

The TRON project

Ken Sakamura outlines the objectives of the TRON project and the possibilities it offers for a global computer network

The TRON (the realtime operating system nucleus) project to design a new computer system for the 1990s began in 1984. The TRON computer architecture for the 1990s is intended to support the use of personal computers as communication machines and many 'intelligent objects' hooked into an HFDS (highly functional distributed system). To support various application needs, four operating system specifications are being designed: ITRON, BTRON, CTRON and MTRON. The project has also designed the TRON VLSI CPU-specification, 32- and 64-bit microprocessor chip. The system being designed will be used to form HFDSs that can act as global networks with data in different languages being passed freely.

microprocessors TRON global networks

 The first electronic digital computer was introduced approximately 40 years ago, and in the years since extensive knowledge about computer science and engineering has been acquired. Not only have computers been toyed with to find out exactly what they can do, but many basic ideas, such as parallel processing and artificial intelligence (AI) have also been probed. In a sense, this work amounts to a long-range feasibility study of how to use computers in society in the future.

A parallel development, the advent of microelectronics, has made possible the production of computers that are small in size and low in cost. This new technology coupled with the computing knowledge gained over the last 40 years has thus led to a position where the 'next step' in the use of computers can be contemplated.

Which 'next step' is the most important in the use of computers? Some say it is AI, while others say it is parallel processing or supercomputing. Needless to say, these are all important topics in the field of computer science to which due attention must be paid.

However, when considered in terms of the history of computing, the most important use computers can now be put is in global computing networks. In this article, such versatile global networks are discussed in terms of highly functional distributed systems (HFDS) — loosely connected networks of heterogeneous computers that together perform services that would be impossible for

each individual computer to perform. In other words, the cooperative network can offer services that are not the simple sum of the services of the individual components.

The building of such networks of computers, called HFDSs, is one of the objectives of the TRON (the realtime operating system nucleus) project^{1, 2}.

The author believes the best way to do this is to rebuild and refine current computer architecture by using the available and predicted microelectronics technology, while, at the same time, paying due consideration to the knowledge of computer science so far gained. The short-term objective of the TRON project is to develop many systems based on the TRON architecture. Combining these systems into a cooperative network that can serve as a truly functional HFDS is the long-term goal.

The computerized society discussed in this paper has been a dream since the early days of computing. However, this dream was abandoned when the technological difficulties involved were realized. Now, with the technology currently at hand, and that which will become available in the near future, the dream has been revived.

History of the TRON project

The original ideas behind the TRON project, which are explained in the subsequent sections, began to take shape around 1982. The project did not formally begin until 1984. Roughly two years later, on 16 June 1986, the TRON Association was formed by a group of Japanese firms, including Fujitsu Ltd, Hitachi Ltd, Matsushita Electric Industrial Co. Ltd, Nippon Telegraph and Telephone Corporation, NEC Corporation, Oki Electric Industry Co. Ltd, and Toshiba Corporation. The Association originally had 30-odd members, but the number has now climbed to more than 100, including Intel Japan K.K., Motorola Inc., Japan Texas Instruments Inc., and IBM Japan Ltd.

How did the project obtain this level of backing? Since openness is one of the prime objectives of the project, the research team under the author's direction at the University of Tokyo has played a central role and received much help from the industrial members of the TRON Association. The project's objective of promoting open competition by providing interface standards and leaving the implementation to manufacturers has been well received by industry. In this way, the project has taken a practical path and obtained cooperation from organizations and individuals throughout the world.

Department of Information Science, Faculty of Science, University of Tokyo, 3-1 Hongo 7-chome, Bunkyo-ku, Tokyo 113, Japan

0141-9331/89/08493-10 \$03.00 © 1989 Butterworth & Co. (Publishers) Ltd

This paper explains what HFDS is, and how its construction is expected to be achieved by developing the necessary underlying technologies. The subprojects of the TRON project are also explained.

THE HIGHLY FUNCTIONAL DISTRIBUTED SYSTEM (HFDS)

When considering the network of computers in the future, account must be taken of two application fields that have become possible thanks to the development of microcomputer chips. These are embedded control applications and personal computers.

The intelligent object

It has become possible to embed computers in various machines as a result of the miniaturization of central processing units (CPUs) and other devices. Such embedded computers extend the usefulness of the host machine. For example, 35 mm cameras, microwave ovens, refrigerators, washing machines, and food processors are just a few of the appliances that nowadays give users a better service as a result of computer control.

In the TRON project, devices, appliances, and machines that have sophisticated or intelligent capabilities as a result of embedded computers are called 'intelligent objects'. The TRON project considers intelligent objects to be an extremely important application of computers to which serious attention must be paid when considering future uses for computers.

The personal computer as a communication machine

The personal computers on the market today have mostly been designed with the needs of the business man and computer enthusiast in mind. They are not really designed for 'everyone', even though the expression 'personal computer' implies that they are.

The 'true' personal computer should be a communication machine that can be used by anyone, including the handicapped. Before today's general-purpose, data-processing personal computers can truly be called 'personal', they must become proficient in the three types of communication. In the TRON project, the term 'electronic stationary' has been coined to describe this future type of personal computer and its accessories.

The three types of communication are:

- communication with oneself
- communication with others
- communication with machines.

'Communication with oneself' essentially means writing a document, drawing figures, or creating any type of new data by means of a computer. This is a creative process based on trial and error in which one communicates (or has a dialogue) with one's inner self in order to fix ideas in a concrete form on the computer, and is most often done with a word processor and drawing programs.

On the other hand, communication with others includes any form of data interchange within a group of

people, and not only ordinary communication over telecommunications networks. This type of computer use requires sophisticated presentation capabilities, such as computer graphics and hypertext functions. It must also be backed by the capabilities to ease communication tasks between people from different cultural backgrounds, the most obvious example being an automatic translation service. Finally, provisions for the handicapped, such as automatic use of Braille for the blind, are also necessary.

