THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 62, No. 1, January 1983
Printed in U.S.A.

The 3B20D Processor & DMERT Operating System:

Overview and Architecture of the 3B20D Processor

By W. N. TOY and L. E. GALLAHER

(Manuscript received March 8, 1982)

The 3B20D Processor is designed to meet a broad range of telecommunication applications. New features such as memory management are incorporated into its design to support a modern operating system. Hardware supports are provided to efficiently execute a high-level language. A major design objective is to meet the stringent reliability requirements of electronic switching systems. The proven technique of self-checking duplex operation forms the basic architecture structure of the 3B20D Processor.

I. INTRODUCTION

The 3B20D Processor is the first member of a family of processors designed for a broad range of Bell System applications. Its development is a natural outgrowth of the continued need for high availability and real-time control of Electronic Switching Systems (ESSs),[1-3] including existing as well as new telecommunication applications. With the rapid growth of integrated-circuit technology, the processor architecture is evolving to include as many features as possible to significantly reduce software development and maintenance costs.

The 3B20D Processor architecture takes advantage of the LSI technology to expand its functionality and yet maintain a high reliability standard. Some of the design goals are to:

(*i*) Achieve highest performance that is consistent with system cost, e.g., provide hardware facilities such as data cache, high-speed interrupt stack, address-translation cache, and microprogramming for critical functions that require too much time in software.

(*ii*) Reduce software complexity, e.g., provide a modern real-time

operating system to manage system resources, thereby creating a more useful and more reliable programming environment for the user.

(*iii*) Reduce programming effort, e.g., provide both an efficient high-level language, such as the C language,[4] and a comprehensive set of software development tools.

(*iv*) Provide a high level of reliability and fault tolerance, e.g., built-in error-detection and correction codes, recovery features, and fault diagnostics.

(*v*) Provide features for system integrity and security, e.g., memory management protection and privileged instructions.

These goals are considered from the viewpoints of both hardware and software architecture in order to realize the most cost effective system for a wide spectrum of applications. Much of the development effort has been directed to achieve these goals.

II. GENERAL DESCRIPTION

High availability is one of the major objectives in the design of the 3B20D Processor. The successful deployment and field operation of many ESS systems have demonstrated the simplicity and robustness of duplex configuration in meeting the ESS reliability requirements.[5] Hence, duplex configuration forms the basic structure for both the hardware and software architecture. Experience gained in the design and field operation of the No. 3A Processor provided valuable input for the 3B20D Processor design.[6]

The 3B20D Processor employs concurrent self-checking design. Extensive checking hardware is incorporated as an integral part of the processor. Faults occurring during normal operation are quickly discovered by detection hardware. This eliminates the need both to run the standby processor in the synchronous mode of operation and to run the fault-recognition program to identify the defective unit when a mismatch occurs. Self-checking implementation simplifies the maintenance program. Reconfiguration into a working system is immediate, without extensive diagnostic programs to determine which subsystem unit contains the fault. Furthermore, the self-checking design will permit more straightforward expansion from simplex to duplex, or multiple processor arrangements.

2.1 Duplex configuration[7]

Figure 1 shows the general block diagram of the 3B20D Processor. The Central Control (CC), the memory, and the I/O disk system are duplicated and grouped as a switchable entity although each CC can access each disk system. The quantity of equipment within the switchable block is small enough to meet the reliability requirement; therefore, the complexity of a recovery program to manage additional

