

Predatory Computing Performance



meiko

A · New · Enabling Technology

◆ Today nearly everybody has access to a personal computer. A few have a personal supermini. Very few have a personal supercomputer. Most do not have quite enough computing power for the job in hand.

Some technical professionals have been fortunate – theirs is a field for which it happens to have been easy to provide tools. Personal engineering workstations have given these individuals sufficient power to work, optimally at their own rates, unhindered by the activities of others who would otherwise be contending for the same resource.

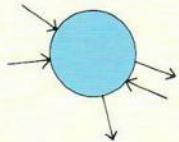
But what of all the technical professionals whose skills happen to involve computationally complex calculations and models, or who make decisions based on the analysis of very large amounts of information – those who need very powerful computers capable of high throughput?

Their turn has come.

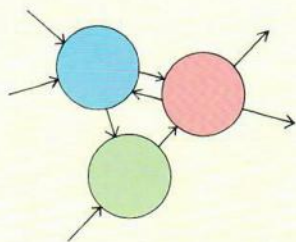
Meiko has a solution that delivers sufficient power, right now, cost effectively – around \$250 a MIP – and demonstrates a uniform expansion path to unlimited power in the future.

The Computing Surface is a flexible, extensible concurrent supercomputer. It exploits the latest VLSI technology to provide virtually limitless computing capabilities.

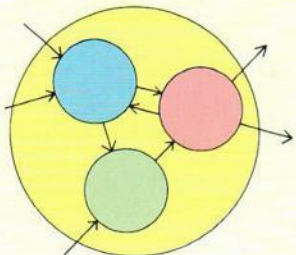
Concurrency is Key



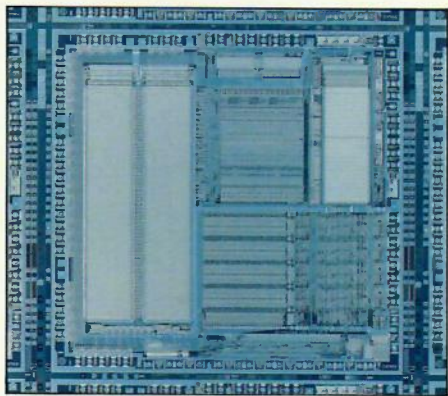
A process is a "black box", with inputs and output, that can process information



Processes can be connected together by channels to build more complex concurrent systems



A collection of processes is itself a process, so that processes may have internal concurrency



The INMOS T414 Transputer

◆ As conventional sequential processors approached the performance limits imposed by the laws of physics, the greatest potential for performance improvement clearly lay with applying many processors to the same task. The intellectual challenge posed was mapping applications onto processors and ensuring the integrity of information – both program and data – exchanged between processors.

Eight years ago the Meiko co-founders participated in the formation of a pioneering team attacking this issue. Intense study and debate determined that Communicating Sequential Processes, postulated by the highly respected Professor C A R Hoare at England's Oxford University, offered the best model to achieve that goal of matching computation and communication with integrity.

Next, this intellectual insight needed to be made a practical reality. Simply plugging more processors onto a common bus quickly runs into problems of contention. Linking processors individually to shared memory merely shifts the bottleneck to the memory. Thus parallel efforts commenced on the design of a semiconductor device to integrate secure communication with processor and memory, and the development of a language needed to create and direct the flows of information.

The results were the transputer™ family of devices and the occam™ programming language. The founders of Meiko were the managers of the design group responsible for the transputer and its peripherals. They have written compilers for the transputer. And to develop the transputer a CAD system with over 200,000 lines of code and customised workstations had to be created and supported by this team. This unique experience made it possible for the Meiko founders to exploit fully the potential of the transputer and occam.

Various topologies were considered. And rejected. Rings, meshes, n-cubes, cylinders, trees, grids, toroids etc all suffered from a major defect. They were compromises. Rigid, fixed topologies that avoided the not inconsiderable difficulties of providing complete flexibility. But different applications require different solutions. And the differences are manifold. Not only do the optimum topologies differ between problems, so do the type of resources required; some are compute intensive, others require rapid access to a large database.

