523-0561787-001731
2 February 1970

COLLINS

instruction book

Volume 1

C-System Principles of Operation

Printed in United States of America

Collins Radio Company | Dallas, Texas

List of Effective Pages

*The asterisk indicates pages changed, added, or deleted by the current change.

Page No.	Issue	Page No.	Issue
Title	Original	5-1 thru 5-23	Original
A	Original	5-24 Blank	Original
i thru vii	Original	5-25	Original
viii Blank	Original	5-26 Blank	Original
1-1 thru 1-5	Original	6-1 thru 6-17	Original
1-6 Blank	Original	6-18 Blank	Original
2-1 thru 2-19	Original	6-19	Original
2-20 Blank	Original	6-20 Blank	Original
2-21	Original	6-21	Original
2-22 Blank	Original	6-22 Blank	Original
3-1 thru 3-15	Original	7-1 thru 7-7	Original
3-16 Blank	Original	7-8 Blank	Original
3-17	Original	7-9	Original
3-18 Blank	Original	7-10 Blank	Original
4-1 thru 4-13	Original	Appendix	
4-14 Blank	Original	(Paragraphs 1 thru 7)	Original

Record of Revisions

RETAIN THIS RECORD IN THE FRONT OF MANUAL.
ON RECEIPT OF REVISIONS, INSERT REVISED PAGES IN THE MANUAL,
AND ENTER DATE INSERTED AND INITIALS..

ASSIGNED TO (JOB TITLE)

LOCATION

REV. NO.	REVISION DATE	INSERTION DATE	BY	REV. NO.	REVISION DATE	INSERTION DATE	BY

table of contents

table of contents (cont)

Page

table of contents (cont)

Page

table of contents (cont)

table of contents (cont)

list of illustrations

list of tables

section 1
introduction

1.1 GENERAL

The C-System is a flexible, expandable hardware system supported by modular software. Time division communication techniques enable single processor installations having limited input/output capacity to be expanded easily into large distributed networks of interconnected processors and peripheral devices. The modular operating software system complements this flexibility and permits the execution of many independent tasks on either a multiprogrammed basis by a single processor or on a multiprocessing basis when more than one processor is installed. The flexibility of both software and hardware allows easy C-System adaptation to a wide range of user requirements. This manual contains a detail explanation of the principles of operation for the C-System center.

1.2 SYSTEM ORGANIZATION

A typical C-System center (figure 1-1) consists of a number of processors, storage units, and peripheral devices interconnected by a high-speed communications facility called the time division exchange. The time division exchange is a switched communications network that allows all processors to have access to all storage facilities and peripheral devices in the center. Communication with devices that require continuous processor service is provided by a time division multiplex facility.

The time division exchange, which operates at 32 million bits per second, uses time division techniques to match its input/output data rate (2, 4, or 8 million bits per second) to that of the peripheral devices. The peripheral devices that operate at high speeds, such as disc files and magnetic tape units, are connected along a coaxial cable loop on the main exchange called loop 1. The slower-speed peripherals (such as cathode-ray-tube displays, line printers, and card readers) are connected by a second loop of coaxial cable called loop 2. This subloop provides the slower data rates (250 to 7.8 thousand bits per second) by subdividing the 2-Mbps data rate from the main exchange. Thus, the data rate from the time division exchange can be tailored to the speeds of the peripheral devices and all devices except disc files can be reassigned dynamically during system operation.

The time division multiplex communications facility connects low- and medium-speed devices directly to a processor and provides for intercenter communications. The multiplex facility operates at 1.2288 million bits per second and uses multiplexing techniques so that it can handle many peripheral devices having various data transfer rates (from 4800 to 153,600 bits per second). Each time division multiplex channel can be used to control teletypewriters, numerically controlled machines, cathode-ray-tube displays, process facilities, and other types of terminal devices.

C-System processor time, data paths, core space, and associated software are divided into independent processor channels that time-share the arithmetic logic and control unit. Consequently, arithmetic logic and control unit time is distributed between channels to provide effective multiprogramming and enables the operating system to automatically distribute work and regulate the workloads within the system. Two processor channels (designated A and B) are used to execute application programs; the A channel is used for major application programs and the B channel is used for minor application programs. Two processor channels (designated the orderwire 1 channel and the service or S channel) are used by the operating system; the orderwire 1 channel is used for processor-to-processor communication of service messages containing control information and the service channel is used for routing of work to processor channels, queueing of work, and controlling the execution of control programs. One processor in the center contains an orderwire 2 channel which contains center control functions including the load regulator, system scheduler, data collection service, and a program to manage the assignment

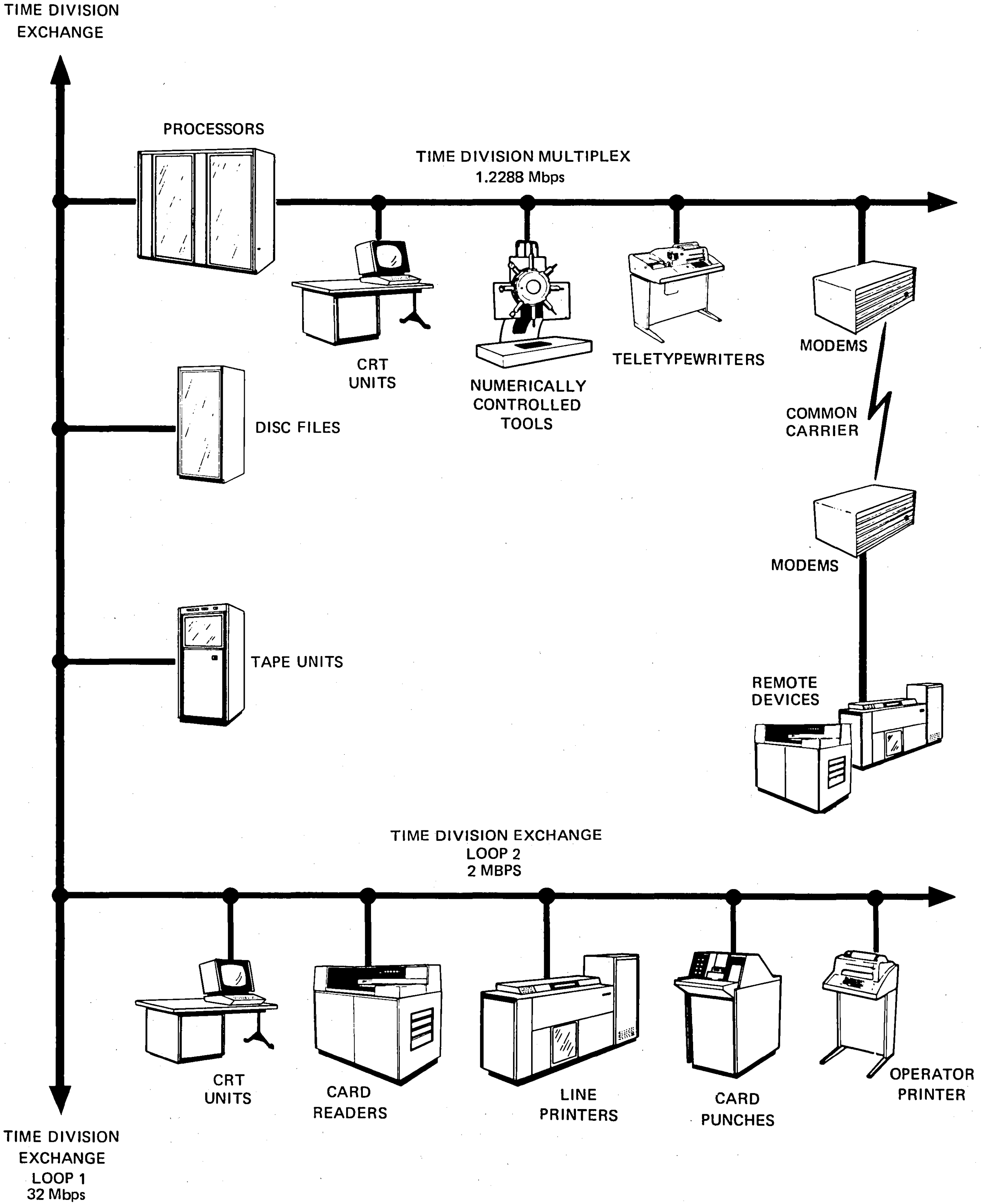


Figure 1-1. System Organization.

of peripheral devices to programs. The remaining processor channel (designated M channel) is used to service the time division multiplex facility.

The programming procedures employed in the C-System involve partitioning the overall activity into separate, well defined, and manageable tasks. Once the task list has been sequenced, the input requirements and the desired outputs are specified for each task. Together, these steps define a logical order of work and form a directed action network. Within the C-System, this logical order of work to be performed is specified by a control program. The control program directs activity within the system at the program calling level. Each task to be performed is specified by an instruction within the control program and may be dispatched to any suitable facility for execution.

As a result, the C-System operating structure allows completely automatic job execution of complex applied systems in a regulated multiprocessor environment. Messages are exchanged between processors and devices for directing work, reporting results, updating job status records and initiating new work. The result is a completely controlled processing environment.

1.3 SYSTEM OPERATION

The C-System software controls the distribution of tasks, provides the service necessary for program execution, and manages all output files. The system software implementation is based on a control program structure that allows input directed control. Control programs define the structural relationships among interrelated subtasks of an overall applied system. A control program consists of Program Control Instructions (PCI) as illustrated at the top of figure 1-2. The nodes represent program control instructions; the interconnecting lines show the required sequence of execution and indicate the outputs of program control instructions that become inputs to other program control instructions.

Each program control instruction contains information which identifies all data files necessary to execute the corresponding program, the location of the program, and other program control instructions that will require files created by execution of this program control instruction. The description and status of the control program is maintained in a control program status record. Each control program is executed by the processor that maintains the control program status record for the control program. Control programs may be initiated externally or on a scheduled basis by a system scheduler. In addition to referencing application programs to be executed, program control instructions may reference other control programs resulting in a nested control program structure.

Every processor in a C-System center may read information from any disc file but may only write to disc file space it has been assigned to manage. As shown in figure 1-2, processor 1 can read from or write to disc file 1, but it may only read from disc file 2; processor 2 can read from or write to disc file 2 but it may read only from disc file 1. The assignment of a whole disc file to a given processor has been illustrated here for simplicity; however, in actual practice disc file space may be segmented allowing several processors to write to one physical disc file but each processor is limited to an assigned segment.

The scheduler is a system program that resides in one processor in a center. At the preassigned time for initiation of a control program, a service message is sent to the processor responsible for the control program. The cognizant processor receives the service message and queues the message for subsequent processing by the service channel. A service program called control program service receives this message from the queue, accesses the control program status record from disc file storage, and builds a service message containing the return address of the cognizant processor, the disc address of the control program status constant, and the location of the calling program control instruction. The service message is then sent to the processor address obtained from the load dispatch table. When the program control instruction has been executed, a return service message is sent to the cognizant processor for the control program status update. Service messages are constructed to direct the execution of subsequent program control instructions to dispatch work throughout the center.

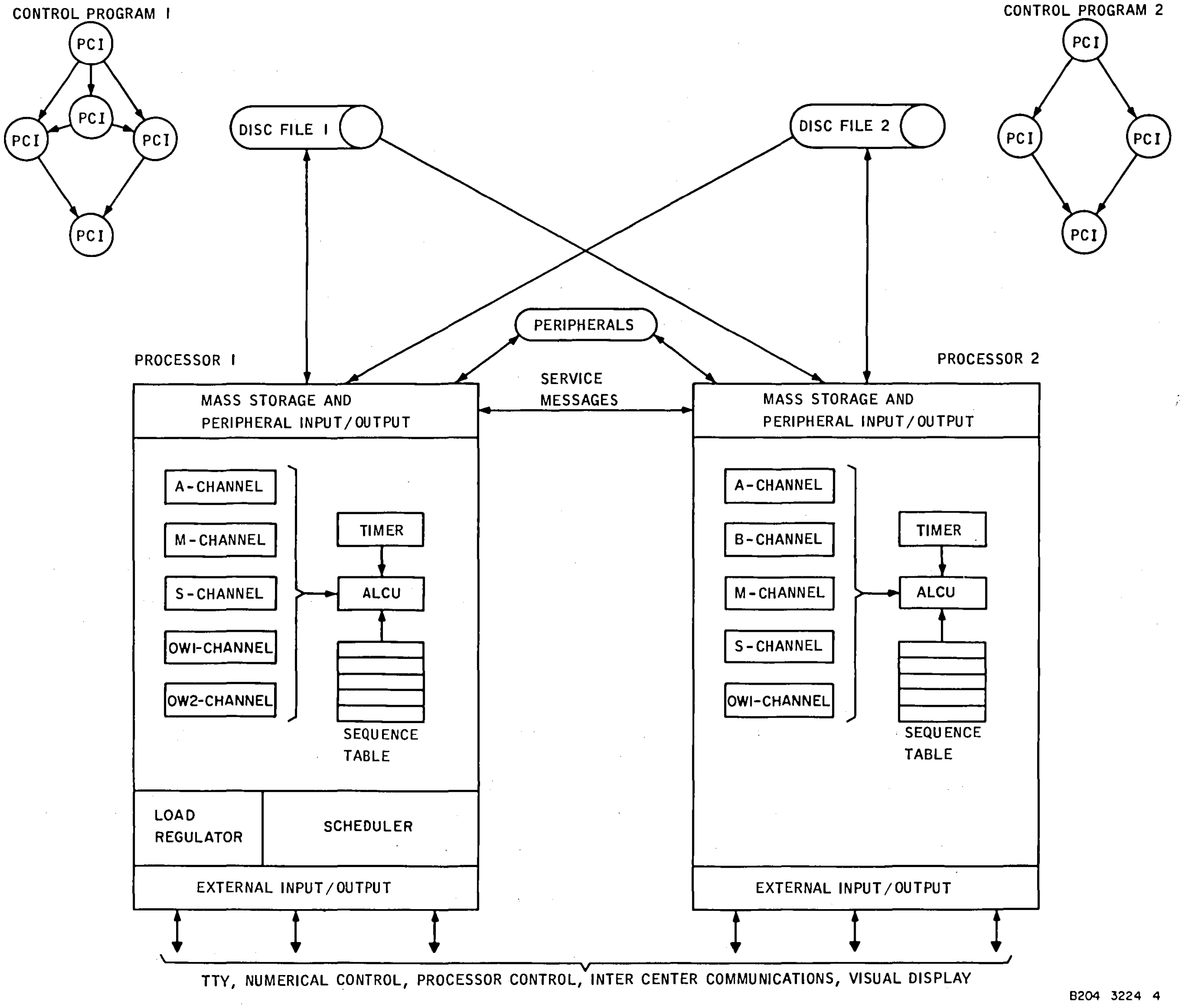


Figure 1-2. System Operation.

When a processor receives a service message to execute a program control instruction, the processor reads the control program status record from the calling processors disc file storage and obtains the particular program control instruction identified in the service message from the control program status record. The program identified in the program control instruction is then executed using input files identified in the program control instruction. The program creates an output file and stores the file on disc. At completion of the program, a return service message is sent to the originating processor noting completion of the program control instruction and specifying the location of the output file. This process continues until all program control instructions in the control program status record have been executed.

Work required to execute programs called by the program control instructions may be dispatched to any processor in the center. Thus, all processors may be involved in dispatching work to other processors. To avoid an unbalanced distribution of work, a load regulator program, which resides in one of the processors, periodically samples work queues throughout the system. If an undesirable balance of work is detected, the load regulator program may send appropriate service messages to various processors altering the work dispatch tables to result in a more evenly distributed work load.

More detailed introductory information is contained in the "C-System General Description" (523-0561-697), "C-System Operating System Software Description" (523-0699-114), and other Collins publications.

section 2
system organization

2.1 INTRODUCTION

This section introduces the hardware organization and operating system software of a typical C-System multiprocessor center.

2.2 SYSTEM HARDWARE ORGANIZATION

Figure 2-4 shows the basic structure of a typical C-System computer center. In general, a C-System is a multiprocessing system with several processors in a center. Processors and mass storage devices are interconnected via the Time Division Exchange (TDX) system. Connections to the exchange are made through terminal units. TU-1 terminal units connect disc files to the exchange; TU-2 terminal units connect processors to the exchange and TU-3A terminal units connect magnetic tape units and operator printers to the exchange. Subloops, designated loop 2, are connected to loop 1 by loop couplers. Card punches, card readers, line printers, CRT's, and operator printers are connected to loop 2 via TU-3B terminal units.

Two processors are shown in figure 2-4. A processor consists of an arithmetic logic and control unit, a magnetic core storage, a transfer link, a processor service unit, a multiplex service unit, and a communications service unit. The transfer link provides the interface between core storage and the other units of the processor. The communications service unit interfaces the processor to the time division exchange system. The multiplex service unit interfaces the processor to the time division multiplex system. The processor service unit is used during initial program loading and provides diagnostic and status monitoring functions during processor operation.

A Time Division Multiplex (TDM) loop may be attached to each processor for connecting low-speed devices to the processor, such as operator printers, CRT's, process control machines, and modems for remote communications. Devices are interfaced with the time division multiplex loop via a multiplex device coupler.

2.2.1 Time Division Exchange System

The C-System uses time division communication techniques for communication between processors, peripherals, and various input/output terminals. As shown in figure 2-4, communication between processors, disc files, tape stands, and related operating equipment is accomplished via time division multiplex techniques on time division exchange loops (loop 1 and loop 2). In loop 1, the loop synchronizer generates a digital pulse train at 32 Mbps. This pulse stream is used to derive sixteen 2-Mbps bit-interlaced communication channels (every sixteenth bit is devoted to one channel). The terminal units identify the channels by detection of a synchronization pulse generated by the loop synchronizer. Data transferred on the loops contain a 32-bit word with four additional bits of supervision. It is also possible to use loop 1 channels together resulting in additional channel rates of 4 Mbps or 8 Mbps.

Certain loop 1 communication channels are dedicated to disc file communication; every disc file normally has a private channel for communication. When a processor needs to access a disc file, its terminal unit waits for a poll. The processor bids for the channel and the channel is granted. The processor transfers only one cell or record (128, 256, 512 or 2048 bytes) and then releases the channel. All other processors are polled before this processor is polled again. Other loop 1 communication channels are designated as working channels and are assignable under control of a software system (device acquisition and control service) to other input/output devices. This concept allows many devices to make use of the available communication channels. The communication channels are assigned to devices such as magnetic tape units, line printers, card readers, etc., only for the duration of time these devices are being used. Party line addressing of devices can be used to allow more than one device to use the same channel.

A 2-Mbps loop 1 channel may be subdivided to provide lower-rate communication channels. The channel subdivision may be accomplished directly on loop 1 or, more commonly, a single 2-Mbps loop 1 channel may be used to drive another loop, designated as loop 2, which then uses the subchannels to communicate with various devices connected to loop 2. Subchannel rates of 7.8125 kbps, 31.25 kbps, 62.5 kbps, 125 kbps, and 250 kbps may be achieved. Loop 2 channels are assignable under the control of device acquisition and control service similar to loop 1 working channel assignments.

In addition to data transfers, the time division exchange system is used to transfer service messages between processors and from processors to peripherals. Orderwire 1 is used for processor-to-processor service message transfer. Orderwire 2 is used for service message transfer between processors and some peripherals. The orderwire channels are subchannels of a loop channel. Orderwire 1 operates at 125 kbps and orderwire 2 operates at 7.8125 kbps. These channels are shared and use party line addressing to enable the device or processor to recognize a call.

2.2.2 Processor

The C-8561A-2 processor consists of an arithmetic logic and control unit, communications service unit, a multiplex service unit, from two to four magnetic core storage units, a processor service unit, and a transfer link.

2.2.2.1 Arithmetic Logic and Control Unit (ALCU)-8561A-2

The arithmetic logic and control unit (figure 2-1) operates at 8 MHz and has a typical instruction execution time of 2.9 to 5.6 μs, including memory access. A total of 72 instructions is available, 65 fixed-wire and 12 executed in a trapped mode. An optional algorithm unit allows hardware execution of all floating point instructions and fixed point multiply and divide.

The arithmetic logic and control unit (figure 2-1) uses a 32-bit accumulator called the A accumulator, a 32-bit accumulator called the B accumulator, or a combination of both, called the D accumulator. Three 18-bit index registers provide storage for address modifiers used in indexing operations.

In the arithmetic logic and control unit, the three main registers communicate with the transfer link through input and output commutators. These are the function register, F, the memory exchange register, Z, and the memory address register, S. The F register holds the 14 bits of the instruction that include the operation code and control bits. This register interfaces with the instruction decoding control and timing circuits. The Z register, 32 bits in length, receives or transmits one word in parallel to the transfer link. All data to or from memory passes through the Z register. The S register, 18 bits in length, transmits the address of each memory word to be accessed through the transfer link.

The P register serves as the program counter or Instruction Address Counter (IAC). This register, 18 bits in length, holds the address of the next instruction to be executed in the program. The instruction address counter increments by 4 to skip from one instruction word to the next (memory is organized with byte addressability).

The 32-bit M register serves as temporary storage for data from the other registers during the machine cycle.

2.2.2.2 Magnetic Core Storage

The C-8561A-2 processor memory has a 2-μs cycle time and an access time of 510 ns. It is expandable from two to four modules. Each module has a capacity of 65,536 bytes, giving a maximum capacity of

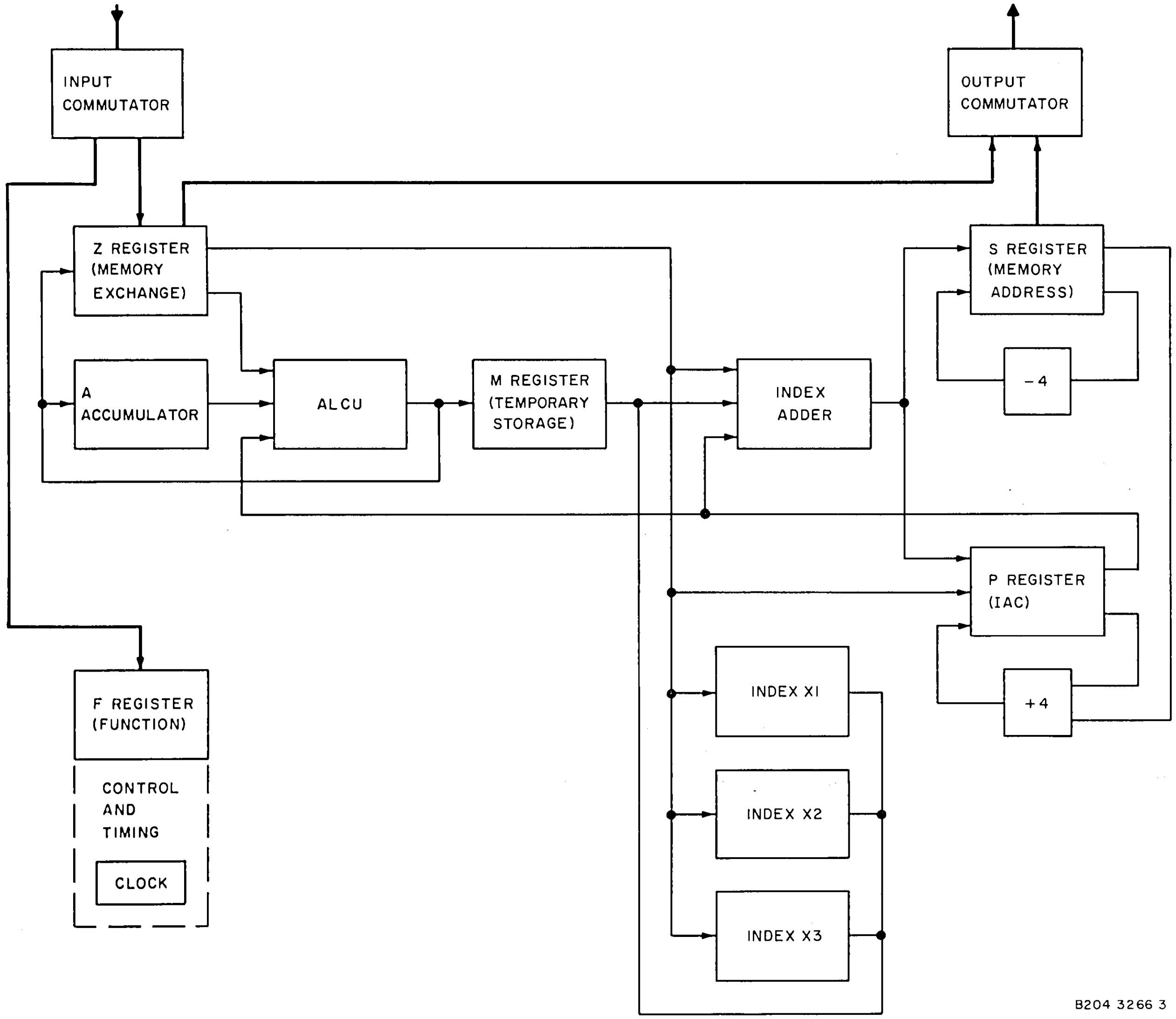


Figure 2-1. 8561A-2 Arithmetic Logic and Control Unit.