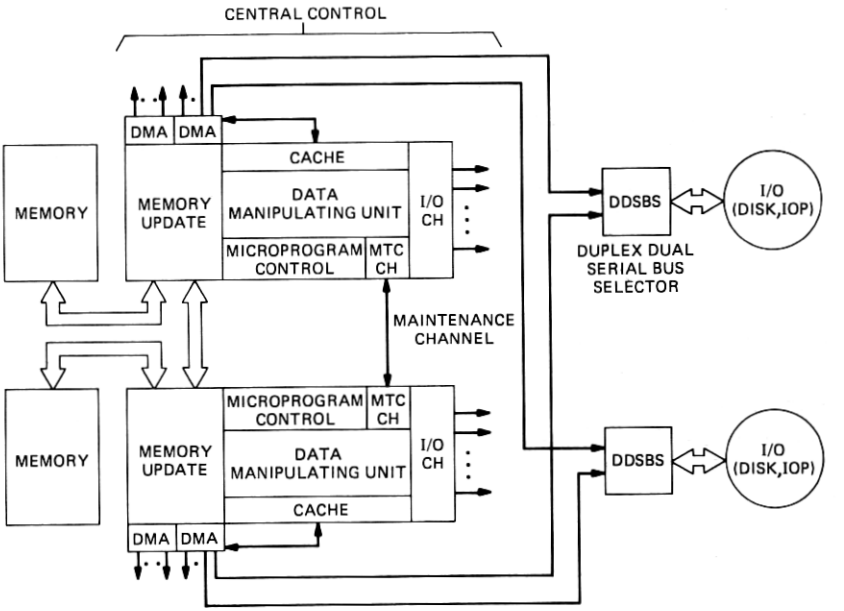

Fig. 1—General block diagram of the 3B20 Processor.

working states is avoided. Although each CC has direct access to both disk systems, this capability is used mainly to provide a valid data source for memory reload under trouble conditions. The processors are not run in the synchronous and match mode of operation as is done in early systems.[1-3] However, both stores (on-line and standby) are kept up-to-date by the memory update hardware concurrent with instruction execution. This is achieved by having the on-line memory-update circuit write into both memories simultaneously when memory data are written by the CC. Under trouble conditions, when the control is switched to the standby processor, its memory will contain up-to-date information without performing a complete transfer from one processor to another. The Direct Memory Access (DMA) circuits interface directly with the memory update circuit in order to have access to both memories. A DMA write also updates the standby memory. Communication between the DMA and the peripheral devices is accomplished by using a high-speed dual-serial channel (DSCH). The duplex dual-serial bus selector (DDSBS) allows both processors to access a single I/O device. For maintenance purposes, the duplex 3B central controls are interconnected via the Maintenance (MTC) channel. This high-speed serial path provides diagnostic access at the microcode level. It has the capability of transmitting a stream of microinstructions to exercise the processor from the on-line processor or from an external unit for diagnostic purposes.

2.2 Peripheral devices[8,9]

A broad range of general-purpose peripheral devices is provided for the 3B20D Processor system. High reliability and maintainability continues to be the design philosophy of the 3B20D peripheral system.

The critical components are duplicated and the software ensures that valid data sources are maintained. The DDSBSs permit controlled switching of a working standby device for a faulty on-line device when duplication of peripheral devices is needed. Some major peripheral devices developed for the 3B20D system are:

(*i*) Moving head disk system—The disk system provides a reliable and flexible mass storage medium for program and data. A backup copy of system programs and critical parameters can be reloaded quickly in the event of a duplex main-store failure. The disk system comprises the Disk File Controller (DFC) and the Moving Head Disk drive (MHD). The DFC interprets and executes commands from the processor to cause information transfer from and to the MHD. Each DFC occupies 1 of 128 channel slots and supports up to 16 MHD drives which are available in 80 and 300 megabyte sizes.

(*ii*) I/O processor (IOP)—The IOP provides the control for a wide range of data-link facilities and is the most flexible of the family of devices. An IOP supports up to 16 Peripheral Controllers (PCs) with each being a microprocessor-based controller programmed to handle a specific terminal or device. For example, one type of PC is the Line Controller (LC); each LC can support up to four independent lines (data links or terminals).

(*iii*) Magnetic tape system—The tape drive accepts the industry-standard (IBM compatible) 9-track tapes at a density of 1600 bits per inch. The tape controller is derived from the basic PC and occupies one of the 16 slots of the IOP.

(*iv*) Scanner/signal distributor (SC/SD)—This device is useful in monitoring and controlling power, equipment states, environment conditions, etc. The SC/SD circuit board provides 48 scan points and 32 signal-distributer points. It occupies one of the PC slots of the IOP. When an IOP is fully equipped with 16 SC/SD circuit packs, a total of 768 scan points and 512 signal-distributor points are provided.

III. SOFTWARE SUPPORT FEATURES[10]

The high cost of designing, updating, and maintaining software dominates the cost of producing computer systems. Considerable attention has been focused on providing various types of support, i.e., high-level language, operating system, and software test, in the development of 3B20D Processor. The combined software and hardware effort has yielded an integrated and cost-effective system.

