THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 62, No. 3, March 1983
Printed in U.S.A.

Traffic Service Position System No. 1B:

System Description

By N. X. DeLESSIO and N. A. MARTELLOTTO

(Manuscript received June 30, 1982)

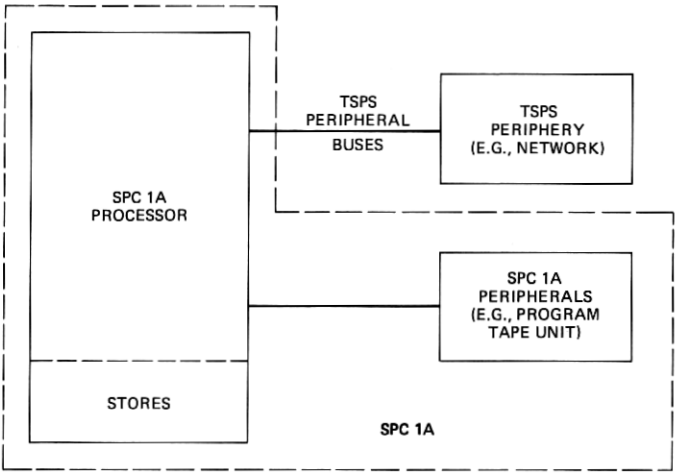
The Traffic Service Position System No. 1B (TSPS No. 1B) is the first field application of the Bell System's new 3B20 Duplex Processor (3B20D) in the emulation mode. The 3B20D Processor replaces the existing Stored Program Control No. 1A (SPC 1A), while retaining the existing TSPS periphery and software. A key factor is the ability to switch between the emulated software and the 3B20D native software within a single process with a single instruction. This allows the flexibility of adding software in either environment as appropriate.

I. INTRODUCTION

The 3B20 Duplex Processor (3B20D), with its associated Duplex Multi-Environment Real-Time (DMERT) operating system, meets the objectives of Traffic Service Position System No. 1B (TSPS No. 1B).[1-3] The 3B20D Processor is significantly faster than the Stored Program Control No. 1A (SPC 1A) and provides the required increase in processor capability. The ability of the 3B20D Processor to emulate allows the retention of most of the existing TSPS No. 1 software and peripheral hardware. In addition, the 3B20D Processor consumes much less energy and is significantly smaller in physical size than the SPC 1A.

II. SYSTEM STRUCTURE

The 3B20D Processor replaces the existing SPC 1A (Fig. 1) while retaining the existing TSPS No. 1 periphery and—through emulation—preserving the existing TSPS software. This software preservation is accomplished by defining (through microcode) one of the four



SPC – STORED PROGRAM CONTROL
TSPS – TRAFFIC SERVICE POSITION SYSTEM

Fig. 1—Existing TSPS system structure.

3B20D instruction sets to be that of the SPC 1A, thus emulating the old processor and allowing existing TSPS software to be transported to the 3B20D Processor almost intact. The ability exists to switch between the emulated instruction set and the 3B20D native instruction set within a single process with a single instruction. This allows the flexibility of adding new software into either environment, as appropriate. For example, some new TSPS No. 1B software, generated to take advantage of the new disk capability and to provide a new interface to maintenance personnel, is written in the C programming language,[4] which compiles into 3B20D native-mode assembly language. Both emulated and native-mode software are run under the DMERT operating system, allowing operating system services to be available to both forms of software. The emulated SPC 1A assembly language code is structured as a single process executing under DMERT.

The existing TSPS periphery is retained and interfaced with the 3B20D Processor through the use of a Peripheral System Interface (PSI) circuit. This unit is designed to interface the TSPS peripheral buses with the 3B20D Central Control Input/Output (CCIO) bus via an Application Channel Interface (ACHI). The PSI duplicates the signals, timing, and error checking of the SPC 1A. This enables TSPS peripheral units to remain unchanged. Future hardware can be added to the existing TSPS peripheral buses or can utilize the I/O processor of the 3B20D Processor to off-load the main processor and to provide fast block data transfers through direct memory access.