Meiko thus created a very fast supercomputer, with totally flexible user determined topology, out of modular subsystems that permit optimisation of compute, store and i/o to particular classes of application; together with straightforward software and tools essential to programming this class of machine.

First shown in July 1985 at SIGGRAPH in San Francisco, since then Meiko has been developing and testing the Computing Surface, its software and support infrastructure, prior to general release in late Summer 1986.

Totally Flexible Topology

◆ To gain the power of the Computing Surface will inevitably involve concurrency, but there is a wide spectrum of approaches which are tractable. Since every single standard computing element is a significant computer in its own right, it can be treated as such. A simple view to take is that of the Computing Surface as a 'processor farm' operated as a multi-task environment with a separate independent task per computing element. A single element will perform many existing applications outright with virtually no modification.

An example is numerically intensive simulation. Often, many simulations have to be performed on the same data set, but with differing starting or operating conditions. This set of simulations is the overall task, but conventionally it is the performance of an individual simulation which is measured and optimised. If each one of the set of simulations is performed simultaneously with the others, with no interactions between tasks, a linear decrease in elapsed time for the overall task will be enjoyed.

The other end of the spectrum is the truly distributed implementation involving a completely fresh look at the problem in question, or indeed tackling a problem which hitherto was looked upon as infeasible with conventional computers.

A Computing Surface is formed by networking many computing elements in an application specific topology. Each computing element is a self-sufficient independent hardware process, with processor, memory and high performance point-to-point communications channels.

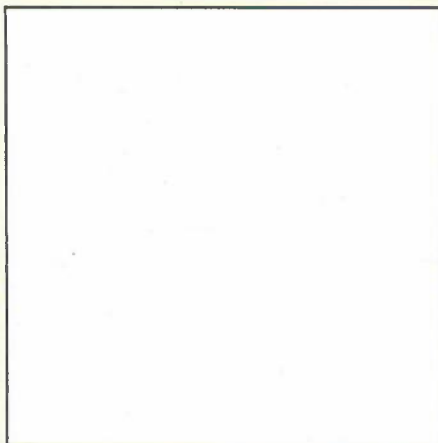
The size, and therefore the power of a Computing Surface, is determined by the quantity and mixture of computing element types.

The Computing Surface has no personality. Users impart their own flavor to the machine. They are not constrained by arbitrary choices made by the developers, however intelligent. Truly optimal configurations can be developed. Most important, they can be fine-tuned as more is learned about an application.

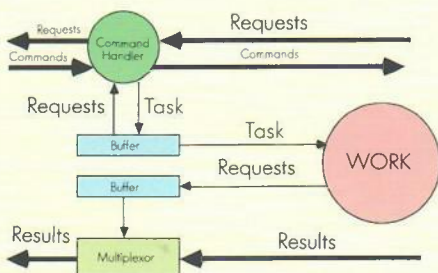
Therefore there is no block diagram for a Computing Surface. Each user develops his own.

In the ray traced example illustrated, the Computing Surface was configured as a distributed rendering engine in which an image is shared between many processors, each one rendering a patch, or number of rasters, and then forwarding its contribution to a display element ready for the next frame. Systems such as this provide ideal examples of how load balancing in concurrent implementations can be achieved; small patches are served on demand to the processors which will require differing amounts of time to render them, depending on the image complexity within the patch.

The Computing Surface is about flexibility. Freedom for implementers to control their own solution space. The support infrastructure which makes this tractable underpins the whole machine. A local host is capable of building a physical map of the entire Computing Surface. Using this map and electronic configuration it wires a machine to a high level specification derived from the application program which is being loaded.



