Review Article

GJAI (2019) 1:5

Global Journal of Artificial Intelligence
(DOI:10.28933/GJAI)

# Transputers Epoch

**Nzenwata Uchenna J*., Adegbenle Adedeji A., and Adedokun Adewale J**

Department of Computer Science, Babcock University, Nigeria

**ABSTRACT**

The transputer (Transistor Computer) was an innovative computer design of the 1980s from INMOS, a British semiconductor company based in Bristol. The transputer was conceived of as a building block for electronic systems comprising a processor, memory and a communication system. The transputer was unique in that each processor had a built-in simple operating system, memory and four high speed (20 Mbit/s full duplex) bi-directional serial links. The transputer is essentially a computer system on a chip. The links on the transputer allow connection to up to four other transputers or peripherals such as video graphics, floppy and hard disc drives, Ethernet networking and standard RS-232 serial ports. In this paper discusses the original purpose of the transputer, the architectural and the network design. It also lay emphasis on the factors that birth the dead end of the tranputer technology and the restoration project.

**Keyword:** Transputer, Ocam, Inmos, Floating Point Unit, Memory Management Unit.

***Correspondence to Author:**
Nzenwata Uchenna J
Department of Computer Science, Babcock University, Nigeria

**How to cite this article:**
Nzenwata Uchenna J., Adegbenle Adedeji A., and Adedokun Adewale J. Transputers Epoch. Global Journal of Artificial Intelligence, 2019; 1:5.

## INTRODUCTION

The word transputer is a derivative name from the Transistor Computer. The transputer was conceived of as a building block for electronic systems comprising a processor, memory and a communication system. Large systems were to be constructed from collections of transputers, each running a program and communicating with other transputers (Quora, n.d).

The Inmos transputer was a British-designed, novel parallel microprocessor architecture from the early1980s. The transputer was unique in that each processor had a built-in simple operating system, memory and four high speed (20 Mbit/s full duplex) bi-directional serial links. The transputer is essentially a computer system on a chip. The links on the transputer allow connection to up to four other transputers or peripherals such as video graphics, floppy and hard disc drives, Ethernet networking and standard RS-232 serial ports (Jaros, Ohlidal and Dvorak, 2005).

The rationale behind the design of the transputer came from the point-to-point connection architecture of the Modular-One computer and the work of Tony Hoare (of QuickSort fame) on Communication between Sequential Processes. His seminal paper led to the design of the Transputer and the parallel programming language Occam by David May. Starting in early 1981 David May designed the Transputer and the novel Occam language and compiler. It was not until 1985 that the first prototype Transputers came off the production line at the Inmos foundry at Newport, Gwent (Borger and Stark, 2012).

Inmos was sold to Thorn EMI under the privatizing Thatcher government in the mid-80s, and later to SGS Thomson, Transputer development continued, but was eventually abandoned. It's not so easy to kill the Hydra-headed Transputer. The site of the original Inmos design centre in Bristol is now owned by STMicroelectronics (May, n.d).

This paper discusses the original purposes of the transputer technology, how they have been addressed over the intervening past years, the architectural designs and network. It also emphasizes the factors that promoted the effacing of transputers and the Restoration movement of transputer.

The study was carried out by performing in-depth study on transputer literatures.

## THE TRANSPUTER TECHNOLOGY DRIFT

Since its introduction around 1980’s according to (Fox, Williams, and Messina, 2014), the transputer has drawn much attention among researchers and design engineers, due its novel architectural features and excellent performance.

(Sheen, Allen, Ripke, and Woo, 1998), stated that the Inmos transputer device range, which started with the T414 in 1983, was continued with a series: the 16-bit T212, T222 and T225. The 32 bit line was extended with the T425, which ran with a faster processor clock, more internal memory and also improved the instruction set and debug architecture. The T800, announced around 1987, included a formally verified IEEE conformant Floating Point Unit (FPU) (quite rare for the period) and was developed soon after the release of the T414.

It was followed by the T801 and T805, introducing among other things the improved software debug also seen on the T425. The M212 was an early trial of an Application Specific Standard Part (ASSP): an MFM disk interface controller version of the T212 (Sheen, et al., 1998).

The T9000, intended as the next in the line, was announced with support for a significantly extended instruction set, a hardware implementation of virtual channels, superscalar performance and clock speeds which once again matched the competition, but for various reasons the chip was delayed, suffered badly from silicon bugs, and eventually canned years late having reached about a quarter of its intended performance (May, 2005).

Finally, the T400 (a 2 link T425 with reduced internal memory) and T450 (a T425 with extra internal memory and enhanced instructions) heralded the "official" end of the line of transputers.

By the time the T450 (later known as the ST20) was in the field, interest in the transputer range as a whole was waning, and having already turned away from occam SGS Thompson announced "last orders" on the transputers themselves and gave the impression it was dropping the line entirely (May, 2005).

