AccessionIndex: TCD-SCSS-T.20230524.003
Accession Date: 24-May-2023
Accession By: Dr.Brian Coghlan
Object name: Orla-I fault-tolerant multiprocessor
Vintage: c.1989
Synopsis: VME-based machine designed for research into checkpointing using Transparent Stable Memory (TSM).

Description:
This item is a VME-based machine called *Orla-I* [1][2][3] designed for research into checkpointing using *Transparent Stable Memory* (TSM). It was explicitly a research machine, designed and built in the late 1980s within the *Advanced Multiprocessor Architectures project* funded by EOLAS ST/89/304 [4]. The principles of fault tolerance using Stable Memory are covered in detail in [5].

A quite complete description of the principles of the design is given in United States Patent No. 5,394,536, see [6]. To summarise:

The term *stable memory* is intended to cover memory, the contents of which is not corrupted under fault conditions such as power failure or faulty processor operation and with which atomic stable memory operations may be performed. A copy and verification sequence is called an atomic transaction if it is designed only to succeed or fail − no other result is allowed, and failure does not result in corruption of data. Without hardware support these operations take time. In U.S. Pat. No. 4,799,186 (Ployette) stable memory is provided by duplicated on-board memory circuits; this results in less time being required by the processor to perform the atomic transactions.

In *Orla-I*, as in U.S. Pat. No. 5,394,536 (Coghlan & Jones), an objective is to provide a stable memory circuit which allows stable memory operations to be carried out concurrently with host processor operations and with very little use of host processor time so that the stable memory operations are effectively *transparent* to the host processor. It is another objective to provide a stable memory circuit that can be part of a main memory of a processor and can behave as conventional memory of increased capacity when stable memory operation is not required and can be dynamically partitioned between conventional and stable memory on a block-by-block basis. Another objective is to allow intelligent data processing operations to be carried out on data involved in a stable memory operation with little overhead requirement of the host processor. A further objective is to allow relatively fast execution of stable memory operations. A still further objective is to provide an inexpensive stable memory circuit.

This *Transparent Stable Memory* (TSM), see Fig.1, comprises a pair of memory banks A and B. Each memory bank comprises arrays each having dual-ported video random access memories (VRAMs). Each VRAM includes a parallel port and a separate serial port. In principle the parallel ports are connected to a processor bus, for connection with a host processor. The serial ports (stable memory ports) are connected to a stable memory bus. There is a stable memory bus for each array of VRAMs. Each stable memory bus connects VRAMs in each bank directly to associated VRAMs in the other bank. Again in principle, there may be any number and possibly only one stable memory bus common to all arrays of VRAMs.

In operation, host processor accesses are carried out via the host processor bus and whenever desired, say, at regular intervals, stable memory control signals are transmitted on the processor bus interleaved with the normal random accesses and these initiate stable memory operations on the stable memory busses. Subsequently, the stable memory operations are carried out without the need for transmission of any information on the processor bus. These operations may include copying of contents of the bank A to the bank B. In one example, a page in the memory bank A is coupled with a page in the bank B to form a single stable page. Host processor accesses are then allowed only into the page in the bank A and the corresponding page in the bank B is inaccessible to the host processor. During stable memory operations, flags may be used to correctly indicate the validity of the copy and to protect against other accesses to these

pages. Because stable memory ports of each VRAM are connected directly to equivalent or associated VRAMs in the other memory bank, there is a very wide serial data path yielding a very high transfer rate.

It will be appreciated that because the circuit is symmetrical either of the memory banks A and B may be used as the accessible bank or both may be used as a conventional memory depending on the instructions which are transmitted on the processor bus from a host processor. Indeed, the memory banks may be dynamically partitioned into stable memory and conventional memory portions depending on instructions received. This is an important aspect, particularly where from time to time stable memory operations are not as important as the storage of as much information as possible. In other cases where such alternatives are not required, it is envisaged that the processor bus may be connected to only one memory bank, the other bank permanently forming the inaccessible memory bank of the stable memory.

There is dedicated stable memory control via a master controller and a slave controller. In principle a host processor can connect via the processor bus to the master controller and the slave controller, while the latter is in turn connected to the memory banks A and B. In operation, the master controller carries out high-level functions and fault-handling. Invocation of stable memory operations is carried out by queuing calls in a memory bank. The master controller passes lower-level duties to the slave controller, which manages each of the two memory banks A and B, and the stable data engine. The master and slave controllers are connected via communication links to other computers, which allow them to be remotely booted, and the booting instructions may be stored in memory bank A or B, The links also allow the master controller to carry out fault recovery operations in isolation from the host processor. This is important in the event that the host processor is faulty.