3.1 High-level language support

The most common approach to increasing software productivity and reducing software maintenance cost is the extensive use of a high-level language suitable for the application. The design of the 3B20D Processor instruction set was based on the fact that C language programs would dominate the programming environment. C is a general-purpose programming language featuring economy of expression, modern control flow and data structures, and a rich set of operators. Many studies were directed to measure and determine the characteristics of a large, diverse sample of C programs. Based on the result of these studies, the instruction set was optimized to be space and time efficient for compiled C programs. Some features provided for the instruction set are concerned with:

(*i*) Symmetrical resources
(*ii*) Addressing modes
(*iii*) Address manipulation
(*iv*) Flexible data structure
(*v*) Stack instructions
(*vi*) Procedural instructions.

From the compiler's viewpoint, the most important attribute of a processor instruction set is regularity. It is the key feature needed to abstract the various processor resources for uniform treatment by the complier. The 3B20D instruction set includes a wide range of address modes—i.e., indexing, direct, indirect—covering various data structures. The treatment of the addressing modes, applied identically to all data types (bytes, half words, full words, and instructions) without exceptions, makes it possible to compile compact and efficient codes.

The subroutine is one of the most important concepts in software. The principal idea in modular, structured programming is the partitioning of large programs into many small, understandable procedures or subroutines. Efficient instructions have been provided to handle subroutine entry and exit in addition to stack manipulation.

3.2 Operating system support

Higher productivity in application programming is made possible by the high-level, simplified facilities provided by the operating system. The Duplex Multienvironment Real Time (DMERT) operating system for the 3B20D Processor is a general manager of processor, memory, input/output, and software processes. The functional description of DMERT will be described in more detail in the next section. This section describes the hardware that has been incorporated into the design to reduce the overhead of the operating system.

As previously indicated, a high-speed address translation cache

memory called the Address Translation Buffer (ATB) is provided to reduce the overhead associated with the address translation function.

Context switching is necessary upon interrupt. A memory stack is provided to facilitate the saving and restoring of the hardware context. In the 3B20D Processor, a local high-speed 8K-byte RAM is provided for this function. The addressing of the stack is part of the kernel virtual address space and has been assigned a fixed segment number and pages 0 to 3. Whenever the kernel virtual address falls into this range, the store operation is directed to the high-speed RAM; otherwise, the virtual address is translated by the ATB and pointed to the main memory. The combination of fixed mapping by special circuit and dynamic address translation by ATB allows the high-speed stack to be extended into the main memory when the use of the high-speed RAM is exceeded.

3.3 Software test support

The software test facility is an option provided at both the microprogram level and the macroprogram level. The Microlevel Test Set (MLTS) is attached to the microcontrol section of the central control. It has the capability of direct access to a support computer system for assembling and loading the writable microstore through the MLTS. The primary purpose of the MLTS used in the development of the 3B20D Processor is for initial debugging and troubleshooting the processor core hardware and, subsequently, the microprogram sequences. Features incorporated into the MLTS allow stepping, freezing, examining, and tracing the execution of a microprogram sequence.

The Utility Circuit (UC), on the other hand, provides a similar set of facilities, except at the macroprogram level for software debugging and troubleshooting. The UC and its associated software form an extensive Test Utility System (TUS) for software testing. A small number of matchers are incorporated into the UC for the tracing and monitoring of a variety of system conditions so that a programmer can observe and follow the execution of a program sequence. Much of the program debugging can take place in real time concurrent with program execution. The UC thus directly extracts and records information such as transfer trace from the internal data buses, thereby "capturing" the history of the machine while it is running at normal speed.

IV. DMERT OPERATING SYSTEM[11]

DMERT is a general-purpose operating system. It is structured as cooperating processes that provide different levels of virtual machines. Protection is built into the structure, preventing these virtual machines from destructively interfering with each other. For processes to cooperate in accomplishing their task, DMERT provides a rich set of