As shown in figure 1 below, other processors were moving on too. Intel's line of x86 processors was steadily growing, with the 80186 embedded system version gaining a lot of popularity, helped by the ready availability of compilers, support tools and chips produced for the burgeoning IBM PC market. Although the x86 range was not particularly powerful, they were quite capable of many of the tasks being asked of them. The Motorola 68000 series, having dropped significantly in price since its release and gained a number of specialized variants, also gained a lot design wins. Both the Intel and Motorola processors had one significant point in their favour – low price. Gained partly through volume and partly through in house fab lines, much of the market is very price sensitive (Hilton and Ivimey, n.d).

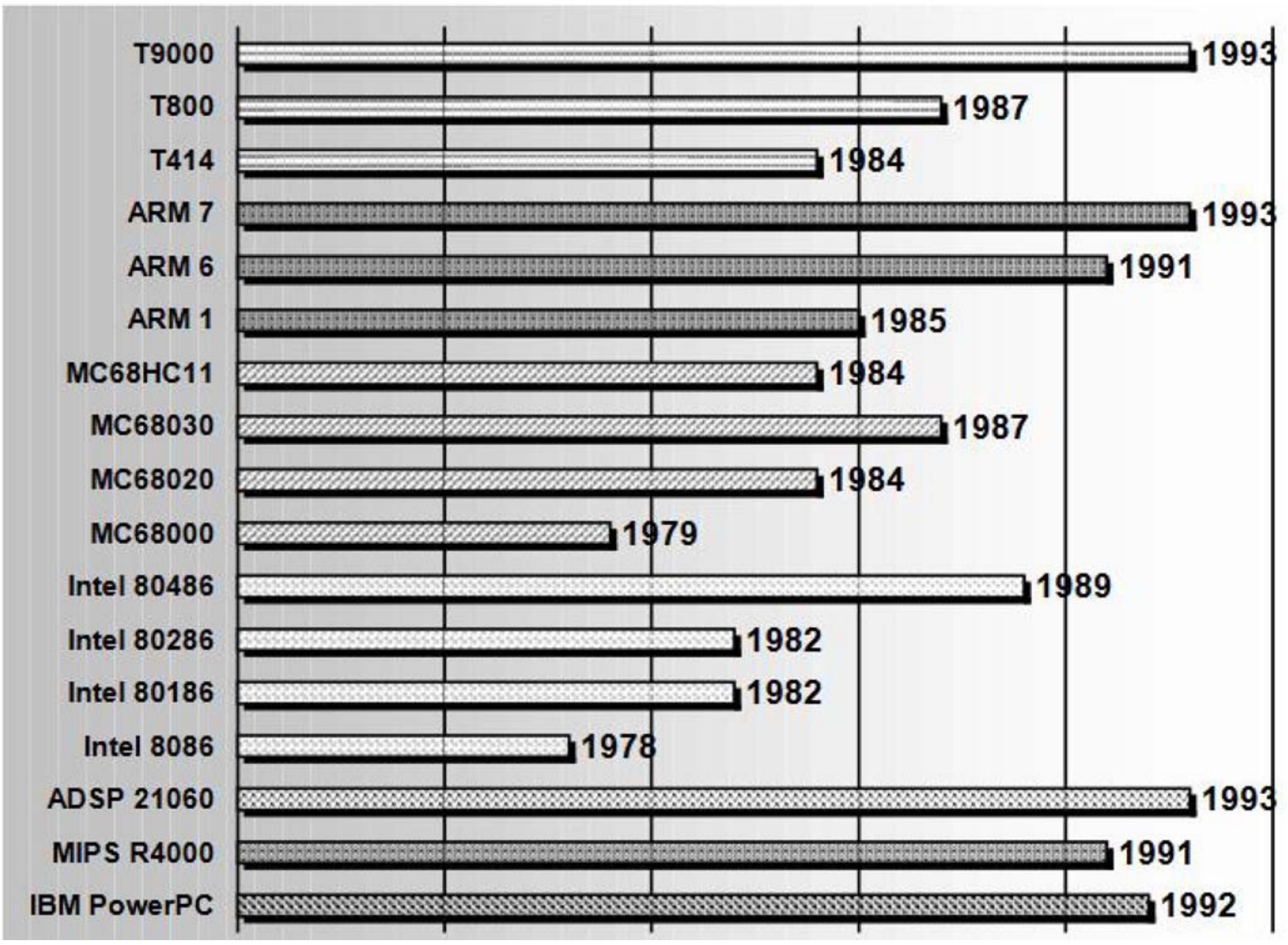

FIGURE 1: This is figure 1. The release dates of various Transputers and some of their competitors (Hilton and Ivimey, n.d).