The stable memory operations carried out by the stable data engine may include copying, comparing, and searches including maximum/minimum, equal/notequal, or inside-limits/outside-limits type searches. For the stable data engine to perform a function, the master controller must co-ordinate the following sequence:

1. *tell the slave controller to transfer from address_A of the memory bank A.*
2. *tell the slave controller to transfer to address_B of the memory bank B.*
3. *tell the stable data engine to process the data.*
4. *wait for completion.*
5. *read results from the stable data engine.*

Serial transfers may be carried out in random fashion starting from any desired bit in the VRAM's serial shift register and for any desired number of bits. Thus, there is complete flexibility in the manner in which stable memory operations are carried out.

To give an example of the speed with which data is transferred between the memory banks A and B, if each bank is constructed from sixty-four 256 k x 4 VRAMs, there can be a 256 bit wide x 40Mbit/sec serial data path between the VRAMs and the stable data engine, which is capable of transferring 1280Mbytes/sec. In *Orla-I* the stable memory port may be bussed between stable memory boards, so the serial data paths are reduced to 32 bits wide to reduce the influence of connector pin failures, resulting in a 32 bit x 40Mbit/sec inter-board data path capable of transferring 160Mbytes/sec *transparently*.

There are a number of ways of performing stable memory operations. For example, with atomic stable memory copying, the following sequence of operations may be performed by the stable data engine 26:

1. *Copy N bytes from bank A address_A to bank B address_B.*
2. *Compare N bytes from bank A address_A to N bytes from bank B address_B.*

This is perhaps the simplest sequence. More comprehensive atomic stable memory operations typically involve more copying. In all cases flags must be released and set in the correct order to nominate the currently valid copy. It will be appreciated that operation of the stable data engine is concurrent with operation of the host processor, and it provides for a much greater data handling capacity for a computer system. It will also be appreciated that the stable data engine may fall within a wide range of complexity, depending on the stable memory operations to be supported. Further, the stable data engine need not intercept the stable memory ports, but may instead simply monitor the stable memory ports, which will then be directly connected between the memory banks and will typically allow faster data transfer.

The memory manager includes protection logic and a pair of port logic circuits (PLCAs) connected to the processor bus. The PCLAs also carry out error detection and control of ports 0 and 1, and in general additional circuits in the stable memory cater for hardware errors and for continuing satisfactory operation in the event of failure of a single component. The memory includes a pair of error detection and correction circuits (EDC), and a pair of port controllers (PCTLs). The EDC circuits perform both error detection and consequent correction of data. The PCTLs handle error logging of the address, syndrome, parity, and the double-bit error flag as well as VRAM control.

The architecture shown in Fig.1 shows that the stable memory was comprised of a multiplicity of two types of PCBs, a memory manager board (Fig.2) and a memory board (Fig.3). Both were very dense multilayer PCBs, designed using hierarchical netlist text files for the *CupidPCB* software [7] written by Dr.Brian Coghlan, and then implemented using the *Multiwire* technology by a Western Australian manufacturer. Figs.4 and 5 illustrate debugging of these boards using VME extender cards.

In the *Orla-I* memory manager the master and slave controllers were both InMOS T800 transputers. The distinction between master and slave was more apparent than real, since after initialization the two could share the same workload as evenly as possible. Each of the two transputers and the VME bus interface connected via two NEC uPD43608 8kB write-through snoopy caches to two multi-master memory busses, i.e. six snoopy caches in total, see Fig.1. The VME bus was managed by a VIC068 controller from VTC Inc. The memory busses were electrically similar to Futurebus-87, driven by DS8397 Futurebus transceivers. Each transputer had four external 20Mbaud links that could be interconnected to a network of other memory managers. Each also had its own NCR 53C90 SCSI-2 interface.

Each memory manager included five Xilinx XC3090 FPGAs, each equivalent to about 9,000 gates. One FPGA contained protection logic for the VME bus. The logic for each transputer was contained in its own FPGA, and similarly the logic for each memory bus was contained in its own FPGA (PLCA, Fig,1). The FPGAs were booted from non-volatile memory with logic configurations designed by Dr.Brian Coghlan and Thomas Labod using the XACT graphical design tools, thanks to generous support from Xilinx.

The *Orla-I* memory board had two 4MB VRAM memory planes accessible over the two memory busses (*port 0* and *port 1*, the outer connectors). Each memory plane was controlled by a NatSemi DP8429 DRAM controller, and fully protected by ECC via an EDC (Fig.1). The serial ports of the VRAM banks were connected via a stable data engine that was intended to manage both checkpointing and associative operations between the two banks and/or a stable data bus (*S port*, the middle connector). The stable data bus was a streaming bus that interconnected all the memory boards, again electrically similar to Futurebus-87. Each memory board included four Xilinx XC3090 FPGAs. The logic for each memory bus was contained in its own FPGA (PCTL, Fig.1), while the logic for the stable data engine was contained in two FPGAs.