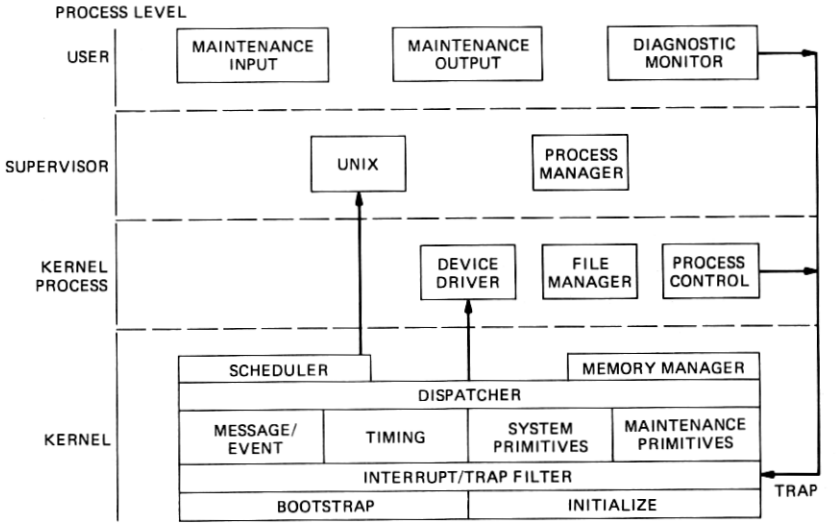

Fig. 2—The DMERT structure.

interprocess communication and synchronization mechanisms, including messages, events, process ports, interprocess traps, and shared memory. By means of these communication primitives, system services can be provided easily to a requesting process.

4.1 Multilayered virtual machines

DMERT provides four levels of virtual machines. They are the kernel, kernel process, supervisor, and user (Fig. 2). Successive levels put additional restrictions on access rights of system resources. This helps free programmers from the details of the physical machine. The higher level may take advantage of services provided by the lower levels.

(*i*) Kernel—The kernel provides the most primitive virtual machine. Programs at this low level are closest to the hardware. They directly control the system hardware and do not have access to other system functions. The kernel services are primitive, yet are efficient in their execution. The user cannot introduce code at this level.

(*ii*) Kernel process—Kernel processes are also strongly hardware related and are structured to provide time-critical processing in a real-time environment. DMERT uses this level for the file manager and device drivers. In addition, there are several special processes that provide scheduling, memory management, and other services. Users may add a kernel process that communicates efficiently with other kernel processes making efficient use of their services.

(*iii*) Supervisor—This third level is comprised of programs that are

generally hardware independent. These processes can use all the services provided by the kernel and its processes. The process manager is implemented in this level. So is a *UNIX** operating system supervisor, which provides time-shared usage of processor hardware through services provided by a scheduler. In general, supervisor segments are not locked in memory and can be moved out onto the disk. Consequently, supervisor processes take a much longer time to dispatch than do kernel processes.

(*iv*) User—All applications programs for which time is not a critical factor are written at the user level. The user process is linked directly with a *UNIX* operating system supervisor in which DMERT treats the supervisor/user pair as a single process. These user programs only see the software environment and services of the supervisor and are well protected from other user processes. Since user-level code must interface its own supervisor to use the lower-level primitives, additional overhead is added for user-level processes.

4.2 Multiple environment support

An application may add code at the level of the kernel process, the supervisor process, or the user. The multilevel structure makes DMERT a flexible system for real-time use. In general, the higher the level, the more services that are available to an application; the lower the level, the more efficient is the program execution. This level structure of virtual machines permits DMERT to manage real-time applications, while at the same time providing the flexibility of a time-sharing system for background tasks. Figure 3 shows how telephone switching software can be allocated to these different levels. By means of this hierarchical execution-level structure, application programs can customize their control and distribution of real time.

The portion of real time that is not utilized by the kernel or kernel processes is time shared among supervisor and user processes. Deferrable jobs such as traffic reports, recent changes, and diagnostics, as shown in Fig. 3, are implemented at the highest user level. The DMERT architecture thus simultaneously supports both a real-time and a time-sharing environment to fully utilize physical resources in the most efficient manner.

V. MAINTENANCE FEATURES

Increased support in this area is most appropriate to facilitate a more reliable and more maintainable system, thereby reducing the maintenance cost. For real-time applications, as in the Electronic Switching Systems (ESS), high availability and uninterrupted opera-

* Trademark of Bell Laboratories.