Hitachi entered the market with its Z80 compatible range of processors – the HD64180 being a significant advance on the ageing Z80, and have carried on with the SH series which was very successful indeed. A number of new designs emerged from Zilog themselves, but the Z8000 and Z800 never caught on in a big way.

During this time, a community of academic and commercial developers had grown around the technology; user groups such as The World occam and Transputer User Group (WoTUG), and The North American Transputer User Group (NATUG) grew and gained large followings. For some time, the community supported several conferences a year averaging 250 delegates each. A lot of experience was gained in the use of the very fine grained parallelism which was offered by the transputer. An interesting observation was made that the programmers with a background in hardware design fared better with the design of these highly parallel systems than did those with a traditional computer science background. This experience has undoubtedly had its effect on the software community at large; a very large number of engineers have at least heard of the transputer, and quite a number of those people are aware of the capabilities of the device. What is a bit sad is that what is remembered best in the industry is the transputer's quirkiness – for example its use of an odd language as well as the rather unusual inclusion of fast communications links in preference to GPIO lines or an RS232 interface – rather than the benefits these things provided (Morse, 2014).

On the commercial front a lot of products were developed around the Inmos suggested TRAM format. This was a PCB with a base unit size slightly longer than credit card size, which could be plugged into a motherboard with connectors along the short sides. Designs requiring a larger area could use multiple TRAM 'units'. The external connectors carried mostly a number of OS Link interfaces. Bringing out address or data bus would have been contrary to the basic principle of the transputer. According to (Furber, 2017), the design of TRAMS was well received and many devices were put on it, including high resolution graphics cards, RS232 and RS432 serial interfaces along with the expected T4 or T8 CPU plus memory.

- The Transputer's Purpose

The transputer is a programmable device on a single chip with a stunning performance. The importance of the transputer is that it provides a higher level of abstraction in the design of information systems, due to its inherent support for multiprocessing (Furber, 2017). (Bhowmick and Prasad, 2017), said opined that multiprocessing is the only way to provide the high performance levels demanded by some present-day applications at a moderate cost. Due to limitations of a physical nature, technology cannot provide sufficient increase in performance, so new techniques are needed. Multiprocessing is definitely the most important one.

The key innovation of the transputer is its inherent concurrency. In all processors multi-tasking is done by software (the operating system kernel) that gives slices of CPU time to every task that is ready to run (time slicing). But with the transputer, however, this multi-tasking kernel and scheduler is already embedded in the microprocessor's microcode, which makes the transputer ideally suited for multi-tasking applications (very fast context switching) (Laplante and Milojicic, 2016).

Transputers during its era saw its purposes in satisfying several applications. (MATSUI, 2015) Identified some of the areas the transputer had gained grounds: High speed multiprocessor systems, Workstations and workstation clusters, Supercomputers, Real time processing, Scientific and mathematical applications, Digital signal processing, Accelerator processors, Distributed databases, System simulation, Telecommunications, Microprocessor applications, Industrial control, Robotics, Fault tolerant systems, Medical instrumentation, TRANSPUTER ARCHITECTURE DESIGN AND NETWORK

The transputer, manufactured by INMOS, is a single chip Very Large Scale Interface (VLSI) device with processor, memory, and communications links. This represents a slight deviation from today's microprocessor architecture. The common features of transputers are: High speed integer processor with micro-codes process scheduler; On-chip fast static memory; Up to four links for communication with other transputers; Internal timers; and External memory interface (Chertovskikh and Rachek, 2014).

Graphics processing, Image processing, Pattern recognition, Artificial intelligence, etc.

- Architectural Design

The INMOS transputer is the first single-chip microprocessor to provide a high speed processor, fast inter-processor communications, and explicit support for multiple processes and multiple processor systems. Transputers are designed to be part of a multiprocessor system, so the performance of an individual processor is not especially critical. If more processing power is needed, more processors can simply be added. Figure 2 below shows the block diagram of a generic transputer.