Communication with machines means controlling machines with a computer. However, when communication using computers is referred to, the first two types of communication are most often discussed. One important application of computers that may not be apparent to the layman is the automatic control of machines in their everyday environment (Figure 1). When large networks of intelligent objects come into use in the future, the computer's ability to communicate with machines will be important to the average person.

The microcomputer chip and the HFDS

The technology that supports the development of intelligent objects and communication machines is the small, inexpensive microcomputer chip. When the above two applications based on microcomputer chip technology come into practical use, the computer will begin to take on new roles in everyday life.

As a result, an explosive increase in the number of computers is inevitable once intelligent objects and communication machines go into production. If the trend continues, and there is no reason to believe that it will not, the number of computers used in an average person's daily life will grow considerably. It is easy to imagine the day when all appliances and even pieces of furniture will contain embedded computers. When that day arrives the number of computers per person will be of the order of hundreds.

Some readers are bound to think such a scenario is unlikely due to cost considerations, and they may have a point with respect to the immediate future. However, the cost of high-performance CPUs will become negligible in the long term. Those who do not believe this need only ask themselves how much they paid for their last calculator.

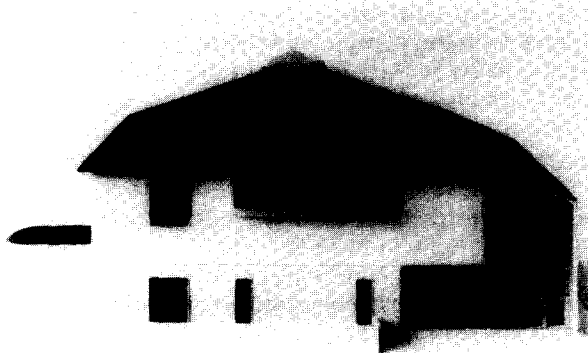


Figure 1. The TRON intelligent house (currently, a house is being built using many intelligent objects to verify the usefulness of intelligent objects. The photo shows the model of the house)

In addition to CPUs, the cost of sensors necessary for intelligent objects will also decrease dramatically as production volume increases, and new technologies such as micromachining mature. Micromachining, the branch of mechanical engineering that deals with the manufacture of miniature mechanisms, offers the potential of building a CPU-sensor hybrid on a single chip.

In conclusion, Figure 2 is a sketch of the predicted computerized society of the future. It is the age of intelligent objects in which everything — from manufacturing equipment, elevators, and other types of machinery, to appliances, pieces of furniture, and even houses themselves — incorporates a microcomputer and sensors and shows intelligent behaviour. It is also the age in which everyone has an easy-to-use communication machine, which they casually employ for many tasks. Whenever the communication machine lacks the power to perform some specialized task, mainframe computers are linked to the HFDS to provide the necessary computing power.

The most important feature of this picture of the computerized society of the future is that every communication machine and intelligent object will be interconnected in some way (Figure 2). Thus, in the computerized society based on the HFDS, people will have at their finger tips both the capability to do something trivial, like calling an elevator, and the potential to perform a complex task, such as the production of an industrial product (i.e. concept design, mechanical design, collection of materials, machining and assembling).

Characteristics of the HFDS

The differences between the HFDS and the distributed systems presently in use are both quantitative and qualitative.

The quantitative difference lies mainly in the number of computers that make up a system. For the most part, the distributed systems of today are local networks made

up of a few dozen computers. In many cases, interaction between the computers in the system is rare, and local computing is the norm.

In the HFDS on the other hand, the number of computers envisaged to interact in one room is over a hundred, and in a large building could exceed a few thousand. It is natural to expect that these buildings will be connected as subnetworks of local or regional networks. As a result, the number of computers in a city could exceed a million, and those scattered throughout a country could surpass a billion. Thus when considering the possibility of a truly global network, the control of hundreds of billions of computers loosely connected in a large network must also be considered seriously.

The qualitative difference between today's distributed systems and the HFDS is that the interaction between computers in the HFDS is more varied than in current distributed systems. For example, in the HFDS the following scenario can exist. Intelligent sensors and intelligent air conditioners in a network talk to each other and decide to obtain the optimal control strategy within a building by creating a model of air flow in a room. The latter task is performed by a supercomputer linked to the HFDS. Communication computers operated by humans can intervene in the control of the air conditioning, and control computers that manage the crucial network nodes can also interact with the network, performing tasks such as load balancing, diagnosing failures, and attempting recovery procedures to keep the network operational.

Requirements of the HFDS

Many underlying technologies are required to build a HFDS such as described above. Some of the more important of these underlying technologies can be listed as follows.

First, communication methods must be established between the components of the HFDS for truly distributed control, i.e. no central control. When considering communication methods or protocols in the HFDS, it is important not to think of communication protocols in narrow terms. This is because to obtain a working HFDS, not only low-level communication protocols, but also high-level data interchange formats and protocols must be defined for highly sophisticated interactions to take place between the component computers.

Second, a man-machine interface (MMI) serves as the base of interaction between the human users and the machines connected to the HFDS. Needless to say, the resulting system must be designed so that it can be easily operated by human users.

Third, to achieve a truly global network, both the HFDS communication protocols and the languages used by different nationalities. In other words, the input of data to be computed and the output of messages from the computer must make sense to any user in any part of the world. Thus the HFDS must have a multilingual capability so that people everywhere can communicate without compromising the use of their mother tongue.