262,144 bytes for one processor. In addition to load or unload memory cycles, the unit has the capability of performing an unload-modify-load function in one memory cycle.

Each module in the memory has its own address and data buffers and it is possible for all four modules to operate concurrently. The alcu, communication service unit, processor service unit, and multiplex service unit may all request the use of the memory independently. If four units request the access of different memory modules, the entire memory can be in operation at the same time. The transfer links operate at a speed that makes this possible; together they can perform the four transfers of 32 bits in 2 μs.

2.2.2.3 Communications Service Unit

The communications service unit and associated terminal unit (figure 2-2) provide six independent data channels to the time division exchange; all channels may operate simultaneously. One of these data channels, the absolute time clock channel, is used to transmit clock signal from the loop synchronizer to all processors in a center. This information is transmitted every 7.8 ms and stored in a fixed location

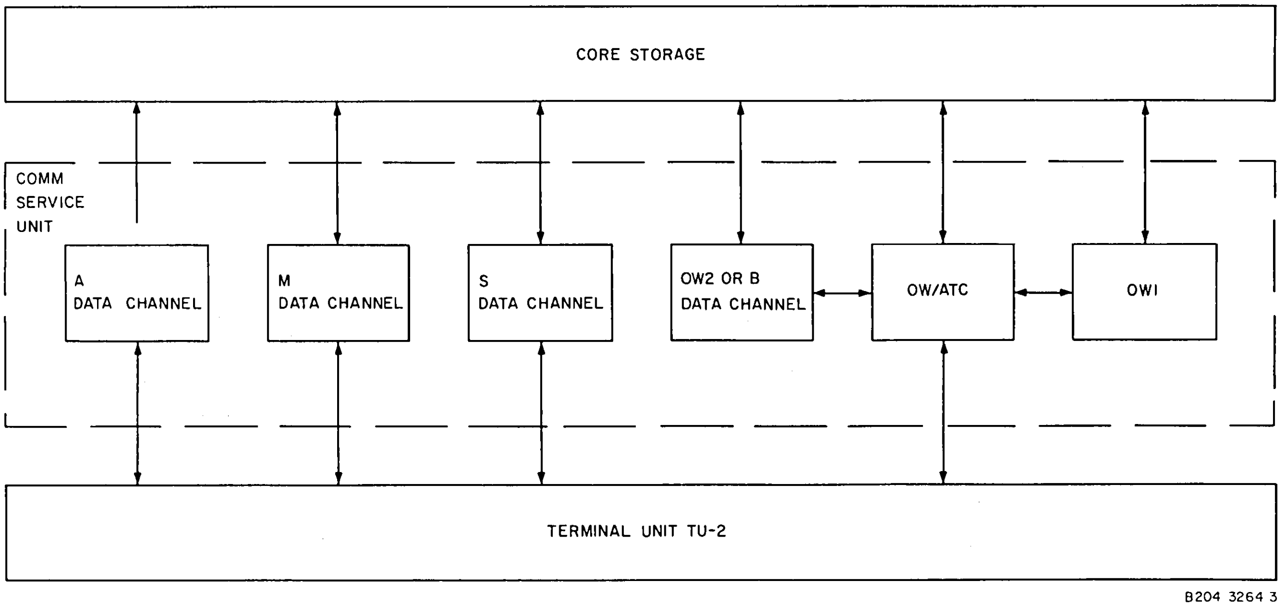


Figure 2-2. Communications Service Unit.

in main core storage for use by system programs. Another data channel is used exclusively for orderwire 1 data channel transmission of service messages between processors. Three of the four remaining data channels are dedicated to the processor channels (programming channels) for communication between programs executed in the processor and peripherals connected to the time division exchange loops such as disc files, card readers, printers, etc. The sixth data channel is used by either the orderwire 2 or the B processor channel. One processor in a center is responsible for orderwire 2 functions including assignment of time division exchange working channels to magnetic tape units and loop 2 devices and necessary control messages to these peripherals. In the remaining processors in the center, this channel is used to support B channel processing.

The communications service unit has direct access to core storage through the processor transfer link. Data records to files and messages for other processors are read from and written to core storage directly under communications service unit control. To instruct the communications service unit, the arithmetic logic and control unit builds device control messages which are 9-word instruction packets in core which the communications service unit reads and executes. The instruction packets contain device control information and core location information for data and status. Once a device control message is built for an input/output transaction or a processor message transfer, the arithmetic logic and control unit is free to continue other work while the communications service unit executes the required information transfer.

2.2.2.4 Multiplex Service Unit

The multiplex service unit drives the serial multiplex loop at 1.2288 Mbps and inserts appropriate synchronizing pulses into the bit stream to identify 256, 4.8-kbps time slots. The time division multiplex loop provides word-interlaced channels that allow simultaneous access between a processor and the medium- or low-speed devices connected to the time division multiplex loop.

All devices on the time division multiplex loop are managed independently and simultaneously, so that each device may be in an arbitrary state of completion with respect to all other devices. This mode of operation is particularly suited to the requirements of many low-speed devices, such as teletypewriter inquiry stations and CRT display devices.

The multiplex service unit allows a device on the time division multiplex loop to communicate with the arithmetic logic and control unit by requesting processor service. The multiplex service unit directs the transfer of data between devices and core storage through the transfer link without arithmetic logic and control unit intervention once the arithmetic logic and control unit has set up the transaction. The multiplex service unit also alleviates the arithmetic logic and control unit of the routine task of monitoring devices on the time division multiplex loop for service requests. When a device needs processing time, an appropriate indication is placed in work queue serviced by the arithmetic logic and control unit. The indicator points to a multiplex status record corresponding to the device requiring service. The multiplex status record is used by the arithmetic logic and control unit and the multiplex service unit to transfer information. A multiplex status record is a 4-word table located in core storage. Each active multiplex status record contains sufficient information for the arithmetic logic and control unit to accomplish the task required by the device. Typical arithmetic logic and control unit tasks include the management and allocation of core storage used for time division multiplex loop data buffers, the transfer of records to and from disc files, and the execution of special device-oriented subroutines. Devices on the time division multiplex loop initiate all communication with their associated processor by transmission of the proper instructions to the multiplex service unit. When a device on the time division multiplex loop is idle, it accepts output information from the computer.

2.2.2.5 Processor Service Unit

The processor service unit performs the functions of controlling initial program loading of the processor, initiating processing, and performing selected diagnostics on processor units. The processor service unit has interface connections with the communications service unit, multiplex service unit, transfer link, arithmetic logic and control unit, terminal unit, and main core storage.

The processor has three manual operating controls which are contained on the front panel of the processor service unit (figure 2-3). The IPL button starts the automatic initial program load sequence, which loads programs from the time division exchange loop. During initial program load, the memory protection feature is overridden. The Initiate (INIT) button initializes the arithmetic logic and control unit without loading a program. Lamps on the panel of the unit indicate that the processor is operating normally (RUN) or that it has failed (Machine Failure Monitor, MFM). An 8-bit lamp-bank indicates which processor function has failed during an initial program load. Upon successful completion of an initial program load, these lights are under program control. The Marginal Voltage Test (MVT) button allows for marginal voltage testing.

The processor service unit controls certain operations of all processor units. These include enabling the arithemetic logic and control unit and communication control equipment, enabling diagnostic mode to all units, controlling marginal voltage tests, forcing memory parity errors in all memory modules, disabling protected memory, controlling interleaved memory mode, and displaying status via the 8-bit lamp-bank on the front of the processor. These functions are controlled by the processor service unit during an initial program load or initiate sequence. After a successful initial program load or initiate, these functions are under the control of software. Every time the arithmetic logic and control unit executes a reset machine failure monitor instruction, the processor service unit will set the above functions to reflect the bits in memory location X'40. This word is referred to as the processor control word.

The processor service unit maintains a processor status word in memory location X'44. This word contains the current status of the fault alarms of all processor hardware.

2.2.2.6 Transfer Link

The transfer link provides the switching for all communication from the arithmetic logic and control unit, multiplex service unit, communications service unit, and processor service unit to core storage. The transfer link performs memory module decoding, resolves contention for a memory module, checks for an invalid address, performs memory protection decoding, gates addresses and data to the memories, and returns data from the memories to the requesting device. A single transfer link module has two source ports and four memory ports permitting two simultaneous paths of communication between two

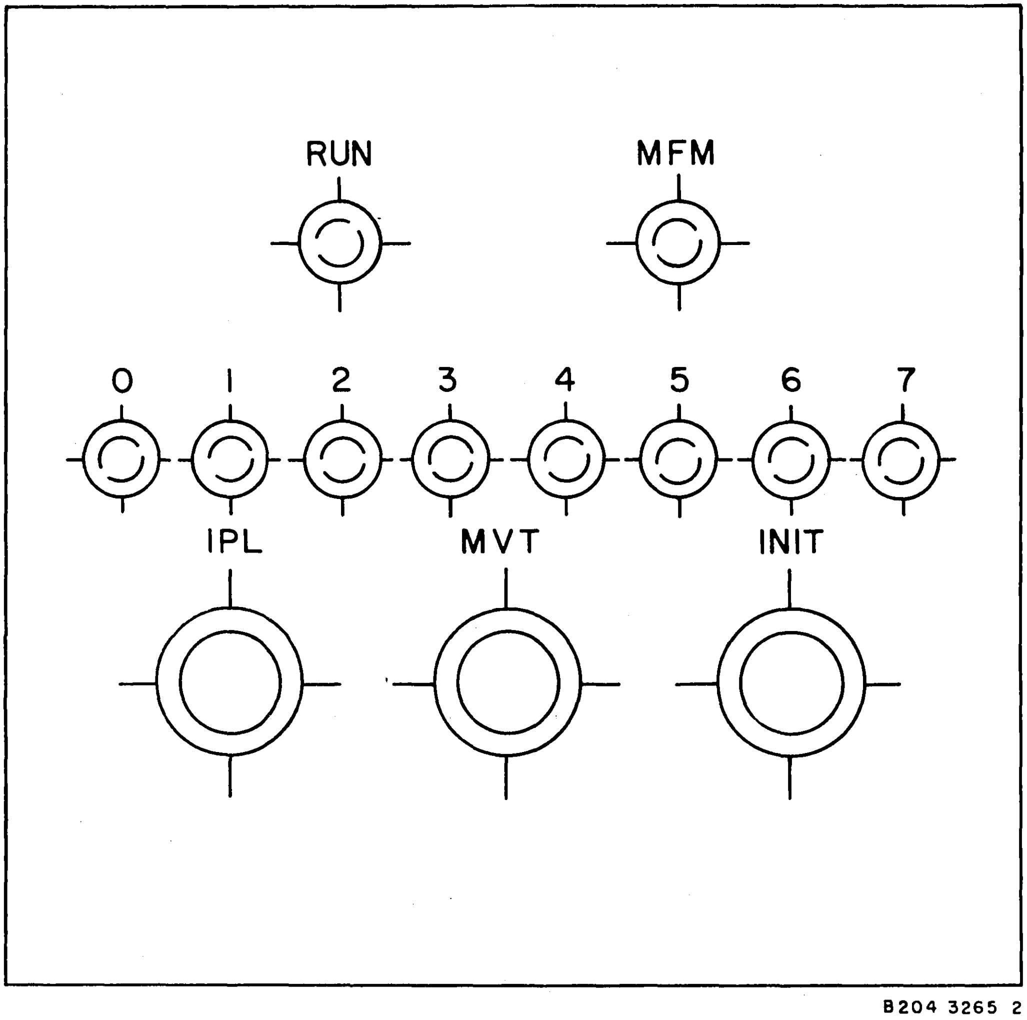


Figure 2-3. Processor Service Unit, Controls and Indicators.

logic modules and any two of four memory modules connected to the transfer link. Up to four logic units may be connected to any of the transfer link module source ports. Also, up to four transfer link modules may be connected in parallel by busing together the memory ports thus allowing eight source ports. If both source ports from the same transfer link attempt to access the same memory module at the same time, the transfer link resolves the conflict on a first in, first out basis. For systems employing more than one transfer link module, a weighted priority between transfer link modules is established by means of the priority interconnect between the boxes.

2.2.3 Disc Files

Two Collins disc files are available for use in the C-System. One is a high-speed, direct-access storage device with a storage capacity of 130 megabytes of data on 26 discs. The unit has an average head positioning time of 93 ms with a 25 ms minimum and 185 ms maximum plus a track verification check requiring 35 to 45 ms. The nominal disc revolution time is 50 ms. There are 81,920 bytes of storage available at 10 fixed-head addresses. Data transfer is serial by byte at a rate of 173,000 bytes per second.

The second unit is a high-speed, random-access device with a storage capacity of 33.5 megabytes on 8 discs. The average head positioning time for this device is 18.4 ms with a 6 ms minimum and 30 ms maximum. The disc revolution time is 25 ms and the data transfer rate is 350,000 bytes per second.

2.2.4 Magnetic Tape Units

The 9-channel magnetic tape unit operates with 1/2-inch magnetic tape. The unit reads in the forward and reverse tape directions and writes in the forward direction at 150 inches per second; the rewind speed is 250 inches per second. Data is stored at a density of 800 bits-per-inch per channel. Eight channels are used for data and one for parity. The data transfer rate is 120,000 bytes per second.

The 7-channel magnetic tape unit operates with 1/2-inch magnetic tape. The unit reads and writes in the forward direction at 150 inches per second; the rewind speed is 360 inches per second. Data is stored at a density of 200, 556, or 800 bits-per-inch per channel. Six channels are used for data and one for parity. The data transfer rate is 30,000, 83,000, or 120,000 bytes per second.

2.2.5 Peripheral Devices

The C-System provides the following optional peripheral devices: card readers, card punches, operator printers, line printers, and CRT display and entry stations. These devices connect to time division exchange loop 2 through TU-3B terminal units.

2.2.6 Time Division Multiplex Loop

The time division multiplex loop provides connection of a large number of low-speed devices to a single computer within the C-System. The loop is physically a coaxial cable which interconnects the various devices in a serial manner and terminates at the multiplex service unit. The basic data rate on the loop is 1.2288 MHz. In contrast to the time division exchange system which is bit interlaced, the time division multiplex loop operates with word interlacing with 256 words (36 bits each) in one frame resulting in a frame length of 9216 bits. Each time division address corresponds to a device channel; the channels are not reassignable as in the time division exchange working channels. A framing pulse generated in synchronization with the first time division address is used to identify the channel words in the loop. Data is inserted and extracted from the channels by the multiplex service unit and devices. A device may be assigned 1, 8, 16, or 32 time division addresses, providing effective channel bit rates of 4.8 kbps, 38.4 kbps, 76.8 kbps, or 153.6 kbps.

The multiplex device coupler provides a standard interface to devices on the time division multiplex loop. The multiplex device coupler performs the functions of time division address recognition, serial extraction and insertion of data into its assigned time division address channel, and logical interpretation and generation of the supervision necessary to communicate with the multiplex service unit.

The loop word format for each of the 36-bit time division addresses consists of a 4-bit supervisory code field and 32-bit operand. The supervisory field contains the supervising codes for multiplex service unit to multiplex device coupler communication. The operand contains the instructions, data, parameters, etc., for the device and programs.

2.3 SYSTEM SOFTWARE ORGANIZATION

Figure 2-5 shows the general flow and interrelationships between the major software elements of the system. All software elements shown in figure 2-5 do not reside in each processor. Orderwire 2 resides in only one processor in a center and that processor normally has no B channel. The exact core allocation depends on the core space required by the M channel and is defined for each center configuration.

2.3.1 Time Division Exchange (TDX) Communication

Communication between the communications service unit data channels and programs running in the processor channel is by means of Device Control Messages (DCM's) which are built by system programs on each of the various channel time. Device control messages are instruction packets directing a data or message transfer; the communications service unit reads and executes the packets. As shown in figure 2-5, each processor channel has two device control message chains; a typical device control message is shown in the upper left-hand corner. There are an arbitrary but fixed number of device control messages in each chain and the entire chain is in a fixed location in core storage with each device control message having a fixed starting address. Each device control message in the chain contains a device control message chain pointer to the next device control message in the chain with the last pointing to the first, thus forming an endless chain. The two device control message chains for each processor channel are serviced alternately by the corresponding communications service unit data channel, with one device control message per chain being serviced on each chain access by the communications service unit.

The actual data and message transfer on the time division exchange system takes place strictly under terminal unit and communications service unit control through execution of the device control message chains. The transfer does not involve the arithmetic logic and control unit except for building the device control messages, data control words, device control words, and monitoring device control messages for transfer completion.

Data Control Words (DCW's) are appended to device control messages. Data control words contain control bits defining the nature of the data or message transfer and the core address of the data to be transferred or the area into which data is to be read. Data control words are also used to point to device command words that identify, for example, the read or write function and the address of a disc file record. A data control word can be used to identify another list of data control words not appended to the device control message but to be executed with this device control message. If such a data control word is encountered in executing a device control message, the data control word chain address is placed in the third and fourth bytes of the second word of the device control message and this address is branched to at an appropriate place in the execution of the data control word list.

The first word of the device control message contains certain control bits and the device control message chain address pointing to the next device control message in the chain. The second word contains the loop 1 and loop 2 time division exchange address of the device which communication is to be established and two bytes for a data control word chain address. The third word contains a from-program address, which references a program that is branched to in the event of an error and that may be given immediate control upon verification of completion of this device control message. This word also contains the response address for storage of device status at completion of the transfer. The fourth word of the device control message is the first data control word to be executed in the transfer.

In addition to the device control message chains for each channel, each processor core has a set of prebuilt device control messages and data control word lists which reference certain tables in the processor core for direct access by the load regulation program via orderwire 1 service messages. In general, a pair of device control messages and data control words (one for reading and one for updating) is required for each table to which the load regulator has access.

2.3.2 Operations Control Program

An operations control table exists in core storage with an 8-word entry for each resident processor channel. A typical entry is shown in figure 2-5. The first word contains channel status bits and the address of the channel save area. A typical save area is shown in figure 2-5. The save area is located at the beginning of the processor channel space and is part of the program status record. The second and third words of the operations control table entry identify the location of the channel work queues and the next-write-positions and next-read-positions for these queues. The Next-Write-Position (NWP) pointer is the next place in the queue to post a request for work; the Next-Read-Position (NRP) pointer is the next entry in the queue to be executed by the channel programs. The next three words in an operations control table entry are device control message chain pointers for chains 1 and 2. The next-read-position pointer is the next device control message to be checked by the operations control program for completion; the next-write-position pointer is the next device control message which can be used to post an input/output request, and the Next Available Cell (NAC) identifies the next device control message to be executed by the communications service unit.

The operations control program, shown in the center of figure 2-5, is the resident software system responsible for allocation of arithmetic logic and control unit time and verification of device control message chains for completion of input/output requests. The operations control program can be entered from a given channel in one of four ways: (1) the channel may run to completion and exit to operations control (OP COMP) in which case the second bit, I, in the operations control table is set to idle and a return address saved; (2) the channel may disconnect busy (OP BUSY) awaiting an input/output completion in which case I is set to not idle and the third bit, B, of the status word in the operations control table is set to busy and a return address saved; (3) the channel may voluntarily relinquish the rest of its time when certain conditions are met in the program execution in which case I is set to not idle and B to checkpoint and a return address saved; or (4) the channel may be interrupted by the interval timer in which case I is set to not idle and B to checkpoint and all arithmetic logic and control unit registers saved in the channel program status record so that the channel may resume processing the next time it gains control of the arithmetic logic and control unit. All methods of entry to the operations control program are used by the A and B channels but, under normal operating conditions, the M and S channels only use entries 3 and 4 while orderwire 1 and 2 channels only use entries 2, 3, and 4. The operations control program works with the operations sequence table which indicates the sequence in which the channels are to be given time. This sequence is arbitrary and may be any order depending on the processing task. Upon entry, the operations control program examines the sequence table for the next channel to service. The idle bit is then checked to determine the channel status. If the channel was idle, the next-read-position pointer and next-write-position pointer for the channel work queues are compared to determine if any channel work has been posted since the channel last had control. If the next-read-position pointer does not equal the next-write-position pointer in either queue, then work is available and operations control exits to the channel. If the next-read-position pointer equals the next-write-position pointer, no work is required and the operations control program checks the sequence table to determine the next channel to be given time. Only the A and B channels use this test since only these channels enter operations control as idle. The other channels manage their own work queues, using the next-read-position and next-write-position as appropriate.

If the channel was not idle the last time it had control, operations control proceeds to verify device control messages for completion on that channel. By monitoring the status bits in the device control messages for the particular channel, the operations control program can detect whether or not any input/output requests have been completed. The B bit of the status word of the operations control table can then be checked to determine if the channel was awaiting a data transfer. If the channel was awaiting a data transfer and none has occurred, the channel is skipped and operations control returns to the sequence table; if at least one device control message had been completed by the communications service unit, operations control will return control to the channel. If the channel had used the voluntary checkpoint as indicated by the third bit of the operations control table status word, control is returned to the channel program regardless if an input/output has been completed. Likewise, if the channel had been interrupted by the timer, the arithmetic logic and control unit is restored to its prior condition and control returned to the channel.

2.3.3 Orderwire 1 Channel

Communication from processor to processor within the C-System is by service messages sent on the orderwire 1 party line communication channel of the time division exchange loop. The orderwire 1 input program has the responsibility of receiving these messages and dispatching them to appropriate work queues to be serviced by the processor on the time of the respective channels. The orderwire 1 input program manages its device control message chain for receipt of orderwire 1 service messages by providing a main core storage buffer for each device control message into which an incoming message can be written. When orderwire 1 is indicated by the operations sequence table and the operations control program detects one or more completed device control messages in the orderwire 1 device control message chain, the orderwire 1 input program is given control. The orderwire 1 input program decodes the operation code of the service message. Certain service messages require immediate processor action. The orderwire 1 input program immediately initiates the appropriate routine called in the immediate service message. All other service messages are either directed to service queue 2, which is a work queue for the orderwire 2 channel, service queue 1 which is a work queue for the S channel, or Multiplex Channel Queue (MCQ) which is serviced on M channel time. For service queue 1, service queue 2, and the multiplex channel queue, a pointer indicating the location of the service message in core is placed in the queue. The orderwire 1 input program reinitializes the device control message by obtaining a new bin for a service message. The orderwire 1 input program exits to the operations control program as busy awaiting an I/O when all device control messages have been reinitialized and does not receive control again until another device control message is complete. The orderwire 1 input program uses the voluntary checkpoint entry to operations control if buffer space is not available for assignment or if the work queues are full, ensuring return of control the next time orderwire 1 is indicated in the sequence table.

2.3.4 Orderwire 2 Channel

Orderwire 2 channel time is used for execution of the scheduler, Data Collection Service (DCS), Device Acquisition and Control Service (DACS), and the load regulator. Requests for service for these channel programs are received from orderwire 1 service messages via service queue 2 and from service messages received via orderwire 2 used only by device acquisition control service. An orderwire 1 service message input decode program decodes the service messages from service queue 2 and posts them for various orderwire 2 channel programs. The orderwire 2 input program maintains input device control messages and service message bins for input service messages. When a message is received, the orderwire 2 input program places the message in a work queue and reinitializes the device control message for receipt of subsequent orderwire 2 service messages. Control is given to the orderwire 2 input program each time a channel program gives up control. A channel program gives up control after it performs a file service request or completes processing of a service message. The orderwire 2 allocator gives each channel, in turn, time to process provided the program is not waiting for the completion of a file service request. Each time the work allocator is given control, the work allocator allows only one function to run.

2.3.4.1 Data Collection Service

Data collection service runs on orderwire 2 channel time and provides applied systems with a means of collecting, categorizing, and distributing raw input data from any point in the system. All data input to data collection service must have a preassigned data collection service code. A data collection service code is assigned by data collection service on receipt of a data collection service code request service message. All input files with the same data collection service code are maintained in a collection file. There is only one collection file per assigned data collection service code. The collection file is a file on disc that contains the input file identifiers for the files associated with a data collection service code. When a data collection service extract service message is received, data collection service returns via a program return service message the address of the collection file which the user may then access to obtain the location on disc or tape of the data needed.

