness which, rather than the rigidity of steel, confers resilience in the design, together with the necessary thermal resistance (normal service temperature is up to 50°C). The material also satisfies the internationally recognised US Underwriters' Laboratory requirement (UL Subject 94) for flammability with a rating of V-0 at 6 mm wall thickness.

Considerations of RFI, EMI and static discharge screening are naturally a prime consideration, and attenuation of 30 dB up to a frequency of 600 MHz is guaranteed. The steel rear panel, which incorporates an air filter, acts as an RFI shield. Screening is effected at the junctions between the mouldings by the use of gasketing and special conductive tape to ensure even pressure. Further protection against RFI/EMI is achieved by zinc-spraying the mouldings or by the use of colloidal paint. The requirements for shielding against static discharge are helped by the opportunities conferred by the use of plastics housing for having the

cabinet intrinsically insulated on the outside, with RFI and EMI considerations satisfied by conductive coating on the inside. The exterior is finally painted in ICL house colours; styling for the entire design was by London Associates.

Perhaps the most remarkable feature of the design is the simplicity with which it is assembled, using no more complicated fastenings than interlocking slots secured by spring clips: a lock is incorporated in the base and door after moulding. The microelectronic equipment and its ancillaries – logic racks, fan, AC mountings etc. – is 'floated' into the final assembly, i.e. there is no rigid connection between the external structure and the electronic equipment it supports. It is this feature that provides the stark contrast between the practice of providing a rigid frame from which the modules etc. are mechanically isolated, demanding extremely close manufacturing tolerances and the employment of craft techniques.

As stated, the use of structural foam results in at least an equivalent degree of mehanical protection through resilience rather than rigidity. The flexing of the cabinet that may be observed during handling is not transmitted to the printed circuit boards and the subsidiary plug-in (daughter) boards and is allowed for in the design. All that is demanded is that the two-part, self-aligning flexible ribbon connections between these boards should remain engaged. The design also permits a very high degree of accessibility.

4 Four different versions

The initial design concept was solely for a cabinet for the basic Level 30 node, which incorporates two order code processors (OCPs), a 16 Mbyte store and two I/O controllers (IOCs). However, at a later stage it was decided to use the same design for carrying peripheral equipment in three further versions. One of these incorporates a high-speed peripheral controller, two 300 Mbyte FDS 300 discs and content address file system (CAFS-ISP); another four FDS 300 discs; and the fourth variant features a high-speed disc controller, CAFS and Faraday cage, to interface with customers' existing ICL MDSS discs.

5 Economic advantages

The successful and revolutionary use of this new design concept, based on structural foam moulding, can already be seen to have resulted in substantial economies for ICL, apart from the undoubted aesthetic attractions for users, in a number of ways. In the first place, there has been a large reduction and rationalisation in the number of components needed for the assembly of both the computer and its peripherals, with consequently minimised stockholding. Secondly, the assembly operations have been deskilled and greatly speeded up. The assembly time for the mouldings of a basic node cabinet (by two unqualified operators, rather than by a team of experienced fitters, as previously) is now no more than 7 min and the total system assembly time has been reduced to 2 hours.

Although the initial tooling cost for the structural foam moulding was heavy, the above factors have more than justified it. The break-even rate over craft-built steel cabinets, allowing for tool amortisation, is estimated at around 3500 sets of mouldings. The cost-effectiveness becomes self-evident bearing in mind the 44% *unit* cost saving over a steel cabinet mentioned above.

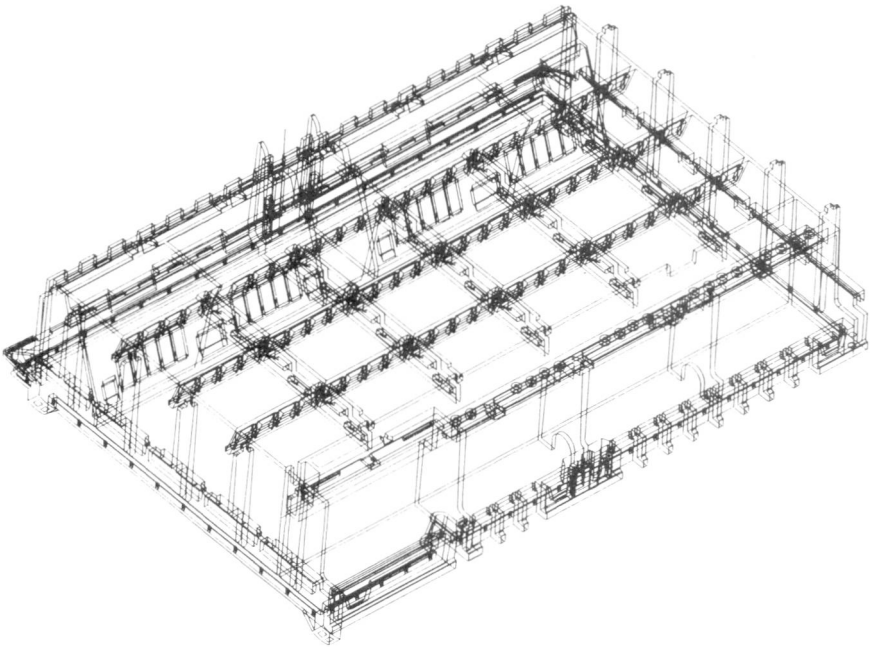

Fig. 1 CV wire frame base model

Manufacturing the Level 30 system I Mercury: an advanced production line

R.K. Shore
ICL Mainframe Manufacturing Centre, Ashton, Manchester

Abstract

The assembly and test of Level 30 are carried out on a specially designed advanced production line called Mercury. This uses the latest automatic warehousing techniques of computer-controlled stacker crane and conveyers, and high bay racking. The design incorporates a unique concept of automatic and manual shutter doors which are primarily for the safety of operators but which also serve to generate commands for crane movements. The application of this type of equipment to an assembly and test line is unique and can give very great savings of space.

1 Design criteria

When it was decided to establish an automated assembly and test line for Level 30 and other products, including the ICL System 25, the following criteria were laid down:

- delivery of high-quality products must be guaranteed
- a fully integrated, computer-controlled system is required, from components to fully tested, temperature-cycled products configured to customer requirements
- handling and routing of cabinets through assembly and test must be automatic
- testing of cabinets must be in a very compact area
- parts resupply must be automatic and based on the Japanese inventory control philosophy known as KAMBAN
- the total area of shop floor occupied must be as small as possible
- the line must be capable of handling up to 15 000 units a year
- the system must be flexible enough to deal with more than one product
- the system as a whole must be aesthetically in keeping with a modern computer manufacturing environment.