However, the multilingual capability of the HFDS must not simply aim to process strings of words using the character sets of the world's languages. It should also be directed toward the portability of a program developed for a given language, such as Japanese, across cultural

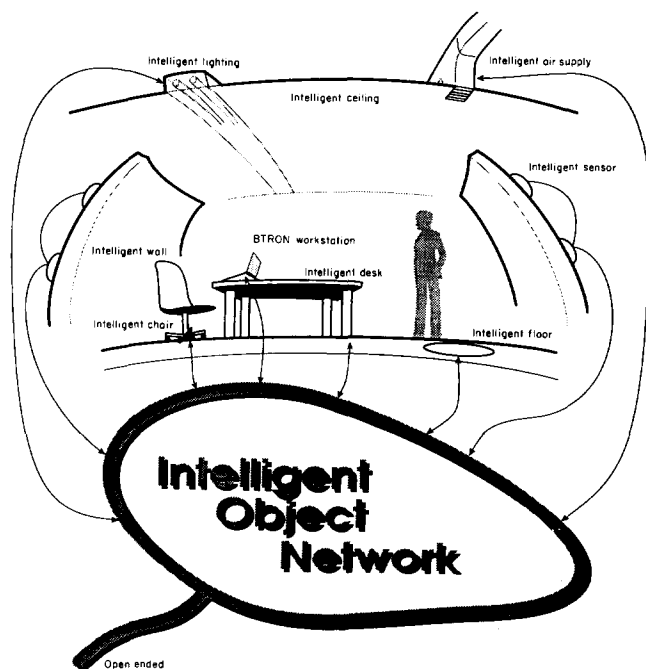


Figure 2. HFDS environment

boundaries, i.e. the program can be used for other languages without requiring extensive rewriting.

When one considers how much time and effort is spent developing software throughout the world today, it is important to provide a way to write computer software that is independent of the natural language it is supposed to handle. Even if an efficient program is available in one language, say English, one cannot expect someone whose mother tongue is French or Japanese to learn English to take advantage of it. It is therefore most desirable for the computer system to resolve this language dependency so that people can use programs in the language of their choice.

An explanation of how each of these underlying technologies for the HFDS has been approached in the TRON project is given.

FEATURES OF THE TRON PROJECT

The total architecture

In the TRON project, the target computer systems range from embedded microcomputer chips to large mainframe computers, all of which are networked together. The goal is to get this wide variety of machines to process in real

time while achieving clean separation of interface specifications and implementation. In addition, high-level utility programs must be written along with carefully prepared training programs.

Needless to say, this goal cannot be satisfied for such a wide variety of operating conditions using a single computer system architecture. Consequently, the following four operating systems (OSs) have been designed for the TRON project: industrial TRON (ITRON) for embedded computer applications in intelligent objects³⁻⁴, business TRON (BTRON) for communication machines to be used by humans⁵⁻⁶, central TRON (CTRON) for mainframe computers and servers in networks⁷⁻⁹, and macro TRON (MTRON) for controlling the computer network itself (Figure 3).

Within each OS family there are various interface standards such as instruction set processor (ISP), OS system calls, high-level utility programs, a character code system, and man-machine interface functions. Great efforts have been made to attain uniformity and consistency in the design of these interface standards in order to eliminate the incompatibilities in many interface definitions of existing computer systems.

The total redesign of the current von Neumann computer architecture has allowed special attention to be paid to performance tuning. For example, the TRON architecture has been designed with special CPU instructions to speed up the specialized operations that are often carried out within realtime operating systems.