Block diagram

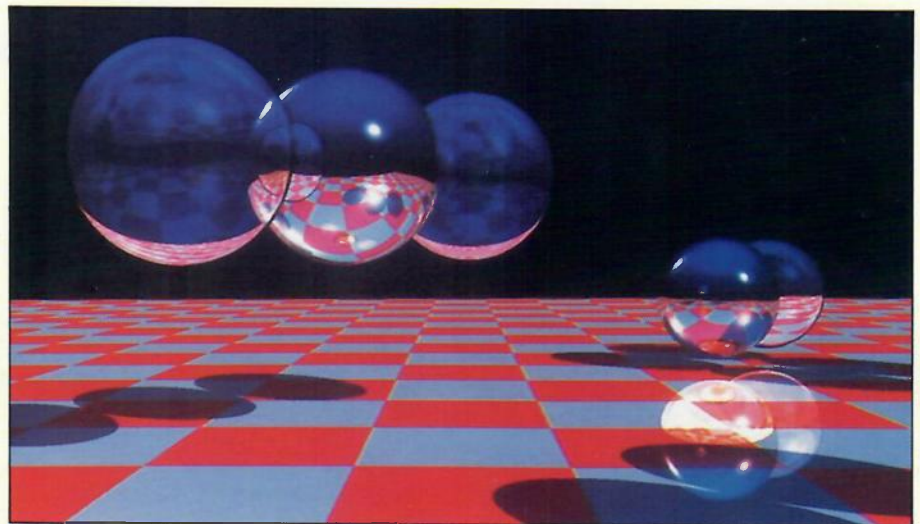


One node of a Load Balancing Task Server
Dark lines are physical communication channels, light lines are logical channels between communicating processes



The support infrastructure can detect hardware or runtime errors in any individual computing element, and perform an immediate analysis – pinpointing a problem by positioning the program source editor at the offending line of code and naming the process instance in which it has occurred. Application diagnostic messages are assured of a guaranteed route to the console or host computer using a communication structure which is independent of, and orthogonal to, the configurable network.

Software is no longer used solely to specify the application program; now it is also used to specify the machine in which the application is executed. The boundaries between hardware and software, processor and memory have been dissolved. Occam specifies connections, communications and computations in one consistent formally based notation. The Transputer is the optimal vehicle to implement the occam model and execute occam code. The Computing Surface is the optimal vehicle to deliver Transputers and support application development.



But the Meiko Computing Surface can also be programmed in many other languages, such as Fortran, C, Pascal etc. Often existing programs can be run substantially without alteration, as the "work" process in the Load Balancing Task Server. Executing such programs merely requires an occam harness to handle communication with other processors (which may be executing copies of the same program), for which tools are provided.

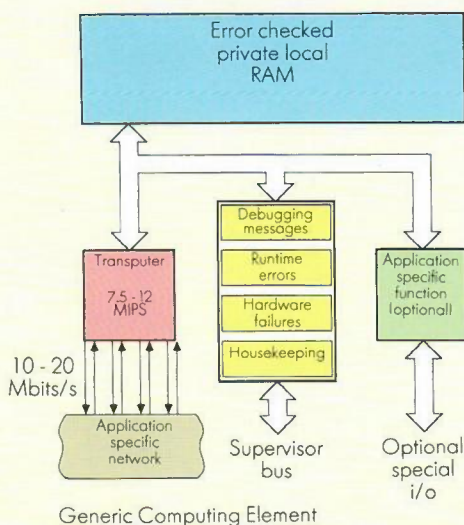
The Development Process is a fully integrated software development toolkit. The user interface is a sophisticated screen editor with direct support for hierarchical program source management by document folding (elision). It provides for automatic recompilation and the ability to load and run distributed multi-processor programs by single function-key commands. Run time errors are detected and reported via the Development Process which opens the program source and positions the cursor at the point where the error has occurred.

Flexible Modules and Subsystems

- Physically a Computing Surface is contained in one or more modules. Connectivity between modules is very unrestricted, allowing the same level of interconnection between computing elements in separate modules as within the same module. These modules are presented in two sizes.

The M40 Computing Surface Module, fully populated, yields an aggregate one thousand one hundred million instructions per second with 42 megabytes of concurrently accessed dynamic RAM. The effective bandwidth to this store is 2.4 gigabytes per second, with a peak bandwidth of 9 gigabytes per second to 300 kilobytes of closely coupled static RAM. This is just one 40" X 20" X 20" aircooled module – an arbitrary number of which may be used together.

A smaller desktop module, the M10, can deliver 250MIPS when fully configured and provides a totally compatible personal supercomputer for workstation or development use.