The design that was given the name Mercury was chosen from several options as giving the best solution overall.

2 Physical characteristics

Mercury is made up of two rows of racking 4 m high and 70 m long; a computer-controlled crane travels between the rows, moving the cabinets through the assembly and test stations and supplying parts to the operators. The layout of the line is shown diagrammatically in Fig. 1. The crane travels the 70 m from one end of the line to the other in approximately 30 s.

Conveyors feed cabinets from the racking to the assembly operators, and power-operated doors prevent access to the crane aisle. Parts for assembly are supplied in containers which are situated within the racking, adjacent to each workstation; containers not yet required are held in buffer storage locations above the workstations.

System and configuration testing are performed within the racking and the layout of the test area enables cabinets to be tested in two tiers. For the temperature cycling certain stations have heaters which swivel over the cabinet air intake, the cabinet's own fan being used to circulate the warm air. The arrangements for handling test data are described in the paper by Rollins and Cullen[1].

With Mercury it has been possible to reduce the floor area needed for the assembly and test operations, work-in-progress and parts storage to only 30% of what would be needed for a conventional production line.

3 Operation

The Mercury line is completely paperless, all instructions being given to the operators by means of visual display units. To ensure effective control over the input of parts laser scanners are used to read bar code labels attached to the containers and the information is used to generate and send to the crane, automatically, a command to move the required containers to the relevant workstations. Bar code readers positioned at each workstation ensure that the operators assemble the correct logic configuration and serialised assemblies into the correct cabinets. The information from these readers is stored, so that a full historical breakdown can be constructed if required.

Major assemblies such as logic boards are preallocated to cabinets by the control computer, which verifies and records the actual assembly when it is made. Thus the system records which assemblies are built into which cabinet, by serial number, and this information is passed to Quality Assurance Defect Analysis via Manufacturing Fault Clearance Control.

The control system monitors all work-in-progress automatically and tracks each cabinet through the various assembly and test stages. A colour-graphics mimic display in the control room is generated automatically and updated in real time (Fig. 2). This displays each cell, its contents and any movements pending.

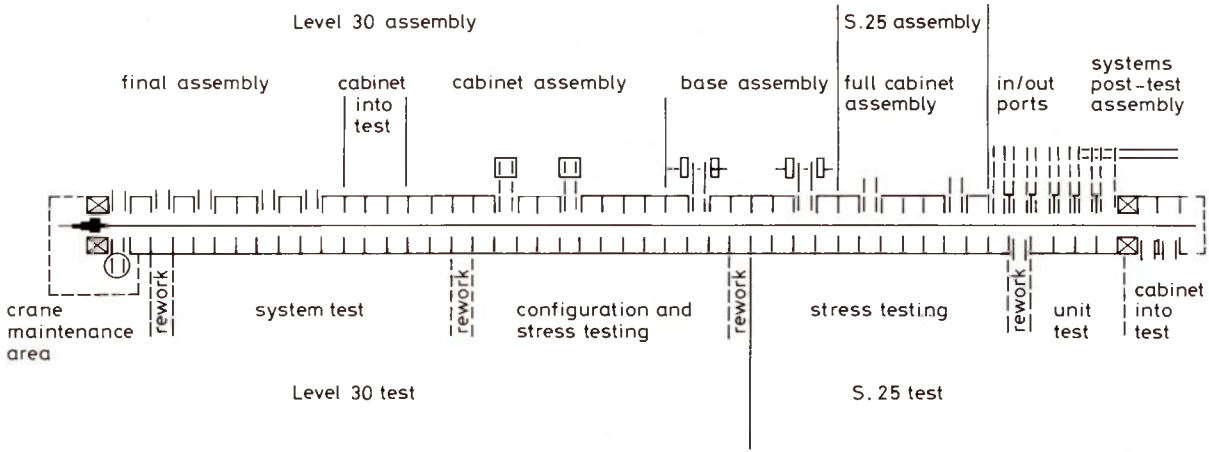

Fig. 1 Layout of the production line

On completion of a particular operation or test stage the door of the station is closed and a signal sent to the computer, which then puts a command on the crane command queue to move the cabinet to its next operation stage. Again, the sending of the signal and the subsequent movement by the crane are entirely automatic.

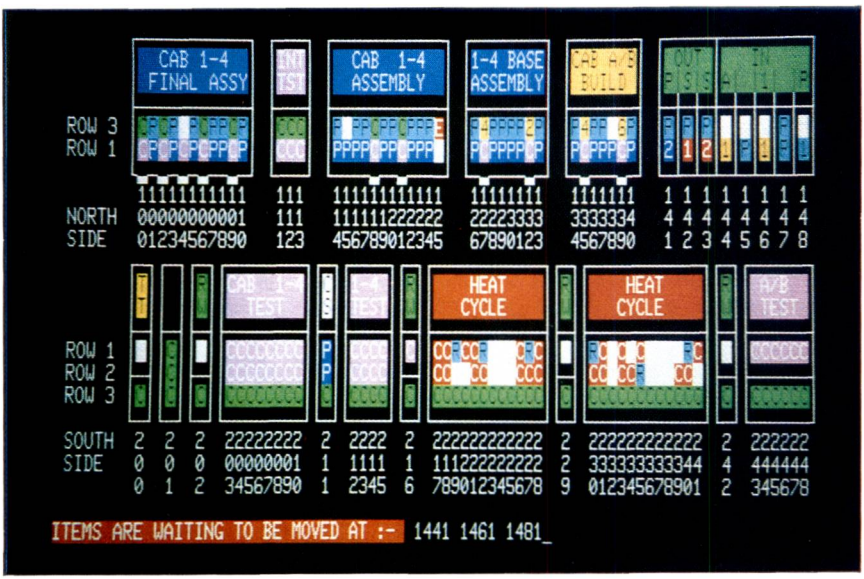

Fig. 2 Colour-graphics mimic display, located in the control room

The control system functions are performed automatically and can be summarised as follows; as has been said, the system is paperless.

- crane control
- routing of cabinets and parts containers
- online communication of build schedule and instructions to operators
- management of pallet buffer stock
- preallocating logic and major subassemblies to cabinets
- validating correct logic and assemblies to cabinets
- validating all data items used
- real-time monitoring of work-in-progress
- real-time production scheduling
- recording serial numbers of components assembled into cabinets
- Management reporting:
  - operation time variations
  - utilisation of work and test stations
  - utilisation of crane
  - work-in-progress inventory
  - throughput and performance statistics
  - downtime
- management of basic data, including control of updates

The system now runs on an ICL System 25; later it will be transferred to a Level 30.