The multiprocessor system, including commercial processor and networking boards, and four independent VME busses, and was built into a custom-built chassis with dual-redundant switchmode power supplies with battery backup for graceful shutdown in the event of mains AC power failure, see Fig.6.

By the end of the project the TSM hardware was part functional. The physical hardware design was completed; four memory manager and five memory boards had been constructed. A large part of the FPGA designs was completed. Some idea of the complexity of the design can be garnered from the fact that each FPGA design was represented by over 20 pages of circuit diagrams. The VME protection FPGA was intended to support capabilities, however the XC3090 did not provide enough logic, so custom headers were designed and made to allow higher-density XC4000-series FPGAs (about 14,000 gates and 2.5 times faster) to be plugged into the XC3090 sockets, but purchase of these FPGAs was never funded. The physical infrastructure, rack, ventilation, disk chassis and card cages and power supply were in place. Apart from ageing commercial VME processor modules used for testing, nothing had been done regarding host processors to support multiprocessing.

Hyen Vui Chung had written very extensive software and simulations to exploit the intended protection and capability logic [10]. Ciarán Bryce had written a prototype Occam *Nucleus* for the transputers [11]; its basic structure was complete, including a substantial body of self-test routines, the SCSI driver, and a large part of the memory management and error handling software, but not tested, although its (perhaps overly fault-tolerant) communications module was complete and tested. The *Nucleus* was unique in being written in Occam, which is intended for loosely-coupled ensemble architectures rather than tightly-coupled shared-memory architectures like *Orla-I*.

The research led to a number of MSc theses [8][9][10], a Final Year project [11], published papers [12][13][14][15], internal reports [16][17][18][19], a campus company (Tolsys Ltd) spun out from the Department of Computer Science in Trinity College Dublin, a *Stable Disk Controller* [20] designed and implemented by Tolsys under the aegis of Esprit Project P5212 (FASST), and conception of a more efficient technique for fault-tolerance, *Switchmode Checkpointing*, that led to another United States Patent, No. 5,574,874 (Jones, Coghlan, O'Carroll) [21] that at the time of writing had been cited 36 times in other patents.

The homepage for this catalog is at: https://www.scss.tcd.ie/SCSSTreasuresCatalog/
Click *'Accession Index'* (1st column listed) for related folder, or *'About'* for further guidance.
Some of the items below may be more properly part of other categories of this catalog, but are listed here for convenience.

Accession Index	Object with Identification
TCD-SCSS-T.20230524.003	Orla-I fault-tolerant multiprocessor. VME-based machine designed for research into checkpointing using Transparent Stable Memory (TSM). c.1989.
TCD-SCSS-T.20121208.096	Tolsys Stable Disk Controller, First Irish-designed fault-tolerant disk controller comprising a TSMR2006 RAID Controller coupled to a TSMV2004 Stable Memory, c.1992.
TCD-SCSS-T.20150217.005	Tolsys DT200.1 Deep Trace Memory Board.
TCD-SCSS-T.20150217.003	Prototype RAID, First RAID chassis built in the Dept.Computer Science, TCD, with five SCSI-1 disks and PC/AT power supply. PSU S/N: 7X705993, c.1990.
TCD-SCSS-T.20121208.094	Experimental SCSI Cluster, 4-node prototype cluster using SCSI as interconnect, 1997.
TCD-SCSS-T.20121208.095	csTCDie Beowulf Cluster, Departmental cluster using 100Mbps Ethernet as interconnect, c.199x.
TCD-SCSS-T.20141120.003	csTCDie Grid-Ireland SCI Cluster, 16-node cluster using 400MB/s SCI switched interconnect, c.1999.

References:

1. Trinity College Dublin, *Orla I fault-tolerant multiprocessor*, see: https://www.scss.tcd.ie/SCSSTreasuresCatalog/hardware/TCD-SCSS-T.20230524.003/TCD-SCSS-T.20230524.003.pdf
   Also see: https://www.scss.tcd.ie/brian.coghlan/orla1.htm

2. Coghlan, B.A., Jones, J.O., *Orla 0 Target Specification*, Internal Report, Department of Computer Science, Trinity College Dublin, 1990.

3. Coghlan, B. A., Jones, J.O., *Orla I: An Advanced, Fault-Tolerant Multiprocessor*, Internal Report TCD-CS-91-09, Department of Computer Science, Trinity College Dublin, 1991.