COMPUTING SURFACE SUBSYSTEMS

- A Computing Surface may be configured as a general purpose computing resource or be optimised to a specific function. A mixture of boards or subsystems is chosen to populate Computing Surface Modules and create a customised solution to a computing problem. A subsystem contains one or more function specific computing elements depending on the physical space occupied by an individual element.

All computing elements conform to the same generic model, giving the important property that they provide a logically consistent interface to the application programmer. The same tools, interfaces and conventions are used with each, for configuration, programming, communication and debugging. This results from their each being based on a Transputer with its own interface to the global Supervisor Bus, and with a quantity of additional private local memory and specialist function unit appropriate to the subsystem.

Every element has eight high speed communication channels which are capable of direct memory access for high speed, low latency message passing to or from other computing elements. Each channel can transfer data at up to 10M bits per second.

The message passing latency is remarkably low for several reasons. The point-to-point communication channels are dedicated, always ready and contention free. Also there is exceedingly little time spent in the scheduling operations commonly associated with message passing since the computing element performs message passing as a single instruction, in which the appropriate process scheduling is achieved in an efficient microcode sequence as opposed to a software kernel scheduling function commonly found in distributed systems.

A · Focused Organization

◆ Meiko has carefully structured an organization to suit the times. The pace of change is accelerating, the world – despite intermittent turbulence - becoming homogeneous. The organisation that succeeds, and prevails, is the one best able to execute any given strategy. Quickly. Effectively.

Consequently Meiko has created a low inertia organisation that spans the globe. Low inertia in order to innovate rapidly, shorten decision and implementation cycles; this requires small autonomous groups fulfilling clearly specified goals. International in order to optimise the opportunities in the three main regions of the industrialised free world from the outset. Just as a tree derives strength from roots spreading in all directions, Meiko Incorporated, Meiko Kabushiki Kaisha and Meiko Limited work in harmony to strengthen each other with their respective skills.

Low inertia is considered critical in the emerging era of massive concurrency. Although Meiko has been intimately involved in this field from the outset, it is only with the advent of machines like the Computing Surface stemming from availability of the Transputer and occam that serious experimentation can occur on any scale. With this new enabling technology placing real and affordable tools within the reach of researchers, rapid strides in algorithm development and hitherto unperceived applications will take place. Meiko considers it essential to be able to incorporate the best of these with unsurpassed speed. Already, in the past twelve months, it has itself gained much experience in load balancing which is already transparently incorporated in the Computing Surface and its associated support tools.

Maintenance is frequently perceived by computing vendors as a source of captive revenue and treated accordingly. Meiko has a different point of view. First, the Computing Surface hardware was designed to be trouble free and reliable. Second, the system has built-in diagnostic and trouble shooting features. Third, Meiko has chosen professional local third-party maintenance organizations to service the Computing Surface. This approach provides an in-place service organization without the start-up problems associated with new and high technology systems.

The short term profits foregone by Meiko through the decision to mould a leaner organization than traditional will be more than outweighed by the ability to innovate faster.

Of course, it helps to be in a sufficiently strong financial position to adopt this policy. Meiko has been consistently profitable since incorporation. As important, it has always generated cash. This is considered crucial in ensuring total freedom to develop what is technically best rather than financially expedient. It is also an independent endorsement of Meiko's skills that it can execute such a strategy, despite being in the traditionally cash consuming development phase.

Like its product, Meiko's organisation is innovative. And, like its product, it delivers.

occam is a trademark of the INMOS Group of Companies.
Copyright 1985 Meiko.