4 Future potential

Mercury is a fully computerised system and offers considerable possibilities for future development. Examples are:

- interfacing to the ICL OMAC system for complete inventory management
- stores management
- automatic kitting for the next day's build
- 'ghost kitting': theoretical allocation of stock to meet predicted production requirements, showing where potential shortages may arise
- process control generally, because the complete system is in effect a very flexible process-control package.

Reference

1 ROLLINS, B. and CULLEN, J.G.: 'Manufacturing the Level 30 system. III The test system', *ICL Tech. J.*, 1985, 4 (3), 339-342.

Manufacturing the Level 30 system II Merlin: an advanced printed circuit board manufacturing system

M. Shubrook
ICL Manufacturing Centre, Kidsgrove, Staffordshire

Abstract

A computer-controlled system is being established at Kidsgrove for the automated manufacture and testing of the multilayer printed circuit boards used in ICL computers; the first phase of this is already in operation. The paper describes the system, with particular reference to the production of the boards for the Level 30.

1 Background

Some 500 different types of printed circuit board are used in the complete family of ICL products, some of these consisting of as many as 22 layers; It was decided in 1983 to set up an automated system for the integrated manufacture and testing of these boards, so as to gain the advantages of quality, quantity production and cost reduction achievable only through automated methods. This was to take place in three stages:

(1) installation of state-of-the art automatic assembly machines for insertion of components
(2) design and implementation of a flexible manufacturing system to link stores and kitting to the assembly and test stations under real-time computer control
(3) development into a fully integrated system incorporating a network of operating systems and automatic equipment and covering all aspects of business from receipt of orders and design information through to manufacture, assembly and test and then to distribution and servicing.

Phase 1 was completed during 1984 and the transfer from manual insertion of components to automatic achieved for all board types. Phase 2 is under way and planned to be completed by summer 1986; the automated stores facility will be operational by the end of 1985, to be followed by materials handling and paperless planning systems for assembly and test. Phase 3 will be achieved by a step-by-step approach to integrate, with full computer control, all the areas

from goods-in to despatch, along with special 'work cells' to control all manual items.

2 Printed circuit boards for Level 30

These are of two main categories:

Large boards ('motherboards'). Five different types, 39 x 52 cm, 14 layers, fine line tracked. The Fujitsu C8K (8000 gate) array chips described in the paper by Suehiro *et al.*[1] are soldered directly on to these boards, and there can be up to 22 of these on a single large board.

Daughterboards. 65 different types, mainly 19 x 10·5 cm, from double-sided to six layers. These carry the standard logic gates and memory devices and plug into the large boards.

The boards are designed in Mainframe Systems Development Division, as described elsewhere in this issue[2]. The design automation suite provides the data for the following operations:

- tracking the boards. This is produced in the form of tapes to drive Gerber plotters
- testing the bare boards, i.e. the boards before any components have been inserted
- component assembly, including the data needed to drive the automatic assembly machines
- testing the assembled boards
- tests for the C8K arrays.

This is the input to the manufacturing process.

3 Production sequence for the Level 30 boards

The main steps are shown diagrammatically in Fig. 1 and summarised below.

3.1 Artwork

Using the Gerber tape the tracks are plotted on photosensitive material and the plot checked to ensure that it is geometrically correct overall between check points to within 75 μm. The actual tracks are then checked to ensure that they are of the correct width and spacing; this is performed by an Oprotech Vision 104 optical inspection system which can detect flaws of 20 μm in a track 125 μm wide, which if undetected might cause the tracks to fuse in use. This check is performed on all copies of all artwork used during production.