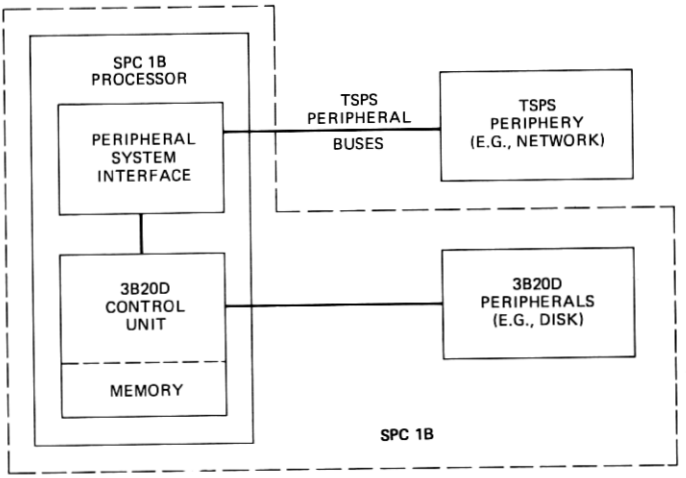
The combination of the 3B20D, PSI, and microcode required to

emulate the SPC 1A is called the SPC 1B (Fig. 2) and is the entity that replaces the SPC 1A. Because the processor replacement is economically attractive both for new installations and for retrofits, techniques have been developed to replace an SPC 1A with an SPC 1B in an in-service office. The resultant system provides significant increases in call processing and main memory capacities, allows the preservation and future growth of existing software and peripheral hardware, and adds modern software and hardware architectures to facilitate future feature introduction.

III. PROCESSOR HARDWARE DESCRIPTION

The 3B20D Processor has been developed as the first member of a family of processors designed for a broad range of Bell System applications. The DMERT operating system provides a comprehensive set of functions associated with management of system resources such as the real-time, memory, input/output, and software processes.

The control unit of the 3B20D (Fig. 3) uses a 32-bit architecture throughout, including the memory buses to main store and an 8K-byte cache. Extensive self-checking logic is employed to ensure immediate detection of errors, thus supporting quick and graceful recovery measures. Hamming correction of all single-bit errors and detection of all double-bit errors are performed by the main memory controller. In addition, data parity is checked on every refresh operation required for the dynamic random access memories (RAMs) used in main



SPC – STORED PROGRAM CONTROL
TSPS – TRAFFIC SERVICE POSITION SYSTEM

Fig. 2—TSPS No. 1B system structure.

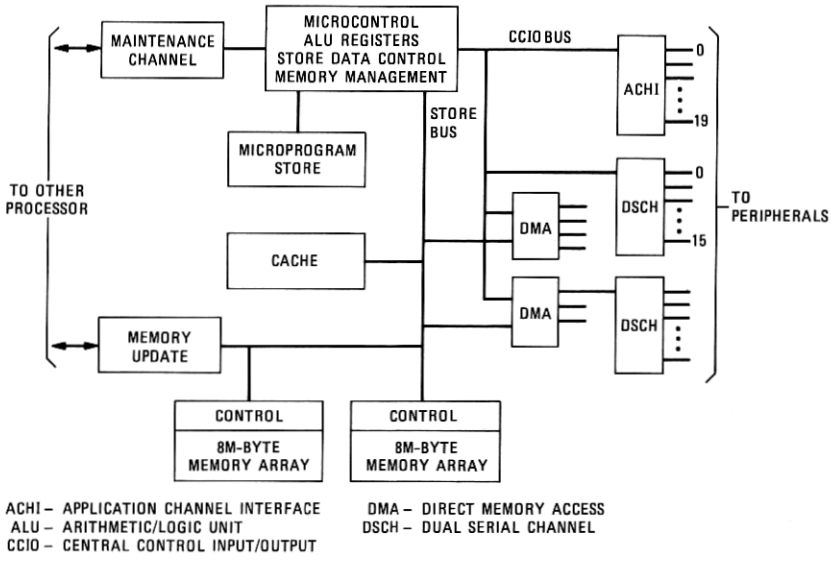


Fig. 3—3B Processor block diagram.