Designed by Proctor + Stevenson Bristol England © 1986
Cover photograph by Charles Summers Jr
(London Museum of Natural History World Wildlife Photographer of the Year 1985)
Aurora USA © 1981
Printed by Oakley Press Bristol England

Meiko Incorporated

6201 Ascot Drive
Oakland, CA 94611
USA

(415) 530 3055
Telex 797748
Fax 415 530 2780

Meiko Incorporated

600 Cummings West Park, Suite 3000
Woburn, MA 01801
USA

(617) 932 0265
Telex 797748

Meiko Limited

Whitefriars
Lewins Mead
Bristol
BS1 2NT
UK

(0272) 277 409
Telex 449731
Fax (0272) 277082



meiko



Configuration Guide

MODULES AND SUBSYSTEMS

Subsystems may be configured within modules with the following power budget

	Module	Capacity	
		+5.0V	-5.2V
Computing Surface Module and Local Host with 9 expansion slots	M10	55A	14A
Computing Surface Module and Local Host with 39 expansion slots	M40	295A	60A
	Subsystem	Load	
Quad Computing Element subsystem	MK009	4.5	0.1
Local Host	MK014	4.5	0.3
Mass Store	MK021	5.0	0.1
Display	MK015	8.0	5.5

EXAMPLE COMPUTING SURFACES

	Configuration module plus subsystems	Total elements	Total MIPS	Total RAM Mbytes	Total Price* \$K
Personal supercomputer	M10 9 MK009	37	275	12	103
High performance graphic workstation	M10 4 MK009 MK015	18	135	8.5	66
Supercomputer increment	M40 39 MK009	157	1175	42	417
Fortran Farm	M40 39 MK021	40	300	315	417
Graphic modeller	M40 30 MK009 8 MK015 MK021	130	975	53	441

* one off end user prices at January 1, 1986

Hotlines:

USA (415) 530 3055,
Japan (03) 238 9807,
UK (0272) 277409

The information supplied in this document is believed to be true but no liability is assumed for its use or for the infringements of the rights of others resulting from its use. No licence or other rights are granted in respect of any rights owned by any of the organisations mentioned herein.

Copyright 1985 Meiko.



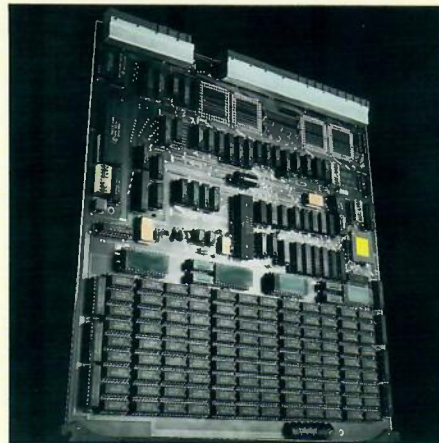
Local Host (IEEE 488)

AVAILABLE

KEY POINTS

Fourth quarter 1985

- ◆ 15MHz 32 bit 7.5 mips processor
- ◆ 3Mbytes 15Mbytes/s error checked RAM and 128Kbytes 10Mbytes/s EPROM
- ◆ eight unidirectional 10Mbits/s autonomous message channels
- ◆ module hardware controller and error handler
- ◆ IEEE 488 and dual RS232 I/O interfaces



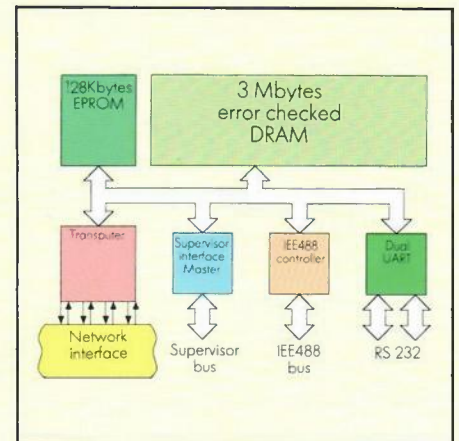
Local Host (IEEE 488 option)

The local host is a high performance self contained general purpose computer, occupying a single slot in the Computing Surface Module. It is programmed and viewed in a manner totally consistent with all other computing elements.

At least one local host is required to perform the house keeping tasks of the module and further local host boards may be added as required.

House keeping tasks include:

- ◆ Global monitor for hardware failures such as a parity error in any of the local memory systems.
- ◆ Master of the (optional) electronic routing network.



- ◆ Control of the hardware reset and post-mortem analysis of all the other computing elements in a module.
- ◆ Hosting the interactive program development environment:
 - accessing files via the IEEE 488 interface
 - supporting compilers
 - topology configuration software, and
 - the Computing Surface loader/ bootstrapper.
- ◆ Runtime error handling linked to source level debugging tools, and global communications handler.