3.2 Manufacture of bare boards

The process plant has the capability of manufacturing all bare boards used by

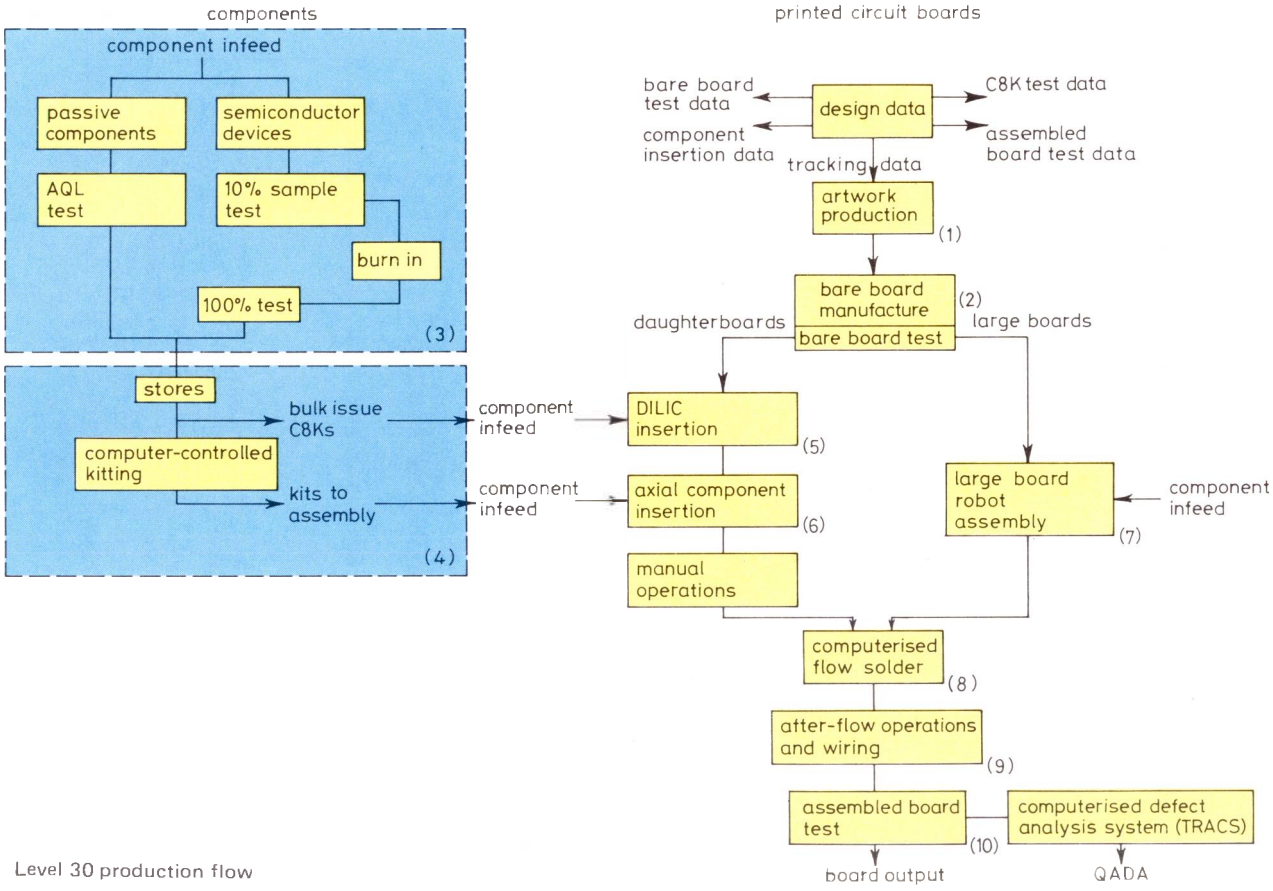

Fig. 1 Level 30 production flow

ICL, from OPD ('One per Desk') to Level 30. Quality is ensured by controlling the process to the very tight BS 9000 standard. The process entails (Fig. 2) plating copper on to laminate which has been printed with the artwork pattern on a photosensitive coating, thus creating the tracking. The various layers of tracking, called logic layers, are then 'glued' together by a bonding process in which a material called Pre Preg is used to fuse the layers together, so forming the board. The board is then drilled and the holes copper-plated so as to make the required connections between the inner layers of tracking, after which the outer circuits are printed and all unwanted copper is etched off. Finally a solder resist is applied to the board, which is then routed to its final shape and tested.

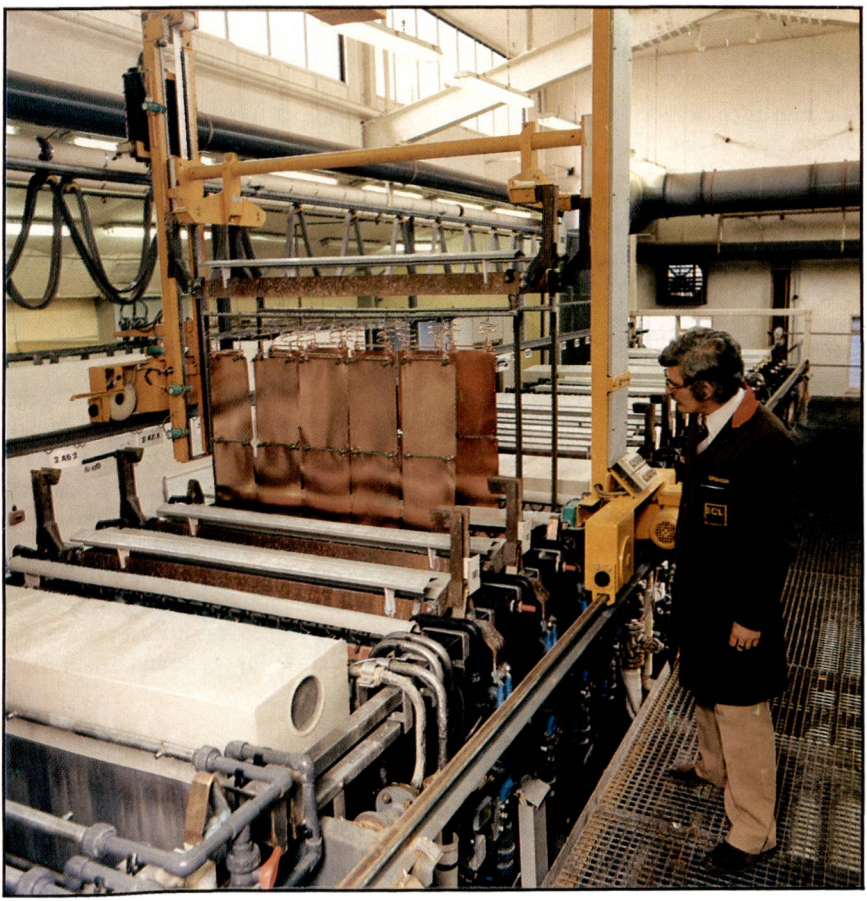

Fig. 2 Process plating line

All boards are continuity tested, both at the logic layer stage and after bonding. A new bare-board tester has been purchased for the Level 30 daughterboards, which has 16 000 test channels and can test four boards at a time. Following the success of the Oprotech 104 for artwork inspection it has been decided to

install an Oprotech 105 high-speed optical inspection system for checking all logic layers before bonding.

3.3 Component testing

This is vital to the achievement of a high-quality final product. All components are tested not merely to the manufacturers' levels but to a level equivalent to that applied to parts used in satellites and other fields where the highest reliability is required. Furthermore, component quality is monitored all through the processes of production at Kidsgrove and assembly at Ashton, and also to any devices failing in the field. All faulty devices are examined and diagnosed, if necessary by sectioning and using electron beam microscopes and other instruments, and information fed back to the vendor during regular reviews.

The aim is to achieve a zero defect policy covering all vendors. At Kidsgrove a failure rate of less than 100 parts per million is being achieved in PCB testing; the Fujitsu C8K gate arrays have a reliability target of a minimum of 10 million hours before fail, and tests so far indicate that at least this will be met. In 1981 the observed failure rate for bought-in silicon devices was 3%; this has now been reduced to 0·3% on a weekly throughput of over half a million devices.

3.4 Stores and kitting

All stores transactions are computer controlled, using the ICL OMAC (On-line Manufacturing Control) system; this is now running on an ME29 but will shortly be transferred to a VME system. About 10 000 transactions a day are handled. OMAC raises all kitting schedules and at present all items except axial devices (for example, resistors, capacitors) are picked manually from stock bins; this will be changed to carousel selection in Phase 2 of the Merlin project. Axial devices are all purchased on bandoliers which are loaded on to a 120-station Universal sequencer; information from the design automation suite is used to produce a sequenced bandolier, ready for insertion. The sequencer also measures component values to ensure that each device has the correct value, type and orientation.

3.5 DILIC insertion

For Level 30 this is handled by a Universal multi-mod automatic insertion machine (Fig. 3). This will insert up to 72 different dual-inline devices with up to 22 pins on 7·6 or 15·2 mm spacing at a rate of up to 4000 an hour. These machines have an automatic load system which enables boards to be fed directly into them, thus avoiding handling and consequently improving quality levels. This pass-through system is fully computer controlled and will allow for integration into the final flexible manufacturing system.