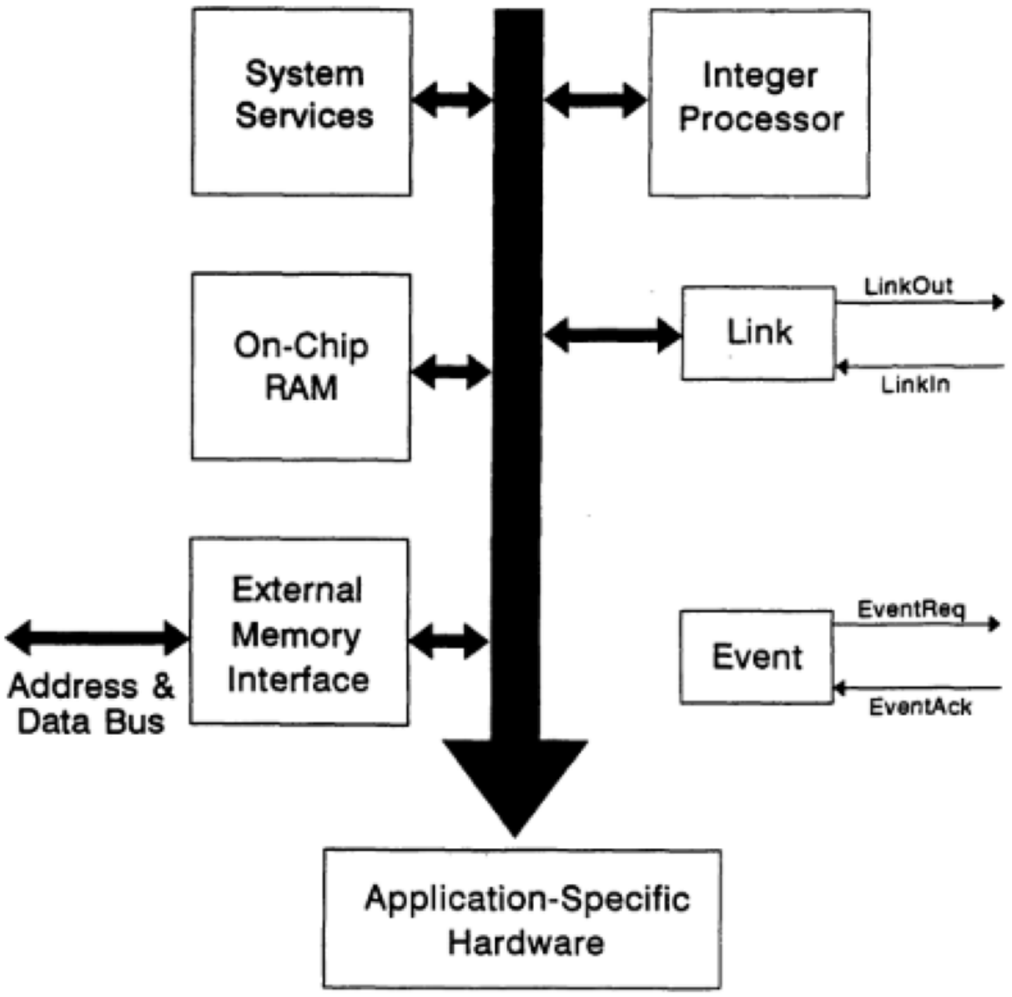

FIGURE 2: This is figure 2. The block diagram of a generic transputer (Transputer, 1993).

The internal design of the transputer is unlike that of any of its predecessors (Fox et al., 2014). The central concept of transputer architecture is that of the process. A process represents an individual thread of control and the transputer switches between running processes to provide the illusion that they are all running simultaneously. This is normally handled by the operating system and called multitasking, but in the transputer, this is implemented in hardware and micro-coding (Haefner, 2018).

According to (Manet and Rousseau, 2016), all transputers have a fast integer processor and many instructions that take only a single cycle of the processor clock to complete. Transputers were manufactured with clock speeds up to 25 MHz. All transputers, however, operate from an external clock speed of 5 MHz. The processor clock is obtained from an internal phase-locked loop multiplier.

- The Floating Point Unit

The Floating Point Unit is that part of a processor which performs floating point arithmetic. Some series of transputers have a 32/64-bit floating point unit that conforms to the IEEE 754-1985 specification. The floating point unit (FPU) has an evaluation stack similar to that of the integer processor, with three registers: FA, FB, and FC. Each of these registers can contain either a 32-bit or a 64-bit number and has a flag to show which of these it does contain. The FPU design is a compromise between maximizing overall processor performance and minimizing chip area. Because of this, the FPU has no flash multiplier or barrel shifter. However, the performance is good, with single and double precision multiplication times of 550 and 1000 nanoseconds respectively, for a 20 MHz device (Cohen and Wang, 2014). The FPU operates concurrently with the integer processor, and thus computation can be speeded up by overlapping integer and floating-point processing. Figure 3 shows the floating point unit.