2.3.4.2 Device Acquisition and Control Service

Device acquisition and control service runs on orderwire 2 channel time and provides the application systems a means of communication between processors and magnetic tape units or loop 2 devices. Certain time division exchange loop 1 communication channels and all loop 2 communication channels are working channels shared among the various peripherals and must be assigned by device acquisition and control service upon request from an application system; it is the responsibility of device acquisition and control service to make these assignments. Device acquisition and control service operates in two modes: (1) the system mode and (2) the device mode. In the system mode, for example, an application program under control of a control program status record encounters a program control instruction requiring communication with a magnetic tape unit or a loop 2 device. An orderwire 1 service message is sent to device acquisition and control service to establish communication. The orderwire 1 input program directs the service message to service queue 2 where it is decoded by orderwire 1 input decode as a device acquisition and control service request. Device acquisition and control service processes the messages, assigns a working channel to the required device, builds an output file containing control data and device status, and identifies the working channel assigned to the device. A return service message is sent to the calling processor identifying the output file which it then accesses, obtains the working channel identification, and begins communication with the device.

In the device mode, device acquisition and control service receives an unsolicited orderwire 2 input message from a loop 2 device requesting to input data into the system. In this case, a program call service message is issued for a device acquisition and control service defined device-to-disc program which will store and forward the input data.

2.3.4.3 Scheduler Program

The main purpose of the scheduler program is to control the initialization of application program systems or portions of application program systems as a function of time. Orderwire 1 scheduler call service messages are directed to the scheduler program through service queue 2 and the orderwire 2 allocator. The service message identifies the orderwire 1 time division exchange party line address of the calling processor, the time the application system is to be initiated, a priority code, and the disc file storage address of the application system control program status record. The received service messages are placed in one of four priority queues in order of activation time by a table generator program which is a nonresident program that runs on a scheduled basis on A or B channel time. Thus, received service messages are placed in a queue to be serviced later by the table generator program if their execution time is later than the next scheduled run of the table generator program; otherwise, the service message is tabled in an advanced activation cell and can be operated on by the scheduler without being filed by the table generator program. When the scheduler program is executed, the current time obtained from the absolute time clock storage location is compared to the activation time specified in the first service message in each queue. If the current time is greater than or equal to the activation time, a return service message is sent via orderwire 1 to the calling processor indicating time to activate the scheduled program.

2.3.4.4 Load Regulation Program

The purpose of the load regulator program is to automatically regulate the work load in the various processors in the system. The load regulation program is executed in the orderwire 2 channel of one processor in a center and dispatches direct access service messages to the various processors in the center. The communications service unit orderwire 1 data channel hardware recognizes the service messages as direct access messages and executes the prebuilt direct access device control messages and data control word lists which reference particular parameters in core storage to be returned to the load regulator. Having sampled each processor, the load regulator program can detect any undesirable distribution of work that may exist in the various channel work queues throughout the center. If such an undesirable balance is detected, the load regulation program will dispatch additional immediate access messages to change dispatch parameters utilized by the various processors in assigning program control instructions for execution.

2.3.5 S Channel

The S channel is used to execute several processor service programs. S channel decode is the program which directs the activity of the S channel by decoding service messages pointed to in service queue 1. The operation code of the service message is mapped to the S channel decode table for the appropriate S channel program entry address, and control is passed to the S channel program. If, in the course of executing an S channel program, it is necessary to checkpoint the program for any input/output, the device control message requesting the transfer is posted, an entry made in an S channel completion chain, and control is returned to S channel decode rather than the operations control program. This allows other S channel programs to execute as required on remaining S channel time. S channel decode always uses the voluntary checkpoint entry of the operations control program, unless a timer interrupt forces relinquishing channel time. When S channel decode is entered from either operations control or an S channel program, it always checks its completion chain first to determine whether or not any requested input/output transactions have been completed. If they have, control is transferred to the program requesting the transfer. After the entire completion chain has been checked, service queue 1 is checked for service messages directing further work.

2.3.5.1 Control Program Service

Control Program Service (CPS) runs on S channel time, is resident in each processor in a center, and is responsible for the supervision of the execution of control programs assigned to that processor. Control programs represent the technique utilized in the C-System for implementing input-directed control for automatic and regulated execution of application system programs in a multiprocessor center. Control programs are physical lists of program control instructions each of which identifies a particular segment of the overall system for execution. The program control instruction also identifies any input files required for execution of the program and points to other program control instructions in the control program that require the execution of this program control instruction before being executed. The program control instruction identifies files which can be released after completion and also contains various control parameters required by control program service in the execution of the control program.

It is possible for program control instructions to call lower level control programs resulting in a nested control program structure. Note that control program service does not manipulate files or the application system programs themselves, rather, it directs the execution of these programs and manipulation of these files through orderwire 1 service messages. For example, if during the execution of a control program the necessary conditions have been satisfied enabling a certain program control instruction to be executed, control program service causes the transmission of an application call service message to a processor in the center. The service message identifies the disc file storage address of the calling control program status record. The service message also identifies the A or B channel work queue to be used. This service message is queued in the appropriate A or B channel work queue of the called processor. When it is reached, the calling control program status record is read from disc file storage and the invoking program control instruction isolated. The program control instruction identifies the application program which is to be accessed and executed using the input files identified in the program control instruction as necessary. Upon completion of the program, a program return service message is sent via orderwire 1 to the calling processor noting the completion and identifying any output files created during execution of the program. Control program service in the calling processor receives this message, updates the control program status record, and initiates execution of subsequent program control instructions in a similar manner. Many other service messages are used by control program service depending on the particular program control instruction to be executed. For example, a program control instruction may require a scheduler call service message to be sent to the system scheduler identifying a time for the scheduler to return a service message. The return service message can then be used to key the execution of another complete control program by making the scheduler program control instruction prerequisite to the execution of the first program control instruction in the control program. Program control instructions can direct the sending of the data collection service file extract service messages by control program service causing particular information to be retrieved from the data collection service files. Messages can be sent to device acquisition and control service for establishing communication with tape units and loop 2 peripherals. As program control instructions are completed and certain disc storage files are no longer needed, disc file storage space release service messages are sent to the space release program executed in the respective processor S channels. In a similar manner, tape release service messages are sent. For nested control programs, control program service sends program return or abnormal return service messages to higher level control programs. In this manner, making use of the various service messages, control program service supervises and directs the complete execution of an applied system. Control programs may be run once or may be cyclic, running on a periodic basis. The latter is implemented by sending a service message to the scheduler upon completion of a control program, directing the scheduler to return a message at a certain time automatically reinitiating the control program.

Control programs may exist in one of two states: (1) as a Control Program Status Record (CPSR) or (2) as a Control Program Status Constant (CPSC). A control program status record is generated from a control program status constant by the control program status record generator program which is a nonresident program run on the B channel of a processor in response to a control program status record generator call service message. This message may originate from another program control instruction, or it may be generated externally and entered, for example, through the time division multiplex loop. The control program status record generator program makes a copy of the control program status constant, placing certain information in the control program status constant creating the control program status record, which is then activated.

2.3.5.2 Space Release

The space release service message routine resides in each processor in a center, runs on S channel time, and releases file space referenced by a file release service message. These service messages are generated by control program service and may come from any processor in the center for release of file space managed by the particular processor receiving the service message. Control program service generates file release service messages upon completion of program control instructions; the messages reference storage files no longer needed. This routine calls upon the common function release file routine to release the file space. The orderwire 1 service message space used by the service message is then released via the common function common space release.

2.3.5.3 A and B Channel Queue Service

A and B channel queue service runs on S channel time and queues service messages directed to the A or B channel. This program is given control by S channel decode upon detection of an A or B channel service message. A and B channel queues consist of one 2048 byte cell each and are located on disc file storage. There are two queues for each channel and the queue cell addresses, next-read and next-write displacements from the top of each queue, are maintained in the operations control table. Because these queues are fixed length, the load regulator program periodically examines them to determine if any of the queues are above a predetermined level. If any queue is nearing capacity, the load regulator program removes the respective channel from the load dispatch tables of all processors in the center. To ensure that loss of queue entries due to queue overflow does not occur, an overflow queue is assigned when regular queues reach a certain level. If an overflow queue is being used, the overflow bits in the channel operations control table entry are set.

2.3.6 A and B Channels

The A and B channel time is used for execution of application programs and certain system program routines. The A and B channel initiation and completion program runs on A and B channel time and is responsible for the processing required for initiation and completion of channel programs. Work for the A and B channels is posted in the A and B channel work queues in the form of service messages by A and B channel queue service which is executed on S channel time. Application program call service messages are originated by control program service in the execution of a control program status record. The application program call service message is posted in the program status record, the calling control program status record obtained from disc file storage, and the calling program control instruction isolated. The program control instruction input list which identifies the input files necessary to execute this program is stored in the program status record and the program listing identified in the program control instruction is obtained from disc file storage and stored in the channel space following the program status record. The application program is then allowed to execute; upon completion, a return service message is sent via orderwire 1 to the calling processor identifying the output file created and allowing update of the control program status record so that subsequent program control instructions in the control program can be executed.

System routines that run on A and B channel time are initiated by program call service messages and are not necessarily initiated directly by a program control instruction nor do they all require completion response. These programs include the following:

a. System checkpoint
b. Device acquisition and control system library support
c. Schedule table generator
d. Control program status record generator program
e. Data collection service table update
f. Disc drain
g. Initialization, backup and recovery
h. Time division exchange message output
i. Disc zone purge
j. Load regulation
k. Time division exchange unsolicited input
l. Tape structure release and restore

2.3.7 M Channel

M channel time is used by the Multiplex Service Program (MSP) to service the requests of devices on the time division multiplex loop, such as CRT display and entry stations and teletypewriters.

The multiplex service program (figure 2-5) consists of a resident Multiplex Channel Control (MCC) program and various subroutines of the multiplex channel. When the multiplex service program is given control of the processors arithmetic logic and control unit facility from the operations control program, multiplex channel control monitors Multiplex Queue 1 (MQ1), Multiplex Queue 2 (MQ2), Multiplex Channel Queue (MCQ), File Transfer Command Queue 1 (FTCQ1), and File Transfer Command Queue 2 (FTCQ2) for channel activity status, invoking the proper routines to service requests as they are encountered. In the absence of any pending multiplex operations or if all outstanding operations are performed before the interval time period has elapsed, multiplex channel control relinquishes the remainder of channel time to the operation control program via the voluntary checkpoint entry (OP CKPT).

The time division multiplex loop is a serial, word-interlaced 1.2288-Mbps communication facility providing 256, 4.8-kbps time slots. Communication between the multiplex service program and devices on the time division multiplex loop is accomplished via the Multiplex Service Unit (MSU) which allows service request inputs and instruction outputs as well as data movement to and from core storage independent of the arithmetic logic and control unit. This is accomplished via a table of Multiplex Status Records (MSR's). There are 256 multiplex status records in the table, one for each time division multiplex time slot with a provision for 512 for full duplex devices. Each multiplex status record (figure 2-5) is four words in length. The first word, called the F word, is used to pass instructions to and from the device and to pass core addresses for field load and store operations. The D word is used for a data buffer when transmitting data a word at a time to or from a device. If an entire field is to be moved, the D word is not used. The R word is used in a link mode operation when two buffer areas or data bins are assigned in main core storage. In this case, the address of the first area is in the F word while the second is in the R word allowing multiplex service unit chaining from one to the other immediately without arithmetic logic and control unit intervention to assign a new bin. The P word is used for various control functions and as pointer to a channel status record.

Multiplex queue 1 and multiplex queue 2 are for queuing device to multiplex service program calls in the time division multiplex system. The queues contain pointers to multiplex status records to be serviced by the multiplex service program. The next-read-position pointer and next-write-position pointer position for these queues are maintained in the M channel operations control table. When a time division multiplex device has an instruction for the multiplex service program, the multiplex service unit places appropriate entries in the corresponding multiplex status record and in multiplex queue 1 or multiplex queue 2. The multiplex queue 1 and multiplex queue 2 entries signal the multiplex service program to service the associated multiplex status record.

Disc file storage data transfers from M channel are handled by M channel subroutines. An input and an output queue, file transfer command queue 1 and file transfer command queue 2, are provided for posting file transfer requests. A file transfer command queue service module of the multiplex service program services these queues directing the building of device control messages as required. The use of completion chains enable the M channel to monitor its own input/output for completion and the channel does not checkpoint busy to the operations control program awaiting input/output.

Orderwire 1 service messages directed to M channel are input through the multiplex channel queue which points to the orderwire 1 service message. Such a message may, for example, direct the output of a data file on disc storage to atime division multiplex device. M channel routines may send orderwire 1 service messages to applied systems via the time division exchange loop. The file transfer command queue service directs the building of the appropriate device control message and monitors its completion.

2.3.8 Common Functions

A common function is defined as a function that may be called directly or indirectly by more than one channel program being executed in either the A, B, M, orderwire 1, orderwire 2, or S channel. All common functions reside in protected memory and are characterized by being reentrant codings. In order to access protected memory, the arithmetic logic and control unit must be running in the privilege mode.

All programs executed in the privileged mode have access to all common functions. Programs executed in the unprivileged mode have direct access only to common functions for which a Branch and Set Return Link Protected (BRLP) code has been defined. The orderwire 1, orderwire 2 and S channels operate in the privileged mode while the A, B and M channels do not except through the BRLP linkage. The following paragraphs describe the common function shown in figure 2-5.

2.3.8.1 Common Space Allocation and Release

Common space allocation and release routines allocate common core space to calling programs and release core space from the calling programs cognizance. Space is allocated in bin size of 4, 8, 16, 32, 64, 128, 256, and 512 words in length. Allocation of core space is provided by the common space allocate routine, and release of core space is provided by the common space release routine. These routines are privileged and cannot be called from outside protected memory except indirectly through other common functions.

2.3.8.2 Disc File Storage Space Allocation and Release

Disc file space allocation and release is accomplished by a service function that provides housekeeping of allocated and released space and by common functions that interface with the channel programs. Disc file storage is partitioned into zones and each zone contains only one cell size. Cell lengths may be 128, 256, 512 or 2048 bytes. The number of cells within a given zone is a function of the cell size and the zone boundaries determined at initialization. Allocation of disc file storage zone space is provided by the zone allocation routine for allocating zones while allocation of disc file storage cell space within a zone is provided by the allocate cell routine. Documentation of active user files is provided by the document file routine. Release of a user file that was previously documented is provided by the release file routine which allows a complete file to be purged once it is declared inactive by the user.

File, zone, and cell space allocated and released is maintained in lists of the respective spaces. As new lists are needed, they are built and maintained on S channel time or by the invoking routine depending on the urgency of the request.

The allocate cell routine can only be invoked by programs operating in protected memory. This means that A, B, or M channel programs cannot obtain disc file storage space except through other common functions such as the file transfer commands. The zone allocation, document file, and release file routines are available to all channel programs via BRLP linkages.

2.3.8.3 Direct File Transfer

The direct file transfer routines are used by channel programs to transfer one cell at a time directly to input/output devices located on the time division exchange loop. Direct transfer of a cell to the processor core from disc or tape is provided by the read direct routine.

Direct transfer of a cell from the processor core to a disc or a tape is provided by a write direct routine. The transfer of a cell to or from disc is to the cell address specified by the user or by the disc storage space allocation routine. The transfer of a cell to or from a tape involves the next sequential record on the tape.

The physical control operation of a tape, such as rewind and backspace, is provided by the tape control transfer routine.

Direct transfer of a cell to devices other than tape or disc (e.g., CRT display and entry stations and line printers, located on the time division exchange loop) is provided by the device control transfer routine.

The direct commands are executed as part of the user program and are available to any channel programs via BRLP codes.

2.3.8.4 Indirect File Transfer

The indirect file transfer routines are common functions that enable channel programs to build and retrieve data files via a common file structure. This allows data from one program in the system to be used by other programs in the system regardless of which processor created the original file. Although disc file storage files have a physical structure different from tape files (random versus sequential access), the indirect file transfer commands make it possible for data files on disc to be derived from data files on tape.

The open input/output file routine initializes the indirect reading or writing of a file to storage. The open input file routine provides a direct file transfer routine to read the file's highest level connector cell into processor core. The routine also builds a bin in core for maintaining location of the current cell being read. The open output file routine builds a high level connector cell, invokes a direct command to write the file's highest level connector cell on disc storage, and builds a bin in core for maintaining the file location of the current cell being written.

The read indirect routine reads a data cell from disc or tape storage to the processor core. The cell can be the next sequential data cell in the file, or a selection process can be invoked to read a specific data cell from the file. The write indirect routine writes a data cell from core to the next available cell in storage. The disc address or relative position of the cell is entered into a system file connector which links together all data cells within a file.

The close file routine causes an output file to be closed but keeps it documented in the system so that it may be used by other channel programs in the system. A read file may be closed (if documented, then the file linkage block pointer is reusable and if tape, then the working channel is released).

The indirect commands are executed as part of the user program and are available to any channel program via BRLP codes.

2.3.8.5 Device Control Message Routines

Device control message routines are those common functions associated with obtaining device control messages and clearing blocked device control message chains (device control message queue).

To obtain device control messages, a privileged program invokes queue device control message routine which gives the address of the next available space in core to build a device control message. The queue device control message routine can only be invoked directly by other programs residing in protected memory.

When a device or channel error is detected by the communications service unit, it returns an indication of the error to the channel program and blocks the device control message chain (device control message queue). The communications service unit does not service any more device control messages in that chain until the error condition is cleared. To clear a blocked device control message chain, the channel program invokes the direct error correction routine so that input/output for that chain can be resumed. The direct error correction routine provides the option of skipping the device control message in error or skipping all device control messages associated with a direct command (that is, device control messages that were built by direct file transfer routines) and servicing only those device control messages associated with indirect file transfer routines or other indirect routines.

The direct error correction routines are executed as part of the user program and are available to any channel program via the BRLP code.

2.3.8.6 Service Message Transfer

Service message transfer is a common function that verifies and initiates a service message transfer request from a channel program. Verification consists of determining whether the service message is destined for an orderwire 1 channel of its own processor or another processor in the center, and determining whether the orderwire 1 code is valid. Initiation consists of building a device control message and initiating its execution by the communications service unit. This routine is available to all channel programs via a BRLP code.

2.3.8.7 Orderwire Translation

An orderwire 1 space release service message is provided in the C-System to release space from a disc managed by a processor. Orderwire translation service provides for the correlation of disc addresses to processor addresses. It accepts the loop 1 address and zone parameters of the disc and provides the orderwire 1 address of the processor responsible for that disc space. This routine is available to all channel programs via a BRLP code.

2.3.8.8 Routing and Conversion

The routing and conversion common function consists of three routines. The routing service routine provides for the conversion of symbolic machine addresses to absolute routing parameters. Every device connected to the C-System is assigned a C-Number to identify its physical address in the system. The routing service routine accepts the symbolic address of the device and converts it to routing parameters of: (1) the orderwire 1 address of a processor that is responsible for the subscriber referred to by the C-Number, (2) the address of a multiplex status record associated with a time division multiplex device, (3) device type and party line address parameters for a time division exchange device, or (4) the file identification of a nonresident service program.

The routing service routine requires a binary number for input. The C-Number conversion routine ASCII to binary operates on C-Numbers in ASCII code and converts them to their binary equivalent. C-Number conversion ASCII to binary can be invoked by any channel program via a BRLP code.

C-Number conversion routine binary to ASCII is utilized to convert the binary number used by the routing service routine to the ASCII code to allow the message to be relayed in characters. This routine can be invoked by any channel program via a BRLP code.

2.3.8.9 Object Program Mapping

Object program mapping routines are common functions that map object programs into processor core. The format of the object program is relocatable as produced by the macro assembler. The relocatable program loader routine provides for the loading of an object program within the upper and lower limits of channel space.

The object text address adjustment routine is capable of detecting relative address references within the text of an object program. By using the relocation base provided by the user, the relative address references are mapped to the channel core space into which the object program is being loaded.

These routines are available to all channel programs via the BRLP codes.

2.3.8.10 Timer 1 Interface

Timer 1 is a 2.4-ms timer available to channel programs. The timer 1 handlers provide the timer 1 interfaces to the unprivileged processing channels (M, B, A). There are four timer 1 routines and all can be called via the BRLP linkage. The declare channel timer 1 entry point routine accepts an address within the invoking channel space to which control is to be transferred upon subsequent timer 1 interrupts and optionally may reset/start timer 1 upon return to the channel. The reset/start timer 1 routine issues the reset/start timer 1 privileged instruction. If an entry point has not been declared by the channel for the timer 1 interrupt, the request is ignored. The stop timer 1 routine issues the stop timer 1 privileged instruction and if an entry point has not been declared by the channel for the timer 1 interrupt, the request is ignored. The start timer 1 routine issues the start timer 1 unprivileged instruction; the request is ignored if an entry point has not been declared by the channel. These routines are available to all channel programs via BRLP codes.

2.3.8.11 Trapped Operation Code Interface

The set channel entry point common function allows for declaration by a channel of an entry point within its own space to which control is to be automatically transferred upon occurrence of a trapped operation code interrupt. This routine is available to all channel programs via a BRLP code.

2.3.8.12 M Channel Linkage

Programs operating in the M channel, such as the multiplex service program and diagnostics, can use some of the privileged common functions and can write to protected memory. The multiplex service program uses the common function release bin to release orderwire 1 service message space. The multiplex service program also requires update of the multiplex queue pointers in its operations control table entry. The M channel diagnostic programs may alter the processor status word in protected memory in the testing of the multiplex service unit and time division multiplex devices. The M channel linkage routine allows M channel programs to use privileged common functions and to write to protected memory while retaining the integrity of a privileged operating system. M channel linkage is called via a BRLP code.