4. Coghlan, B. A., Jones, J.O., Chung, H.V., Labod, T., and Bryce, C.J., *Advanced Multiprocessor Architectures*, Final Report for EOLAS Strategic Research Grant ST/304/89, Department of Computer Science, Trinity College Dublin, 1992.

5. Coghlan, B., Fabregat, G. [Eds], *Fault Tolerance using Stable Memory*, accepted for publication by Springer-Verlag before they were taken over by Bertelsmann, but discontinued afterwards. It is preserved as Internal Report TCD-CS-1999-75, pp.1-243, Department of Computer Science, Trinity College Dublin, see: https://www.scss.tcd.ie/SCSSTreasuresCatalog/hardware/TCD-SCSS-T.20230524.003/TCD-CS-1999-75.pdf

6. Coghlan, B.A. and Jones, J.O., *Stable memory circuit using dual ported VRAM with shift registers in a multiple memory bank setup for high speed data-transfer*, United States Patent 5,394,536, priority date 11th September, 1989, granted 28th February, 1995, see: https://www.scss.tcd.ie/SCSSTreasuresCatalog/hardware/TCD-SCSS-T.20230524.003/USpatent-5394536.pdf Last browsed to on 24-May-2023.

7. Trinity College Dublin, *CupidPCB*, Department of Computer Science, see: https://www.scss.tcd.ie/brian.coghlan/cupidp.htm

8. Dierdre O'Donovan, *The Design and Implementation of Transputer Code for a Transparent Stable Memory*, MSc Thesis, Department of Computer Science, Trinity College Dublin, 1988-90.

9. Kevin Carey, *An associative engine for transparent stable memory*, MSc Thesis, Department of Computer Science, Trinity College Dublin, 1988-90.

10. Hyen Vui Chung, *Memory protection using capabilities*, MSc Thesis, Department of Computer Science, Trinity College Dublin, 1990-94.

11. Bryce, C.J., *A Reliable Communications Module and the Orla Nucleus*, Final Year Project, Department of Computer Science, Trinity College Dublin, 1991.

12. Coghlan, B.A., Jones, J.O., *Stable memory for a disk write cache*, Microprocessing & Microprogramming, p53-80, Vol.41, 1995.

13. Jones, J.O., Coghlan, B.A., Jones, J.O., *Stable Disk : A Fault-Tolerant Cached RAID Subsystem*, In: *Hardware and Software Architectures for Fault Tolerance*, pp78–90, Vol.774, LNCS, Springer-Verlag, 1994.

14. Coghlan, B.A., Jones, J.O., *Stable memory using transputers*, Transputer Conference, Glasgow, August 1992.

15. Horn, C., Coghlan, B., Harris, N., Jones, J.O., *Stable memory--another look*, In: Karshmer, A., Nehmer, J. (Eds.), *Proc.Int.Workshop in Operating Systems of the 90s and Beyond*, pp.171-177, Vol.563, LNCS, Springer-Verlag, July 1991.

16. Coghlan, B.A., Jones, J.O., *Transparent Stable Memory*, Internal Report TCD CS-91-16, Department of Computer Science, Trinity College Dublin, 1991.

17. Coghlan, B.A., Jones, J.O., *The Case for Transparent Stable Memory*, Internal Report TCD-CS-91-21, Department of Computer Science, Trinity College Dublin, 1991.

18. Coghlan, B.A., Jones, J.O., *Stable Memory: Another Look*, Internal Report TCD-CS-91-19, Department of Computer Science, Trinity College Dublin, 1991.

19. Coghlan, B.A., Jones, J.O., *Transparent Stable Memory Using Transputers*, Internal Report TCD-CS-91-20, Department of Computer Science, Trinity College Dublin, 1991.

20. Trinity College Dublin, *Stable Disk Controller*, see:
https://treasures.scss.tcd.ie/hardware/TCD-SCSS-T.20121208.096/TCD-SCSS-T.20121208.096.pdf
Also see: https://www.scss.tcd.ie/brian.coghlan/tv2004.htm

21. Coghlan, B.A. and Jones, J.O., *Memory Checkpointing*, United States Patent 5,574,874, priority date 3rd November, 1992, granted 12th November,1996, see:
https://www.scss.tcd.ie/SCSSTreasuresCatalog/hardware/TCD-SCSS-T.20230524.003/USpatent-5574874.pdf
Last browsed to on 24-May-2023.