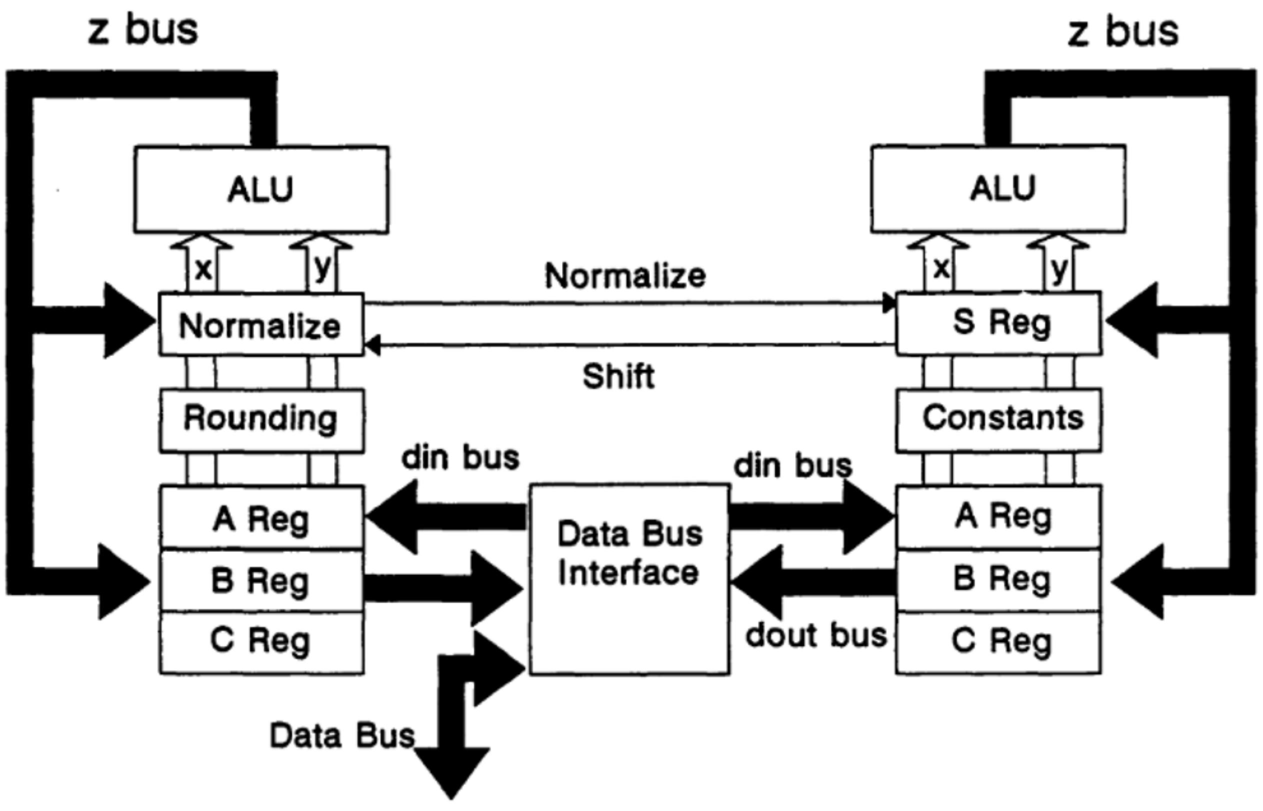

Figure 3: This is figure 3. The Floating Point Unit (Transputer, 1993)

- The Transputer Network

A typical Transputer has four hardware communication ports, permitting a variety of configuration patterns when a number of these processors are linked together. This was demonstrated by (Hass, Kuila and Shahid, 2017), where applications requiring distributed processing from a single stream of control data, the most effective arrangement can be shown to be a ternary tree, providing hierarchical control. While a simple tree structure provides short path lengths between the arbitrary nodes, a modified ternary tree can also achieve this between siblings at the same level, thus increasing the scope for achieving both efficiency and flexibility in the flow of data between transputers. Four of these single elements, a total of 16 transputers, are mounted on a standard 3U printed circuit board as shown in Figures 4 and 5. The transputers are permanently hardwired to each other.

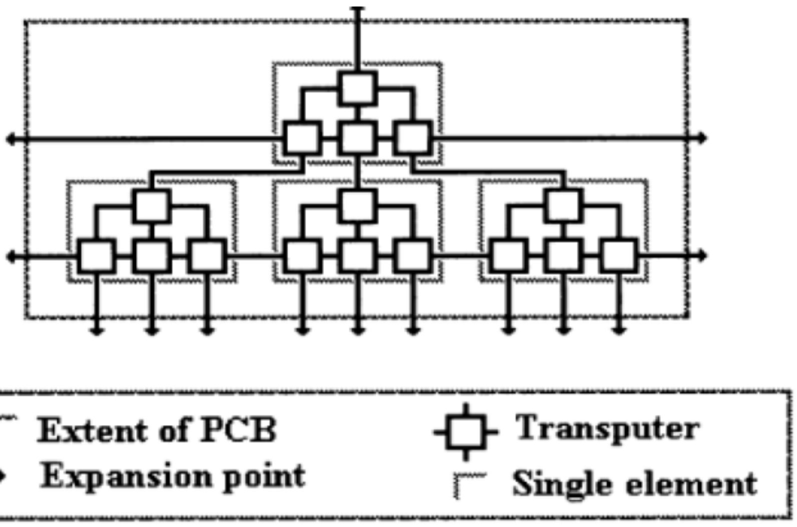

Figure 4: This is figure 4. Basic Network Topology of a 16-Transputer Board (Itagaki et al., 2018).