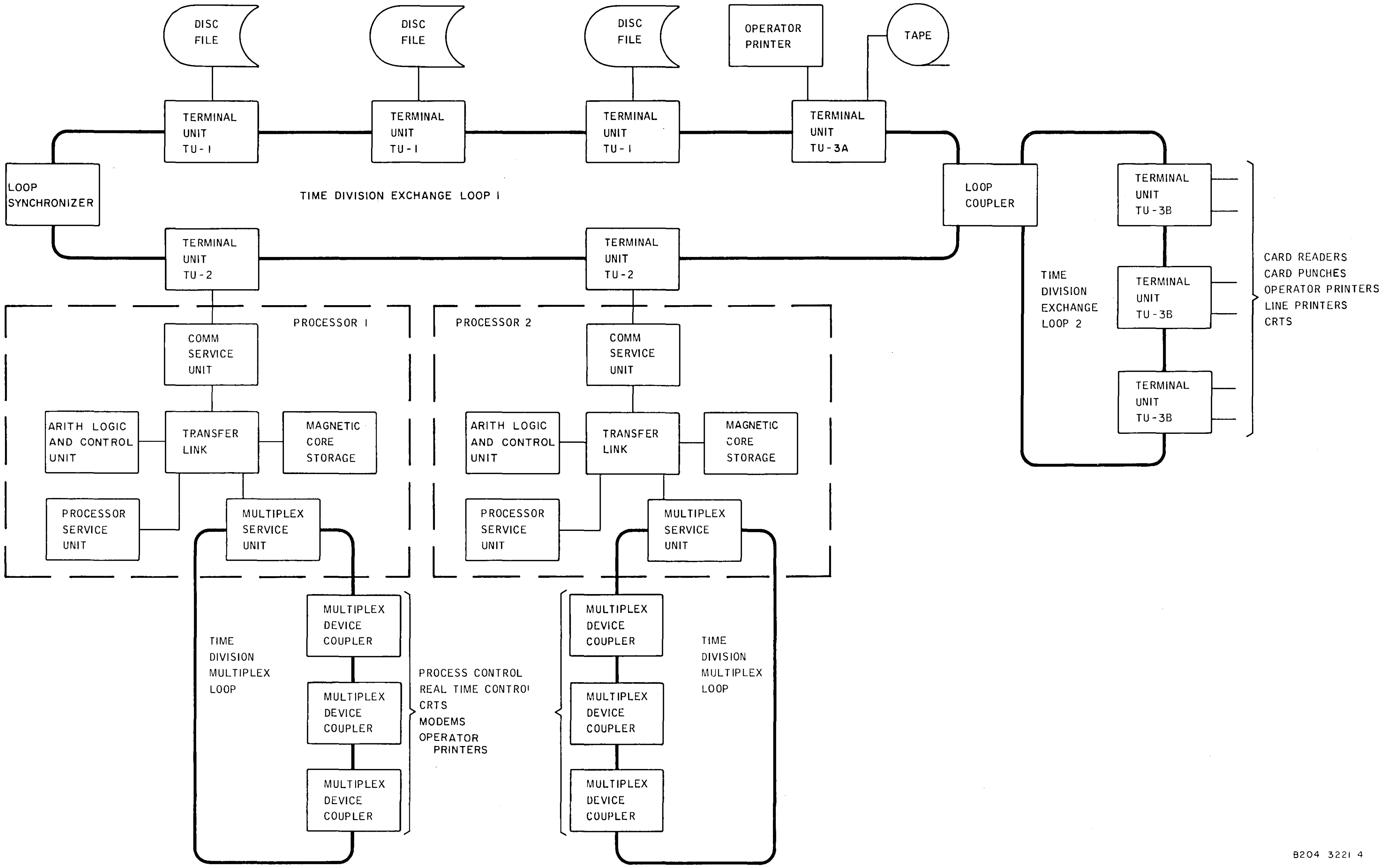
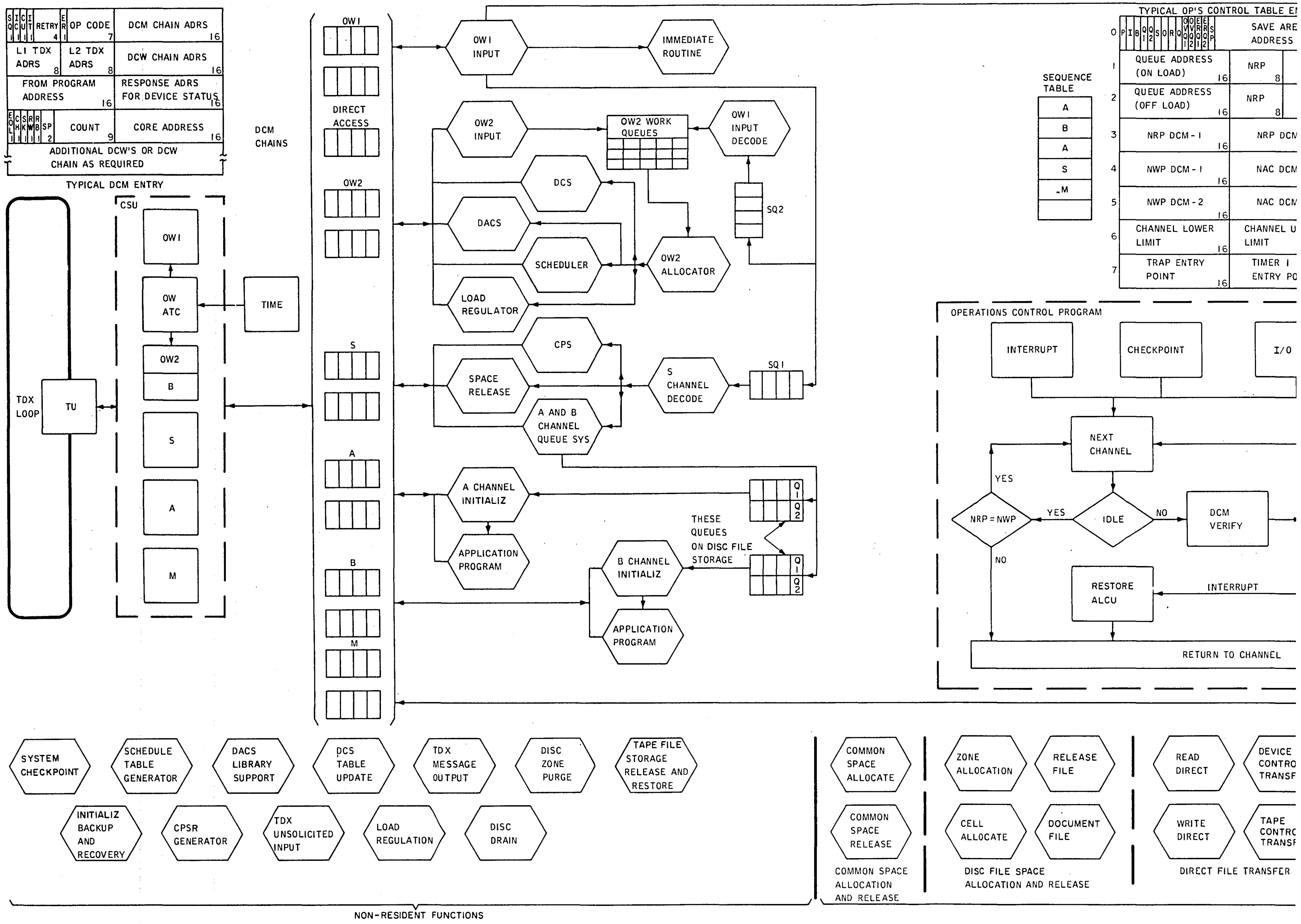


Figure 2-4. Typical C-System Computer Center.


S Q 1
I C 1
C U 1
I T 1
RETRY 4
E R 1
OP CODE 7
DCM CHAIN ADRS 16
L1 TDX ADRS 8
L2 TDX ADRS 8
DCW CHAIN ADRS 16
FROM PROGRAM ADDRESS 16
RESPONSE ADRS FOR DEVICE STATUS 16
E O L 1
C H 1
S K 1
R W 1
R B 1
SP 2
COUNT 9
CORE ADDRESS 16
ADDITIONAL DCW'S OR DCW CHAIN AS REQUIRED
TYPICAL DCM ENTRY
DCM CHAINS
TDX LOOP
TU
CSU
OWI
OW ATC
OW2
B
S
A
M
TIME
OWI
DIRECT ACCESS
OW2
S
A
B
M
OWI INPUT
IMMEDIATE ROUTINE
OW2 INPUT
OW2 WORK QUEUES
OWI INPUT DECODE
SQ2
DCS
DACS
SCHEDULER
OW2 ALLOCATOR
LOAD REGULATOR
CPS
SPACE RELEASE
S CHANNEL DECODE
SQ1
A AND B CHANNEL QUEUE SYS
A CHANNEL INITIALIZ
APPLICATION PROGRAM
Q1
Q2
THESE QUEUES ON DISC FILE STORAGE
B CHANNEL INITIALIZ
Q1
Q2
APPLICATION PROGRAM
TYPICAL OP'S CONTROL TABLE EN
0
P
I
B
Q1
Q2
S
O
R
Q
OVQ1
OVQ2
ERQ1
ERQ2
SP
SAVE ARE ADDRESS
1
QUEUE ADDRESS (ON LOAD) 16
NRP 8
2
QUEUE ADDRESS (OFF LOAD) 16
NRP 8
3
NRP DCM-1 16
NRP DCM
4
NWP DCM-1 16
NAC DCM
5
NWP DCM-2 16
NAC DCM
6
CHANNEL LOWER LIMIT 16
CHANNEL U LIMIT
7
TRAP ENTRY POINT 16
TIMER 1 ENTRY PO
SEQUENCE TABLE
A
B
A
S
M
OPERATIONS CONTROL PROGRAM
INTERRUPT
CHECKPOINT
I/O
NEXT CHANNEL
YES
NRP = NWP
YES
IDLE
NO
DCM VERIFY
NO
RESTORE ALCU
INTERRUPT
RETURN TO CHANNEL
SYSTEM CHECKPOINT
SCHEDULE TABLE GENERATOR
DACS LIBRARY SUPPORT
DCS TABLE UPDATE
TDX MESSAGE OUTPUT
DISC ZONE PURGE
TAPE FILE STORAGE RELEASE AND RESTORE
INITIALIZ BACKUP AND RECOVERY
CPSR GENERATOR
TDX UNSOLICITED INPUT
LOAD REGULATION
DISC DRAIN
NON-RESIDENT FUNCTIONS
COMMON SPACE ALLOCATE
COMMON SPACE RELEASE
COMMON SPACE ALLOCATION AND RELEASE
ZONE ALLOCATION
RELEASE FILE
CELL ALLOCATE
DOCUMENT FILE
DISC FILE SPACE ALLOCATION AND RELEASE
READ DIRECT
DEVICE CONTRO TRANSF
WRITE DIRECT
TAPE CONTRO TRANSF
DIRECT FILE TRANSFER

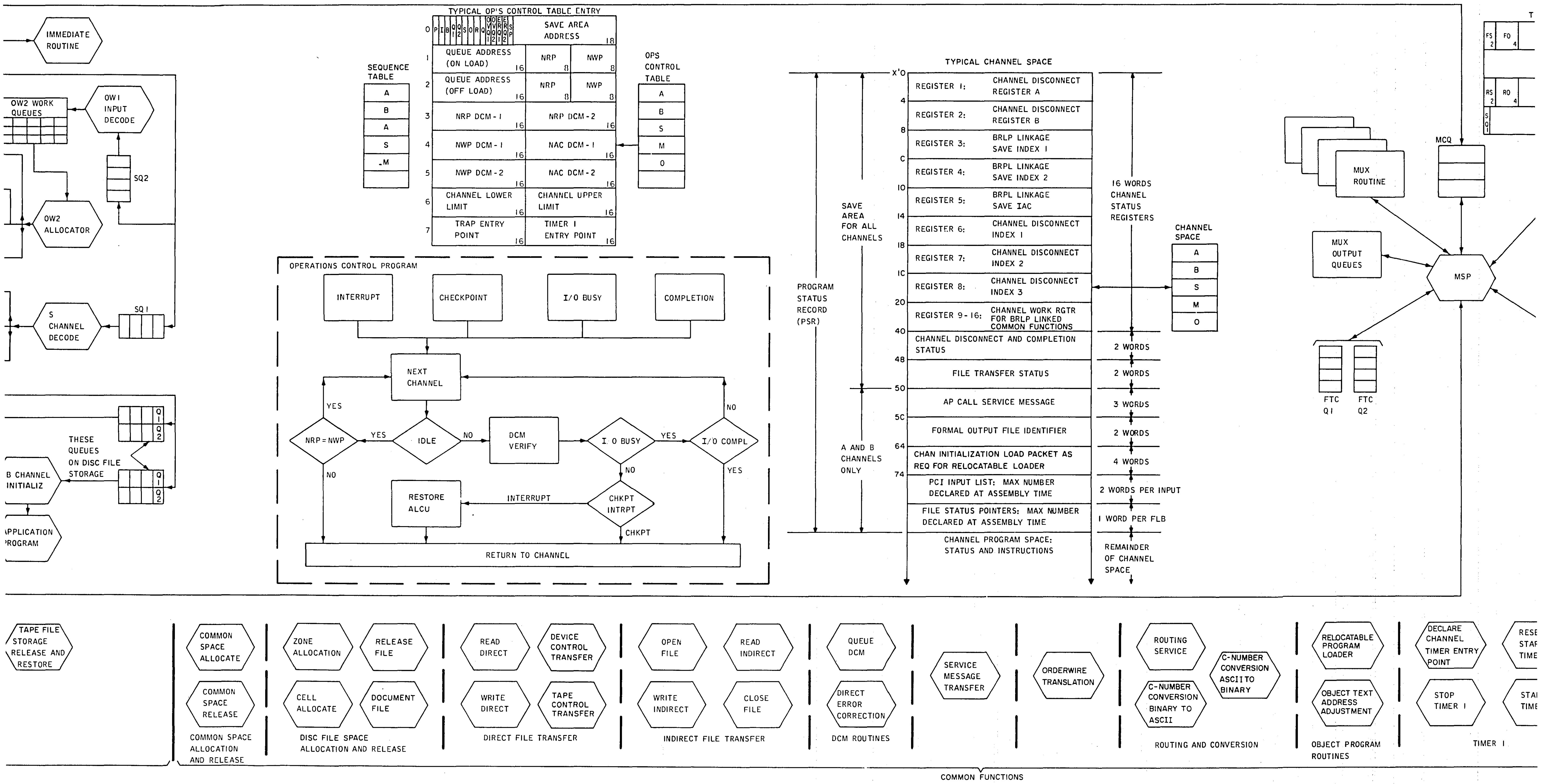


Figure 2-5. System S

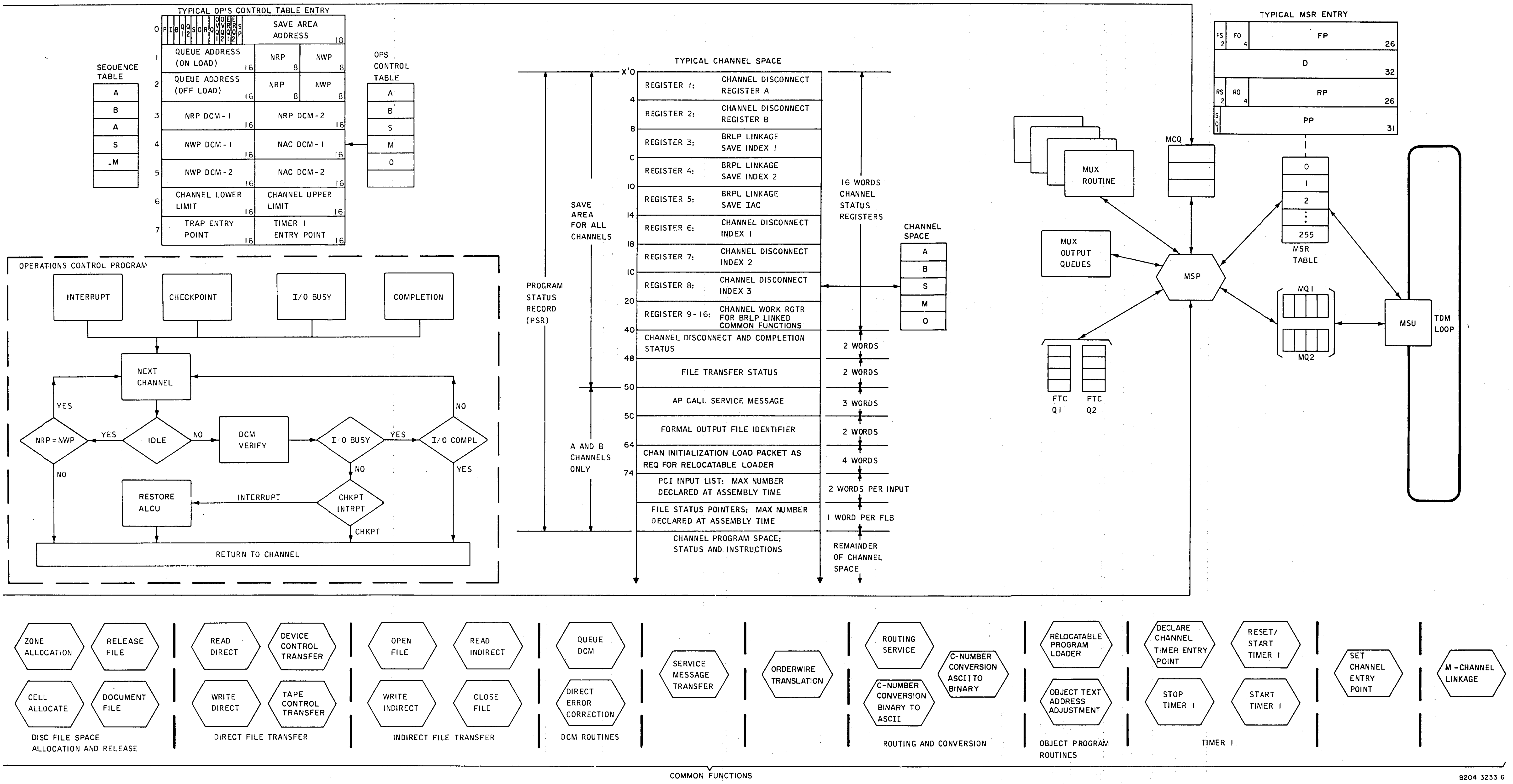


Figure 2-5. System Software.

section 3

processor operation, organization, and control

3.1 INTRODUCTION

This section describes the basic operation, organization, and control of a C-System processor. The organization of a processors core storage is shown with key interface tables. The operations control program which is responsible for arithmetic logic and control unit time sharing is described in detail.

3.2 PROCESSOR OPERATION

The following paragraphs describe the data formats used in a C-System processor, the concepts of privileged mode operation and protected memory, the execution of trapped operation codes, and the interrupts which may occur during the execution of a program.

3.2.1 Data Formats

A C-System computer has a basic data format of 8-bit bytes and 32-bit words. A byte can represent a character in an 8-bit code, or a byte can be one-fourth of a 32-bit word. Two bytes can make a halfword, and four bytes make a full word. In some instances, a double word of 8 bytes is used.

1 byte	=	8 bits
1 halfword	=	16 bits or 2 bytes
1 word	=	32 bits or 4 bytes
1 double word	=	64 bits or 8 bytes

Core storage is byte addressable. Only certain instructions manipulate bytes or characters; however, core storage is accessed by words and numerous instructions manipulate words. Word manipulations provide maximum speed for arithmetic and data handling operations. Byte manipulations, on the other hand, enable operations on variable length data and coded characters.

Words and halfwords are referenced (addressed) by the first (leftmost) byte. The length of the data is implied by the operation. Words, halfwords, and bytes must be properly aligned. Even addresses are halfword boundaries, and addresses divisible by 4 are word and double word boundaries. All addresses are byte boundaries. Table 3-1 shows the alignment restrictions.

3.2.2 Privileged Mode and Protected Memory

The processor operates in the privileged or unprivileged mode. To inhibit interrupts or to write to an area in protected memory, the processor must be in the privileged mode. The privileged mode is automatically entered whenever an interrupt occurs. The orderwire 1 channel, orderwire 2 channel, and S channel operate in the privileged mode while the A, B, and M channels do not operate in the privileged mode except through special linkage instructions. Protected memory includes at least the first 512 word locations (hexadecimal 00000 through 007FF). It can be expanded in modules of 512 words up to 16,384 words depending on system requirements.

The common functions reside in protected memory, thus, an interface between the unprivileged channels and the common routines must be provided. This is accomplished through a Branch and Set Return Link to Protected area instruction (BRLP). Execution of the BRLP instruction invokes the BRLP linkage handler routine that invokes the common routine defined by the BRLP code. Table 3-2 lists the common functions and the operations control entry points that can be accessed by use of the BRLP code. Other

Table 3-1. Alignment Restrictions.

DATA UNIT	ADDRESS OF LEFTMOST BYTE
Byte	Any address
Halfword	Even addresses
Word	Addresses divisible by 4
Double word	Addresses divisible by 4
Instruction word	Addresses divisible by 4

Table 3-2. BRLP Codes.

BRLP CODE	BSX/B MNEMONIC	SEMANTICS
0	OP ABNR	Channel Abnormal Completion
4	OP NORM	Channel Normal Completion
8	OP BUSY	Channel Disconnect Busy
12	OP CKPT	Channel Disconnect Checkpoint
16	CF SCR	Single Cell Release
20	CF FTO	Open Input/Output File
24	CF FTD	Read/Write Direct
28	CF FTI	Read/Write Indirect
32	CF FTC	Close File
36	CF DFM	Document File
40	CF RFM	Release File
44	CF RTS	Routing Service
48	CF RPR	Relocatable Program Loader
52	CF SMT	Service Message Transfer
56	CF SCT	Set Channel Trap

Table 3-2. BRLP Codes (Cont).

BRLP CODE	BSX/B MNEMONIC	SEMANTICS
64	CF TCT	Tape Control Transfer
68	CF DCT	Device Control Transfer
72	CF DEC	Direct Error Correction
76	CF ZAL	Zone Allocation
80	CF MLINK	M Channel Linkage
84	CF OTS	Orderwire Translation Service
88	CF CACB	C-Number Conversion - ASCII to Binary
92	CF CBCA	C-Number Conversion - Binary to ASCII
96	CF RLOC	Object Text Address Adjustment
256	---	Declare Channel Timer 1 Entry Point
260	---	Reset/Start Timer 1
264	---	Stop Timer 1
268	---	Start Timer 1

common functions (queue device control message, allocate cell, common space, single cell release) are not available to unprivileged processor channels except indirectly through the common functions listed in table 3-2. Calls from programs operating in the privileged mode are made via the Branch and Save Index Register (BSX) or Branch (B) instructions and the mnemonic shown in table 3-2.

3.2.3 Trapped Operation Code Execution

A set of instructions are available in the C-System assembly language that are not fixed-wired but are implemented by a trapping mechanism. Upon detection of a trapped operation code, the arithmetic logic and control unit status is stored in fixed locations in protected memory and control is transferred to the trapped operation code handler routine residing in protected memory. This function provides for transfer of control to the trapped entry point declared in the operations control table of the channel executing on the arithmetic logic and control unit when the trapped operation code was encountered.

3.2.4 Interrupts

Three types of interrupts can occur during program execution: program interrupt, parity error interrupt, and timer interrupt. The following paragraphs describe the interrupts.

3.2.4.1 Program Interrupt

A program interrupt occurs under any of the following conditions:

a. An invalid address is detected.
b. An invalid operation code is detected.
c. A direct control instruction is attempted with the processor in the unprivileged mode.
d. A write into protected memory is attempted with the processor in the unprivileged mode.

An invalid address is detected when a memory address exceeds the highest address in the implemented memory or when a write to protected memory is attempted with the processor in the unprivileged mode. A program interrupt causes the program interrupt handler to be invoked and status to be saved at a fixed location in core storage.

3.2.4.2 Parity Interrupt

A memory parity interrupt occurs when a memory parity error is detected during a memory read initiated by the arithmetic logic and control unit. This interrupt causes status to be saved and the parity interrupt handler to be invoked. The memory interrupt cannot be masked.

3.2.4.3 Timer Interrupts

There are two types of timer interrupts used in the C-System. The first is a timer 0 interrupt; timer 0 is the basic channel timer for the implementation of the arithmetic logic and control unit time sharing. This timer allows a channel approximately 24 ms of run time before interrupting the channel causing operations control to be entered and the next entry in the sequence table serviced.

Timer 1 is a 2.4-ms timer available to the user of unprivileged processing channels A, B, and M and is used to time any function for which a 2.4-ms period is appropriate. Timer 1 is implemented by declaring an entry point to which control is transferred at the end of the period by a hardware interrupt. Timer 1 is reset and started or stopped upon command.

If the channel gives up its arithmetic logic and control unit time voluntarily by either the busy or checkpoint call upon the operations control program, timer 1 will be inoperative upon reinstatement of the channel (a reset/start timer 1 call initiates it). The occurrence of a timer 0 interrupt (involuntary checkpoint of channel) is transparent to the channel, other than a possibility that up to 4.8 ms of channel time would pass before a timer 1 interrupt. The arithmetic logic and control unit time required by the common functions called by the channel is not included in the 2.4-ms timer 1 interval.

3.3 CORE STORAGE ORGANIZATION

A C-System processor may use either two or four magnetic core storage units, each with a 16,384-word storage capacity. Depending on the processor environment and functions to be performed, certain software channels are resident in core storage. The exact order and particular combination, in which the software elements exist in core varies from processor to processor; see figure 3-2. Not all software elements identified in the figure can exist in one processor because of core size limitations; for example, only one processor in a center requires an orderwire 2 channel and that processor does not have a B channel.

To illustrate the system flexibility, consider the multiplex channel. The channel buffer requirements are a function of the number of devices being serviced on the time division multiplex loop; this dependence reflects the need for data buffers in core for input and output of data from the devices. Depending on the time division multiplex loop loading, it may be desirable to delete the A or B channel to obtain more core storage for buffer space for the M channel. Also, a processor with no time division multiplex loop would require no multiplex channel space. Other variations are possible and the final configuration depends on the users requirements.

The first X'4FF (hexadecimal) bytes of core storage are fixed address and contain the processor interface table, the operations control table, Multiplex Queue 1 (MQ1), and Multiplex Queue 2 (MQ2) for the multiplex channel, and pointers to the direct access device control messages.

The device control message chains are next. There are two device control message chains for each communications service unit data channel plus the direct access device control messages. The program and parity interrupt handlers follow next; then the 15-entry operations sequence table, the trapped operation code handler, the BRLP handler, the timer 1 handler, the operations control program, A/B channel initiation and completion, and the common functions.

The channel space for each of the resident channels is located next in core. Each channel space consists of the program status record plus the channel program space. Service queue 1 and service queue 2 are maintained in the S channel space. The M channel queues, except multiplex queue 1 and multiplex queue 2, and the orderwire 2 channel work queues are similarly maintained in the respective channel spaces. (The A and B channel work queues and the multiplex channel output queues are maintained on disc.) The multiplex status records are maintained just ahead of the M channel spaces.

Of the software shown in figure 3-2, the software through the orderwire 1 channel space is required in every processor of a fully operational multiprocessor C-System center. This software currently occupies approximately 20K words of core storage. The amount of core storage required for the remaining channels is shown in the figure. No more than three channels from the A, M, orderwire 2, B1, and B2 channels may be resident in a single processor.

3.3.1 Processor Interfaces

Interfacing the arithmetic logic and control unit, communications service unit, processor service unit, and multiplex service unit with programs and data in core storage requires the maintenance of certain interface information. Key elements in this interface are the processor interface table, the operations control table and the operations sequence table, the device control message, and the multiplex status record and multiplex queues. Detailed definitions of the processor interfaces are provided in the appendixes.