Hotlines:

USA (415) 530 3055,
Japan (03) 238 9807,
UK (0272) 277409

The information supplied in this document is believed to be true but no liability is assumed for its use or for the infringements of the rights of others resulting from its use. No licence or other rights are granted in respect of any rights owned by any of the organisations mentioned herein.

Copyright 1985 Meiko.

meiko

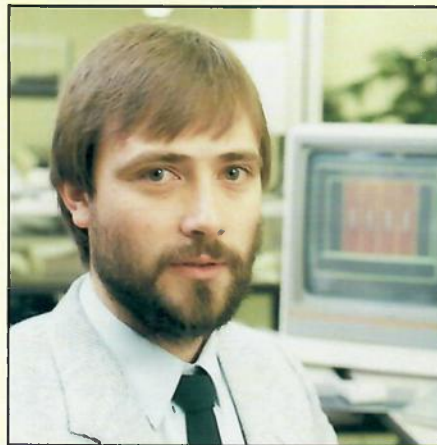
Mass • Store

AVAILABLE

Second quarter 1985

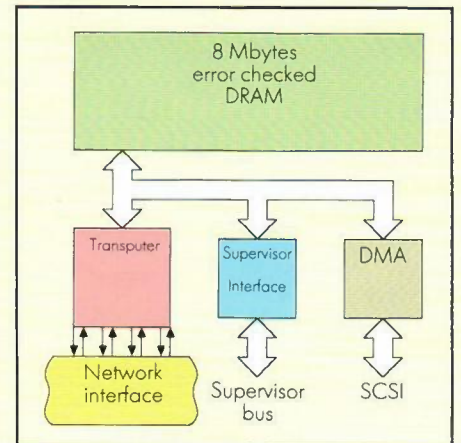
KEY • POINTS

- ◆ 15MHz 32 bit 7.5 mips processor
- ◆ 8Mbytes 15Mbytes/s error checked RAM
- ◆ eight unidirectional 10Mbits/s autonomous message channels
- ◆ 2Mbytes/s DMA controlled SCSI disk and peripheral interface
- ◆ global communications and error handling



layout in progress at time of publication

The mass store element provides three levels of memory hierarchy: 2048 bytes of single cycle static RAM for frequently accessed local variables, 8 megabytes of four cycle, error checked, dynamic RAM and a memory mapped 2 megabytes/s DMA controlled SCSI interface. 100 Mbyte, 500 Mbyte winchester discs and 1-2 Gbyte laser discs are available for direct attachment to this interface.



Multiple mass store elements may be used to form an intelligent database with local search, match and cache capabilities.

Alternatively this subsystem can be viewed simply as an enhanced memory computing element for use in applications involving large or unpredictable data structures.

Hotlines:
USA (415) 530 3055,
Japan (03) 238 9807,
UK (0272) 277409

The information supplied in this document is believed to be true but no liability is assumed for its use or for the infringements of the rights of others resulting from its use. No licence or other rights are granted in respect of any rights owned by any of the organisations mentioned herein.

Copyright 1985 Meiko.



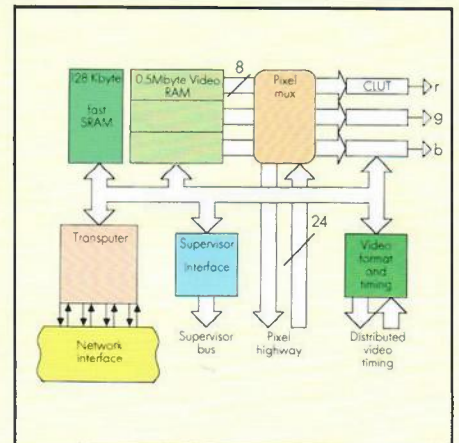
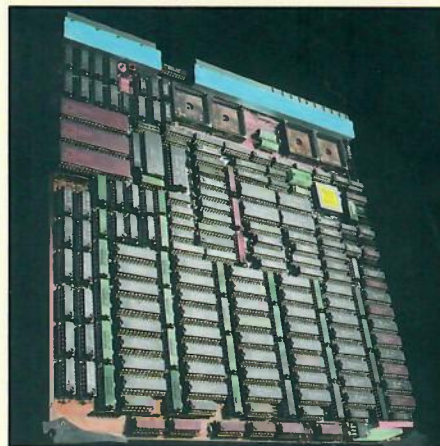
Display