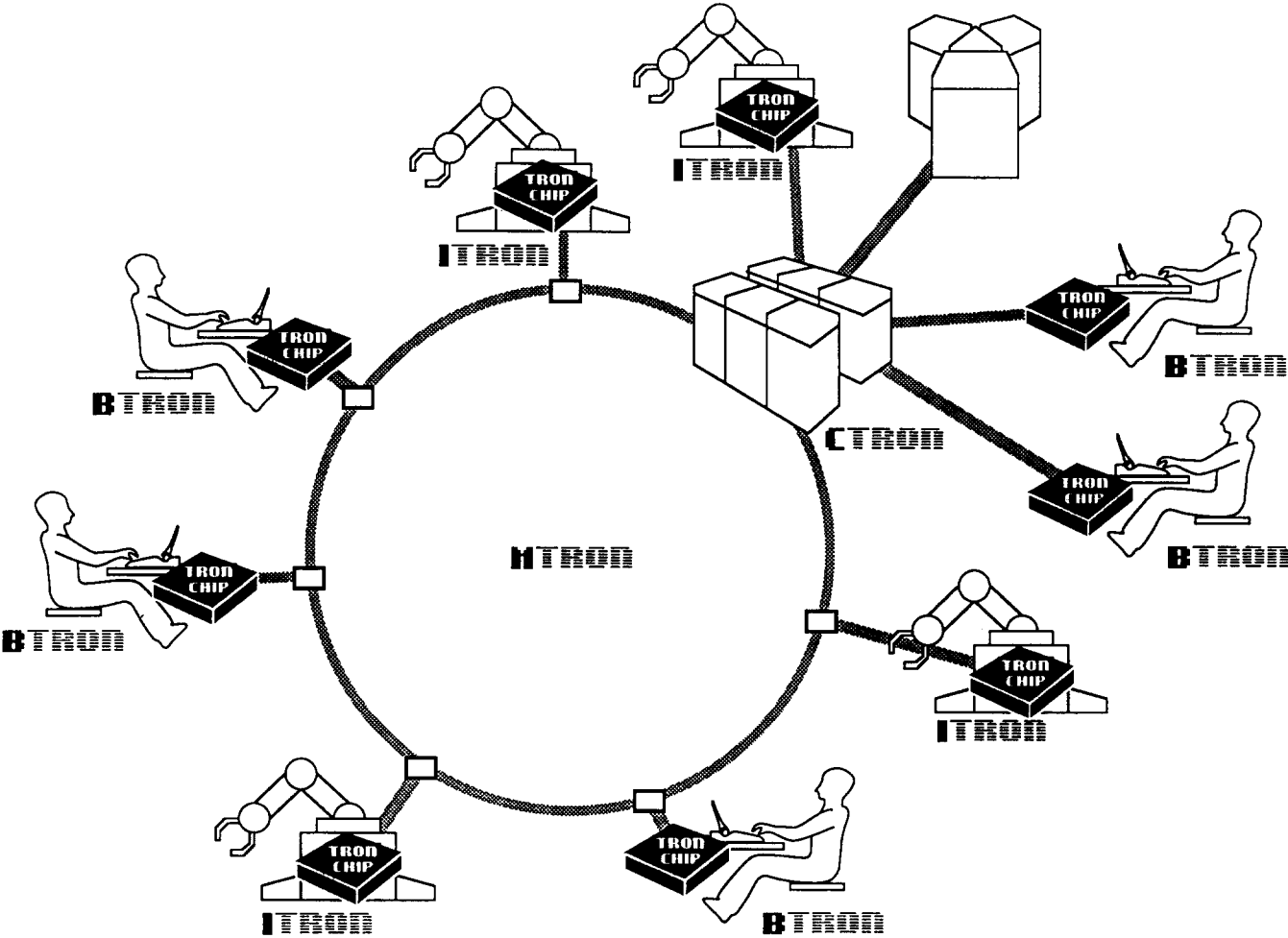


Figure 3. Overview of the TRON system architecture

Since this performance tuning is being undertaken with a clear picture of the total architecture at hand, the resulting system should be efficient, as well as easy to understand and maintain.

Open architecture

The value of a network as large as the HFDS is directly proportional to the number of components connected to it, and, hence, the greater the number the greater the value. However, this means that users should be able to connect and replace computer components easily in the network, something that is only possible if compatibility is assured for components produced by different manufacturers. To achieve this, the aim is to establish an 'open architecture'.

'Open architecture' in the TRON project means that the results of the project, in particular the set of interface specifications, will be made available at a nominal fee (to cover the costs of publishing, and so forth).

In making this information available 'for free', it is hoped that the TRON system architecture will be adopted by many manufacturers. The existence of well-defined standards is beneficial to both the manufacturers and the users, because the compatibility of the products based on open standards will expand the market as a whole and encourage suppliers and users alike to adopt the standards.

Weak standardization

When considering standardization it is very important to define exactly what must be standardized.

In the TRON project, the interfaces between the layers of the hierarchical architecture are standardized, but the TRON specifications do not spell out the method of implementation. As long as the interface standard is followed, different layers of different origins can be used together.

Allowing for variations in implementation is important for incorporating advanced technology, and allows the system to be adapted to various applications while assuring healthy competition between commercial implementors.

The TRON project standards are referred to as 'weak standards'. As this term implies, the TRON architecture is a general architecture and is not tied to any particular hardware or software. For example, the instruction set for the TRON VLSI CPU, known as instruction set processor (ISP), specifies the type of instructions, but not the hardware implementation.

The TRON project aims to establish an open market where many players can compete under the umbrella of weak standards. This contrasts with the present market situation where the necessity to maintain compatibility with existing systems can, and does, preclude the incorporation of new technology, even though the new technology brings new features.

THE TRON ARCHITECTURE

Organization of the TRON architecture

Before the HFDS can be achieved, its components must be built one step at a time. For this reason, a family of four

OSs has been defined in the TRON project.

The four OSs have been classified by what they communicate with — ITRON communicates with machines, BTRON communicates with human users, CTRON communicates with machines based on ITRON and BTRON, and MTRON communicates with the aggregate of these machines.

The OSs have been classified like this because emphasis has been placed on the realtime processing capability of computers. (This is why TRON stands for the realtime operating system nucleus.) For example, a typical application for ITRON will require a response time of the order of a millisecond. On the other hand, a human operator using a BTRON computer does not need as fast a response time. Moreover, when the nature of the tasks performed by the various OSs is considered, another difference becomes obvious. The kind of tasks handled on computers controlled by ITRON tend to be simple. However, the tasks handled on BTRON-based computers tend to be symbolic or graphical processing, which can be quite complex.

ITRON

ITRON is the OS architecture for embedded computers. It is a realtime, multitask OS specification that will be used in intelligent objects to control machines and to sense data from the outside world.

The major objective in the design of ITRON was to shorten the task dispatch time. The specification will work best with the TRON VLSI CPU, but ITRON-based OSs can also be implemented on existing VLSI CPUs.

ITRON has an extensive set of functions. For example, three different mechanisms are provided for synchronizing tasks. Although one mechanism is theoretically sufficient, the provision of three eases programming and tunes performance. Unnecessary functions can be removed by system regeneration procedures, which means that the resulting application size can be kept to the minimum.

A variation of ITRON, called μ ITRON, is currently being designed for single-chip microcomputers and 8-bit microcomputer chips with small RAMs and ROMs.

BTRON

BTRON is the OS system architecture and MMI specifications for workstations where man-machine interaction takes place (Figure 4). Human interaction with HFDS is via BTRON-based machines, so it is important that a consistent interface is provided on every BTRON computer. The *TRON Design Guideline* has been compiled for MMI, a set of design guidelines for manufactures to follow in implementing their MMIs, so that users can easily operate a wide variety of BTRON-based computers. Such standardization naturally decreases the difficulties users encounter when learning to operate a new machine.

In the design of BTRON, a data model was introduced called the real/virtual object model to represent and treat data in a consistent way. This data model can represent most of the useful data structures used on computers. In this model, an accumulation of data is referred to as a real object, which is referenced by a tag called virtual object. Just as the text figures in a conventional document can, a



Figure 4. An 80286-based computer that runs BTRON/286, an OS based on BTRON specifications. The machine shown has a special video processor unit (VPU) that makes it possible to superimpose video images from external sources on the screen

virtual object can be embedded in a real object as a standard unit of data.

The real/virtual object data model is also the basis for the TRON application databus (TAD), which is used to provide as much data compatibility as possible. In the HFDS, data exchange is important, and the real/virtual object model and TAD provide the basis for the smooth exchange of data over the network.

BTRON specifications call for multilingual processing, and include Japanese language processing as a part of this compatibility. This multilingual processing is expected to make it easy to adapt BTRON-based computers to many different languages.

The high-level functions provided in BTRON will reside in the external shell of BTRON. As a result, application programmers will be able to incorporate these functions easily in their applications, and it follows that the use of these functions will standardize the MMI and data handling on BTRON-based computers.