There is also a Universal box cap inserter for other boards which use 2·5 mm spaced radial devices and 7·6 mm spaced DILICs (dual-inline integrated circuits); but this is not used for Level 30.

Fig. 3 Universal multi-mod automatic insertion machines

3.6 Axial component insertion

This is done by a Universal variable centre-distance inserter which uses the presequenced bandolier and will insert devices at a rate of 12 000 per hour. The pass-through system is used here also.

NB: There are always a small number of devices which, because of size or shape, cannot be inserted automatically; connectors are an example. At present there are a number of manual assembly stations, with separately issued kits, to deal with these; in the later stages of Merlin development these will be replaced where possible by robot stations.

3.7 Robot assembly of large boards

There are currently no autoassembly machines capable of handling the C8K

(179-pin grid) gate arrays, large electrolytic capacitors, small decoupling capacitors and connectors simultaneously; it was therefore decided to develop a special robot assembly station for these operations, which would give an improvement in quality over manual assembly.

The cell is equipped with a Toshiba robot controlled by an ICL Personal Computer using the ICL autohandling system and bar-code scanner. It is loaded with up to eight boards, which can be a mix of any types. The first board is placed on a conveyor and moved into position, where the bar-code scanner reads the board type from the bare board and causes a program to be loaded into the computer. The robot arm then places the gate arrays, dispensed from a 64-station carousel, on the board. Both types of capacitors are inserted by the robot direct from bandoliers, and two different types of connector are fitted, as required by the particular board.

Fig. 4 is a photograph of the robot. The system allows for further integration into a flexible manufacturing system.

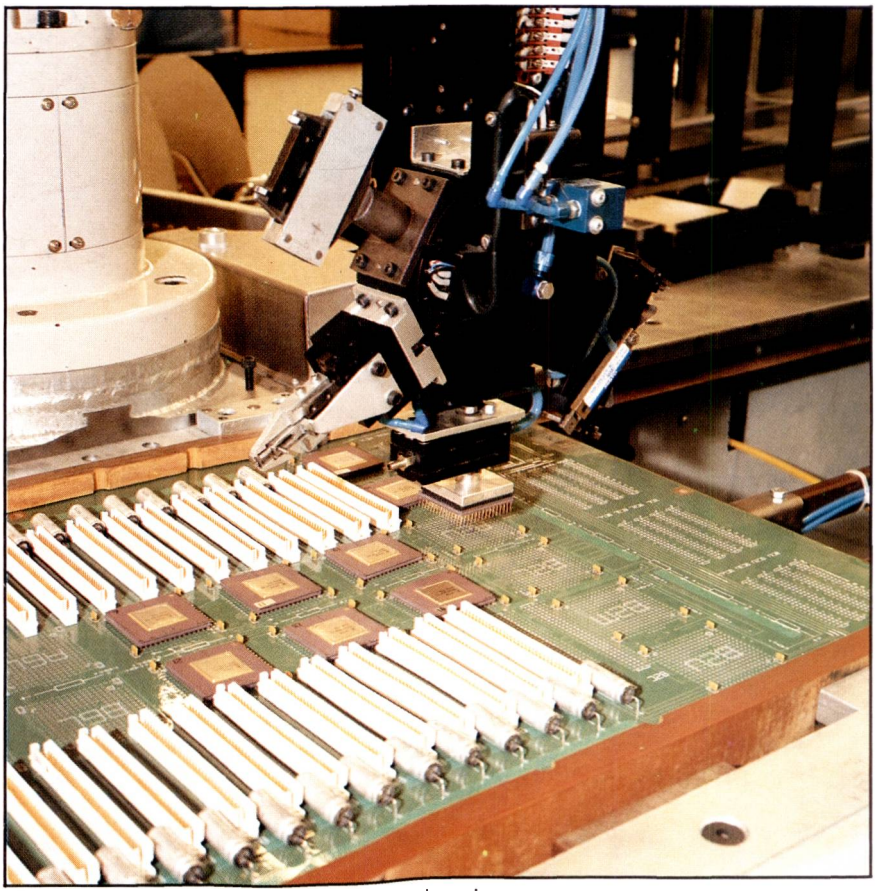

Fig. 4 Robot used in the assembly of large boards

3.8 Computer-controlled flow soldering

A state-of-the-art wave-soldering machine has been installed recently, with computer control with features such as flux viscosity and giving optimum solder quality for each type of board automatic. The solder module can be equipped with dual waves of different profiles which can be used independently; a sophisticated air-knife feature removes fine-line solder bridges.

The overall control here will be by an ICL Personal Computer and again allows future integration into the flexible manufacturing system.

3.9 After-flow operations

There are a few devices that cannot be fitted before the flow soldering because they may be damaged during either the flow or the wash-off; these are handsoldered at after-flow assembly stations. It is also necessary on occasions to fit wires required for modifications; to ensure correct placement of such wires Universal Logpoint machines have been installed, programmed with start and finish locations for each wire. The board is placed in a jig and a light beam indicates the termination points in sequence; this information is also displayed on a VDU.

3.10 Testing the assembled boards

The Kidsgrove PCB test area contains some 50 machines used for testing approximately 10 000 boards a week, of 500 different types and in batch sizes from 1 to 300. Some 15 of these machines are used for Level 30, from commercial automatic equipment to special equipment designed and built by ICL for testing storeboards, optical transmitter/receiver boards and the four boards contained in the node support computer.

The majority of boards first undergo a 'cold test' which checks for continuity and short circuits; this uses very-low-level (200 mV) signals, which will not damage any devices on the boards if there are any short circuits. The boards are then put through a full functional test to ensure that all defects are removed before despatch. At the end of the test cycle there is a 100% final inspection, after which the boards are packed in static-safe bags for despatch.

4 Quality statistics – 'TRACS'

TRACS means Test, Repair and Analysis Control System. This is a real-time computer-controlled system for logging defects discovered in testing boards. The test machines are connected together by means of a local area network (LAN). Every time a board is tested its serial number, read by a bar-code reader, is entered. The results of the test are fed along the LAN and logged into a local database. If the test is failed this information is used at repair stations, via a VDU, to analyse the cause of the failure and to enter the corrective actions to be taken, this information also being loaded into the local database. The board

will be retested after repair. All this information is loaded into the company-wide quality assurance defect analysis (QADA) database daily, so that full PCB history is available not only in Kidsgrove but to any site with access to QADA.

TRACS provides also a real-time warning of any failure parameters, such as repeated faults, being exceeded; this shortens the time from observation of defect to feedback of information relating to any problem involving assemblies or components.