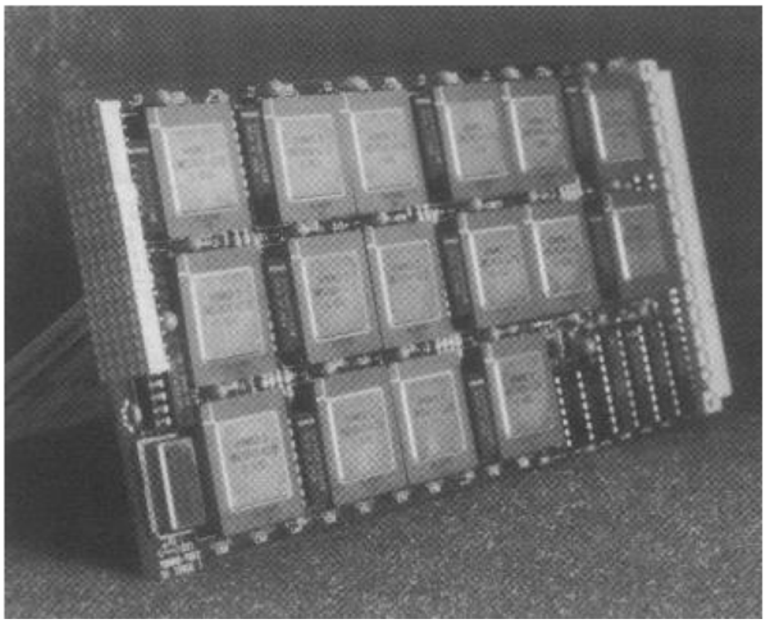

Figure 5: This is figure 5. Printed Circuit Board with 16 Transputers (Itagaki et al., 2018).

The rationale behind this design feature is that a real-time distributed system should not require a large amount of on-board memory for inter mediate data storage. Each transputer thus uses only its internal 4 kB memory for programming purposes. This results in a total of 160 transputers with 640 kB of internal memory distributed across the network in the case of the 10-board system, as shown in figure 6, and a maximum processing power of 1.4 GIPS. The absence of local external memory necessitates compact algorithms for execution at audio sampling rates and the use of a compact code such as Occam, a programming language designed specifically for the transputer family.

- OCAM as transputer's Instruction Set

In the words of (Furber, 2017), the transputer does not readily fall into either the CISC (complex instruction set) or the RISC (reduced instruction set) categories. It has a simple instruction set, called Occam, and tends to be viewed as a RISC processor. However, it is much more than a RISC processor because of the functionality built into the chip to support high level concepts such as processes, timers, and inter-process communication. Programmers used to programming other microprocessors may find programming the transputer to be a strange experience (Bull, 2016). There is only a small number of registers that are organized as a stack, and all instructions are stack, rather than register oriented. There is little concept of condition codes, only limited instructions for accessing memory, and more sophisticated memory-addressing modes (Heath, 2014).

The language created by Inmos is the Ocam. It was based on research by (He, Joseph and Hoare, 2015) that described a mathematical language called Communicating Sequencing Processes (CSP). CSP describes the interactions of processes interacting through the exchange of signals or messages. In CSP, the internals of a process are opaque, as are the events. In occam, CSP processes are implemented as tasks, and events as messages communicated over one- to-one links. This mapping retains an extremely important feature of CSP: processes can be composed. For an instance, if two processes x and y interact together and with some environment, the external behaviour can be modeled as a process z. If y is replaced with another process implementing its external behavior, you also end up with z so long as x and y are composed. Composition of processes is believed to be essential to the ability to effectively design systems. Consider how it would be if you had to use a particular finger to switch on a light. In real world, things do exhibit composition.

## THE EFFACEMENT OF TRANSPUTER

The transputer invention was a realistic implementation multiprocesses. It was intended to provide high performance at low cost. The main idea behind the transputer was quite simple: instead of creating a very complex processor, the transputer consisted of a family of chips. Each chip had a very simple design and multiple chips could be wired together to form an entire computer. Each transputer chip was in fact some kind of a microcontroller and was able to boot and operate by itself, it had its own RAM, a serial bus and an embedded real-time OS. Amidst the achievements of the transputer's era, there were problems that promoted the effacement of the transputer. These problems birth the replacement of the transputer with sophisticated approach to multiprocessing: the Intel core microprocessors.

Occam was developed to specifically support the development of the fine grained parallel processing environments supported by the transputer model and typical of those found in embedded systems. It also supports extensive checking of programs, threaded or not (Haefner, 2018).

## FACTORS THAT PROMOTED THE EFFACEMENT OF THE TRANSPUTER

There are ample number of factors, amidst the pool of the transputer architecture applications that militated against its permissive use through the generations of computing processor. Some of the major reasons are identified in this study.