memory, thus ensuring that even infrequently used addresses are periodically checked for integrity. The 3B20D Processor uses a 24-bit virtual address, which is converted to a 24-bit physical address using a paged segmentation scheme. The 16M-byte address space is divided into 128 segments, each having up to 64 pages of 2K bytes each. Memory protection can be provided on either a segment or a page basis. Physical memory is growable in 512K-byte increments, to a maximum of 16M bytes. The main-memory access time is 525 nanoseconds, while the cache access time is 250 nanoseconds. Memory communication provides a byte-addressing capability with byte, half-word, full-word (32-bit), and move block options as part of the instruction repertoire. A memory management unit performs virtual-to-physical address mapping and main-store access protection. A high-speed, two-way, set-associative memory called the Address Translation Buffer (ATB) is provided to reduce the overhead associated with the address translation function. The ATB is divided into eight sections that are assigned to processes by software.

The Central Control (CC) is microprogrammable with the capability of executing a variety of instruction sets. Up to four instruction sets can be selected dynamically. The microstore uses a 64-bit word length with up to 16K words of high-speed, bipolar programmable read-only memory (PROM) or RAM available. A variable microcycle ranging from 150 to 300 nanoseconds is employed to optimize execution times. The native instruction set of the 3B20D Processor was designed to be

compatible with the C programming language. It optimizes the execution and memory-space utilization of the language while including instruction-level support for all C-language data types and control structures.

Figure 4 is a general block diagram of the 3B20D Processor. Two basic connections exist between the duplicated Control Units (CUs). One is an update connection that serves to keep the off-line CU's memory completely up to date. The second connection is a maintenance channel over which diagnostics of the off-line CU are performed. The Central Control and memory are duplicated and grouped as a switchable entity. The I/O and disk systems can be accessed by either CU through duplex intelligent controllers. The Disk File Controllers (DFCs) are normally both active in order to keep the data on the disks identical. Thus, under trouble conditions, either disk can support system operation. Unlike the SPC 1A Processors, the CUs are not run in a synchronous matching mode. Instead, 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 control is switched to the standby CU, its memory will contain up-to-

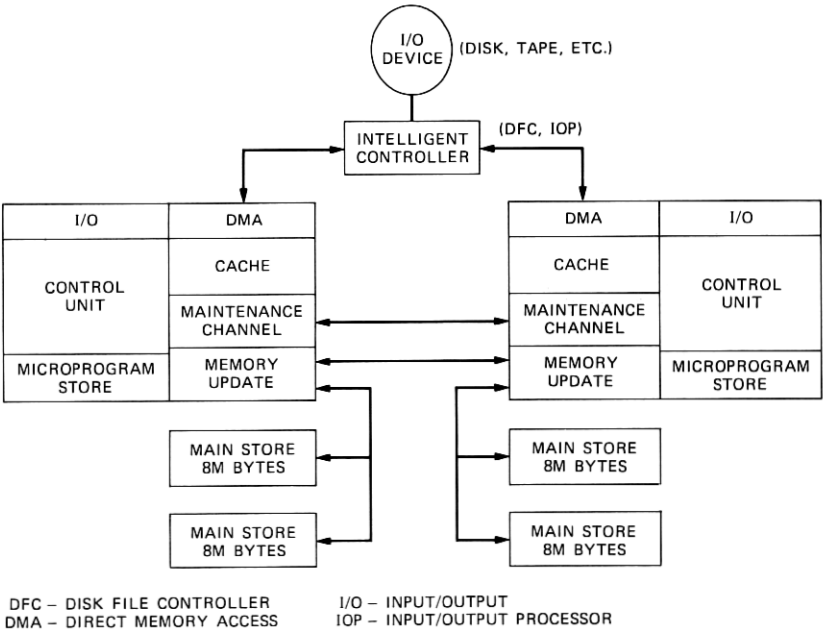


Fig. 4—General block diagram of the 3B20 Duplex Processor.

date information without performing a complete transfer from one CU to another.