To optimize the MMI, a keyboard has been designed with a new layout for inputting Japanese-language characters, and the Dvorak system¹⁰ for inputting alphabetic characters (Figure 5). The layout for Japanese-language input was determined by analysing the frequency of individual characters in a large number of documents, which makes the resulting layout easier to use while

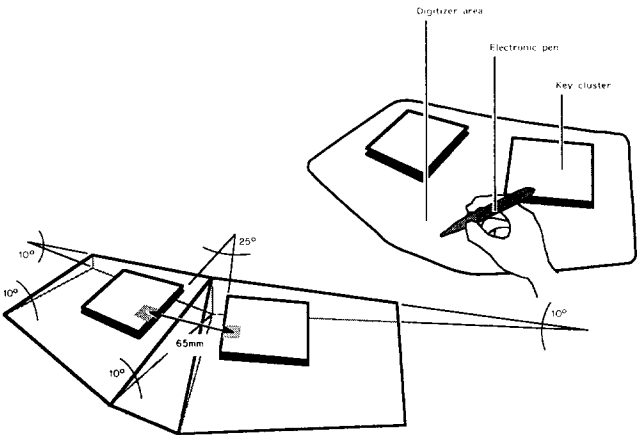


Figure 5. The TRON keyboard unit

causing less fatigue. A digitizing pen has also been specified as a standard pointing device, which can additionally be used for inputting drawings and hand-writing (Figure 6).

A subset version of BTRON specifications, called μ BTRON has also been designed for use on small, specialized machines¹¹.

CTRON

The HFDS cannot be realized with ITRON and BTRON alone. This is because specialized services and high-performance computing cannot be provided solely by using computers based on ITRON and BTRON. Some examples of services beyond the capability of ITRON and BTRON are large databases, numerical computation requiring supercomputers, and network services of various sorts. CTRON is used to run the specialized servers and mainframe computers connected to the HFDS that provide these services.

Since CTRON applications will be very specialized and varied in scope, the approach taken has been to build a specialized version of CTRON for each application by expanding the core set of functions (Figure 7).

MTRON

MTRON will hold the key to realizing the HFDS. It is the OS for the HFDS, namely, the large computer network consisting of a great number of computers that interact with each other in a complex way.

The day when every appliance and machine has a computer in it is still a long way off, and computer scientists have not paid much attention to the possibility that a large number of interconnected computers can bring about qualitative changes in the way computers are used. It is true that there have been some implementations of, and research on, computer systems with several tens of thousands of computers. However, the computing network in these cases is homogeneous, and the geographical span of the computers involved was a small room at the largest.

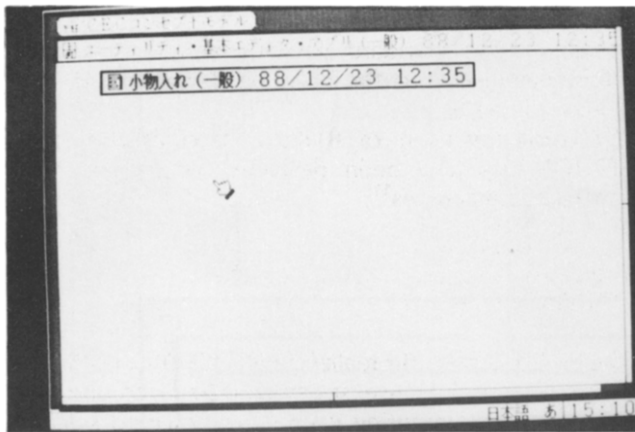
MTRON research will have more significance once the results of the other subprojects of the TRON project, such as ITRON, BTRON, and CTRON, become widely available. Research on MTRON is currently at the stage of defining an acceptable model of computing for the HFDS so that control methods can be devised.

RELATED TOPICS

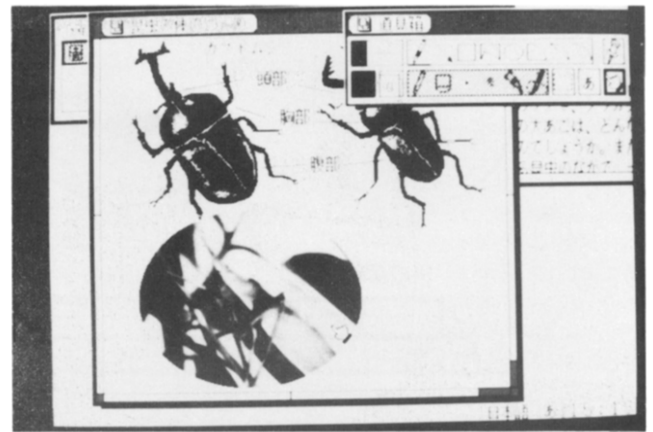
The TRON VLSI CPU

The ideal way to build the HFDS is to design a general-purpose microcomputer chip family and then use this in communication machines and intelligent objects. Since having a single underlying CPU is attractive, a subproject was begun to design a CPU, which resulted in the specification for the TRON VLSI CPU^{12,13}.