- Computing Power Scalability

As the computing power increases overtime, there came a need for a miniature architecture which birth the microchips. The latching of transistors became a bulky technology to be laid on a mother board of a system. The idea of developing a high speed processor with compressed IC board became essential for the growth of microprocessor speed. Also, since a high computing power gained by the combination of multiple transistors can be replication in a small high speed microprocessor, transputers application diminished due to the large size of trasputers network.

- Need for Superscalar Processing

According to (Denning and Lewis, 2017), growing internal parallelism has been one driving force behind improvements in conventional CPU designs. Instead of explicit thread-level parallelism, as is used in the transputer, CPU designs exploited implicit parallelism at the instruction-level, inspecting code sequences for data dependencies and issuing multiple independent instructions to different execution units. This is termed superscalar processing. Superscalar processors are suited for optimising the execution of sequentially constructed fragments of code. Given these substantial and regular performance improvements to existing code there was little incentive to rewrite software in languages or coding styles which expose more task-level parallelism.

Unlike the transputer architecture, the processing units in these systems typically use superscalar CPUs with access to substantial amounts of memory and disk storage, running conventional operating systems and network interfaces. Resulting from the more complex nodes, the software architecture used for coordinating the parallelism in such systems is typically far more heavyweight than in the transputer architecture (Agullo et al., 2017).

- The Inmos Patent Right Transfer

Inmos improved on the performance of the T8 series transputers with the introduction of the *T9000* (Haase and Pester, 2013). The T9000 shared most features with the T800, but moved several pieces of the design into hardware and added several features for superscalar support.

Long delays in the T9000's development meant that the faster load/store designs were already outperforming it by the time it was to be released. It consistently failed to reach its own performance goal of beating the T800 by a factor of ten. When the project was finally cancelled it was still achieving only about 36 MIPS at 50 MHz. The production delays gave rise to the quip that the best host architecture for a T9000 was an overhead projector. This was too much for Inmos, which did not have the funding needed to continue development. By this time, the company had been sold to SGS-Thomson, now STMicroelectronics, whose focus was the embedded systems market, and eventually the T9000 project was abandoned (Clark, 2006).

When Inmos was sold to Thorn EMI under the privatising Thatcher government in the mid-80s, and later to SGS Thomson, Transputer development continued, but was eventually abandoned.

- Poor Memory Management Unit (MMU)

Another major problem of the transputer was the lack of an MMU or virtual memory support, which prevented UNIX to be ported to the transputer architecture. Although there were ports of some UNIX-like OSes.

The Ocam Wane

The occam language, known as the transputer language or instructions set, was though hampered by the lack of compilers for other chips, it waxed as real time systems developers discovered its expressive power. However, as the transputers lost their speed advantage against other processors, people found them less and less practical, and the use of occam waned too.

- The Restoration Movement of Transputer.

It was after Inmos was sold out that the idea of Transputer restoration project came up. After the termination of T9000 project, there was a comprehensive redesigned 32-bit transputer intended for embedded applications, the ST20 series, using some technology developed for the T9000. The ST20 core was incorporated into chipsets for set-top box and Global Positioning System (GPS) applications. According to (May, 2005), ST20 was not strictly a transputer, but it was heavily influenced by the T4 and T9 and formed the basis of the T450, which was arguably the last of the transputers.

The kernel of the idea to breathe life back into a forgotten piece of British technological genius was given by Paul Walker, of 4Links Ltd, who worked with the Inmos Transputer in the early days and took forward some of its design ideas to help create *SpaceWire*, a serial communication technology used throughout the space industry (Chintalapati, 2016).

According to (Project Aims, n.d), the transputer Restoration Project aims to bring into service and maintain two fully functioning Inmos Transputer Development System units. The first unit contains 64 processors and runs a demonstration of parallel processing of this 1980's technology using a Mandelbrot Set generator and a Ray Tracing simulation. Also, a system containing two banks of 12 processors with dedicated video cards and monitors to drive a dual flight simulator.

A longer term project is to restore a large system based around a network of 100 T9000 Transputers that were originally used for fast data capture at the European Organization for Nuclear Research (CERN).

## SUMMARY

The intention of this paper was to review and summarize the era, and the idea of the transputers technology which birth the implementation of today's multicore microprocessors.

We have looked at the basis of the transputer: the elements of simplicity and in integration that made it a useful and very powerful processor in its own time. We noted here that the transputers were designed to satisfy their original purposes. We have also identified the occam language as the instruction set of the transputer which Inmos designed with the transputer to provide high level access to these facilities.

We looked at how the transputer evolved and addressed over the intervening past years, here, we discussed the architectural design and the network of transputers.