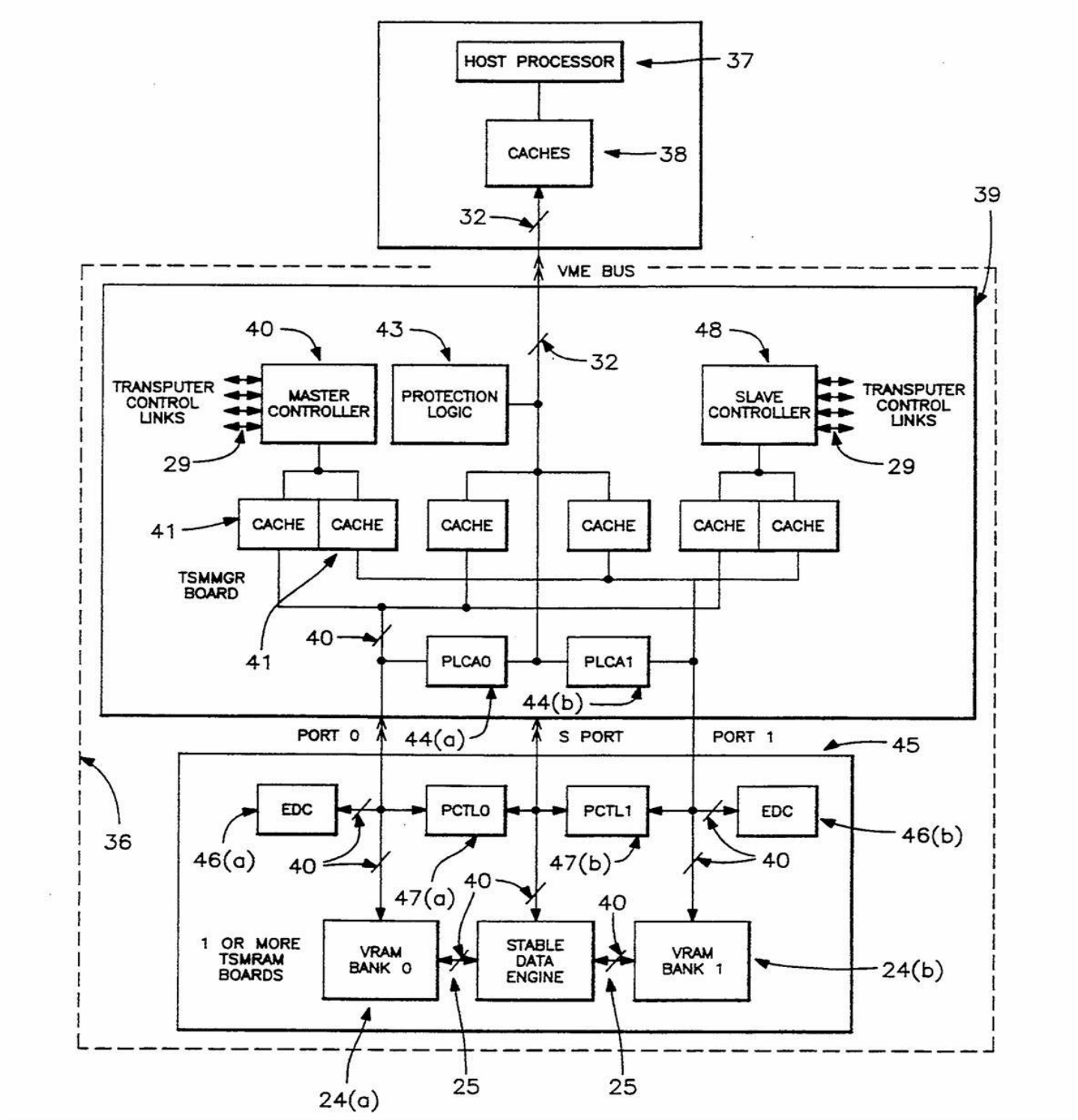

Figure 1: Orla-I architecture.
From: United States Patent 5,394,536, Fig.5.