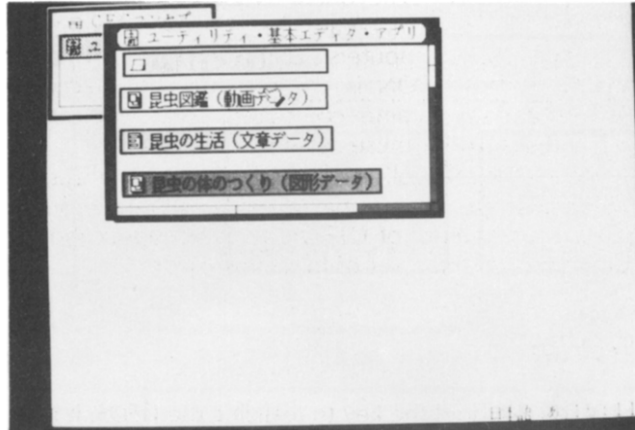
The architecture for this CPU has been designed with extensibility in mind to take advantage of future technology.



a



d



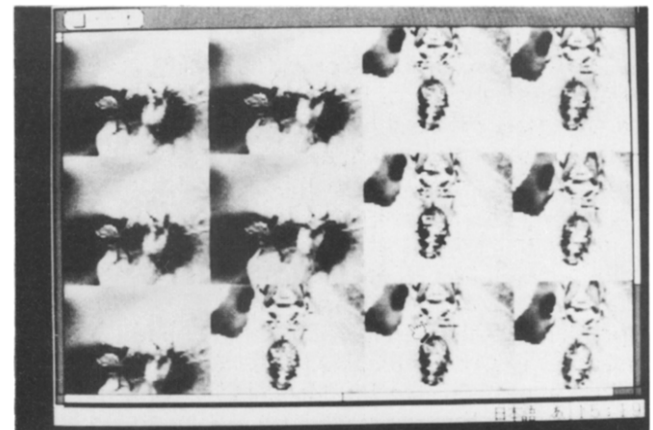
b



e



c



f

Figure 6. An example of the BTRON/286 computer screen. **a**, the initial screen after start up. **b**, the initial screen of the computer-aided instruction (CAI) demonstration program, **c** video, graphics, and text windows opened simultaneously on the screen, **d**, an irregularly shaped video window embedded in the graphics window, which was opened using system tools, **e**, full-screen moving video image, and **f**, multiple windows frozen together

It uses 32-bit addressing at present, but extension to 64-bit is already defined.

The TRON VLSI CPU has instructions to accelerate certain operations of the ITRON and BTRON OSs. In addition, its instruction format is optimized so that frequently executed instructions can be encoded compactly.

The CPU is offered in various forms, from low-cost versions to be used in intelligent objects, to high-performance versions for engineering workstations with virtual memory.

In early 1988, the first commercial implementation of the TRON VLSI CPU was announced. This was the GMICRO/200 (Figure 8a) by Hitachi Ltd.¹⁴, which supports virtual memory. Low-cost ASIC versions of the TRON VLSI CPU, the TX1 (Figure 8b) from Toshiba Corporation¹⁵ and the GMICRO/100 from Mitsubishi Electric Corporation¹⁶, and a high-performance version for engineering workstations, the GMICRO/300 from Fujitsu Ltd.¹⁷, have also been announced.

Since the design of the 32-bit version of the TRON VLSI CPU has been completed, work on the design of CHIP64,