3B20D Processor peripheral units are connected to the CC via the Direct Memory Access (DMA) unit. The DMA does not interface directly with peripheral units, but rather communicates with two intelligent subsystems, a Disk File Controller, and an Input/Output Processor (IOP). The CC builds job blocks for a peripheral unit that in turn notifies the CC upon job completion. The parallelism afforded by the autonomous processing capabilities of the DFC and IOP frees the CC for other work. Communication to both the DFC and the IOP is via Dual Serial Channels (DSCH) that allow any peripheral to operate with either CC of a duplex pair. Each DFC is capable of supporting 16 movable-head disk drives of 300M-byte capacity. Each IOP is capable of supporting a wide variety of peripherals, such as nine-track tape units, printers, synchronous and asynchronous data links, maintenance terminals, scanner/signal distributors, and custom network interfaces. Peripherals also may be connected to the Central Control Input/Output (CCIO) bus via an interface such as the Application Channel Interface (ACHI) used to communicate with the TSPS No. 1 periphery that was retained.

IV. OPERATING SYSTEM DESCRIPTION

DMERT is the real-time operating system for the 3B20D Processor. This operating system was developed concurrently with the hardware and uses a multiple-environment approach where time-critical code coexists with time-shared software.[5] That is, one environment supports real-time response while another environment provides a time-shared interface similar to that of the *UNIX** operating system.[6] The architecture of the operating system is process oriented; this allows applications to write software at the level most productive for each task. A process is an instance of a program executing on the processor and is characterized by a separate virtual address space. The multiple environments are implemented via a kernel that supports three levels of processes, described below. The DMERT kernel provides the most primitive virtual machine as it handles hardware interrupts, timer interrupts, and operating system traps. In all cases, the kernel saves the state of the interrupted process, provides whatever service is requested, and then restores the state of the interrupted process. The first level of process, known as a kernel process, is offered limited services by the DMERT kernel and is dispatched because of real-time events such as interrupts. The second level of process, known as

* Trademark of Bell Laboratories.

supervisor, is offered more services by the DMERT kernel in the area of I/O and dynamic memory allocation. Supervisor processes support a third level of process called a user-level process. An example of this is a *UNIX* operating system process controlled by a kernel-level process. In order for processes to cooperate in accomplishing their tasks, DMERT provides a set of interprocess communication and synchronization mechanisms including messages, events, process ports, interprocess traps, and shared memory. These interprocess communication primitives are fundamental to the DMERT structure.

I/O is accomplished by a kernel process, known as a driver. A unique driver is provided according to the type of I/O device, such as disk or IOP. Drivers receive I/O requests in the form of messages. When the I/O is completed, the message is returned to the requesting process. An intermediate supervisor process known as the file manager stands between disk users and the disk driver. This process implements a logical file system on the disk for those processes that care to use it. Files can be created, opened, closed, grown, and deallocated through the file manager. Hierarchical directories of files such as those found in the *UNIX* operating system are supported.

A set of processes and a special IOP peripheral controller provide a modern craft interface for the 3B20D and the TSPS No. 1B. A split-screen cathode ray tube (CRT) and a printer interface are utilized. The top portion of the screen is the status and display portion. Various status and display pages can be called upon demand. A special page that provides basic machine-control features such as initialization requests is provided by peripheral firmware. The lower portion of the screen is utilized for terminal I/O messages.

The continuous operation aspects of the 3B20D Processor are supported by a number of processes that are an integral part of the DMERT operating system. These processes handle error interrupts, control processor switches, and provide I/O to common peripherals such as disks; they also run equipment diagnostics and audit key data structures for consistency. Reference 7 contains a detailed description of both the 3B20D Processor and DMERT.