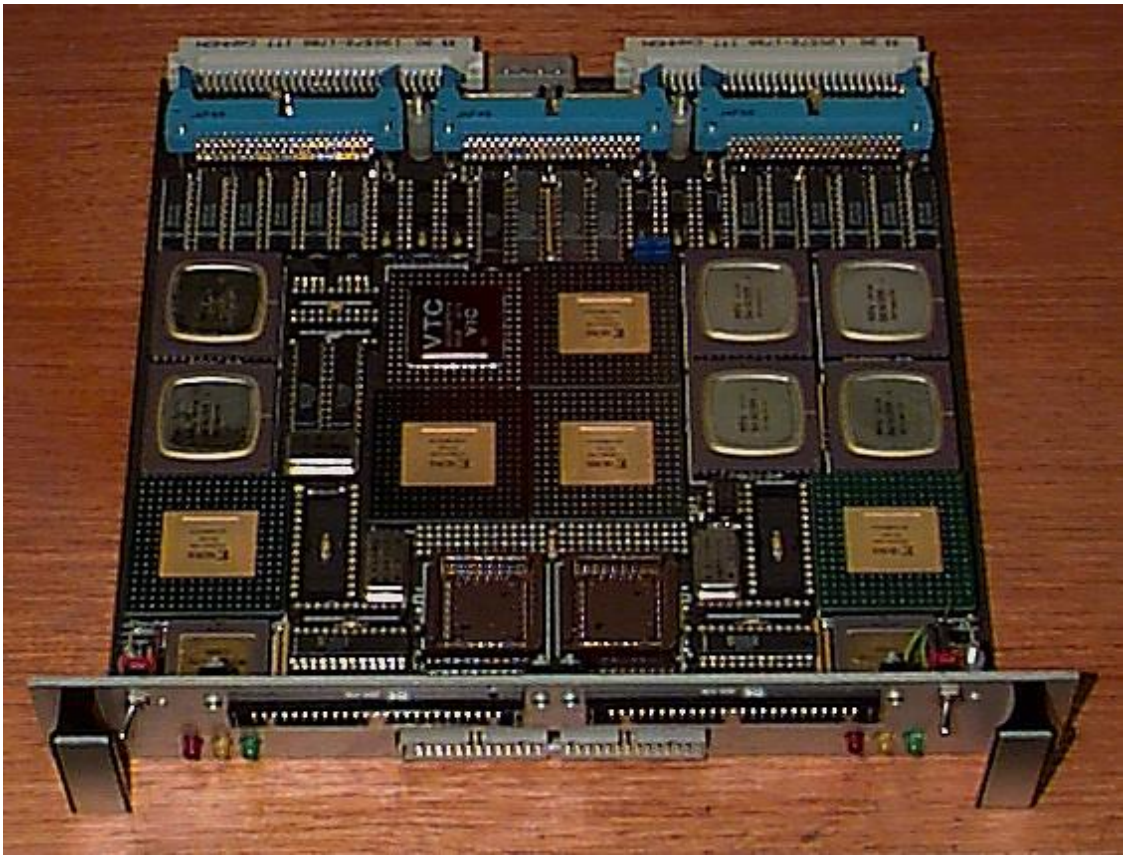

Figure 2: Orla-I Transparent Stable Memory (TSM), memory manager PCB.