AVAILABLE

KEY • POINTS

First quarter 1986

- ◆ 15MHz 32 bit 7.5 mips processor
- ◆ 128KBytes private SRAM; 1.5Mbytes dual ported display store
- ◆ eight unidirectional 10Mbits/s autonomous message channels
- ◆ 70MHz pixel rates; 200Mbytes/s external pixel highway
- ◆ CCIR/RS-343 compatible video with programmable sync generator supports interlace and non-interlace



The display memory appears in the address space of a powerful 32 bit processor. The display format is very flexible and can, for example, be treated as a single frame of 0.5M 24bit pixels, or 1.5M 8bit pixels. Three 256 entry colour lookup tables drive the monolithic 8bit per gun DAC's, giving easy colour correction of

rendered images. Alternatively, for animated applications, the display may be used as two 0.75M 8bit per pixel frame buffers.

Multiple Display Elements may be ganged to provide a larger format frame store, or to increase drawing bandwidth.

Hotlines:
USA (415) 530 3055,
Japan (03) 238 9807,
UK (0272) 277409

The information supplied in this document is believed to be true but no liability is assumed for its use or for the infringements of the rights of others resulting from its use. No licence or other rights are granted in respect of any rights owned by any of the organisations mentioned herein.

Copyright 1985 Meiko.



Computing Element

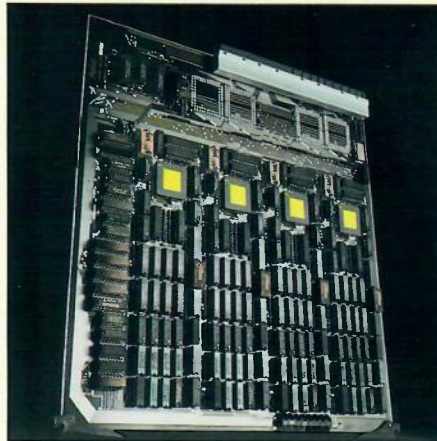
AVAILABLE

KEY POINTS

Fourth quarter 1985



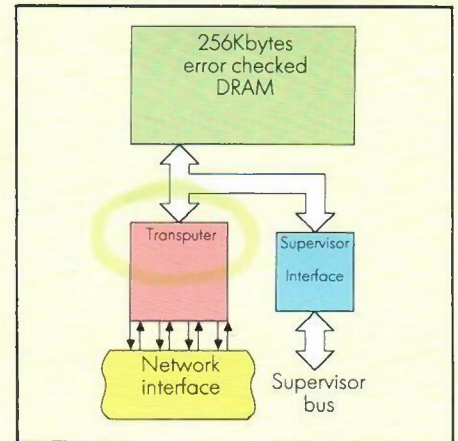
- 15MHz 32 bit 7.5 mips processor ~ 0.5MWIPS (whetstones)
- ◆ 256Kbytes 15Mbytes/s error checked RAM
- ◆ eight unidirectional 10Mbits/s autonomous message channels
- global communications and error handling
- four computing elements per Subsystem



Quad Computing Element Board

This is the standard computing element; supplied four per board. Its VLSI 32 bit processor was designed specifically for use in multiprocessor point-to-point connected applications, and optimised for the execution of compiled high level languages.

High performance message transferring and process scheduling capabilities are primitive functions of the processor instruction set – there are no software kernel overheads.



Occam II is the language which was developed in parallel with this processor, designed to allow the programmer to express concurrency as easily as sequentiality, and to pass messages as easily as performing an assignment. It provides the cement that holds co-operating processes together. These processes may themselves be written in occam, or a variety of conventional languages such as C, FORTRAN or Pascal.