5 Summary

The decision taken by ICL to centralise the production of printed circuit boards makes it feasible to introduce the greatest possible amount of automation into the process. This brings the advantages not only of economy of scale but also of assured quality and great potential for future development. The process has already been automated, extensions are planned and are in progress and possibilities for the future are being continually monitored.

References

1 SUEHIRO, Y., MATSUMURA, N., SUGIURA, Y., YAMAMOTO, M. and HOSHIKAWA, R.: 'Development of 8000-gate CMOS gate arrays for the ICL Level 30 system', *ICL Tech. J.*, 1985, **4** (3), 289-300.

2 HEWSON, M. and HU, C.H.: 'Design automation tools used in the development of the ICL Series 39 Level 30 system', *ICL Tech. J.*, 1985, **4** (3), 307-318.

Manufacturing the Level 30 system
III The test system

B. Rollins and J.G. Cullen
ICL Mainframe Manufacturing Centre, Ashton, Manchester

Abstract

Testing the final product is a vital part of any manufacturing process and is especially important when large volumes are concerned. The paper describes the automated test process set up in connection with the Mercury production line for the Level 30 system.

1 Requirements

The Mercury line for the production of Level 30 systems in high volume has been described in the paper by Shore[1]. It was obvious from the start that the assembly process would have to be complemented by some sophisticated and efficient test system and that this would have to be incorporated as a key element into the automated production strategy. The requirements laid down for this were as follows:

- test standards to give a high-quality product
- high volume throughput
- control of assembly and statistics for the complete throughput
- statistical analysis of test results and forecasting of trends
- fault diagnosis and feedback of fault information
- minimal operator intervention
- minimal paperwork.

Discussion with the design and development authorities when Level 30 was being planned led to a proposal for a system providing central control of assembly and test stages which would automatically and immediately process the data generated there and provide immediate visibility of any problems that arose. As well as being required to reduce operator intervention to a minimum, the system was to allow a reduction in skill levels to be made in the assembly and test areas.

The Manufacturing Fault-Clearance Control (MFCC) system has been designed to meet these requirements.

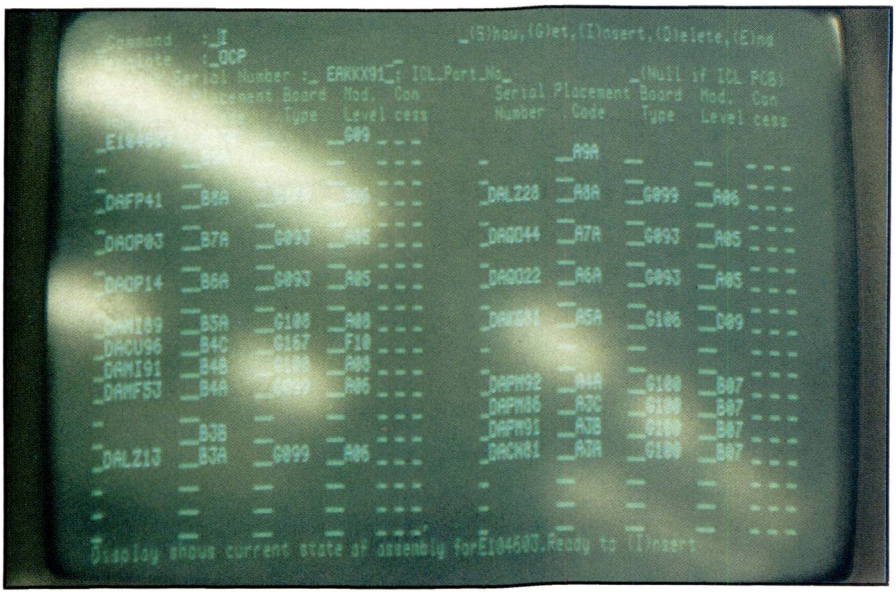

Fig. 1 MFCC-driven assembly template for a Level 30 order code processor (OCP). This directs the operator during assembly and logs serial numbers into their correct positions on the large PCB

2 The MFCC system

Before describing this we must say a few words about the other data-processing systems to which it is linked. These are

- LQS local quality system. This runs on an ICL 2966 at Ashton; it handles all the information relating to build, assembly, test and fail of ICL 2966 systems in production and test areas and the failure and repair statistics for Level 30. It was designed in-house, using the ICL Quickbuild product.
- QADA quality assurance and defect analysis. This is a company-wide database, held at Hitchin in Hertfordshire. Information is fed into this from all ICL factories; it can be accessed not only from ICL centres but also from customer sites by means of the online support and maintenance (SAM) software running on customers' machines.

MFCC is a specially designed package running on an ICL Superdual 2966 and covering the following main areas.

2.1 Assembly data

Each assembly point for logic hardware is under the direction of MFCC by means of screen-formatted displays (Fig. 1) adjacent to the assembly points in both the production line and the test area. Information is input automatically from bar code readers, leading to a complete equipment schedule consisting of a

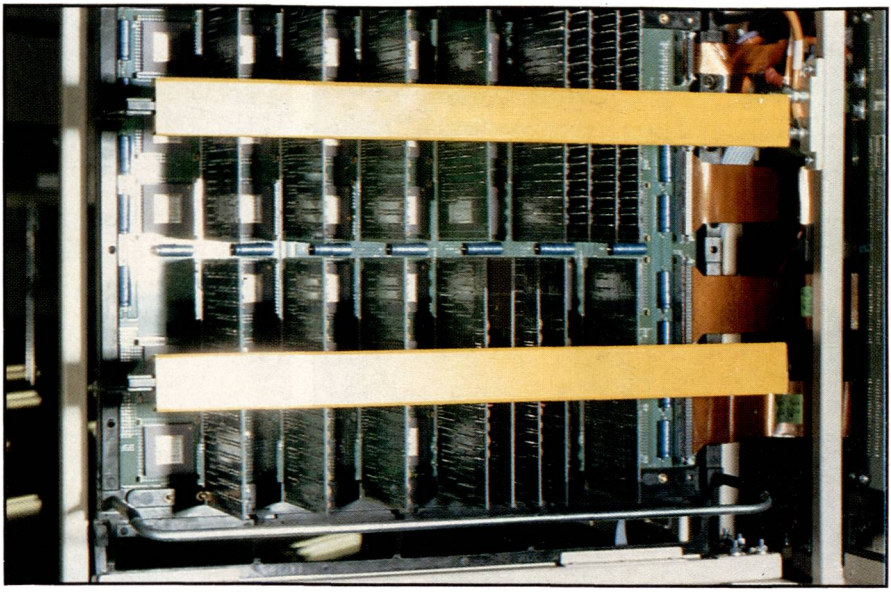

Fig. 2 Assembled OCP large PCB awaiting test, showing daughterboard subassemblies

hierarchy of serialised parts from subassemblies to completed cabinets and giving full details of everything to be delivered to the customer.