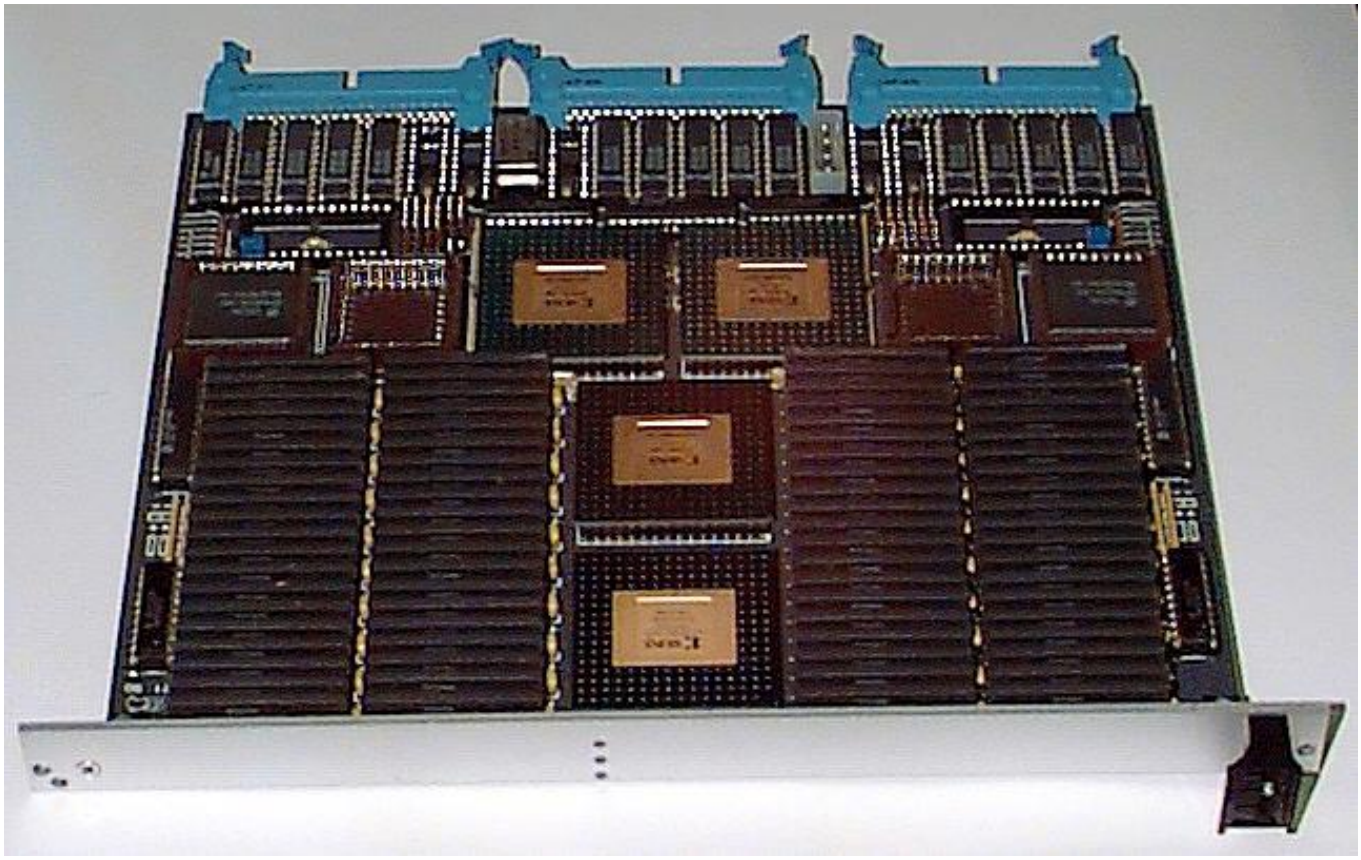

Figure 3: Orla-I Transparent Stable Memory (TSM), memory PCB.