3.3.1.1 Processor Interface Table

The processor interface table contains the fixed core addresses of certain software routines as well as fixed locations for storage of information required during operation. The first X'2F bytes of this table are spare. Word X'30 is used by data channel 4 and the orderwire absolute time clock modules of the communications service unit; for direct access device control messages, the orderwire absolute time clock module places the core address of the required device control message in this location; for normal orderwire 1 service messages, the Next-Available-Cell pointer (NAC) from the orderwire 1 operations control table is inserted. Byte X'38 contains the data channel 4 core boundary which corresponds to the orderwire 1 channel upper and lower limits as called for in the operations control table.

Byte locations X'40 through X'47 of the processor interface table are used by the processor service unit for status. The 4 bytes starting at byte location X'40 are the Processor Control Word (PCW) and are used to control the state of the processor service unit. The Processor Hardware Status Word (PHSW) starts at byte location X'44 and stores processor hardware status.

The word starting at byte location X'48 is used by the absolute time clock unit for storing the system time word transmitted by the loop synchronizer.

Byte locations X'4C through X'7F of the processor interface table are used by orderwire 2 for recording busy-idle status of time division exchange working channels; the TU-2 corresponding to the orderwire 2 channel continually requests status from all TU-3's connected to the center when the orderwire 2 time division exchange channel is idle. The continuing status request may be interrupted at any time by a bid for orderwire 2. Each valid working channel has a unique bit in this 13-word status request. Byte locations X'80 through X'9F are spares.

Byte locations X'A0 through X'BF are used to interface the program interrupt, parity interrupt, timer 1 interrupt, and Branch and Set Return Link Protected (BRLP) interrupt handlers. Eight bytes are used by each handler. The first four bytes point to the location in core storage of the handler routine and the last four allow storage of the previous instruction address from the program being executed and arithmetic logic and control unit control information.

The word starting at byte location X'CO is the 32-bit B accumulator. The word starting at byte location X'C4 is used by the algorithm unit for manipulation of the A accumulator of the arithmetic logic and control unit.

The four words starting in a byte location X'DO are used by the trapped operation code handler and identify the core location of the handler as well as providing storage area for arithmetic logic and control unit status when executing a trapped operation code.

The word starting at byte location X'F8 (OPSTAC) is maintained by the operations control table and is the core address of the current operations sequence table entry. OPSLOW is also maintained by the operations control program and is the starting address of the current channel program status record.

3.3.1.2 Operations Control Table and Operations Sequence Table

The operations control table occupies core storage from byte location X'100 through X'1FF and contains space for eight entries. Each entry is eight words in length and a typical entry is shown in figure 3-2. The first word of each entry contains status bits used by the operations control program to record channel status and the beginning address of the channel space. The second and third words contain the locations of the channel work queues along with their Next-Read-Position (NRP) and Next-Write-Position (NWP). For S channel, the second and third words refer to Service Queue 1 (SQ1) and Service Queue 2 (SQ2); for M channel, to multiplex queue 1 and multiplex queue 2; and for A and B channels, the A and B channel work queues on disc storage. The orderwire 1 and orderwire 2 channels do not use these entry points except for orderwire 2 using service queue 2 for receipt of orderwire 1 service messages. The next three words are pointers to the channel device control message chains next-read-, next-write-, and next-available-cell-positions. The channel space upper and lower limits are in the seventh word. The eighth word contains the trap entry point and timer 1 entry point for the trapped operation code handler and timer 1 handler.

The operations sequence table contains up to 15 entries to identify the sequence in which the channels are to be given channel time. The entries are the core addresses of the respective operations control tables.

3.3.1.3 Device Control Messages

In addition to the normal Device Control Message (DCM) chains, direct access device control messages exist which are used by certain system programs to read or write directly to certain tables in core; the load regulator is a user of these device control messages. Byte locations X'400 through X'4FF contain pointers to these device control messages.

The Next-Read-Position (NRP), Next-Write-Position (NWP) and Next-Available-Cell (NAC) are maintained in the operations control table for each normal device control message chain in each channel. The next-read-position is the next location the operations control program investigates to determine if a device control message has been serviced by the communications service unit; the next-write-position is the next location available to build a device control message, and the next-available-cell is the next device control message to be accessed in the device control message chain by the communications service unit.

The first word of the device control message contains status and control bits and the device control message chain address. The second word contains the time division exchange loop 1/loop 2 address of the device to be communicated with and the Data Control Word (DCW) chain address. The third word contains the from-program-address to which control may be transferred by operations control when verifying completion of this device control message. Control is transferred when an input/output operation error occurs and is detected by the communications service unit or when the channel program initiating the input/output operation directs immediate return of control to the from-program upon operations control verification of the completed input/output operation by setting the connector update bit. The third word also contains space for recording device status at completion of the data transfer.

The remaining area of a device control message is used for device command words and data control words which identify the transfer to be completed and the core address to which the data is to be written to or read from. Data control words may also point to other data control words located elsewhere in core. When such a data control word is encountered, the address is placed in the data control word chain address area of the second word and then referenced at an appropriate time in the execution of device control message chains.

3.3.1.4 Multiplex Status Records and the M-Queues

Multiplex Queue 1 (MQ1) and Multiplex Queue 2 (MQ2) are the M channel work queues. The next-read-and next-write-positions for these queues are maintained in the M channel operations control table entry. Multiplex queue 1 is used as an input queue. The multiplex service unit places the identification of the multiplex status record to be serviced by the multiplex service program in multiplex queue 1. The multiplex status record contains sufficient information for the multiplex service program to perform the directed action. Multiplex queue 2 is used as an output queue. The multiplex service unit places the identification of the multiplex status record to be serviced by multiplex service program in multiplex queue 2. There are 256 possible entries in both multiplex queue 1 and multiplex queue 2 and 256 multiplex status records can be allocated for time division multiplex devices with provisions for 512 multiplex status records for full duplex operation.

3.3.2 Channel Spaces

The space for each of the processor channels is located after the common functions. A typical channel space (figure 3-2) consists of a Program Status Record (PSR) and a channel program space. The contents of the channel program for each processor channel space are shown in the figure. The M channel program status record is preceded by the Multiplex Status Records (MSR). The first eight words of the program status record are used for storing arithmetic logic and control unit registers after a timer interrupt or during the execution of a BRLP code. Words 1, 2, 6, 7, and 8 are used to save work registers A and B and index registers 1, 2, and 3. Words 3, 4, and 5 are used for saving index registers 1 and 2 and the instruction address counter during execution of a BRLP command. Words 9 through 16 are used as channel work registers for the BRLP linked common functions.

The file transfer status is used by the A and B channels to store parameters relative to files being read or written by the channel. The A and B channels have five additional program status record entries between the file transfer status and the channel program space. The first entry is the Application Program (AP) call service message that directs the execution of the application program currently being run and identifies the processor to which a return message is sent upon completion of the program. The output file identifier identifies any output files created by the execution of this program and is returned to the invoking control program as part of the return service message. The next space is reserved for the channel initiation and load packet used by the relocatable program loader to load the application program into core. The next entry is the Program Control Instruction (PCI) input list from the calling control program status record which identifies the files required for execution of the application program. File status pointers are maintained for the A and B channel that contain information identifying the current location in the reading or writing of a structured file by the indirect file transfer commands.

3.3.3 Common Core

Common core is available to privileged routines through the common space allocate common function. Release of common core is via the common space release common function. Common core is allocated in bin sizes of 4, 8, 16, 32, 64, 128, 256, and 512 words.

3.4 OPERATIONS CONTROL PROGRAM

Control of processor time and monitoring of device control message chains for completion of time division exchange input/output is accomplished by the operations control program. When the operations control program is entered, it examines the operations sequence table, operations control table, and device control message chains to determine where to return control. Figure 3-3 shows essential steps taken by the operations control program. A typical operations control table entry and a typical device control message chain are also shown.

3.4.1 Basic Operations Control

The operations control program may be entered under one of four conditions: (1) timer 0 interrupt (OP INT), (2) A or B channel completion of an application program being executed (OP COMP), (3) channel awaiting a time division exchange data transfer relinquishes the remainder of its interval time (OP BUSY), or (4) channel voluntarily relinquishes the remainder of its time when certain conditions exist in the execution of the channel programs (OP CKPT). The operations control program resides in protected memory and only entrances (3) and (4) may be used by an unprivileged program via a BRLP instruction.

A timer interrupt during execution of the operations control program sets a switch and control is immediately returned to the operations control program at the point it was executing. Periodically, the operations control program tests this switch to determine if a timer 0 timeout has occurred. If the switch is set, operations control completes the operation in progress and returns to the sequence table to determine which channel to service next, thus avoiding operations control program interrupts. Note the difference between this operation and interrupt masking which causes the interrupt to be completely ignored.

Timer 0 is started for all entries to operations control. For an interrupt entry, the arithmetic logic and control unit registers and channel status are stored in the channel space. The second bit (I) of the operations control table indicates if a channel was idle or not idle the last time it passed control to operations control. If this bit is a 1, the channel was idle; if the bit is a 0, the channel was not idle. This bit is set to 1 by the completion entry (OP COMP) to operations control. It is a 0 for all the other entries. The third bit of the first word indicates whether the channel is awaiting a time division exchange data transfer; it is set to a 1 if the channel is awaiting such a transfer. This corresponds to the OP BUSY entry to operations control. For the OP COMP, OP INT and OP CKPT entries, it is set to a 0.

Having set the proper status bits and saved the registers if necessary, the operations control program reads and advances the operations sequence table pointer. The operations sequence table is a 15-entry table. Each entry points to an operations control table entry to be serviced. The last bits of the first word of the operations control table identify the location of the channel space in core storage.

The first word from the operations control table entry for the channel indicated by the operations sequence table is accessed, and the I bit is checked to determine the idle status of the channel when it last had control. If the channel was idle, the operations control program continues to process the operations control table for any work that has been posted for this channel since the channel last had control. The third and fourth bits (Q1 and Q2) are system regulation bits and may be set by the load regulator program to block either queue 1 or 2 from further execution. If both work queues are inhibited, operations control returns to the sequence table to determine which channel to service next. If only one of the queues is inhibited, the other is serviced. If neither queue is inhibited, the S bit is checked to determine which of the two queues to service next. For the queue selected, the Next-Read-Position (NRP) is compared to the Next-Write-Position (NWP). The next-read-position is the next entry in the queue to be processed and the next-write-position is the next location in the queue to enter a request for service. If the next-read-position is equal to the next-write-position, no work has been posted in that queue and the alternate queue is tested if its inhibit bit is not set. If the next-read-position equals the next-write-position for both working queues, there is no new work for the channel and the operations control program returns to the sequence table to determine the next channel to be serviced. If work is found in either queue, the operations control program exits to that channel. The S bit is switched by the operations control program to ensure alternate service of the queues. Only the A and B channels use this test for new work in operations control. All other channels manage their own work.

If the channel was not idle, the device control message complete counter is set to 0.

The operations control program next investigates bits 11 and 12 (E1 and E2) of the first word of the operations control table. The device control message chain error bits for chains 1 and 2 are set by the operations control program when an unresolved error results in the execution of a device control message input or output request execution by the communications service unit. If both chains are blocked, the communications service unit ceases servicing the device control message chain until the error condition is cleared; control must be returned to the channel program. Before returning control, a check determines if an interval timeout has occurred since the timer was reset and started. If it has, the operations control program returns to the operations sequence table to obtain the next channel to be executed. If there has not been an interval timeout, the operations control program returns to the channel program after determining whether the last relinquishing of time was voluntary or involuntary. For an involuntary interrupt, the arithmetic logic and control unit registers and status must be restored. For the voluntary interrupt, only the Instruction Address Counter (IAC) directing which instruction to be executed next must be restored. The instruction address counter and voluntary/involuntary information are obtained from the channel disconnect and completion status entry of the channel program space of the program status record.

If one or both of the device control message chains is not blocked by an error, the operations control program proceeds with verification of completion status of the channels device control message chains. If both device control message chains are free of error, they are checked alternately, one device control message at a time. The operations control table next-read-position for the selected device control message chain is checked to determine which device control message to investigate. Verification of a device control message involves checking whether or not the device control message has been serviced by the communications service unit, updating the completion count and device control message next-read-position for each chain when each device control message is completed, and invoking a from-program indicated by the address in the third word of the device control message if an error has occurred during the device control message execution by the communications service unit or if the connector update bit of the device control message is set requiring branching to the from-program by operations control. The connector update bit is set by the program constructing the device control message if it is desired immediate return from the operations control program after the device control message execution. One user of this branch is the indirect read and write routine which may be reading many cells from disc or tape file storage or searching a file for some application system.

After verification of each device control message by operations control, a test for a timeout occurrence is made. Device control message verification continues until either a timeout occurs, both channel device control message chains have been completely verified, or a device control message with an input/output error is encountered. After device control message verification, operations control decodes the B bit of the operations control table to determine if the channel was awaiting an input/output (OP BUSY). If the channel was not awaiting a data transfer, a test for an interrupt is made and the same path followed as when both device control message chains were blocked. If the channel was awaiting a data transfer, the device control message completion count is checked to determine if any device control messages were verified as complete during device control message verification. If at least one device control message was verified, an interrupt test is made. If no interrupt has occurred, control is returned to the channel. If an interrupt has occurred, the B bit in the operations control table is switched to reflect involuntary interrupt and operations control resets and starts the timer and advances to the next channel indicated in the sequence table.

If no device control message has been verified, a test is made to determine if a device control message execution error was discovered that could not be corrected by the from-program. If such a device control message execution error had occurred, the error bit in the operations control table (E1 or E2) corresponding to the device control message chain in which the device control message resides has been set during device control message verification and control is returned to the channel whether or not other device control messages have been verified as complete. If a device control message permanent error has not been detected, the timer is reset and started and the next channel indicated in the sequence table serviced by operations control.

3.4.2 Device Control Message Verification

A typical device control message chain is shown in figure 3-3. This chain contains only two device control messages; a chain may contain any number of device control messages, but the number is determined at assembly, space provided, and the individual device control messages are chained together with each device control message pointing to the next in the chain. Device control messages are built on channel time in space allocated by the queue device control message function. The next place to build a device control message is maintained in the next-write-position pointers in the channel operations control table. The common function queue device control message routine verifies that this space is available and is not awaiting verification or communications service unit execution. The next-available-cell pointers in the operations control table identify the next device control message to be executed by the communication service unit and is updated by the communication service unit.

When device control message verification is entered, the Service Queue (SQ), Immediate Chain (IC), Connector Update (CU), and Error bits (ER) in the first word are examined. These bits, together with the device control message chain next-available-cell and next-read-position pointers in the channel operations contrl table determine the course of action to be taken. Table 3-3 presents a detailed summary of the action taken by device control message verification depending on the status of these bits and the next-read-position and next-available-cell.

The communications service unit processes a device control message if the service queue bit is set to 0. Upon completion of the device control message processing, the communications service unit sets the service queue bit to 1. If the immediate chain bit is set to a 1 and the service queue bit to a 0, the communications service unit chains around the device control message and sets the service queue bit to a 1. The communications service unit always updates the next-available-cell in the operations control table after servicing or chaining around a device control message. If the communications service unit detects an error in the input/output operation, the error bit is set, the service queue bit set to a 1, and the next-available-cell is not updated. When the communications service unit encounters a service queue bit equal to a 1, the communications service unit has completed the device control message chain; that is, no more device control messages remain to be verified and the communications service unit stops servicing the chain. The communications service unit continues to monitor the service queue bit and when the service queue bit is set to a 0, the communications service unit services the device control message. Since the next-available-cell is not updated if an error occurs, but the service queue bit is set to a 1, an error causes the communications service unit to cease servicing the chain until the error is resolved and the chain cleared by setting the service queue bit to a 0.

As shown in figure 3-3, the service queue bit is the first bit tested by device control message verification. The service queue bit of a device control message is always set to 0 by the processor when a device control message is built and is set to a 1 by the communications service unit when executing the device control message. If the service queue bit is a 0, the communications service unit has not processed this device control message. Note that the device control messages in a given queue form a cyclic chain as

Table 3-3. DCM Chain Verification.

STATUS BITS				CHAIN POINTERS NAC = NRP	INTERMEDIATE ACTION BY DCM VERIFICATION	NRP UPDATED	COMPLETE COUNT UPDATED	CHAIN COMPLETE
SQ	IC	ER	CU					
0	NC	NC	NC	NC	None (No verification required)	No	Yes	No
1	0	0	0	NC	Set IC bit to 1.	Yes	No	Yes
1	0	0	1	NC	Branch to from program; set IC bit to 1.	No	No	Yes
1	0	1	NC	NC	Branch to from program; from program either clears error or declares a permanent error and sets the error bit for the chain in the operations control table.	No	Yes	No
1	1	NC	NC	Yes	None	Yes	Yes	No
1	1	NC	0	No	None, continue.	No	No	Yes
1	1	NC	1	No	Branch to from program.	No	No	Yes

Note 1: NC means not checked.

Note 2: There is a minimum of two DCM's per chain for verification.

Note 3: DCM status bits initial state is
SQ = 1, IC = 1, ER = 0, CU = 0
DCM chain pointers are initially equal (NAC = NRP = NWP)

shown in figure 3-1. At initialization, the next-read-position equals the next-available-cell position and the next-write-position and the service queue bit is a 1 in all device control messages. When the first device control message is built, the next-write-position is incremented and the service queue bit is set to a 0 in this device control message. The communications service unit then services this device control message, changes the service queue bit to a 1, and updates the next-available-cell in the operations control program. This device control message is then verified by the operations control program and the next-read-position is updated. The system software is so constructed that the next-write-position, next-available-cell and next-read-position always remain in that order around a device control message chain; the next-available-cell never passes the next-write-position, and the next-read-position never passes the next-available-cell. (At most, they can be equal.) Hence, if the service queue bit is a 0 at the next-read-position, then the next-read-position must equal the next-available-cell and it can be concluded that there are no completed device control messages in the chain; that is, there have been no input/output transactions completed that have not been verified by operations control. In this case, processing of this device control message chain by the operations control program is complete.

If the service queue bit is a 1, the next bit tested is the immediate chain bit. Assume that this bit is a 0 and the error bit has not been set by the communications service unit. If the connector update bit is set, device control message verification branches to the from-program address indicated in the third word of the device control message. If the connector update bit is not set, the device control message verification updates the complete count. In either case, the next-read-position is updated and the immediate chain

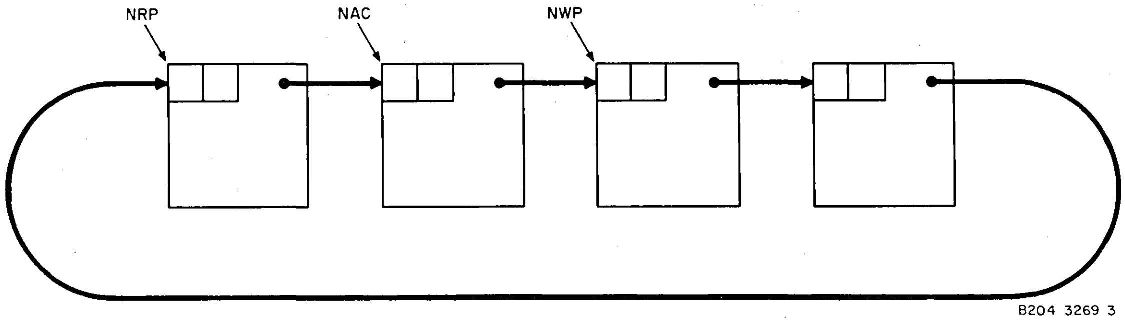


Figure 3-1. NRP, NAC, and NWP Relationships.

bit is set to 1. The device control message verification is not complete at this point; operations control services the alternate device control message chain next, and then operations control returns to the next-read-position. As long as the service queue bit is a 1, the immediate chain bit is a 0, and the error bit is a 0, this procedure continues until device control message verification encounters the service queue bit, a 0 indicating that the next-read-position equals the next-available-cell or the service queue bit a 1 and the immediate chain bit a 1, indicating that device control message verification has completed the chain. In this case, the next-read-position equals the next-available-cell and verification of the chain is complete.

The service queue bit equal to a 1 and the immediate chain bit equal to a 1 can be encountered by device control message verification in another manner. It is possible, in an error condition, for the handler to set the service queue bit to a 0 and the immediate chain bit to a 1, causing the communications service unit to chain around the device control message, setting the service queue bit to a 1. Hence, device control message verification may encounter the service queue bit equal to 1, the immediate chain bit equal to a 1, and the next-read-position not equal to the next-available-cell. In this case, the connector update bit is tested to determine whether or not to branch to the from-program. The next-read-position is updated and verification continues.

If the service queue bit is a 1 and the immediate chain bit is a 0, the error bit is tested. If the communications service unit detected an error during the data transfer, the error bit is set to a 1 and device control message verification branches to the from-program. In this case, the from-program tries to correct the error by retrying the transfer. If the error cannot be corrected, the error bit in the operations control table is set to indicate a permanent error and the channel program is given control if it was waiting for a data transfer. Whether or not the error is cleared during device control message verification, verification on this chain is terminated.

3.4.3 Timer Interrupts

In addition to directing allocation of arithmetic logic and control unit time, and verification of completed device control messages, the operations control program handles timer 0 and timer 1 interrupts. Upon occurrence of either a timer 1 or timer 0 interrupt, control is transferred to the interrupt location in the operations control program which then determines the course of action to be followed.

3.4.3.1 Timer 0 Interrupts

A timer 0 interrupt during a channel time always causes branching of control of the OP INT entry of the operations control program unless the interrupt occurs during execution of the operations control program. In this case, a switch is set to 1 to indicate that the interrupt has occurred and control is returned to operations control. Operations control periodically checks this switch to determine whether or not an interrupt has occurred while it has been executing.

3.4.3.2 Timer 1 Interrupts

Timer 1 is a 2.4-ms timer available to unprivileged processor channels. Bit 0 of the operations control table indicates whether a channel is running as privileged (1) or unprivileged (0). Bits 6 and 7 of the status word of the operations control table entry for each channel indicate the status of timer 1 at interrupt as follows:

Bit	Definition
6	0 = Timer 1 inoperative 1 = Timer 1 operative
7	0 = Timer 1 stopped 1 = Timer 1 running

These bits are set by the timer 1 common functions as follows:

COMMON FUNCTION	BIT 6	BIT 7
Declare Channel Timer 1 Entry Point	0	0
Reset/Start Timer 1	1	1
Stop Timer 1	1	0
Start Timer 1	1	1

A timer 1 interrupt is ignored if it occurs during execution of the operations control program, or if the channel has not declared an entry point or if an entry point has been declared and bit 6 is a 0. If the channel is executing in the privileged mode, an entry point has been declared and bit 6 is set to 1. Timer 1 is set to 0 and stopped and control is returned to the channel at the interrupt address. If an entry point has been declared, the channel is executing in an unprivileged mode and bits 6 and 7 are both 1, operations control sets bits 6 and 7 both to 0 and returns to the declared entry point indicated in the operations control table entry; if bit 7 is 0, timer 1 is reset and stopped and the channel returned to at the interrupt address.

On return to a channel which was executing in the unprivileged mode and involuntarily checkpointed by timer 0 with the timer 1 running, timer 1 is reset to 0 and started and the channel returned to at the interrupt address. If bit 6 was set but the channel was operating in the privileged mode when interrupted, timer 1 is reset to 0, stopped, and the channel returned to at the interrupt address. If bit 6 was set, the channel operating unprivileged and timer 1 was not running (bit 7 = 0), timer 1 is reset and stopped before return to the channel from an involuntary checkpoint.

Table 3-4 summarizes the actions taken by operations control programs for the various timer 1 interrupt conditions.

3.4.4 Channel Uses of Operations Control

The A, B, S, M, and orderwire channels use different entries to the operations control program. Any channel may experience a timer 0 interrupt causing operations control to be entered via the interrupt entry (OP INT). The OP BUSY entry is used by the A and B channels and the orderwire 1 channel. When this entry is used, a channel does not receive control again until at least one device control message has been executed by the communications service unit and verified by device control message verification as either completed or in permanent error. The orderwire 2, S and M channels do not use the OP BUSY entry because they service many functions on their channel time and may have new work to accomplish regardless of the status of a particular time division exchange data transfer request. To keep track of input and output under these conditions, the orderwire 2, S, and M channels maintain a high level completion chain. The orderwire 1 channel uses the OP BUSY entry when it has disposed of all orderwire 1 service messages received and has reinitialized the device control messages for receipt of further messages; a device control message is executed by the communications service unit and verified by

Table 3-4. Timer 1 Interrupt Handling.