Some of the factors that contributed to the dead end of the transputer legacy were identified. We also discussed how these factors hindered the progress of the transputers' operations. Finally, we reported that a transputer restoration project to bring into service and maintenance two fully functioning Inmos Transputer Development System units.

## REFERENCES

1. What is a transputer_ - Quora. (n.d.).
2. Jaroš, J., Ohlídal, M., & Dvořák, V. (2005, Oct.). Evolutionary design of group communication schedules for interconnection networks. In *International Symposium on Computer and Information Sciences* (pp. 472-481). Springer, Berlin, Heidelberg.
3. Börger, E., & Stärk, R. (2012). *Abstract state machines: a method for high-level system design And analysis*. Springer Science &Business Media.
4. David May, parallel processing pioneer • The Register. (n.d.).
5. Fox, G. C., Williams, R. D., & Messina, G. C. (2014). *Parallel computing works!*. Elsevier.
6. Sheen, T., Allen, A. R., Ripke, A., & Woo, S. (1998). oc-X: an optimizing multiprocessor occam system for the PowerPC. In *Architectures, Languages and Patterns for Parallel and Distributed Applications*. IOS Press.
7. May, D. (2005). CSP, occam and Transputers. In *Communicating Sequential Processes. The First 25 Years* (pp. 75-84). Springer, Berlin, Heidelberg.

8. Hilton, C., & Ivimey, C. R. Legacy of the transputer. *emergence*, *80186*, 19.
9. Morse, H. S. (2014). *Practical parallel computing*. Academic Press.
10. Nocetti, D. F. G., & Fleming, P. J. (2012). *Parallel processing in digital control*. Springer Science & Business Media.
11. Furber, S. B. (2017). *VLSI RISC architecture and organization*. Routledge.
12. Bhowmick, A., & Prasad, C. G. V. N. (2017). Time and cost optimization by grid computingover existing traditional IT systems in business environment. *Int J*, *5*, 93-98.
13. Laplante, P., & Milojicic, D. (2016, Oct). Rethinking operating systems for rebooted computing. In *2016 IEEE International Conference on Rebooting Computing (ICRC)* (pp. 1-8). IEEE.
14. MATSUI, K. (2015). Towards Transputing with Parallella!!.
15. Chertovskikh, A., & Rachek, I. (2014, Oct.). Using Transputer Computing Systems at the BudkerInstitute of Nuclear Physics. In *Computer Technology in Russia and in the Former Soviet Union (SoRuCom), 2014 Third International Conference on* (pp. 86-88). IEEE.
16. Transputer, T. H. E. (1993). 93 12 13 077'.
17. Haefner, J. W. (2018). Parallel computers and individual-based models: an overview. In *Individual based models and approaches in ecology* (pp. 126-164). Chapman and Hall/CRC.
18. Manet, P., & Rousseau, B. (2016). "Tile-based processor architecture model for high efficiencyEmbedded homogeneous multicore platforms." *U.S. Patent No. 9,275,002*. Washington, DC: U.S. Patent and Trademark Office.
19. Cohen, R., & Wang, T. (2014). Intel embedded hardware platform. In *Android Application Development for the Intel® Platform* (pp. 19-46). Apress, Berkeley, CA.
20. Hass, D. T., Kuila, K., & Shahid, A. (2017). *U.S. Patent No. 9,596,324*. Washington, DC:

U.S. Patent and Trademark Office.

21. Itagaki, T., Manning, P. D., Purvis, A., Purvis, A., Road, S., & Dh, D. (2018). Distributed Parallel Learned from a 160- Transputer Network Processing : Lessons, *21*(4), 42–54.
22. Bull, M. (2016). *Students' Guide to Programming Languages*. Elsevier.
23. Heath, S. (2014). *Microprocessor Architectures: RISC, CISC and DSP*. Elsevier.
24. He, J., Josephs, M. B., & Hoare, C. A. R. (2015). A theory of synchrony and asynchrony.
25. Denning, P. J., & Lewis, T. G. (2017). Exponential laws of computing growth.
26. Agullo, E., Aumage, O., Faverge, M., Furmento, N., Pruvost, F., Sergent, M., & Thibault, S.P. (2017).Achieving high performance on supercomputers with a sequential task-based programming model. *IEEE Transactions on Parallel and Distributed Systems*.
27. Haase, G., & Pester, M. (2013). A Brief History of the Parallel Dawn in Karl-Marx Stadt/Chemnitz. In *Advanced Finite Element Methods and Applications* (pp. 1-26). Springer, Berlin, Heidelberg.
28. Clark, I. (2006). Microprocessor Course Part I Processor History and Selection.
29. Chintalapati, L. V. B. (2016). Integration of Mission Control System, On-board Computer Core and spacecraft Simulator for a Satellite Test Bench.
30. Project Aims The National Museum of Computing. (n.d.).