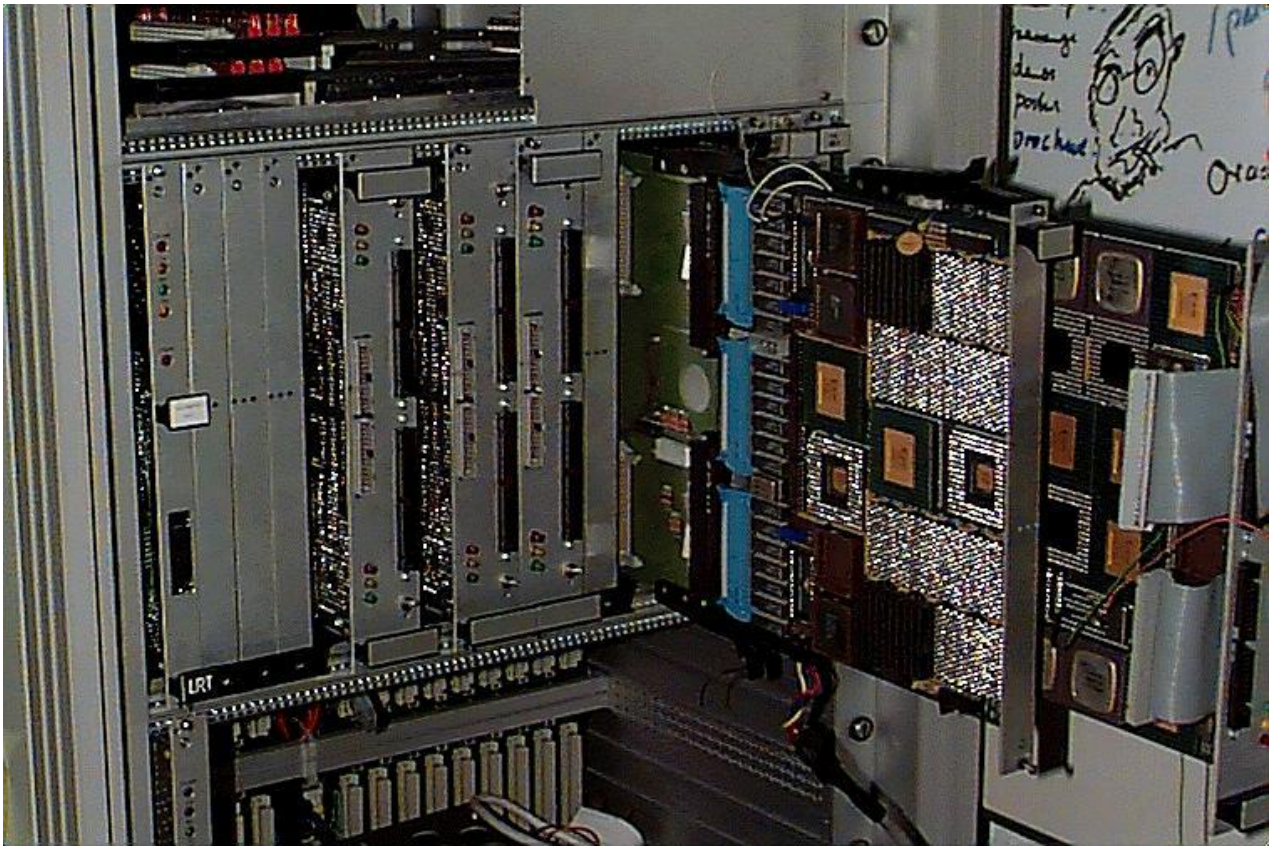

Figure 4: Orla-I Transparent Stable Memory (TSM) debugging.

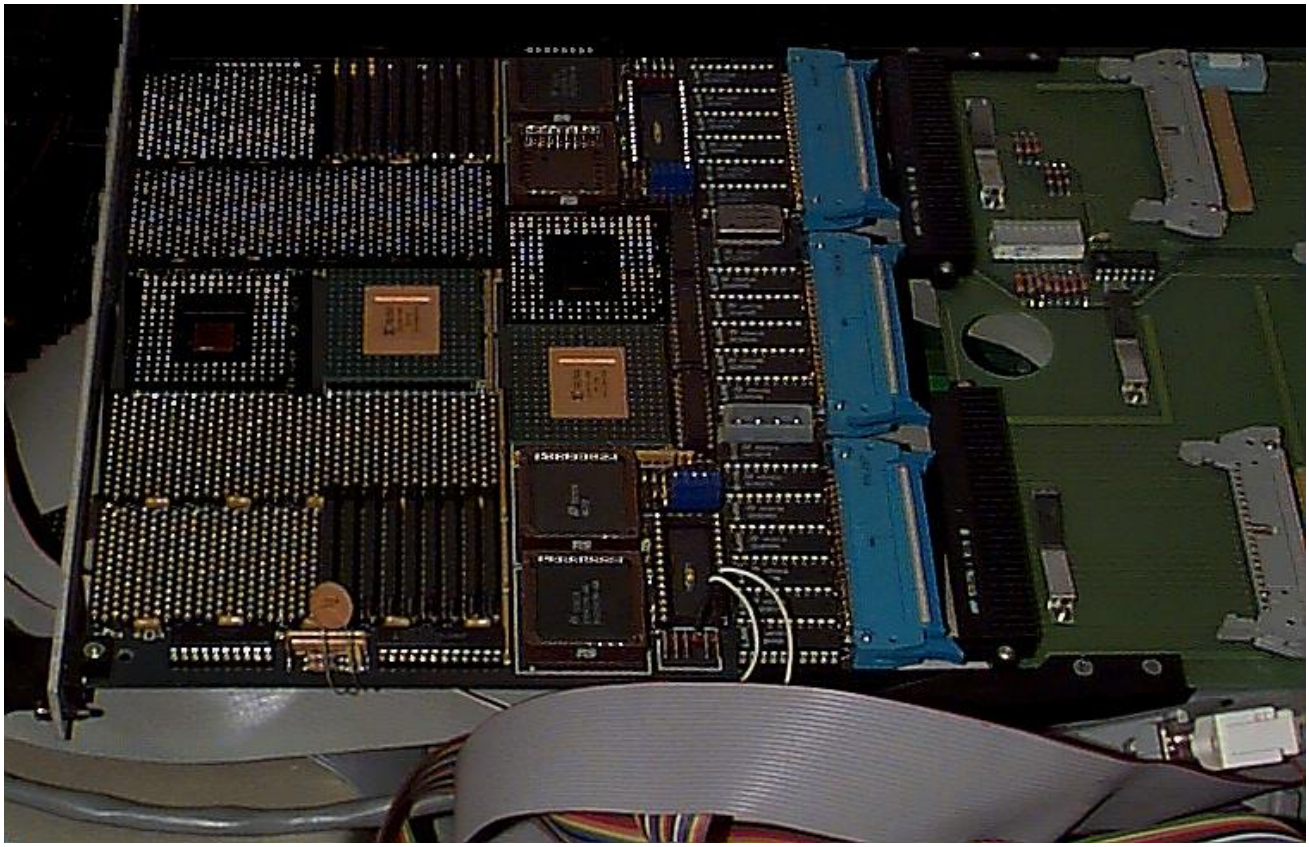

Figure 5: Orla-I Transparent Stable Memory (TSM), memory PCB debugging.

Figure 6: Orla-I dual-redundant power supply.