INTERRUPT	OPS-TABLE ENTRY		ACTION
	WORD 7	WORD 0	
	16-31	0 6-7	
TIMER 1 (Occurring during channel time)	=0	0 --	Return to old Instruction Address Counter in unprivileged mode.
	=0	1 --	Return to old Instruction Address Counter in privileged mode.
	≠0	0 0-	Return to old Instruction Address Counter in unprivileged mode.
	≠0	1 0-	Return to old Instruction Address Counter in privileged mode.
	≠0	0 10	Reset/Start/Stop Timer 1. Return old Instruction Address Counter in unprivileged mode.
	≠0	1 1-	Reset/Start/Stop Timer 1. Return old Instruction Address Counter in privileged mode.
	≠0	0 11	WORD 0 (6-7) 00. Return to common entry in unprivileged mode. Set old Instruction Address Counter to Program Status Record work register.
TIMER 1 (Occurring while in Ops-Control	-	- --	Return to old Instruction Address Counter in privileged mode.
TIMER 0	≠0	0 10	Reset/Start/Stop Timer 1 Exit to channel in unprivileged mode.
	≠0	0 11	Reset/Start Timer 1 Exit to channel in unprivileged mode.
	≠0	1 1-	Reset/Start/Stop Timer 1 Exit to channel in privileged mode.

device control message verification and control then passed to the orderwire 1 channel. Under these conditions, the orderwire 1 channel does not receive control until a service message has been received.

The voluntary (OP CKPT) entry is used by all channels and is the normal entry from the S, M, and orderwire 2 channels. Using this entry, no test for new work is made and control is returned to the channel regardless of the results of device control message verification. The S, M, and orderwire 2 channels monitor their own completion status and do their own testing for new work. Orderwire 1 uses this entry when it cannot obtain bin space for reinitialization of device control messages or when it cannot place a service message in a channel work queue because the queue is full. In both cases, orderwire 1 requests operations control to try again the next time it comes up in the sequence table.

The A and B channels are the only channels that use the completion entry (OP COMP) to operations control. When an application program has completed execution on A or B channel time, it branches to the A/B channel initiation/completion routine that sends a return service message to the calling control program status record noting the completion and location of any output files. The A/B channel initiation/completion routine transfers control to the OP COMP entry of the operations control program. The channel status is set to idle and the next channel indicated in the sequence table is serviced. When the A or B channel occurs again in the sequence table, a test for new work is made and control is returned if an unserviced entry is found in either queue 1 or queue 2.

Only the OP CKPT and OP BUSY entries are available from the unprivileged channels via a BRLP linkage. The OP COMP entry is used only by A and B channel initiation and completion.

PROCESSOR INTERFACE TABLE

(BYTE)	FUNCTION			
00-0F	SPARE			
10-1F	SPARE			
20-2F	SPARE			
30-3F	DC4 NAC	SPARE	DC4 BOUNDS	SPARE
40-4F	PCCW	PHSW	ATC	OW2 STATUS
50-5F	OW2 STATUS			
60-6F	OW2 STATUS			
70-7F	OW2 STATUS			
80-8F	SPARE			
90-9F	SPARE			
A0-AF	PROGRAM INTERRUPT		PARITY INTERRUPT	
B0-BF	CLOCK INTERRUPT (IPL)		BRLP INTERRUPT	
C0-CF	B REGISTER	A REGISTER	SPARE	
D0-DF	TRAPPED OPCODE INTERRUPT			
E0-EF	SPARE			
F0-FF	SPARE		OPSTAC	OPSLOW

4 WORDS

MSR TYPICAL ENTRY

FS 2	FO 4	FP 26
D 32		
RS 2	RO 4	RP 26
SQ 1	PP 31	

OPERATION CONTROL TABLE

CORE LOC (BYTE)	ENTRY NO	CSU DATA CHAN ASSIGNMENT	FUNCTION
100-11F	0	0	SERVICE CHANNEL
120-13F	1	1	MUX CHANNEL
140-15F	2	2	A CHANNEL
160-17F	3	3	B CHANNEL OR ORDERWIRE 2
180-19F	4	4	ORDERWIRE 1
1A0-1BF	5	NONE	SPARE
1C0-1DF	6	NONE	SPARE
1E0-1FF	7	NONE	SPARE

15 ENTRIES

0 STATUS BITS	1 2 CHAN SPACE ADRS	
4 QUEUE CELL ADRS (ON LOAD)	6 NRP	7 NWP
8 QUEUE CELL ADRS (OFF LOAD)	10 NRP	11 NWP
12 NRP DCM 1 (TX)	14 NRP DCM 2 (RX)	
16 NWP DCM 1 (TX)	18 NAC DCM 1 (TX)	
20 NWP DCM 2 (RX)	22 NAC DCM 2 (RX)	
24 CHAN LOWER LIMIT	26 CHAN UPPER LIMIT	
28 TRAP ENTRY POINTS	30 TIMER 1 ENTRY POINT	

TYPICAL OPS CONTROL TABLE ENTRY

1. SET C
2. ROUTI
3. COMM
4. ZONE
5. ALLOC
6. DOCUM
7. RELEA
8. SINGL
9. DIREC
10. QUEUE
11. SERVI
12. RELOC
13. DIREC
14. INDIRE
15. M CH
16. ORDEF
17. C-NU
18. C-NU
19. OBJEC
20. DECLA
21. RESET
22. STOP
23. STAR

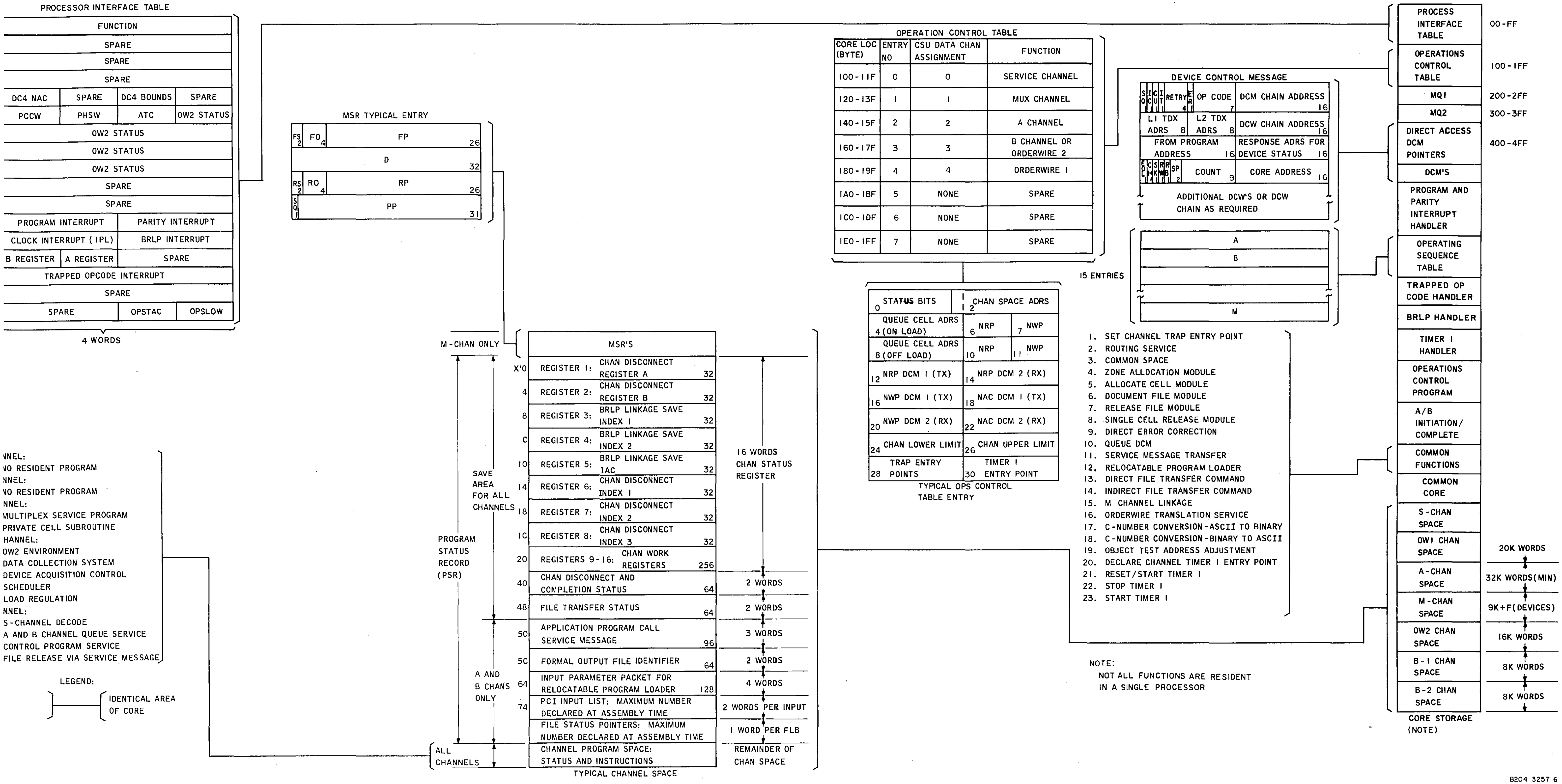
NOTE:
NOT AL
IN A SI

A-CHANNEL:
NO RESIDENT PROGRAM
B-CHANNEL:
NO RESIDENT PROGRAM
M-CHANNEL:
MULTIPLEX SERVICE PROGRAM
PRIVATE CELL SUBROUTINE
OW2-CHANNEL:
OW2 ENVIRONMENT
DATA COLLECTION SYSTEM
DEVICE ACQUISITION CONTROL
SCHEDULER
LOAD REGULATION
S-CHANNEL:
S-CHANNEL DECODE
A AND B CHANNEL QUEUE SERVICE
CONTROL PROGRAM SERVICE
FILE RELEASE VIA SERVICE MESSAGE

LEGEND:
IDENTICAL AREA OF CORE

TYPICAL CHANNEL SPACE

M-CHAN ONLY		MSR'S		
PROGRAM STATUS RECORD (PSR); SAVE AREA FOR ALL CHANNELS	X'0	REGISTER 1: CHAN DISCONNECT REGISTER A	32	16 WORDS CHAN STATUS REGISTER
	4	REGISTER 2: CHAN DISCONNECT REGISTER B	32	
	8	REGISTER 3: BRLP LINKAGE SAVE INDEX 1	32	
	C	REGISTER 4: BRLP LINKAGE SAVE INDEX 2	32	
	10	REGISTER 5: BRLP LINKAGE SAVE IAC	32	
	14	REGISTER 6: CHAN DISCONNECT INDEX 1	32	
	18	REGISTER 7: CHAN DISCONNECT INDEX 2	32	
	1C	REGISTER 8: CHAN DISCONNECT INDEX 3	32	
	20	REGISTERS 9-16: CHAN WORK REGISTERS	256	
	40	CHAN DISCONNECT AND COMPLETION STATUS	64	2 WORDS
	48	FILE TRANSFER STATUS	64	2 WORDS
A AND B CHANS ONLY	50	APPLICATION PROGRAM CALL SERVICE MESSAGE	96	3 WORDS
	5C	FORMAL OUTPUT FILE IDENTIFIER	64	2 WORDS
	64	INPUT PARAMETER PACKET FOR RELOCATABLE PROGRAM LOADER	128	4 WORDS
	74	PCI INPUT LIST: MAXIMUM NUMBER DECLARED AT ASSEMBLY TIME		2 WORDS PER INPUT
		FILE STATUS POINTERS: MAXIMUM NUMBER DECLARED AT ASSEMBLY TIME		1 WORD PER FLB
ALL CHANNELS		CHANNEL PROGRAM SPACE: STATUS AND INSTRUCTIONS		REMAINDER OF CHAN SPACE



NNEL:
NO RESIDENT PROGRAM
NNEL:
NO RESIDENT PROGRAM
NNEL:
MULTIPLEX SERVICE PROGRAM
PRIVATE CELL SUBROUTINE
HANNEL:
OW2 ENVIRONMENT
DATA COLLECTION SYSTEM
DEVICE ACQUISITION CONTROL
SCHEDULER
LOAD REGULATION
NNEL:
S-CHANNEL DECODE
A AND B CHANNEL QUEUE SERVICE
CONTROL PROGRAM SERVICE
FILE RELEASE VIA SERVICE MESSAGE

LEGEND:
IDENTICAL AREA OF CORE

OPERATION CONTROL TABLE

CORE LOC (BYTE)	ENTRY NO	CSU DATA CHAN ASSIGNMENT	FUNCTION
100-11F	0	0	SERVICE CHANNEL
120-13F	1	1	MUX CHANNEL
140-15F	2	2	A CHANNEL
160-17F	3	3	B CHANNEL OR ORDERWIRE 2
180-19F	4	4	ORDERWIRE 1
1A0-1BF	5	NONE	SPARE
1C0-1DF	6	NONE	SPARE
1E0-1FF	7	NONE	SPARE

TYPICAL OPS CONTROL TABLE ENTRY

0 STATUS BITS	1	2 CHAN SPACE ADRS
4 QUEUE CELL ADRS (ON LOAD)	6 NRP	7 NWP
8 QUEUE CELL ADRS (OFF LOAD)	10 NRP	11 NWP
12 NRP DCM 1 (TX)	14 NRP DCM 2 (RX)	
16 NWP DCM 1 (TX)	18 NAC DCM 1 (TX)	
20 NWP DCM 2 (RX)	22 NAC DCM 2 (RX)	
24 CHAN LOWER LIMIT	26 CHAN UPPER LIMIT	
28 TRAP ENTRY POINTS	30 TIMER 1 ENTRY POINT	

TYPICAL CHANNEL SPACE

	MSR'S		
X'0	REGISTER 1: CHAN DISCONNECT REGISTER A	32	16 WORDS CHAN STATUS REGISTER
4	REGISTER 2: CHAN DISCONNECT REGISTER B	32	
8	REGISTER 3: BRLP LINKAGE SAVE INDEX 1	32	
C	REGISTER 4: BRLP LINKAGE SAVE INDEX 2	32	
10	REGISTER 5: BRLP LINKAGE SAVE IAC	32	
14	REGISTER 6: CHAN DISCONNECT INDEX 1	32	
18	REGISTER 7: CHAN DISCONNECT INDEX 2	32	
1C	REGISTER 8: CHAN DISCONNECT INDEX 3	32	
20	REGISTERS 9-16: CHAN WORK REGISTERS	256	
40	CHAN DISCONNECT AND COMPLETION STATUS	64	2 WORDS
48	FILE TRANSFER STATUS	64	2 WORDS
50	APPLICATION PROGRAM CALL SERVICE MESSAGE	96	3 WORDS
5C	FORMAL OUTPUT FILE IDENTIFIER	64	2 WORDS
64	INPUT PARAMETER PACKET FOR RELOCATABLE PROGRAM LOADER	128	4 WORDS
74	PCI INPUT LIST: MAXIMUM NUMBER DECLARED AT ASSEMBLY TIME		2 WORDS PER INPUT
	FILE STATUS POINTERS: MAXIMUM NUMBER DECLARED AT ASSEMBLY TIME		1 WORD PER FLB
	CHANNEL PROGRAM SPACE: STATUS AND INSTRUCTIONS		REMAINDER OF CHAN SPACE

1. SET CHANNEL TRAP ENTRY POINT
2. ROUTING SERVICE
3. COMMON SPACE
4. ZONE ALLOCATION MODULE
5. ALLOCATE CELL MODULE
6. DOCUMENT FILE MODULE
7. RELEASE FILE MODULE
8. SINGLE CELL RELEASE MODULE
9. DIRECT ERROR CORRECTION
10. QUEUE DCM
11. SERVICE MESSAGE TRANSFER
12. RELOCATABLE PROGRAM LOADER
13. DIRECT FILE TRANSFER COMMAND
14. INDIRECT FILE TRANSFER COMMAND
15. M CHANNEL LINKAGE
16. ORDERWIRE TRANSLATION SERVICE
17. C-NUMBER CONVERSION-ASCII TO BINARY
18. C-NUMBER CONVERSION-BINARY TO ASCII
19. OBJECT TEST ADDRESS ADJUSTMENT
20. DECLARE CHANNEL TIMER 1 ENTRY POINT
21. RESET/START TIMER 1
22. STOP TIMER 1
23. START TIMER 1

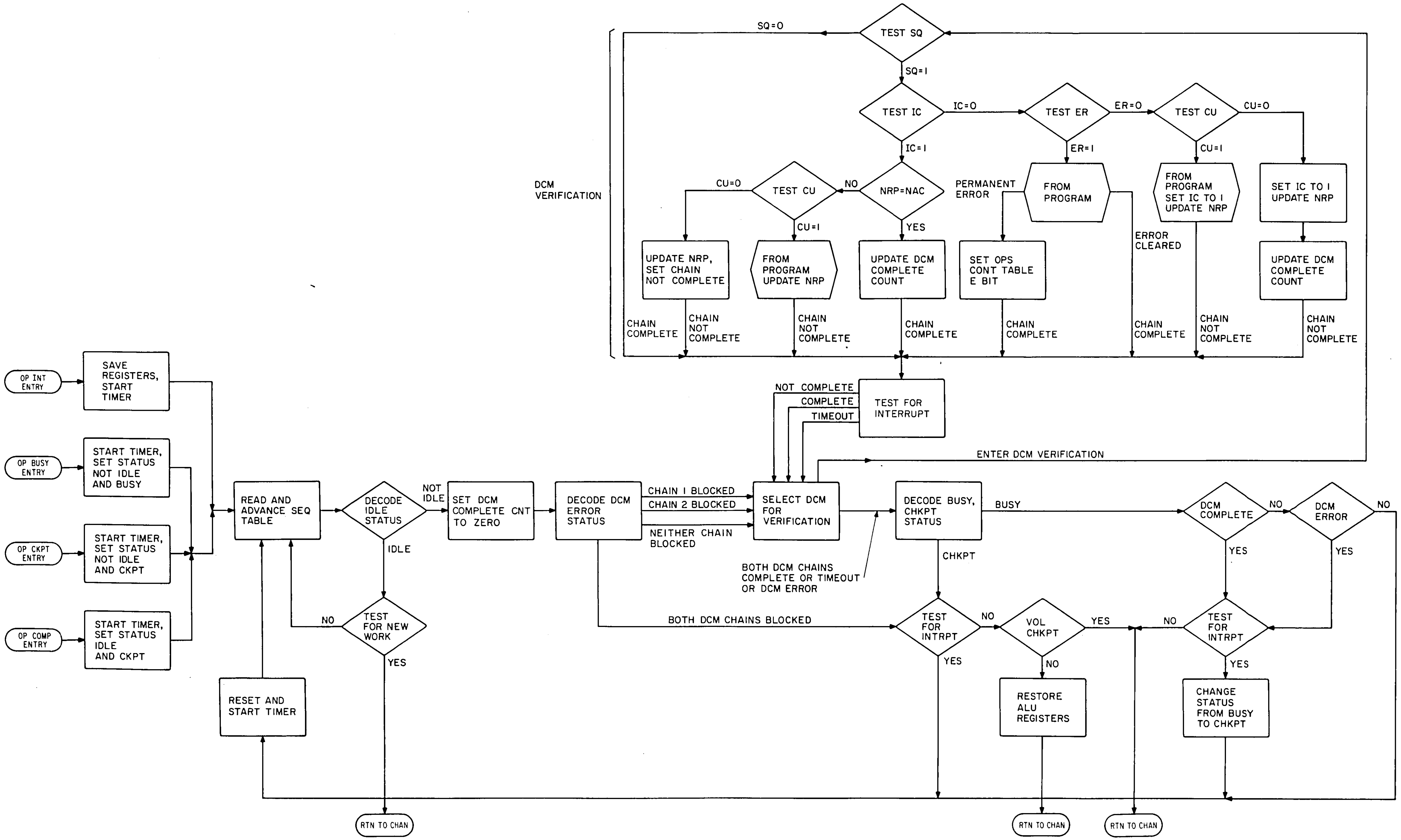
NOTE:
NOT ALL FUNCTIONS ARE RESIDENT IN A SINGLE PROCESSOR

CORE STORAGE (NOTE)

PROCESS INTERFACE TABLE	00-FF	
OPERATIONS CONTROL TABLE	100-1FF	
MQ1	200-2FF	
MQ2	300-3FF	
DIRECT ACCESS DCM POINTERS	400-4FF	
DCM'S		
PROGRAM AND PARITY INTERRUPT HANDLER		
OPERATING SEQUENCE TABLE		
TRAPPED OP CODE HANDLER		
BRLP HANDLER		
TIMER 1 HANDLER		
OPERATIONS CONTROL PROGRAM		
A/B INITIATION/COMPLETE		
COMMON FUNCTIONS		
COMMON CORE		
S-CHAN SPACE		
OW1 CHAN SPACE		20K WORDS
A-CHAN SPACE		32K WORDS (MIN)
M-CHAN SPACE		9K+F(DEVICES)
OW2 CHAN SPACE		16K WORDS
B-1 CHAN SPACE		8K WORDS
B-2 CHAN SPACE		8K WORDS

B204 3257 6

Figure 3-2. Core Storage Organization.


OP INT ENTRY
SAVE REGISTERS, START TIMER
OP BUSY ENTRY
START TIMER, SET STATUS NOT IDLE AND BUSY
OP CKPT ENTRY
START TIMER, SET STATUS NOT IDLE AND CKPT
OP COMP ENTRY
START TIMER, SET STATUS IDLE AND CKPT
READ AND ADVANCE SEQ TABLE
RESET AND START TIMER
DECODE IDLE STATUS
NOT IDLE
IDLE
TEST FOR NEW WORK
NO
YES
RTN TO CHAN
SET DCM COMPLETE CNT TO ZERO
DECODE DCM ERROR STATUS
CHAIN 1 BLOCKED
CHAIN 2 BLOCKED
NEITHER CHAIN BLOCKED
BOTH DCM CHAINS BLOCKED
SELECT DCM FOR VERIFICATION
BOTH DCM CHAINS COMPLETE OR TIMEOUT OR DCM ERROR
TEST FOR INTERRUPT
NOT COMPLETE
COMPLETE
TIMEOUT
ENTER DCM VERIFICATION
DCM VERIFICATION
TEST SQ
SQ=0
SQ=1
TEST IC
IC=0
IC=1
TEST ER
ER=0
ER=1
TEST CU
CU=0
CU=1
NRP=NAC
NO
YES
TEST CU
CU=0
CU=1
UPDATE NRP, SET CHAIN NOT COMPLETE
FROM PROGRAM UPDATE NRP
UPDATE DCM COMPLETE COUNT
PERMANENT ERROR
FROM PROGRAM
SET OPS CONT TABLE E BIT
ERROR CLEARED
FROM PROGRAM SET IC TO 1 UPDATE NRP
SET IC TO 1 UPDATE NRP
UPDATE DCM COMPLETE COUNT
CHAIN COMPLETE
CHAIN NOT COMPLETE
CHAIN NOT COMPLETE
CHAIN COMPLETE
CHAIN COMPLETE
CHAIN COMPLETE
CHAIN NOT COMPLETE
CHAIN NOT COMPLETE
DECODE BUSY, CHKPT STATUS
BUSY
CHKPT
DCM COMPLETE
NO
YES
DCM ERROR
NO
YES
TEST FOR INTRPT
NO
YES
VOL CHKPT
YES
NO
RESTORE ALU REGISTERS
RTN TO CHAN
RTN TO CHAN
TEST FOR INTRPT
NO
YES
CHANGE STATUS FROM BUSY TO CHKPT

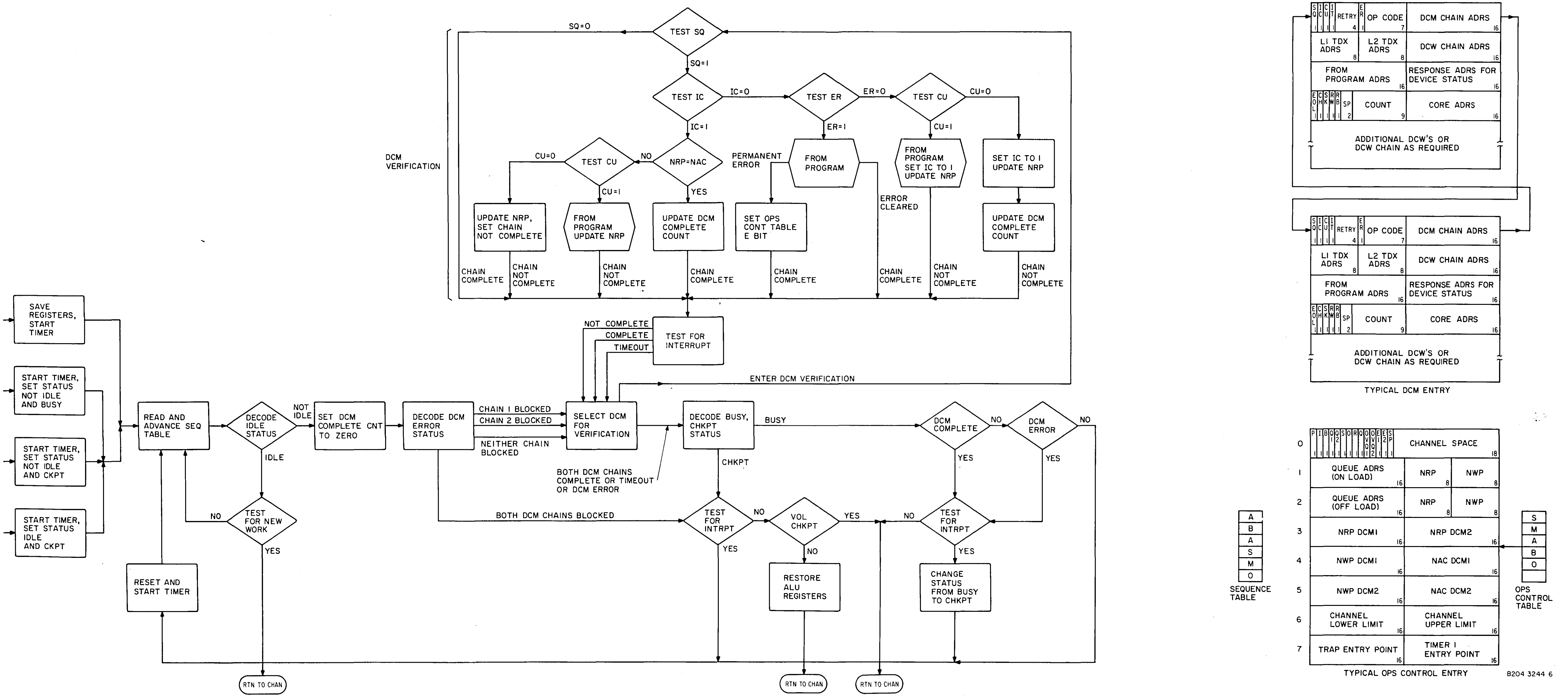


Figure 3-3. Operations Control Program.

section 4
service messages

4.1 GENERAL

Service messages in the C-System transfer information between processors and peripheral devices connected to the system. Service messages are transmitted via the orderwire 1 and orderwire 2 time division exchange channels; the orderwire 1 channel operates at 125 kbps while the orderwire 2 channel operates at 7.1825 kbps. Messages transmitted via the orderwire 1 channel (orderwire 1 service messages) are processor-to-processor messages. Orderwire 2 service messages (service messages transmitted via the orderwire 2 channel) are transmitted between processors and peripherals in the system.