2.2 *Test control and monitor*

All test stations, of which there are 25 at present, are connected to MFCC by local area network (LAN) and communications links. Engineers and technicians assemble the logic to be tested under MFCC direction (Fig. 2); when this is completed MFCC takes control and enters a predetermined automatic test sequence consisting of engineering test programs, VME-based controller tests and finally a special in-house VME benchmark trial suite which carries out a complete system test of all the logic. Customer configurations are then brought together and subjected to tests before despatch, including testing at elevated temperatures. MFCC monitors the results of these and in the event of a failure tells the engineer which is the failing part to change.

2.3 *Diagnosis*

Built into MFCC is a database of predicted fail patterns ('fingerprints'), together with the ability to expand this as more information is fed in. This Known Error Log (KEL) provides a constantly up-to-date record of problems encountered in the Ashton manufacturing environment; it is merged regularly with the central maintenance database (MDB) compiled from field engineers' reports, so allowing further refining of predictions on failing parts. MFCC presents diagnostic infor-

Fig. 3 Opened-out node test rig after assembly and prior to installation in cabinet

mation to the engineers and so gives them the ability to use additional engineering tests for investigation of faults.

2.4 Fault feedback

Information on diagnosed faults is fed back to earlier stages and to feeder plants to improve test cover and therefore to give a higher quality product. Direct logging of fail data into LQS provides a means of monitoring current failures and so makes trends visible and alerts attention to imminent problems.

3 Summary

The overall effect of this software-based control of assembly, test, data capture and data handling is to give a virtually paperless system with a built-in ability to learn from the results of experience fed back into it. It is thus a powerful prodution tool which is constantly refining its predictions.

Reference

1. SHORE, R.K.: 'Manufacturing the Level 30 system. I Mercury: an advanced production line', *ICL Tech. J.*, 1985, **4** (3), 325-329.

Patent applications arising out of the Level 30 project

The following is a list of the patent applications that have been made in connection with the work on this project. GB and EP indicate applications for British and European patents, respectively.

This information was provided by ICL Patents Services, Stevenage.

Public writes EP 83 300 516
N.L. Vince
Multinode processing system in which shared data is replicated in each node and kept up-to-date by broadcasting messages.

Signalling hardware fails GB 83 13 479
B.J. Proctor
Parity check signals from chips are inverted in alternate clock beats. This allows failure of clock supply to chips to be detected.

Diagnostic loop selection GB 83 15 751
P.L.L. Desyllas, A. Spillar
Diagnostic loops are connected in a tree-like structure and selected by sending addresses serially through the tree.

Double go-ahead GB 83 11 274
D. Bell, S.O'Connor
Ring network using duplicated pattern as a token, to give greater security.

Slave validity mechanism GB 83 23 716
P.L.L. Desyllas, N. Holt
Each time the current tag register is incremented, a fraction of the slave store is cleared so that by the time the CTR recycles all the slave will have been cleared.

Soft control of store GB 83 26 885
S.J. Delahunt
Store control signals (RAS, CAS etc.) are produced by a writable control store, so that they can easily be retimed.

State step error check GB 83 21 373
J. Howarth
Checking a sequential circuit by comparing a code representing the predicted state with code representing actual state.

Micro-bus handshake GB 83 29 805
R.J. Ogden
When processor is not using I/O controller bus, it loops requests from I/O controller back to it as acknowledgment signals.

Refresh address generator GB 83 21 374
B.D. Wells
Use of linear feedback shift register to generate refresh address for dynamic RAM. Uses less chip area than a counter.

Address fan-out GB 83 21 372
M.J. Ratcliffe
Method of fanning out addresses to memory chips so that, if one fan-out circuit fails, no more than two chips in row will be wrongly addressed, which can be corrected by Hamming code.

Star network EP 84 302 628
S.O'Connor, D. Bell, T.R. Fox, P. Townsend
Star network in which central unit receives and forwards messages, with token-passing protocol.

Chip test mode EP 83 306 426
G.M. Bradshaw, P.L.L. Desyllas, K. McLaren
Diagnostic loops can be divided into mini loops each extending between separate pairs of pins to allow rapid testing of chip.

Grid array PCB GB 84 21 093
G.L. Thompson, B. Gudgeon, K.J. Wombwell
Pins of a pin-grid-array package are received in non-plated holes in a circuit board with connections to surface pads only.

Zip tube entry GB 84 04 479
S.H. Martin
Cable entry into side of zip-tube enables cabling between units to be accommodated in low height space.

PCB polyimide capping GB 84 21 094
G.L. Thompson
Printed circuit Board with polyimide-glass layer carrying pads for connection to pins of package.

Manchester decoder GB 85 06 100
T.R. Fox
Manchester decoder with tuned circuit to assist jitter rejection.

Notes on the authors

D.W. Ashcroft Processing node of the ICL Series 39 Level 30 system

Derek Ashcroft was first an apprentice and then an instrumentation engineer at AEI, Trafford Park, Manchester (one of the companies that now forms part of GEC). After 1965 he worked first for the Instrumentation Division of English Electric at Stafford and later for a part of the Philips group at Cambridge. He joined ICL in 1970 as a section leader on the 2970 control, followed by responsibilities in the design and development of 2966, later becoming responsible for the Level 30 processor. He now manages the Medium System Sector under the Director of Hardware Engineering at West Gorton.

P. Broughton Input/output controller and local area networks of the ICL Series 39 Level 30 system
The store of the ICL Series 39 Level 30 system

Phil Broughton has a B.Sc. in Electronics Sciences from Southampton University. After graduating in 1970 he joined ICL to work on the hardware design of large and medium mainframe machines; as a logic designer he was one of the key engineers for the 2980 and its development into the 2982, later working on the design of a large pipeline machine proposed as a successor to the 2980. In the Level 30 project he has managed the infrastructure sector, with responsibility for the design of the store, the I/O couplers (HSC, MSC, LSC) the clock and the printed-circuit-board technology.