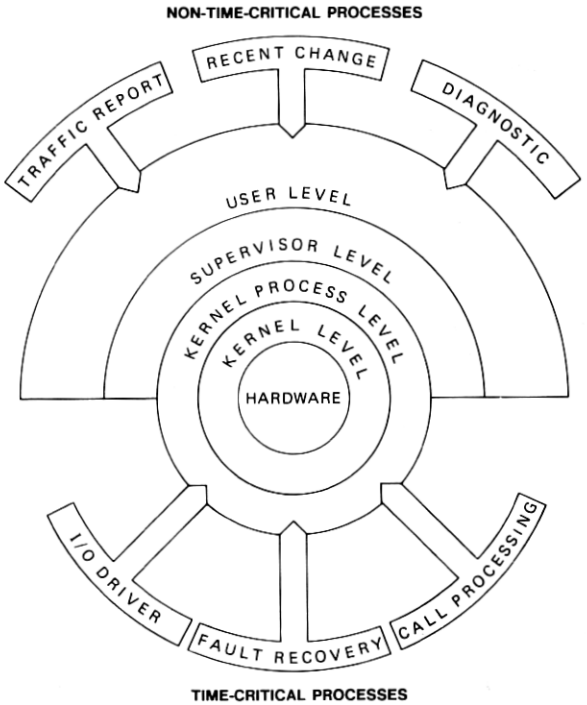

Fig. 3—Example of DMERT multienvironment structure.

tion is essential. This requires the system to function correctly even when a fault is present and maintenance is being performed. The 3B20D is designed to meet the ESS standard so that the expected amount of accumulated processor downtime does not exceed an average of 2 minutes per system per year.[5] Software and hardware are designed to function jointly to ensure that reliability objectives are met. Software features include such components as fault recovery programs, audits, and diagnostics.[12] Hardware features include redundant processors, error-detection circuits, maintenance-access controls, and diagnostic microcode. These components contribute to the effective maintenance design.

VI. SUMMARY

The 3B20D Processor is a high-availability system capable of supporting a broad spectrum of applications. A comprehensive set of software tools and facilities is provided to improve programming productivity and also to reduce the cost of software development and maintenance. The hardware architecture is designed to efficiently support high-level languages, particularly the C language.

A wide range of general-purpose peripherals have been provided with the 3B20D Processor. Some of these are the moving head-disk system, magnetic tape system, high-speed printer, scanner and signal distributor, and data terminals.

DMERT is the duplex multienvironment operating system for the 3B20D Processor. It was designed concurrently with the hardware to meet the high-availability demands of real-time switching and telecommunication systems. DMERT provides a set of procedures that enables users to efficiently share the 3B20D Processor and the physical resources such as processor time, storage space, and peripheral devices. The multiple environment permits time-critical code to coexist with time-shared software as background tasks.

An important provision in the 3B20D Processor is a complete set of maintenance facilities, from error detection through fault recovery and diagnostics. Approximately 30 percent of the internal central control logic is devoted to self-checking. This allows concurrent error detection and immediate recovery. The combined hardware and software features give an integrated package of maintenance facilities to meet the high ESS reliability requirements.

VII. ACKNOWLEDGMENTS

The design of the 3B20D Processor was accomplished through the combined efforts of many designers in Bell Laboratories and Western Electric. The authors wish to acknowledge the leadership of R. J. Watters who helped direct the early development of the 3B20D Processor system, and the design efforts of C. W. Hoffner, D. E. Gearhart, C. M. Narayanan, D. J. Matter, and D. J. McGraw.

REFERENCES

1. "No. 1 ESS," B.S.T.J., *43*, No. 5 (September 1964).
2. "No. 2 ESS," B.S.T.J., *48*, No. 8 (October 1969).
3. "No. 1A Processor," B.S.T.J., *56*, No. 2 (February 1977).
4. B. W. Kernighan and D. M. Ritchie, *The C Programming Language*, Englewood Cliffs, NJ: Prentice-Hall, 1978.
5. W. N. Toy, "Fault Tolerant Design of ESS Processors," *Proc. IEEE*, *66* (October 1978), pp. 1126–45.
6. T. F. Storey, "Design of a Microprogram Control for a Processor in an Electronic-Switching System," B.S.T.J., *55*, (February 1976), pp. 183–232.
7. M. W. Rolund, J. T. Beckett, and D. A. Harms, "3B20D Central Processing Unit," B.S.T.J., this issue.
8. A. H. Budlong and F. W. Wendland, "3B20D Input/Output System," B.S.T.J., this issue.
9. R. E. Haglund and L. D. Peterson, "3B20D File Memory System," B.S.T.J., this issue.
10. B. R. Rowland and R. J. Welsch, "Software Development System," B.S.T.J., this issue.
11. M. E. Grzelakowski, J. H. Campbell, and M. R. Dubman, "DMERT Operating System," B.S.T.J., this issue.
12. R. C. Hansen, R. W. Peterson, and N. O. Whittington, "Fault Detection and Recovery," B.S.T.J., this issue.