In addition to the normal orderwire 1 and orderwire 2 service messages, direct access service messages are transmitted via the orderwire 1 time division channel. A direct access service message causes the communications service unit orderwire 1 data channel to read or write information directly from or to core storage, rather than initiating processor or peripheral device action. This feature is used in sampling work distribution and changing system operating tables in core storage.

4.2 SERVICE MESSAGE FLOW

The general flow of Service Messages (SVM) in the C-System is shown in figure 4-1. Orderwire 1 (OW1) service messages originate in the A, B, S, M and Orderwire 2 (OW2) processor channels; the orderwire 1 processor channel does not transmit service messages. Orderwire 2 service messages are only used in the orderwire 2 channel by Device Acquisition and Control Service (DACS) for communication with peripherals. All service messages are prepared for transmission by channel programs. The channel programs build a File Transfer Command (FTC) which is a 4-word packet directing the transmission of the service message. The common function Service Message Transfer (SMT) is then called to verify the file transfer command and call the common function Queue Device Control Message (Queue DCM) routine for allocation of device control message space. The queue device control message routine returns the address of the allocated device control message if space is available. Service message transfer then uses this space to build a device control message directing transmission of the service message by the Communications Service Unit (CSU). The information for building the device control message is contained in the file transfer command packet transferred to service message transfer by the channel program. The file transfer command packet (figure 4-1) identifies the message as an orderwire 1 or orderwire 2 service message (bit A), the device control message chain to use (bit B), the address of the Data Control Word list (DCW ADRS) which points to the service message to be transmitted, the Party Line Address (PLA) for orderwire 1 transfer, the device command word for orderwire 2 transfer, words for storage of device and channel status, and a transfer complete indication bit (CC).

Orderwire 1 service messages originated by the A, B, and M channels are transmitted from either device control message chain 1 or chain 2 in the respective channels. Orderwire 1 service messages originating in the S channel are transmitted via S channel device control message chain 1. S channel device control message chain 2 is used for transmission of orderwire 1 service messages originated by orderwire 2 channel programs. Orderwire 2 service messages, originated by orderwire 2 channel programs, are transmitted via the orderwire 2 device control message chain 1.

Since the S channel monitors its own data transfer completions via a completion chain, a dummy transfer command packet is placed in the S channel completion chain for transmission of the orderwire 1 service messages via the S channel device control message chain 2. A from-program address in the dummy packet indicates an orderwire 2 program which sets up the parameters for file transfer and calls service message transfer to build the device control message. When S channel decode detects the completion bit set in the command packet, control is returned to the orderwire 2 program which is executed on S channel time.

All orderwire 2 messages are transmitted between Device Acquisition and Control Service (DACS) operating on orderwire 2 channel time and peripheral devices connected to L2 loops through TU-3B terminal units and peripheral devices connected to the L1 loop through TU-3A terminal units.

All orderwire 1 service messages received by a processor are input via orderwire 1 device control message chain 2. The orderwire 1 input program maintains this device control message chain and distributes the received messages to the appropriate work queues in the processor. When the orderwire 1 channel is encountered in the sequence table by operations control, the orderwire 1 device control message chains are verified by device control message verification. A completed device control message indicates that an orderwire 1 service message has been received and requires service by the orderwire 1 input program. If at least one device control message is verified as complete, operations control gives control to the orderwire 1 input program. The orderwire 1 input program disposes of the received service message in one of four ways. If it is an immediate message, the corresponding immediate program is called and allowed to execute during orderwire 1 channel time. If the message is for an orderwire 2 channel program, a pointer to the message is placed in Service Queue 2 (SQ2). If the message is for S channel, A channel, or B channel programs, a pointer is placed in Service Queue 1 (SQ1) identifying the message. For M channel service messages, the pointer is placed in the Multiplex Channel Queue (MCQ).

Having disposed of all service messages, the orderwire 1 input program obtains new service message bins from the common function space allocation routine and reinitializes the device control messages. The orderwire 1 input program then relinquishes the remainder of its time by exiting OP BUSY to the operations control program and does not receive control again until at least one message has been received. If the orderwire 1 input program cannot obtain new bins for service messages or cannot dispose of received messages because of full work queues, it uses the voluntary checkpoint entry (OP CKPT) to operations control, guaranteeing return of control the next time the channel appears in the sequence table.

Messages directed to service queue 1 are serviced by S channel decode on S channel time. When a message intended for the A or B channel is encountered, channel queue service is invoked. Channel queue service writes the service message to the appropriate A or B channel work queue on disc file storage. (An overflow queue is created if required.) The next-write- and next-read-positions for the A and B channel work queues are maintained in the operations control table. The next-write-position is incremented when a new entry is posted in a channel work queue. Status bits in the operations control table status word (bits 11 and 12) indicate the use of overflow queues. The operations control program monitors the next-read- and next-write-positions in the operations control table entries for the A and B channels. Every time an action initiated by a service message for an A or B channel is completed, the channel exits to the operations control program. When that channel appears again in the operations sequence table, the operations control program compares the next-read- and next-write-positions for the channel work queues to determine if any service messages are waiting for that channel.

Service queue 1 service messages may also be directed to Control Program Service (CPS) or may direct disc file storage release for files no longer needed in the file space managed by this processor. Disc file storage release is accomplished via an S channel routine called file release via service message. S channel decode invokes control program service or file release via service message as appropriate upon receipt of these messages.

An orderwire 1 service message directed to multiplex channel queue is serviced by the Multiplex Service Program (MSP) on M channel time. These messages direct the ouput of a file to a device on the time division multiplex loop.

Orderwire 1 service messages directed to service queue 2 are destined for orderwire 2 channel programs. These messages are decoded by orderwire 1 service message input decode in the orderwire 2 channel environment and directed to the proper work queue for Data Collection Service (DCS), Device Acquisition and Control Service (DACS), the system scheduler, or the load regulator.

Orderwire 2 service messages received via the orderwire2 data channel of the communications service unit are from system peripherals and are received via the orderwire 2 device control message chain 2 by the orderwire 2 input program. The orderwire 2 input program decodes the received service messages andplaces the pointers in the device acquisition and control service work queues. Orderwire 2 input then obtains a bin for further receipt of orderwire 2 service messages and reinitializes the device control messages. The orderwire 2 input routine does not work with the operations control program in the same manner as the orderwire 1 input routine; the orderwire 1 input routine uses the input/output busy (OP BUSY) entry to operations control, only receiving control after one or more service messages are verified as having been received and needing the service of orderwire 1 input. This presents no problem for the orderwire 1 channel because orderwire 1 input is the only program executed in this channel. In contrast, the orderwire 2 channel is used by several programs and always uses the voluntary checkpoint (OP CKPT) entry (except for interrupt) to operations control guaranteeing return regardless of data transfer status or service message receipt. Orderwire 2 input uses device control message verification in operations control by setting the connector update bit in the device control messages used for receipt of orderwire 2 service messages. The setting of the connector update bit causes device control message verification to branch immediately to the from-program address indicated in the device control message which in this case is orderwire 2 input. Orderwire 2 input then decodes the message, reinitializes the device control message, and returns to device control message verification in operations control. Device control message verification continues until all device control messages completed have been verified and then control is given to the orderwire 2 channel for execution of channel programs.

Each orderwire 1 service message is preceded on the time division exchange loop by a word of the following format:

Party Line Address of Called Processor 8	Spare 16	Displacement 8

The party line address is a code identifying the processor to receive the service message. The displacement field is 0 for a normal service message, and the device control message to be used is located by the next-available-cell pointer in the operations control table. If the displacement is not 0, the service message is a direct access service message. This is a special device control message which is never in the device control message chain. The displacement is added to a fixed base address in the communication service unit to locate the appropriate device control message for the service message. The fixed device control messages contain the proper data control words and core addresses to allow the various table-read and table-write direct access functions. The table-read direct access function is the only service message that transmits data to the calling processor.

4.3 ORDERWIRE 1 SERVICE MESSAGES

The orderwire 1 service messages listed in table 4-1 may be classified according to the hexadecimal operation codes as follows:

OPERATION CODE	DESTINATION
00 - 3F	Orderwire 1 Immediate Program
40 - 3F	Service Queue 1

OPERATION CODE	DESTINATION
80 - 9F	Service Queue 2
A0 - BF	Multiplex Channel Queue
C0 - FF	Service Queue 1

Detailed definitions of the orderwire 1 service message formats are given in the appendix. The basic functions of each of the orderwire 1 service messages are discussed in the following paragraphs.

Table 4-1. Service Messages.

OP CODE	SVM NAME	ORIGINATING FUNCTION	RECEIVING FUNCTION
		IMMEDIATE OPERATIONS	
01	Replace Output Tape Return	DACS	OW1 Immediate Function
02	Initialize Ops Control Table	Load Regulator	OW1 Immediate Function
		SERVICE CHANNEL FUNCTIONS	
40	CPSR Call	CPS	CPS
42	Program Return	A or B Chan. Complete, Channel Completion CPS, MSP, OW2	CPS
43	CP Abnormal Return	CPS	CPS
44	AP Abnormal Return	Service program application program A or B Channel Completion MSP, OW2	CPS
45	Space Release	CPS, service programs application programs	Disc file storage allocation and release
47	CPSR Generator Return	CPSR Generator	CPS
4D	Initial CPSR Call	CPSR Generator, MSP	CPS
4E	Redirect	Recovery, Service Message Issue	CPS
		OW2 CHANNEL FUNCTIONS	
80	Tape Release	CPS	DACS
81	DCS Update	All Programs	DCS
82	DCS Extract	CPS	DCS

Table 4-1. Service Messages (Cont).

OP CODE	SVM NAME	ORIGINATING FUNCTION	RECEIVING FUNCTION
83	DACS Request	CPS	DACS
84	Replace Output Tape	Application Programs	DACS
85	Scheduler Service Call	CPS	Scheduler
86	Scheduler Table Generator Return	Schedule Table Generator	Scheduler
87	DCS Code Delete	Application Programs, CPS	DCS
88	DCS Code Request (Unload Allowed)	Application Programs, CPS	DCS
8B	DCS File Return	System Update and Checkpoint Program	DCS
8C	DCS Collection Inhibit	System Update and Checkpoint Program	DCS
8D	DACS Routing Request	Service Programs, Application Program, external inputs	DACS
90	DCS Extract and Save	CPS	DCS
91	Processor Initialization	Initialization Program	Load Regulator
92	Processor Removal	Backup and Recovery	Load Regulator
93	Disc Initialization	Initialization Program	Load Regulator
94	Disc Removal	Backup and Recovery	Load Regulator
95	Overload	Channel Queue Service Disc File Storage Allocation	Load Regulator
96	Update Tables	Aps Program Load Regulator	Load Regulator
	M CHANNEL FUNCTIONS		
A0	I/O Call 1	CPS, MSP, Application Programs	MSP
A1	I/O Call 2	MSP	MSP
A4	TDM Device Initialization Response	CPS	MSP

Table 4-1. Service Messages (Cont).

OP CODE	SVM NAME	ORIGINATING FUNCTION	RECEIVING FUNCTION
		B CHANNEL FUNCTIONS	
C0 D0	Application Program Call	CPS	AP Initialization
C8 D8	Program Call	CPS, MSP, DACS	B Channel
		A CHANNEL FUNCTIONS	
E0 F0	Application Program Call	CPS	AP Initialization
E8 F8	Program Call	CPS, MSP, DACS	A Channel
		DIRECT ACCESS	
N/A	Direct Access	Load Regulator, Initialization Recovery	CSU

4.3.1 Immediate Operations

A service message which does not require a file transfer may be processed immediately as an immediate operation. The following are immediate operation service messages:

a. Replace Output Tape Return (operation code X'01)
This service message is issued by device acquisition and control service in response to a replace output tape service message. This service message is used to return to an applications program the identifiers for the new output tape.

b. Initialize Operations Control Table (operation code X'02)
Load regulation service sends this service message to the orderwire 1 service function to modify the status indicators in the operations control table entry referenced in the service message. Only the A, B, or M channel operations control table entries may be modified in this way.

4.3.2 Service Channel Functions

Various orderwire 1 service messages are received by S channel decode. Upon receiving an orderwire 1 service message, S channel decode reads the service message operation code and initiates a particular C-System routine for further processing of the received service message.

The following seven service messages result in S channel decode initiating the Control Program Service (CPS) routine:

a. Control Program Status Record Call Service Message (operation code X'40).
When received by control program service, this service message is used to initiate the execution of a particular control program which is resident on disc. The service message contains information needed by control program service to read the calling control program status record and to isolate the calling program control instruction. Contained in the calling program control instruction

is the disc address of the called control program status record (related to the control program to be executed) and various input data to be used by control program service in the execution of the control program.

b. Program Return Service Message (operation code X'42).
This is the service message normally returned to control program service after the operation called for by the execution of one of its program control instructions has been successfully carried out. All functions which may be called by a program contol instruction return this service message, except the control program status record generator call. Information needed by control program service for reading the control program status record from disc and for isolating the program control instruction which originated the operation just performed is contained in the service message. Also contained in the service message is either 64 bits of literal data created during the operation or the disc address of the created data which is loaded into the output field of the program control instruction by control program service. Control program service then updates the control program status record, writes it back to disc, and sends any service messages created during the update.

c. Control Program Abnormal Return (operation code X'43).
The control program abnormal return service message is generated if there is an uncorrectable error in the control program execution and if the control program is not a high level control program. Contained in the service message is the disc addresses of the calling control program status record and of the control program status record in error. Upon receiving this service message. control program service discontinues execution of the calling control program status record and, when all outstanding program control instructions have completed, invokes the recovery program contol instruction if one is provided. If there is no recovery program control instruction, control program service abnormally completes the calling control program status record.

d. Application Program Abnormal Return Service Message (operation code X'44).
This service message is sent any time an error occurs during the execution of an application program. Upon receiving this service message, control program service discontinues execution of the initiating control program status record and, when all outstanding program control instructions have been completed, executes the recovery program control instruction, if one is provided. If no recovery program control instruction is provided, control program service abnormally completes the calling control program status record.

e. Space Release Service Message (operation code X'45).
When this orderwire 1 service message is received by S channel decode, it invokes the disc file storage allocation and release program. This service message contains the disc address of the initiating control program status record and the disc address of the disc storage file to be released. The service message may be built due to a control program service (program control instruction call), service program, or application program request. When initiated, the space release program releases the file whose disc address is contained in the service message.

f. Control Program Status Record Generator Return Service Message (operation code X'47).
The generator return service message is sent by the control program status record generator to a program that issued a control program status record generator call service message that called for a cyclic control program status record to be built from a Control Program Status Constant (CPSC). Contained in the service message are the addresses of the calling control program status record that sent the generator call and of the generated control program status record. Control program service uses the addresses to update the calling program control instruction and to issue a control program status record call service message. The control program status record call service message then causes the control program of the generated control program status record to be executed.

g. Initial Control Program Status Record Call Service Message (operation code X'4D).
The initial control program status record call service message is normally created by the control program status record generator program executed in the B channel of a processor, in response to a control program status record generator call service message. This service message is responsible for initiating a control program that is to be non-cyclic, i.e., its related control program status record will be released after the program has run once. This service message is received by the control program service routine resident in the processor responsible for execution of the control program. The service message contains the disc address of the control program status

record which has been created by the generator program. Control program service, after receiving the service message, loads the generated control program status record and initiates execution of its control program.

h. Redirect Service Message (operation code X'4E).
This service message is included as a facility for reissuing service messages that cannot be performed by the receiving processor. Service message issue and the recovery routine are the only two programs capable of building this service message. Contained in the redirect service message is the operation code X'4E and the remainder of the service message to be reissued. Upon receiving this service message, control program service reads the originating control program status record to determine the type of service message to be reissued and then reissues the service message to another processor.

4.3.3 Orderwire 2 Channel Functions

Orderwire 1 service messages directed to orderwire 2 programs are placed in service channel queue 2. Orderwire 1 service message input decode then places these messages in the appropriate orderwire 2 program input queue for service by the orderwire 2 channel function.

The following service messages initiate orderwire 2 channel functions:

a. Space/Tape Release Service Message (operation code X'80).
This service message is directed to device acquisition and control service to effect the release of disc and tape files. For the disc file case, device acquisition and control service generates a space release service message (operation code X'45) return to the file release via service message routine located in the S channel. For the tape file case, device acquisition and control service initiates the update of the master tape library maintained on disc to place the indicated tape reels in a released state and then generates a file release service message to release the reels list or connector list maintained on disc for the indicated tape file.

b. Data Collection Service Update (operation code X'81).
This service message is used by programs to effect the addition of an input file to a specified data collection system file. The file identifier is added to either the tape entry or disc entry of the data collection file, and thus is available for the next extract performed on its data collection service code.

c. Data Collection Service Extract and Delete (operation code X'82).
This service message is sent to data collection service by control program service to obtain the address of the collection file referenced by the specified data collection service code. The address of the collection file is returned via a program return service message. The program return service message collection file address is 0 if no updates have been received for this file. The data collection file address is normally deleted; however, an option is provided to allow retention of the address in data collection service for system assigned data collection service codes.

d. Device Acquisition and Control Service Request (operation code X'83).
Device acquisition and control service request service messages are issued by control program service to provide for mounting tapes, both input and scratch; to provide for operator readying of peripheral equipment such as card readers, CRT's, and line printers; to obtain a list of reels applicable to a tape file (a reels list); and to enter unassigned tapes in the center library. Device acquisition and control service processes the request service message and builds an output file that contains parameters applicable to the particular request and returns the address of the output file to control program service via a program return service message (operation code X'42).

e. Replace Output Tape (operation code X'84).
The replace output tape service message is sent to device acquisition and control service by an applications program when the end of an output tape is reached. This service message specifies the library zone where the tape reel is to be placed and initiates the mounting of a new scratch tape. Device acquisition and control service directs the mounting of the new tape, writes a one-word header on the tape, and then generates a replace output tape return service message (operation code X'01) which is an orderwire 1 immediate service message.

f. Scheduler Service Call (operation code X'85).
The scheduler service call service message is sent to the scheduler program by control program service. The activation time for an application system is contained in this message. The scheduler program sends a program return service message (X'42) when the activation time has arrived, thus allowing the application system to be initiated.

g. Scheduler Table Generator Return (operation code X'86).
This service message is sent by the table generator program to the scheduler program to signal that a new updated schedule file has been created and to give the scheduler program the next call time for the table generator program system. Upon receipt, the scheduler program replaces the current schedule table file with the new file.

h. Data Collection Service Code Delete (operation code X'87).
This service message, which may be sent to data collection service by an application program or control program service. causes a data collection service code to be made inactive and data in the file to be placed under data collection service code 01.

i. Data Collection Service Code Request (operation code X'88).
Application programs and control program service may request assignment of data collection service codes by sending the service message to data collection service. Data collection service issues a data collection service code and responds with a program return service message (X'42). If there are no data collection service codes available, an abnormal program return service message (X'44) is issued.

j. Data Collection Service File Return (operation code X'8B).
This service message is used by the system update and checkpoint program to return the address of the new data collection service file or a data collection service code collection file to data collection service after a disc to tape unload has taken place. Data collection files which remain inactive are unloaded to tape and this service message issued.

k. Data Collection Service Collection Inhibit (operation code X'8C).
The system update and checkpoint program issues this service message to inhibit update of a data collection service file. Service messages directed to this data collection service code are queued up for processing. Periodically this data collection service code file is checked to determine if the inhibit has been removed.

l. Device Acquisition and Control Service Routing Request (operation code X'8D).
This service message may be issued either internally by control program service or an application program or externally by a time division exchange or time division multiplex device to initiate a program call service message (X'C8, D8, E8, or F8) to a utility program capable of delivering a specified message to a device. The device to which the message is to be delivered may be specified or if not specified an appropriate device is allocated.

m. Data Collection Service Extract and Save (operation code X'90).
This service message may be issued by control program service only for those data collection service codes which specify no unload to tape and which were assigned by the system. Data collection service responds with a program return service message (X'42) containing the address of the single cell collection file or 0 if a file does not exist. The file is saved as opposed to the delete action for a data collection service extract and delete service message (X'82).

n. Processor Initialization (operation code X'91).
This service message is issued to the load regulation program by the initialization program to provide data which identifies the processor; its assigned disc(s); disc format; queue threshold values for the processor; and exchange number assigned to the processor. The load regulation program uses this data to update the center configuration tables, set zone and queue threshold values for the processor and update center routing tables.

o. Processor Removal (operation code X'92).
The backup and recovery program issues this service message to load regulation service to signal the taking of a processor off-line. This service message provides necessary data for the load regulation program to update the center configuration and routing tables, deleting the removed processor from the tables and updating disc assignments and time division multiplex channel assignments.

p. Disc Initialization (operation code X'93).
This service message is issued by the initialization program to load regulation service when a disc file is brought on-line. This service message contains the necessary information for load regulation to update the center configuration and routing table.

q. Disc Removal (operation code X'94).
The backup and recovery program issues this service message to load regulation service to signal the taking of a disc file off-line. This service message provides necessary data for the load regulation program to update the center configuration tables and remove the associated processor from load dispatch tables.

r. Overload (operation code X'95).
This service message is used by channel queue service to notify load regulation service of a channel queue overload condition. Disc file allocation also uses this service message to notify load regulation of a disc file storage zone overload condition.

s. Regulator Tables Update Return (operation code X'96).
This service message is used by load regulation to send new tables to load regulation. This service message contains the disc file address of the new tables file. Upon receipt of this service message the load regulation program replaces the old tables with new ones referenced in the service message.