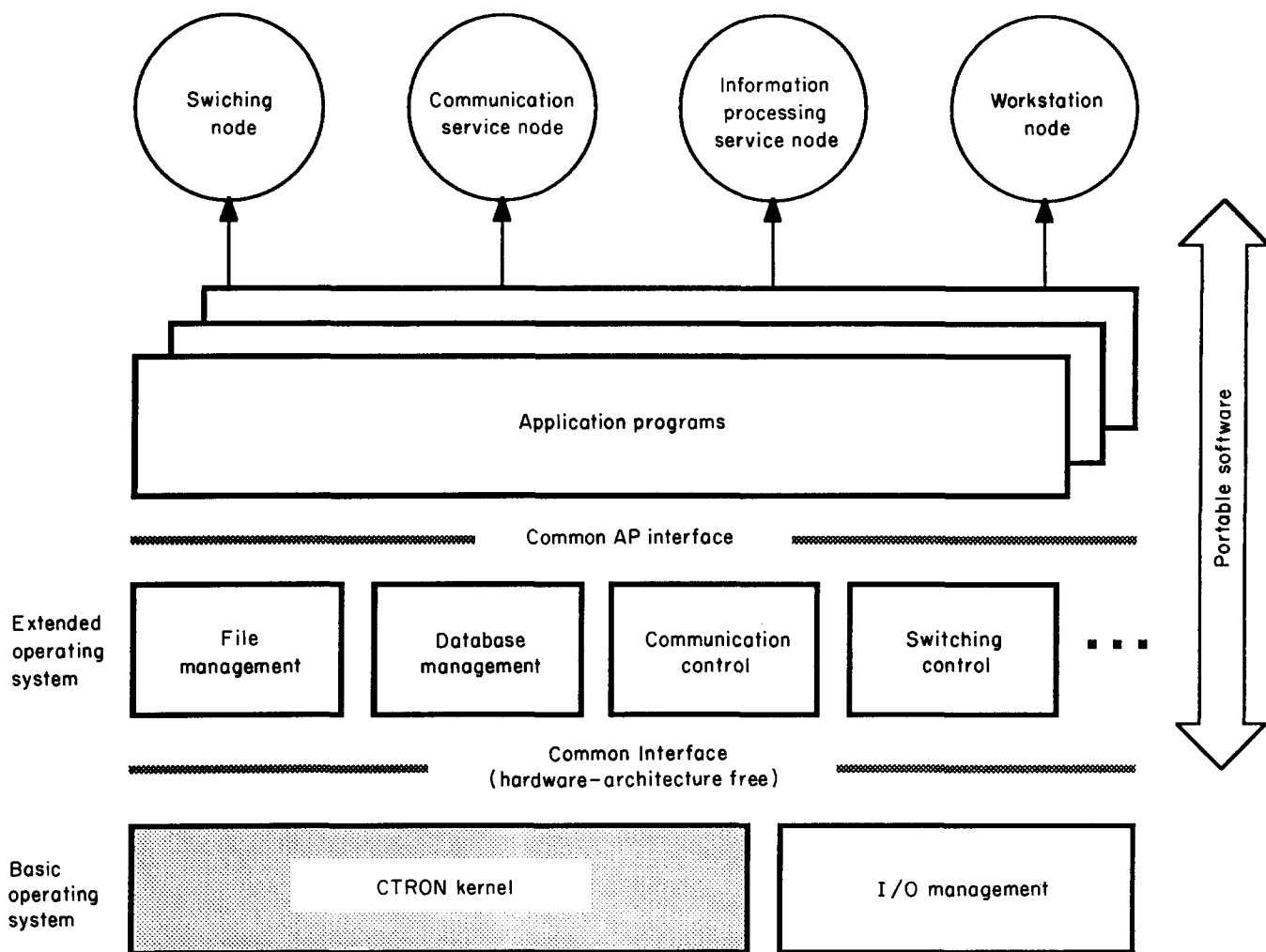


Figure 7. Software configuration of the CTRON-based system

the 64-bit version of the CPU has begun. Aside from the CPU itself, support chips such as a floating-point unit (FPU), a direct memory access controller (DMAC), an interrupt request controller (IRC), and a cache tag controller (CTC) are now being designed, and software support tools such as a compiler and an assembler are also being prepared. These support chips and software tools will become commercially available at about the same time as the CPU.

Multilingual capability

In the TRON project, TAD defines a framework for handling the characters of different languages. The intention is to support the character sets of all languages, and, what is more, to support the character sets of extinct languages that are of importance in academic research. The present lack of a standard code system for the character sets of extinct languages is creating confusion in the computer operations of research institutes. The standardization of the character code should eliminate this confusion.

In designing TAD to handle multiple character sets, three objectives were set. The first was the processing of multilingual documents. However, it is not enough just to assign a code to each character when multiple languages

are handled simultaneously. This is because words are written differently from one language to the next. In English, for example, words are written from left to right, but in Arabic it is the reverse. Line-breaking rules also differ according to language.

The second objective was to define a space-efficient code system. The goal of handling the character sets of the major languages of the world requires that codes be devised for approximately 100 000 characters. By using a simplistic code system, 3 byte would be required to distinguish all these characters, which would have been cumbersome. However, it was decided to optimize the character code system by taking advantage of the fact that most documents contain just one language, and that in most languages 1 byte is sufficient to label characters.

The final objective was to make application programs independent of a particular character set or language so that it would be easy to port programs from one country to another. Therefore, provisions have been made to absorb the differences in the handling of the character sets of the different languages at the OS level in order to free programs from this burden as much as possible.

Another ambitious plan involves automatic translation on TRON-based computers. The intention is to incorporate an automatic translator within the TRON OS so that the functions of machine translation can be used from within application programs. The aim is to use such functions to

translate program messages automatically when the message in the user's language is missing. This automatic translator could be used with an electronic mailer, which would filter incoming messages before presenting them to users. An intermediate representation of semantics that can be used as a universal language for translation purposes is being experimented with. One of the long-term objectives of the TRON project is a natural-language front-end processor using the same technology.

Some details of how multiple character sets from different languages are handled through TAD are given below.

Separation of rules

One factor central to the design effort was the compilation of the language-dependent rules governing data processing so that provisions could be made for them in TRON-based OSs. These rules must be constructed in a way that allows them to be switched on/off according to the language chosen for use.

Language-dependent algorithms are those for input and output (display and formatting) and sorting. The input algorithm translates the signals from input devices, such as the keyboard, into the particular characters of the language being used, and the forming algorithm determines how characters look and where they are placed in the output.

It was decided that the best way to handle language-dependent rules would be to combine the input, formatting, and sorting algorithms with other language-dependent processing rules in packages that can be changed as the language in use changes. Such packages make applications highly portable between countries.

Language specifier code

In order to switch from one language to another, language-specific codes are employed to indicate exactly which language is being used.

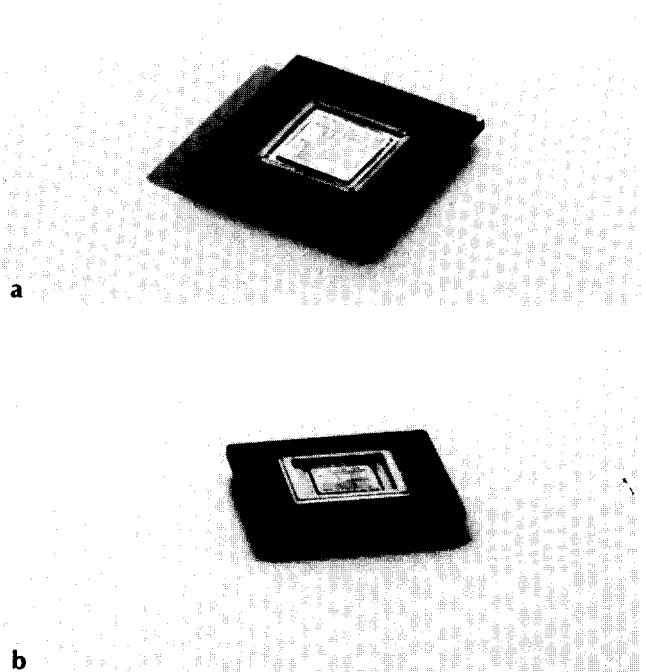


Figure 8. The TRON VLSI CPU. **a**, GMICRO/200 by Hitachi Ltd, and **b**, TX1 by Toshiba Corporation

Separation of display and storage

Textual data are separated into two categories: display and storage. Display refers to the way the data are represented on a piece of paper or a computer display, while storage refers to the way the data are stored. In the TRON system, display can change even while the underlying storage remains the same. For example, when sophisticated formatting rules are employed, the letter 'f' followed by 'i' will be combined into a ligature when printed. Therefore, in TAD, display is dynamically generated from storage according to the rules being employed at the time.

Hierarchy in the language environment

The TAD language-specific environment is comprised of four hierarchical layers: Language, Group, Script, and Font (Figure 9). Language is the layer where the notion of 'natural language' is directly handled. The Group layer is for handling the stored text as character code. Script is a family of similar characters. It has been determined that 39 Scripts can cover the major character sets used across the world. Font means the recognized characteristic shape of a particular drawing style of characters, such as Helvetica and Courier for English, and Mincho for Japanese. For details, interested readers are referred to Reference 18.