Hotlines:
USA (415) 530 3055,
Japan (03) 238 9807,
UK (0272) 277409

The information supplied in this document is believed to be true but no liability is assumed for its use or for the infringements of the rights of others resulting from its use. No licence or other rights are granted in respect of any rights owned by any of the organisations mentioned herein.

occam is a trademark of the INMOS Group of Companies.

Copyright 1985 Meiko.



End·User Price·Summary



M10
local host + 9 additional subsystem slots
integral power and forced air module
5V @ 60A, -5.2V @ 14A output
110/ 220Vac 50/60Hz 750W input
19" rack mountable or free standing
10"h x 17.5"w x 19"d

\$13,000 Q1/86



M40
local host + 39 additional subsystem slots
integral power and forced air module
5V @ 300A, -5.2V @ 60A output
220Vac 50/60Hz 2750W input
19" rack mountable or free standing
36.5"h x 17.5"w x 19"d

\$27,000 Q1/86



Electronic Configuration

to upgrade from miniature twisted pair jumper configuration to software controlled electronic topology (can be retrofitted)

add 15% Q3/86
to hardware
total



Development process

DEC VAX/VMS fileserved or
IBM PC/AT MSDOS fileserved
Hierarchical folding source code editor with
integrated occam II compiler,
distributed system loader and
source level runtime error locator

\$5,000 Q1/86



Sequential language compilers

C
FORTRAN 66
FORTRAN 77
PASCAL
BCPL

per compiler \$2,000
Q2/86
Q1/86
Q3/86
Q2/86
Q1/86

Prices valid at January 1, 1986



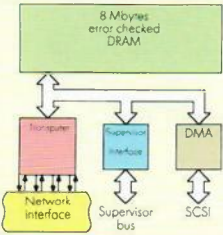

Hotlines:
USA (415) 530 3055
Japan (03) 238 9807
UK (0272) 277409

The information supplied in this document is believed to be true but no liability is assumed for its use or for the infringements of the rights of others resulting from its use. No licence or other rights are granted in respect of any rights owned by any of the organisations mentioned herein.

occam is a trademark of the INMOS Group of Companies. DEC, VAX and VMS are trademarks of Digital Equipment Corporation. IBM PC/AT is a trademark of International Business Machines Corporation. MS-DOS is a trademark of Microsoft Incorporated.

Copyright 1985 Meiko.

End·User Price·Summary

		Price per subsystem	Planned availability
	Computing 15MHz 32 bit 7.5 mips processor 265K bytes 15Mbytes/s error checked dynamic memory Four elements per subsystem	\$10,000	Q4/85
	Local host IEEE 488, local host SCSI 15MHz 32 bit 7.5 mips processor 128Kbytes 15Mbytes/s EPROM 3Mbytes 15Mbytes/s error checked dynamic memory module hardware error handler 2 RS232 serial interfaces IEEE 488 or SCSI interface bus options	\$6,000 \$6,000	Q4/85 Q2/86
	Mass store 15MHz 32 bit 7.5 mips processor 8Mbytes 15Mbyte/s error checked dynamic memory 2Mbyte/s DMA controlled SCSI disk and peripheral interface	\$10,000	Q2/86
	Display 15MHz 32 bit 7.5 MIPS processor 1.5Mbytes dual ported display store 70MHz pixel rate, 200Mbyte/s external pixel bus programmable display format triple 8 bit lookup tables driving 8 bit DACs CCIR/RS-343 compatible video programmable sync generator	\$13,000	Q1/86

Hotlines:
 USA (415) 530 3055,
 Japan (03) 238 9807,
 UK (0272) 277409

The information supplied in this document is believed to be true but no liability is assumed for its use or for the infringements of the rights of others resulting from its use. No licence or other rights are granted in respect of any rights owned by any of the organisations mentioned herein.

occam is a trademark of the INMOS Group of Companies. DEC, VAX and VMS are trademarks of Digital Equipment Corporation. IBM PC/AT is a trademark of International Business Machines Corporation. MS-DOS is a trademark of Microsoft Incorporated.

Copyright 1985 Meiko.