V. SOFTWARE STRUCTURE

The software structure of the TSPS No. 1B system was governed by three major design goals. First, the software architecture had to maximize the call-handling capacity. Second, steps were taken to maintain the SPC 1A programming environment as closely as possible in order to allow maximum utilization of existing software support utilities. Third, the existing TSPS software was to be emulated with a minimum of modifications. This guideline precluded unnecessary redesigns or restructures of the current field-proven software.

The emulation of the SPC 1A at the instruction level by the microcoding of its instruction set and at the system level by the PSI has allowed most TSPS programs to be executed on the 3B20D Processor with minimal modifications. The TSPS emulated code has been incorporated into a single, large, high-priority kernel process under DMERT.

The single process structure was dictated primarily by the existing tightly coupled nature of the TSPS software. As on the SPC 1A, all TSPS data and programs share and have access to the entire address space and communicate through data structures that reside in memory. Retaining these structures maintained the goal of exact emulation and avoided interprocess communication overhead detrimental to achieving the required performance gain.

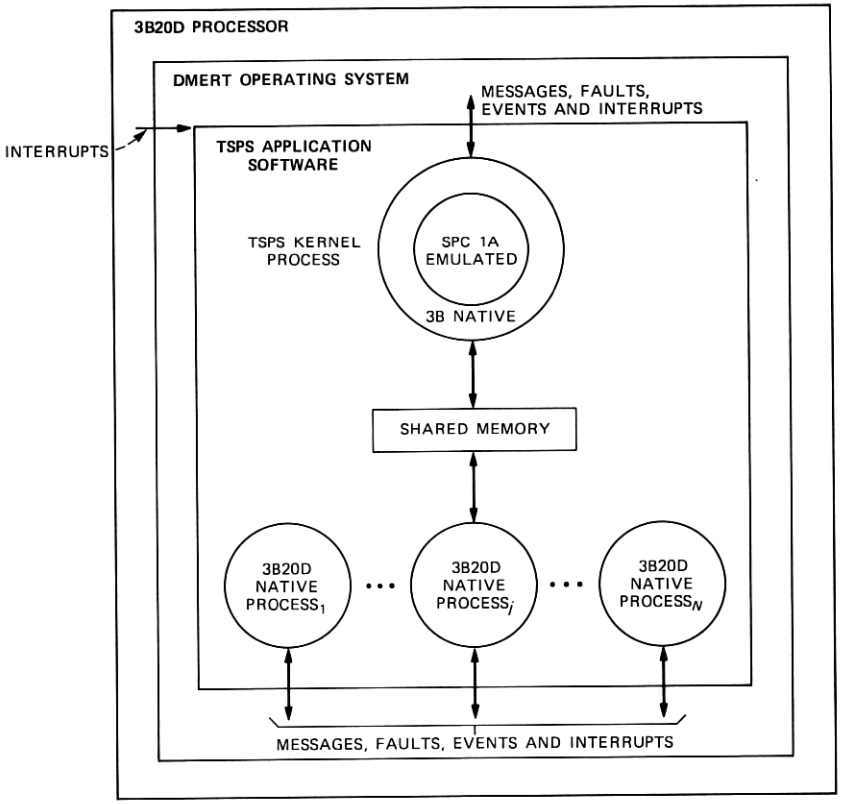
In addition to the emulated code, the TSPS kernel process contains native-mode (C-language) code that provides several capabilities. First, it provides a standard C-language data structure interface to the operating system and to other processes. Native code resident in the TSPS kernel process also works in conjunction with emulation microcode to implement system-level emulation of the SPC 1A interrupt structure within the actual interrupt structure defined by the 3B20D and DMERT. In addition, native code resident in the TSPS kernel process has eliminated the need to introduce new instructions, such as system calls, not available in the emulated instruction set.