SUMMARY

It may seem too ambitious to some to overhaul and rebuild the computer systems of today. However, efforts to maintain the *status quo* without incorporating the changes that have become necessary with the passage of time have complicated many aspects of today's computer systems. As a result, it is now impossible to move on to the next stage of computer use, the HFDS, without negotiating these complications.

The best way to do this, the author believes, is to rebuild and refine the current computer architecture by using available and predicted microelectronics technology, while, at the same time, paying due consideration to the knowledge of computer science gained so far. The short-term objective of the TRON project is to make many systems based on the TRON architecture available in the 1990s. Combining these systems into a cooperative network that can serve as a truly functional HFDS is the long-term goal.

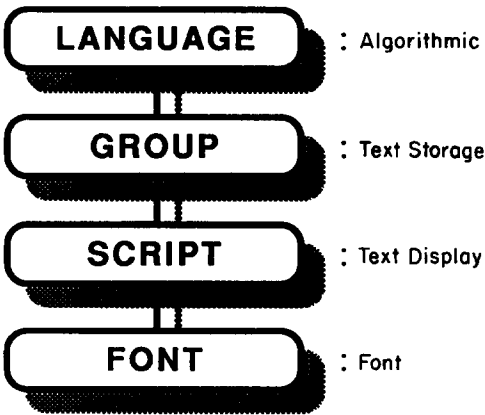
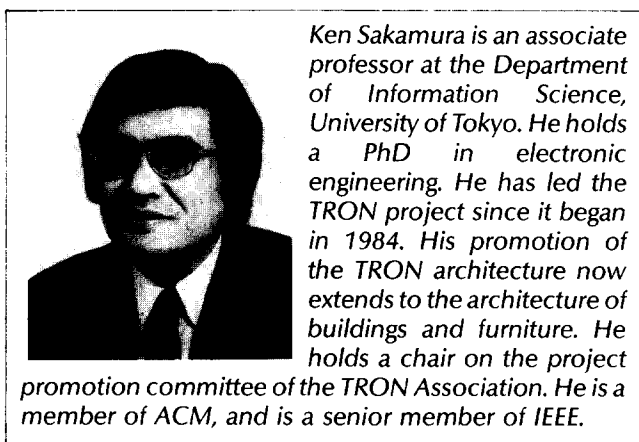


Figure 9. Layers in the TAD language-specific environment

REFERENCES

Only representative papers in English are given. Interested readers are advised to read *TRON Project 1987*, which is the proceedings of the Third TRON Project Symposium held in 1987, and *TRON Project 1988*, which is the proceedings of the Fifth TRON Project Symposium held in 1988.

- 1 **Sakamura, K** 'The objectives of the TRON project' in *TRON Project 1987* Springer-Verlag, Heidelberg, FRG (1987) pp 3-16
- 2 **Sakamura, K** 'The TRON Project' *IEEE Micro* Vol 7 No 2 (April 1987) pp 8-16
- 3 **Monden, H** 'Introduction to ITRON, the industry-oriented operating system' *IEEE Micro* Vol 7 No 2 (April 1987) pp 45-52
- 4 **Sakamura, K** 'ITRON: An overview' in *TRON Project 1987* Springer-Verlag, Heidelberg, FRG (1987) pp 29-34
- 5 **Sakamura, K** 'BTRON: An overview' in *TRON Project 1987* Springer-Verlag, Heidelberg, FRG (1987) pp 75-82
- 6 **Sakamura, K** 'BTRON: The business-oriented operating system' *IEEE Micro* Vol 7 No 2 (April 1987) pp 53-65
- 7 **Ohkubo, T et al.** 'Configuration of the CTRON kernel' *IEEE Micro* Vol 7 No 2 (April 1987) pp 33-44
- 8 **Sakamura, K** 'CTRON: An Overview' in *TRON Project 1987* Springer-Verlag, Heidelberg, FRG (1987) pp 153-156
- 9 **Wasano, T et al.** 'Design of CTRON' in *TRON Project 1987* Springer-Verlag, Heidelberg, FRG (1987) pp 157-172
- 10 **Yamada, H** 'A historical study of typewriters and typing methods: from the position of planning Japanese parallels' *J. Info. Processing* Vol 2 No 4 (1980) pp 175-202
- 11 **Sakamura, K et al.** 'Applying the μ BTRON bus to a music Lan' *IEEE Micro* Vol 8 No 2 (April 1988) pp 60-66
- 12 **Sakamura, K** 'TRON VLSI CPU: Concepts and architecture' in *TRON Project 1987* Springer-Verlag, Heidelberg, FRG (1987) pp 199-238
- 13 **Sakamura, K** 'Architecture of the TRON VLSI CPU' *IEEE Micro* Vol 7 No 2 (April 1987) pp 17-32
- 14 **Inayoshi, H et al.** 'Realization of Gmicro/200' *IEEE Micro* Vol 8 No 2 (April 1988) pp 12-21
- 15 **Miyata, M et al.** 'The TX1 32-bit microprocessor: performance analysis and debugging support' *IEEE Micro* Vol 8 No 2 (April 1988) pp 37-46
- 16 **Tomisawa, Q et al.** 'Design Considerations of the GMICRO/100' *TRON Project 1987* Springer-Verlag (1987) pp 249-258
- 17 **Itoh, M** 'Architecture characteristics of GMICRO/300' *TRON Project 1987* Springer-Verlag (1987) pp 273-280
- 18 **Sakamura, K** 'Multi-language character sets handling in TAD 1' *TRON Project 1987* Springer-Verlag, Heidelberg, FRG (1987) pp 97-112



Ken Sakamura is an associate professor at the Department of Information Science, University of Tokyo. He holds a PhD in electronic engineering. He has led the TRON project since it began in 1984. His promotion of the TRON architecture now extends to the architecture of buildings and furniture. He holds a chair on the project

promotion committee of the TRON Association. He is a member of ACM, and is a senior member of IEEE.