J.G. Cullen Manufacturing the Level 30 system. III The test system

Graham Cullen has HNC with Endorsements in Electronics and Computers; his main interest is in methods and services for the support of production. At the Ashton factory he implemented the services to support the installation and commissioning first of the 2966 and then of Level 30. He was also responsible for introducing the MFCC (Manufacturing Fault-Clearance and Control) into Ashton.

M. Hewson Design automation tools used in the development of the ICL Series 39 Level 30 system

Mike Hewson, C.Eng. MIEE, joined English Electric as a student apprentice at Kidsgrove and moved to Manchester as Sector Manager in the Design Automation organisation. He was responsible for all the design automation software used in the Level 30 project. He has recently left ICL and has joined Marconi Electronics Development Ltd. at Lincoln.

R. Hoshikawa Development of 8000-gate CMOS gate arrays

Ryusuko Hoshikawa graduated from Hokkaido University, Sapporo with an MS degree in Electronic Engineering. He has been working on IC design since he joined Fujitsu in 1968. He has been manager of the MOS Logic IC Design Department.

H.C. Hu Design automation tools used in the development of the ICL Series 39 Level 30 system

Chi Hu has a B.Sc. degree in Engineering from London University, and C.Eng, MIEE. He joined ICT, one of the constitutent companies of the future ICL, at Stevenage in 1966 and has been in Design Automation ever since. As Project Manager in this field he played a major role in the Level 30 project on most CAD software matters, and was one of the team that worked closely with Fujitsu on the C8000 development.

J.A. Maddison The high-speed peripheral controller for the Level 30 system

John Maddison graduated in Physics and Business Management from Nottingham University in 1965 and joined English Electric Computers, one of the constituent companies of the future ICL, as a test programmer. He works in the I/O Development Systems Segment, where he has overall responsibility for the testing strategies for such projects as the High Speed Peripheral Controller in the development, manufacturing and customer environments. He has a keen interest in the use of ICL wordprocessors for the production of documentation for development projects.

S.H. Martin Design and manufacture of the cabinet for the ICL Series 39 Level 30 system

Sid Martin has an HNC in Electronics. His major interest has always been in the field of industrial design, in particular the problems of ergonomics, environmental design and designing to cost. He has been concerned with packaging of electronic systems, cooling and screening, power distribution and design and interconnection of printed circuit boards. He designed the power cabinet for the Elliott 803, was responsible for the physical design of the ICL 2903 and for the common hardware cabinets used on all the 2900 systems, and has been responsible for the physical design of the Level 30 system.

N. Matsumura Development of 8000-gate CMOS gate arrays

Nobutake Matsumura graduated from Tohoku University with a B.S. degree in Electronic Engineering. He has been working on design of MOS integrated circuits since he joined Fujitsu in 1972.

R.J. Metcalfe Development route for the C8K 8000-gate CMOS array

Roy Metcalfe has an HND in Electronics and Grad. IERE. In the Level 30 project he has been concerned with the interface between the design work and the Design Automation tools, in particular with the specification of the additional tools found to be needed and the initial use and validation of these; also with the provision of an advisory service on the use of the DA tools. Previously he had worked on the development of the ICL Distributed Array Processor (DAP).

B. Rollins Manufacturing the Level 30 system. III The test system

Brian Rollins has C.Eng. and MIERE qualifications and his main interest is in production testing. He was responsible for introducing into Ashton the production of 2966 enhancements, Level 30 and Level 80; in the Level 30 project he is Production Control Manager for manufacturing operations at Ashton.

R.K. Shore Manufacturing the Level 30 system. I Mercury: an advanced production line

Rob Shore holds a full Technological Certificate in Production Engineering, and has main interests in the general field of automated production, including systems for materials handling, flexible manufacturing, control and management information. He is Project Manager for the Mercury production line for Level 30 and was responsible for the implementation of Phase 1.

M. Shubrook Manufacturing the Level 30 system. II Merlin: an advanced printed circuit board manufacturing system

Malcolm Shubrook served a full apprenticeship in Electronics at the Royal Aircraft Establishment, Farnborough and holds the City & Guilds Certificate in Electronics. His main interest is in the manufacture and testing of printed circuit boards, with emphasis on ensuring full cover up to the unit and system stages. He has been with ICL for the past seven years, during which time he has been concerned with the manufacture and testing of 2950 and 2966, and in the Level 30 project has been responsible for introducing the manufacture and testing of the printed circuit boards into Kidsgrove.

C.J. Skelton Overview of the ICL Series 39 Level 30 system

Colin Skelton graduated from University College London with an honours degree in Physics. He designed servosystems and fire control electronics with Hawker Siddeley Dynamics (now British Aerospace) before joining ICL in 1971, and since then has managed the development of the successful 2960, 2956 and 2966 range of mainframe systems. In the Level 30 project he has been responsible for the design and development of the hardware system. He is now consortium project manager for the new fifth generation parallel processing architecture project.

Y. Suehiro Development of 8000-gate CMOS gate arrays

Yoshiyuki Suehiro graduated from Osaka University with B.S degree in Electrical Engineering. He has been working on the design of MOS integrated circuits since joining Fujitsu in 1976.

Y. Sugiura Development of 8000-gate CMOS gate arrays

Yoshihido Sugiura graduated from Kelo University, Tokyo with an M.S. degree in Electronic Engineering. He has been working on the development of IC CAD systems since he joined Fujitsu in 1975.

R.E. Thomas Development route for the C8K 8000-gate CMOS array

Dick Thomas qualified as an electrical engineer and has had very broad experience in the computer industry in both hardware and software, having worked on research, development, commissioning, support, marketing and customer applications. These last have been as diverse as a management information system for a large consumer goods company and real-time military control for NATO.

B.C. Warboys VMC nodal architecture: a model for the realisation of a distributed system concept

Brian Warboys has worked for 'ICL' since joining English Electric on graduation from Southampton University with a degree in Mathematics in 1963. For the great majority of that time he has concentrated on the design and development of operating systems and from the start of 2900 until recently was chief architect of the VME system. This led to an awareness of the need for, and an interest in, a study of the software engineering requirements for such developments. With time, this work and responsibility have widened to include all aspects of system developments; At present he is Director of Engineering, Office Systems Division and is Visiting Professor of Information Technology at Stirling University; last year he was appointed the first ICL Fellow.

M. Yamamoto Development of 8000-gate CMOS gate arrays

Minoru Yamamoto graduated from Chiba University and received the MS degree in Electrical Engineering. He joined the semiconductor laboratory of Fujitsu Laboratories in 1969 and then joined the MOS Logic Design Department in 1983. He has been a section manager developing gate arrays as well as application-specific ICs.