All entries into the TSPS kernel process are through native code, which then calls the emulated code as a subroutine with a single instruction. Certain functions exist in the current TSPS software that would have required extensive modifications to emulate because of machine dependencies. In these cases, new replacement native code was written as subroutines that are called with a single instruction from the emulated code. In cases where completely new functions were implemented, native-mode processes separate from the TSPS kernel process were generated. An example of this is the TSPS File System Interface, which provides disk file system access as a replacement for functions previously implemented using the program tape unit of the SPC 1A.

The control structure of the TSPS call processing and peripheral maintenance software, being emulated, is almost identical to what exists on the SPC 1A. The major difference is that rather than running continuously as on the SPC 1A, the high-priority TSPS kernel process voluntarily must give up control of the machine periodically to allow lower priority processes to run. This is done by requesting periodic time-outs from DMERT. After the specified time has passed, the TSPS kernel process is reentered. The native code responds to the event and causes the emulated code to resume where it had left off.

The real-time breaks are not noticeable to the emulated software. The combined DMERT and TSPS priority structure is such that the TSPS kernel process dominates control of real time.

The process structure of TSPS No. 1B is summarized in Fig. 5. The DMERT operating system running on the 3B20D Processor provides a high-level, multiprocess environment for TSPS application processes. In this environment, the TSPS kernel process appears to the DMERT operating system to be identical to other native-mode processes that utilize the facilities of messages, faults, events, and interrupts in communicating with the operating system and—through the operating system—with other processes. The TSPS kernel process, in some instances, also communicates with other TSPS application processes through shared memory. An example, noted above, is the TSPS File System Interface. Within the TSPS kernel process, the combination



SPC – STORED PROGRAM CONTROL
TSPS – TRAFFIC SERVICE POSITION SYSTEM

Fig. 5—TSPS No. 1B process structure.

of emulation microcode, PSI, and native code creates an SPC 1A environment, thereby shielding the emulated code from the details of the specific machine and operating system on which it is running.

VI. SUMMARY

The processor capability was increased by replacing the existing SPC 1A with the 3B20D Processor while retaining the existing TSPS periphery and using emulation to preserve the existing TSPS software. A key aspect in the software architecture is the ability to execute either the emulated instruction set or the 3B20D native instruction set, and to switch between the two within a single process with a single instruction. Thus, it is possible to add new software to either environment and thereby increase the future flexibility of the system. Because the processor replacement is economically attractive both for new installations and for retrofits, techniques have been developed to replace an SPC 1A in an in-service office.

VII. ACKNOWLEDGMENT

The design of the 3B20D Processor and TSPS No. 1B required the cooperative efforts of a large number of people in Bell Laboratories, Western Electric, and AT&T. The authors wish to acknowledge the contribution of all of the team members whose work is summarized here.

REFERENCES

1. R. E. Staehler, "Traffic Service Position System No. 1B: Overview and Objectives," B.S.T.J., this issue.
2. N. X. DeLessio, J. R. Kane, M. W. Rolund, J. M. Scanlon, and R. E. Staehler, "The 3B Duplex Processor System and Its Application to TSPS," 10th Int. Switching Symp. Proc., September 21–25, 1981.
3. N. A. Martellotto, "An Operating System For Reliable Real-Time Telecommunications Control," Fourth Int. Conf. on Software Eng. for Telecommun. Switching Systems Proc., University of Warwick, England, July 20–24, 1981.
4. B. W. Kernighan and D. M. Ritchie, *The C Programming Language*, Englewood Cliffs, NJ: Prentice-Hall, 1978.
5. H. Lycklama and D. L. Bayer, "*UNIX* Time-Sharing System: The MERT Operating System," B.S.T.J., *57,* No. 6, Part 2 (July 1978), pp. 2049–86.
6. D. M. Ritchie and K. Thompson, "The *UNIX* Time-Sharing System," B.S.T.J., *57,* No. 6, Part 2 (July 1978), pp. 1905–29.
7. J. M. Scanlon, "3B20D Processor & DMERT Operating System: Prologue," B.S.T.J., *62,* No. 1, Part 2 (January 1982), pp. 167–9.