4.3.4 M Channel Functions

There are three service message types designated as M channel functions which are received by the multiplex service program.

a. Input/Output Call 1 Service Message (operation code X'A0).
The input/output call 1 service message is used by any program wanting to output a message file through M channel. The receiving M channel reads the header from the specified message file, validates the header, and attempts to route all the addresses contained in the header.

b. Input/Output Call 2 Service Message (operation code X'A1).
The input/output call 2 service message is used by the multiplex service program to output a message file through other M channels. The receiving M channels attempt to output those messages local to their exchanges.

c. Time Division Multiplex Device Initialization Response (operation code X'A4).
The time division multiplex initialization response service message is used by control program service to request the opening of a working channel. The multiplex service program attempts to open the channel and sends a program return service message to the calling control program status record (through control program service). If the opening of the channel is unsuccessful, an applications program abnormal return service message is sent.

4.3.5 A and B Channel Functions

Two service message types are received by the A and B channels of a processor. They are the following:

a. Application Program Call Service Messages.
The application program call service messages are built by control program service upon request from the program control instruction being executed. They are received by S channel decode which loads them into the appropriate A or B channel queues dependent upon the service message operation code. Operation codes X'C0 and X'D0 denote application program calls for the B channel on-load and off-load queues respectively while operation codes X'E0 and X'F0 denote application program calls for the A channel on-load and off-load queues respectively. The action taken by the channel receiving a service message of this type is to call the application program initiation routine.

b. Program Call Service Messages.
These service messages may be issued by control program service, the multiplex service program, or device acquisition and control service to have a nonresident service function executed in an A or B channel. The operation codes for B channel on-load and off-load routines are X'C8 and X'D8,

respectively while for the A channel they are respectively, X'E8 and X'F8. An example of such a service function is the control program status record generator call service message which is sent to a B channel on-load queue.

4.3.6 Direct Access Functions

Direct access to tables in processor core is provided via orderwire 1 for reading and updating tables. This function is primarily used by the load regulator to control processor work queue levels. Provisions have been made for up to 128 direct access device control message pointers. This means that 128 different direct access service messages may be defined. The content of direct access service messages varies depending on the particular processor center.

4.4 ORDERWIRE 2 SERVICE MESSAGES

Orderwire 2 service messages provide processor to device and processor to operator communication in the C-System. Orderwire 2 service messages issued by the processor are built by device acquisition and control service.

There are three types of orderwire 2 service messages. These are operator setup, orderwire 2 bid message input, and assign working channel. The following paragraphs describe each type.

4.4.1 Operator Setup

Device acquisition and control service issues an operator setup message to send setup directions for the requested devices to the operator. The message is directed to an operator printer which may be operating on either time division exchange loop 1 or loop 2.

For tape unit requests, device acquisition and control service sends a message for each tape in the request. This message is printed on a sticker which is attached to the tape reel for control and identification of that reel.

For loop 2 peripheral device requests, device acquisition and control service displays the operator setup service messages on an operator printer equipped with a preprinted label form that can be removed and used for routing or logging purposes.

4.4.2 Orderwire 2 Bid Message Input

Orderwire 2 bid message input service messages are transmitted by the device to the processor in response to an operator setup service message in which case this message input without a corresponding operator setup message is considered an unsolicated message.

In the solicited case, the operator follows the device setup directions and presses an initiating switch which causes the device to transmit the orderwire 2 bid message input service message to the processor. This message contains the device party line address, device type, control data, and device status.

In the unsolicited case, the control data field contains information which identifies this as an unsolicited message.

The remainder of the message is identical to that for a solicited message.

For both cases, the assign working channel function of device acquisition and control service is invoked.

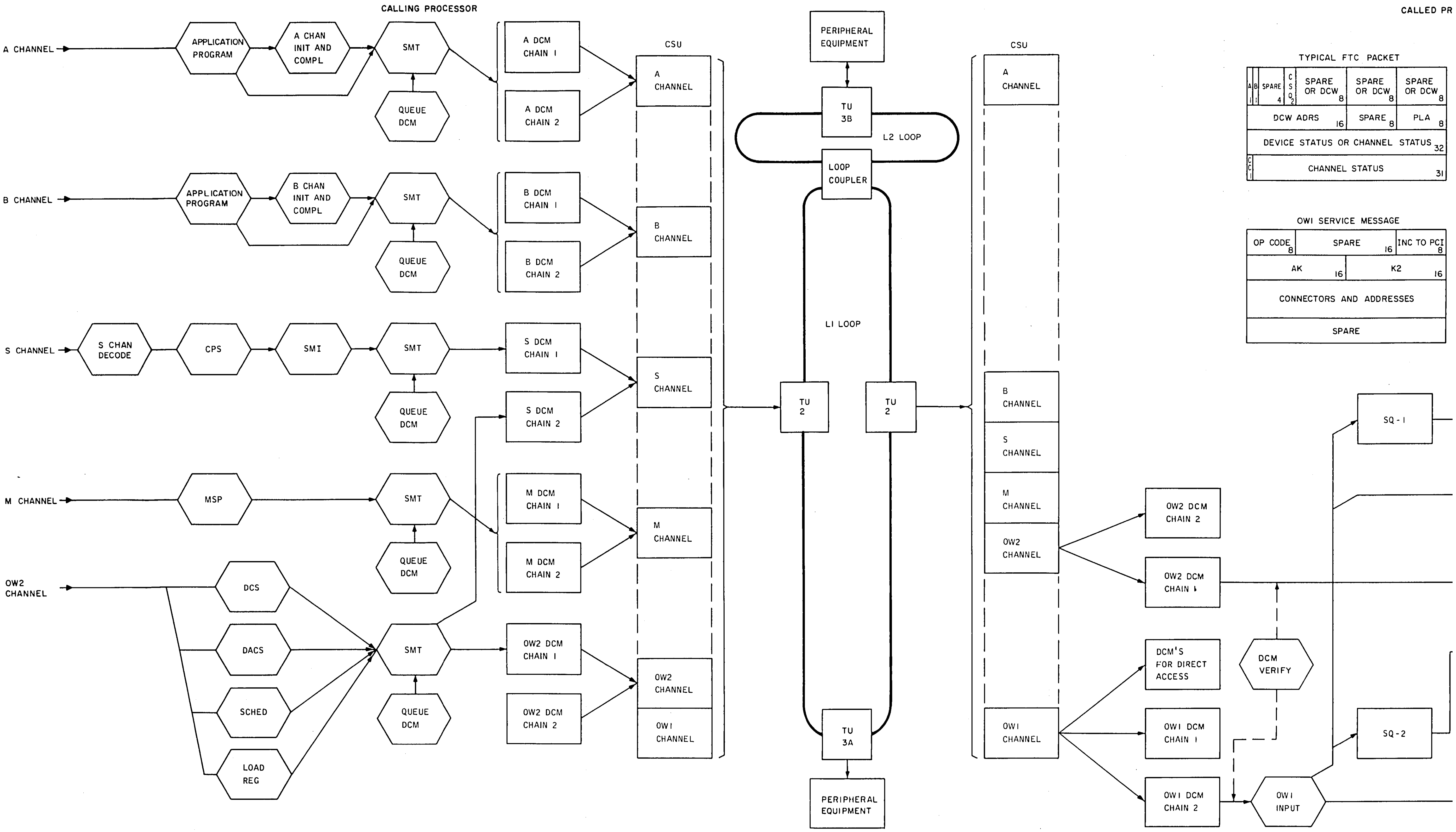
4.4.3 Assign Working Channel

This service message is a 1-word device command message which causes the loop 1 or loop 2 device to assume the working channel address contained in the message and switch from orderwire 2 mode to working channel mode.

service messages

In the solicited case, device acquisition and control service generates an orderwire 1 program return service message containing the disc file address of an output file which contains the assigned working channel and other information such as tape file identifiers, device status, etc.

In the unsolicited case, device acquisition and control service assigns a working channel to the device and builds a program call service message which contains the Time Division Address(es) (TDA) assigned to the device, its party line address, and the device status word. The program call service message invokes a B channel program which proceeds to service the device, building file transfer command packets, and issuing them to direct commands to effect communications with the device.



TYPICAL FTC PACKET

<table>
<tr><td>A 1</td><td>B 1</td><td>SPARE 4</td><td>CSO 2</td><td>SPARE OR DCW 8</td><td>SPARE OR DCW 8</td><td>SPARE OR DCW 8</td></tr>
<tr><td colspan="5">DCW ADRS 16</td><td>SPARE 8</td><td>PLA 8</td></tr>
<tr><td colspan="7">DEVICE STATUS OR CHANNEL STATUS 32</td></tr>
<tr><td>CCI</td><td colspan="6">CHANNEL STATUS 31</td></tr>
</table>

OWI SERVICE MESSAGE

<table>
<tr><td>OP CODE 8</td><td colspan="2">SPARE 16</td><td>INC TO PCI 8</td></tr>
<tr><td colspan="2">AK 16</td><td colspan="2">K2 16</td></tr>
<tr><td colspan="4">CONNECTORS AND ADDRESSES</td></tr>
<tr><td colspan="4">SPARE</td></tr>
</table>

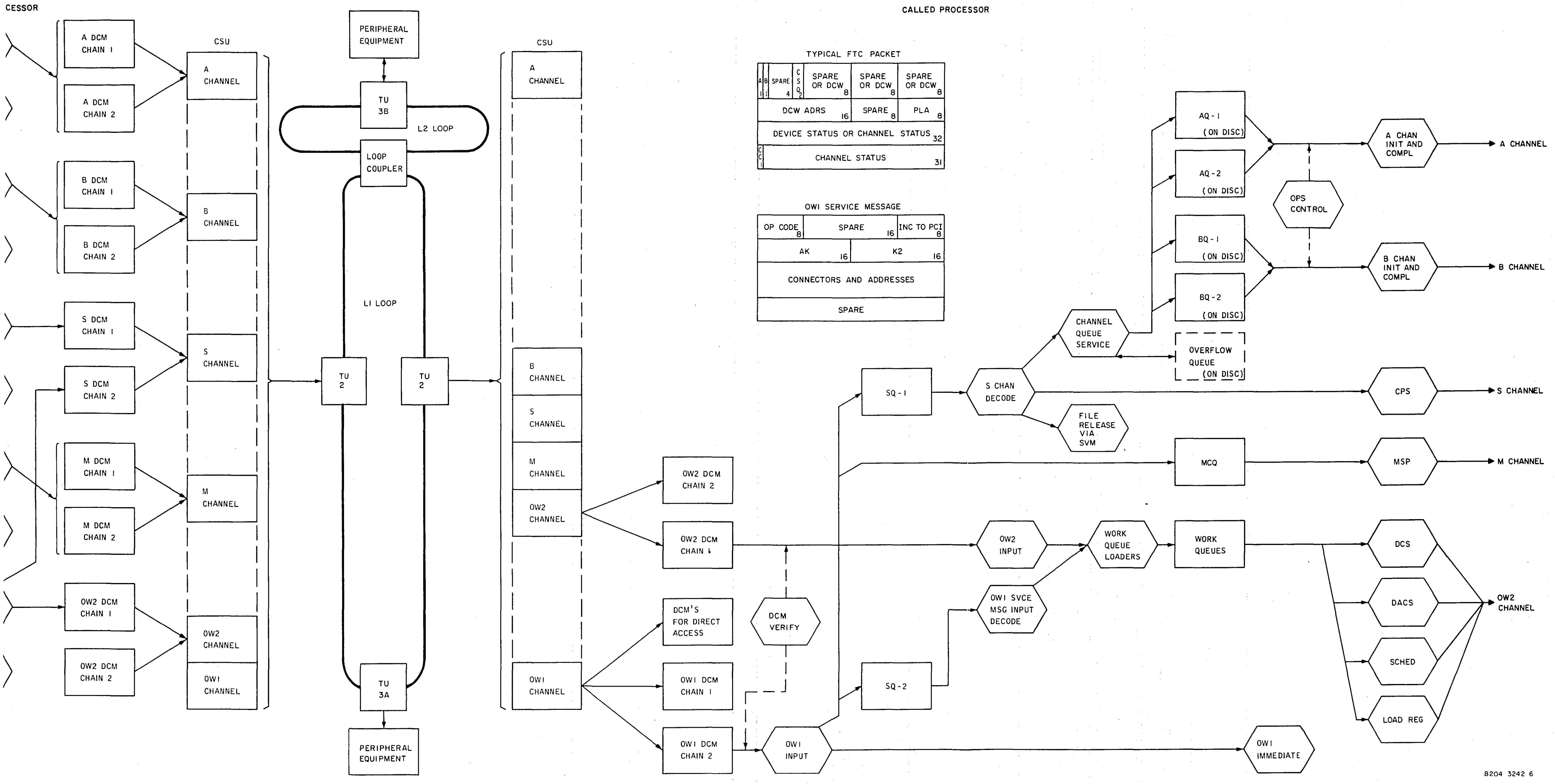


Figure 4-1. Service Message Flow.

section 5
time division exchange loop input/output operation

5.1 INTRODUCTION

The C-System uses the arithmetic logic and control unit to initiate input/output transactions; however, the actual input/output operation or transfer of data occurs independent of the arithmetic logic and control unit. Thus, the arithmetic logic and control unit can be executing other instructions while the input/output operation is taking place.

Input/output operations via the time division exchange loop involve the transfer of information between a processor and another processor and between a processor and a peripheral device such as a disc file, a magnetic tape, a card reader, or a line printer. The transfer of data to and from processor core storage is controlled by the communications service unit in the processor, using device control messages to determine the required input/output operations.

Figure 5-12 shows a two processor system with a disc file, magnetic tape, and operator printer. The figure also shows the user command packets, the programs involved in device control message initialization, the linkages between the device control messages and data areas, and the format of the various device control messages for several input/output transactions. These transactions are typical transactions required in C-System input/output operations.

The first transaction shown is a disc file write transaction. The figure shows the File Transfer Command (FTC) packet which contains the necessary parameters and pointers to allow the write direct file transfer routine (FTDI) to initiate the disc file write transaction. Also shown is a Data Control Word (DCW) list which contains pointers to the data to be written from core to disc. In this case, the data is to be written from noncontiguous areas of core. Also shown is the operations control table segment which contains the device control message pointers. The device control message is shown as an entry in the A channel device control message chain to be executed by the associated data channel. Also shown is the status packet which consists of the Device Status Word (DSW) and the Channel Status Word (CSW). The data channel places these words at this location upon acknowledgement from the device and data channel A.

The second transaction shown is an orderwire 1 service message being transmitted via the service channel data channel. The figure shows the file transfer packet, the various pointers, the device control message, (built by Service Message Transfer (SMT) in this case) and the operations control table segment.

The third transaction shown is an orderwire 1 service message input. The figure shows the receiving of an Orderwire 1 (OW1) service message via the orderwire 1 data channel. The message is read into core in an area pointed to by the device control message entry (SVMINADR) which is initialized for input by the orderwire 1 input. The figure shows that orderwire 1 input receives the service message and either invokes an orderwire 1 immediate function or makes an entry in the appropriate queue.

The last segment of the figure shows the transactions necessary to effect an Orderwire 2 (OW2) time division working channel address assignment. The figure shows the orderwire 2 bid input service message being received via device control message chain 2 of the orderwire 2 data channel. Note that the third word of the orderwire 2 service message, the device status word is stored as the fifth word of the device control message instead of being appended to the orderwire 2 service message. The message in this case is a solicited message issued in response to a tape setup request.

The figure shows that the orderwire 2 input program services the input service message and makes an entry in the device acquisition and control service work queue. This queue is subsequently serviced by device acquisition and control service which then initiates the transmission of the assign working channel service message as shown in the figure by building a file transfer command packet. The assign working channel service message is built by the orderwire 2 file service controlled and places it in the fifth word of the device control message.

5.2 CHANNEL ACQUISITION

Before a time division exchange loop input/output operation can take place, a time division exchange communication channel must be acquired. A time division exchange communication channel is established via a poll-bid-grant sequence initiated by a communications service unit data channel. The terminal unit assigned responsibility for channel control transmits the poll word on the time division exchange loop allowing the processors to bid for the channel. The data channel continuously monitors its assigned device control messages for input/output operations, reads the time division exchange address, and requests the terminal unit to acquire this channel (time division exchange address).

Upon receiving the data channel request, the terminal unit monitors the channel poll and bids for the channel. If the channel is granted, the terminal unit signals the data channel which is now in control of data transmission on this channel of the time division exchange loop. Upon completion of the transaction, the processor terminal unit disconnects from the channel, starting a new polling sequence.

5.3 INPUT/OUTPUT OPERATIONS

Input/output operations include communications between processors for transfer of service messages over the orderwire 1 time division exchange channel, between processors and tape or disc files for the transfer of data, between processors and low-speed peripheral devices for the transfer of data, and between processors and devices for transfer of service messages over the orderwire 2 time division exchange channel.

5.3.1 Orderwire 1 Service Message Transaction

Orderwire 1 service messages are generated by service programs and application programs to initiate control programs, provide program return status, initiate control functions, and perform other functions requiring communication between processors. Orderwire 1 service messages are generated by programs running in S, M, A, B and orderwire 2 channels. The service messages are sent on a time division exchange channel designated orderwire 1. The orderwire 1 channel is a 125-kbps party line channel recognized by all processors in a center. Each processor is assigned a party line address to identify its calls on the orderwire 1 time division exchange channel.

As shown in figure 5-12, the user initiating the service message builds a File Transfer Command packet (FTC) whose address is passed to the common function Service Message Transfer (CF SMT) during the calling sequence, a Data Control Word (DCW) pointed to by the file transfer command packet, and the orderwire 1 service message to be transferred via the orderwire 1 time division exchange channel. The data control word contains the address of the service message.

Note that orderwire 1 service messages are transmitted only by the A, B, M, and S data channels. Orderwire 1 service messages initiated by orderwire 2 channel programs are transmitted via the S channel data channel. Orderwire 1 service messages are received by the orderwire 1 data channel. The orderwire 1 input program initializes the device control messages of the unsolicited message.

5.3.2 Disc File Storage Transaction

High-speed data communication with disc files is used to read and write files, build and read queues, and read and write programs. File service requests are directed to the direct file transfer routines, either directly or by the indirect file transfer routines.

Figure 5-12 shows how the direct file transfer routines (FTD) are used to initiate the disc file transaction. The user in this case is the set of indirect file transfer routines. The indirect file transfer routines build a file transfer command packet whose address is passed to the direct commands. The file transfer command contains a pointer to the data control word list which defines the data cell to be written or the area into which the data cell is read. Figure 5-12 shows a data control word list which is noncontiguous and in which the data cell to be written to the disc file consists of data located in noncontiguous core locations.

The first two words of the file transfer command packet passed to the indirect file transfer routines are identical to those of the file transfer command packet passed to the direct file transfer routine. The third and fourth words differ as shown in figure 5-1.

The direct file transfer routines use the file transfer command packet to build the device control message required to effect the data transfer. Device command messages for all disc file transactions are built by use of the direct file transfer routines.

5.3.3 Tape File Storage Transaction

The process of initiating a tape file transaction is identical to initiating a disc file transaction once the required tape reel has been loaded anda working channel assignment made. To initiate a tape file transaction, the user, via a controlprogram, mustsend an orderwire 1 service message to Device Acquisition and Control Service (DACS) requesting the loading of the proper tape reel(s). Device acquisition and control service issues an operator printer message to the operator. After completion of the requested action, the operator presses the READY switch of thetape unit which initiates an orderwire 2 service message. This message conveys the tape unitparty line address, the mounted tape identifier (AK-RKN), and device status to device acquisition and control service. The device acquisition and control service then sends

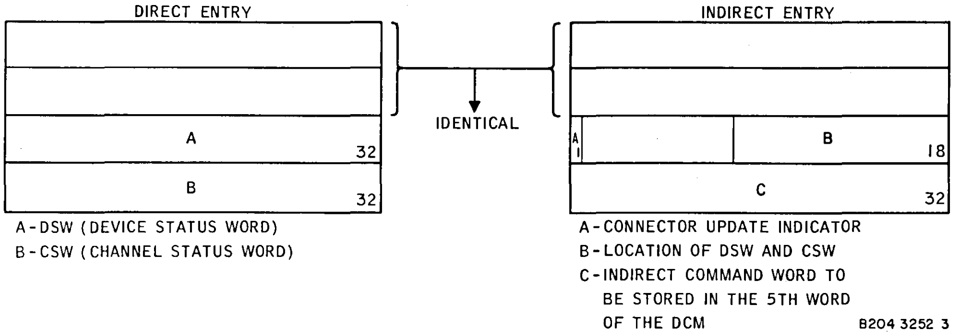

DIRECT ENTRY
INDIRECT ENTRY
IDENTICAL
A
32
B
32
A
I
B
18
C
32
A-DSW (DEVICE STATUS WORD)
B-CSW (CHANNEL STATUS WORD)
A-CONNECTOR UPDATE INDICATOR
B-LOCATION OF DSW AND CSW
C-INDIRECT COMMAND WORD TO
BE STORED IN THE 5TH WORD
OF THE DCM
B204 3252 3

Figure 5-1. File Transfer Packets.

an orderwire 2 service message assigning one of the available working channels to the terminal unit that interfaces this tape unit. With the tape mounted and working channel assignment completed, device acquisition and control service builds a return service message (program return) that references a device acquisition and control service output file containing the necessary parameters to allow initiating the tape file transaction.

5.3.4 Peripheral Device Transactions

Initiating communication with peripheral devices is identical to initiating tape file communication when the input/output request is solicited; that is, when the device setup is initiated by the orderwire 1 service message derived from the control program, device acquisition and control service request. An unsolicited input/output request is initiated by the device. The device transmits an orderwire 2 bid message input to the device acquisition and control service processor. The control field of the bid message allows device acquisition and control service to discriminate between solicited and unsolicited calls.

If the control data field of a received orderwire 2 service message indicates the request is unsolicited, the device acquisition and control service assigns a working channel to the device and builds a program call service message which contains the Time Division Address (TDA) assigned to the device, its party line address and device status word.

The program call service message invokes a B channel program which then proceeds to service the device, building file transfer command packets, and issuing them to the direct file transfer routines to effect communication with the device.

5.3.5 Orderwire 2 Service Message Transactions

Orderwire 2 service messages are used to transmit operator printer messages, to enable device-to-processor communication, and to assign working channel addresses.

As shown in figure 5-12, to initiate transmission of an Orderwire 2 (OW2) service message, the user builds a File Transfer Command (FTC) packet which has the same format as that used to initiate an Orderwire 1 (OW1) service message by an orderwire 2 program. The orderwire 2 file service controller detects the device acquisition and control service request as an orderwire 2 service message, builds a device control message, and calls Service Message Transfer (CF SMT) to verify the device control message and place it in the orderwire 2 channel device control message chain 1.

In the case of an orderwire 2 service message, the transmission is via the orderwire 2 data channel. Orderwire 2 device control message chain 1 is only used for transmitting orderwire 2 service messages. Note that all orderwire 2 file transfer requests are completed via the S channel data channel. Orderwire 2 input service messages are received via orderwire 2 channel device control message chain 2. Each time the orderwire 2 channel is indicated by the operations sequence table, the device control message verification routine of the operations control program checks for a completed device control message in chain 2 and extends control to the orderwire 2 input program. The transfer of control to orderwire 2 input by the device control message verification routine is achieved by setting the Connector Update (CU) bit when the device control message is initialized for input.

The orderwire 2 input program checks for an error in the device control message. If an error exists, orderwire 2 input clears the device control message, sets the Connector Update (CU) bit, and returns control to the operations control program. The device control message is now available for input and the service message just received must be reissued by the sender because orderwire 2 input has discarded the message in error. If no error exists, the Device Acquisition and Control Service (DACS) work queue is loaded and the next-read-position updated. The queue device control message routine is then called to reinitialize the device control message and to allocate an 8-word service message input buffer. The connector update bit is set and control is returned to the device control message verification routine which continues to verify the chain until all device control messages have been verified. Control is then given to the orderwire 1 input program. The orderwire 1 input program services S channel queue 2 and gives control to the work allocator which allocates time to all modules in the orderwire 2 channel. Device acquisition and control service is given control and subsequently services the orderwire 2 service message entry in its work queue. This, for example, might be solicited orderwire 2 bid message input as shown in figure 5-12 which causes device acquisition and control service to respond with an assign working channel orderwire 2 service message.

5.3.6 Direct Access Transaction

Direct access to tables in a processor core is provided via orderwire 1 for reading and updating the tables of interest to the load regulator. Each processor has in its core a set of previously built device control messages and data control word lists which reference these tables. In general, a pair of device control messages and data control words (one for reading, one for updating) is required for each table which the load regulator accesses. A device control message address table containing the addresses of the previously built device control messages is constructed in a fixed core location and its address wired in the communication service unit. Any table can be read or updated by selecting the appropriate device control message address from the device control message address table.

Figure 5-2 shows an example of the device control messages and data control words required to write a record from one processor to another via the direct access mechanism.

The device command word fields are as follows:

A - Receiving processor's orderwire 1 party line address

B - Spare

C - Device control message table displacement (0 for normal service message)

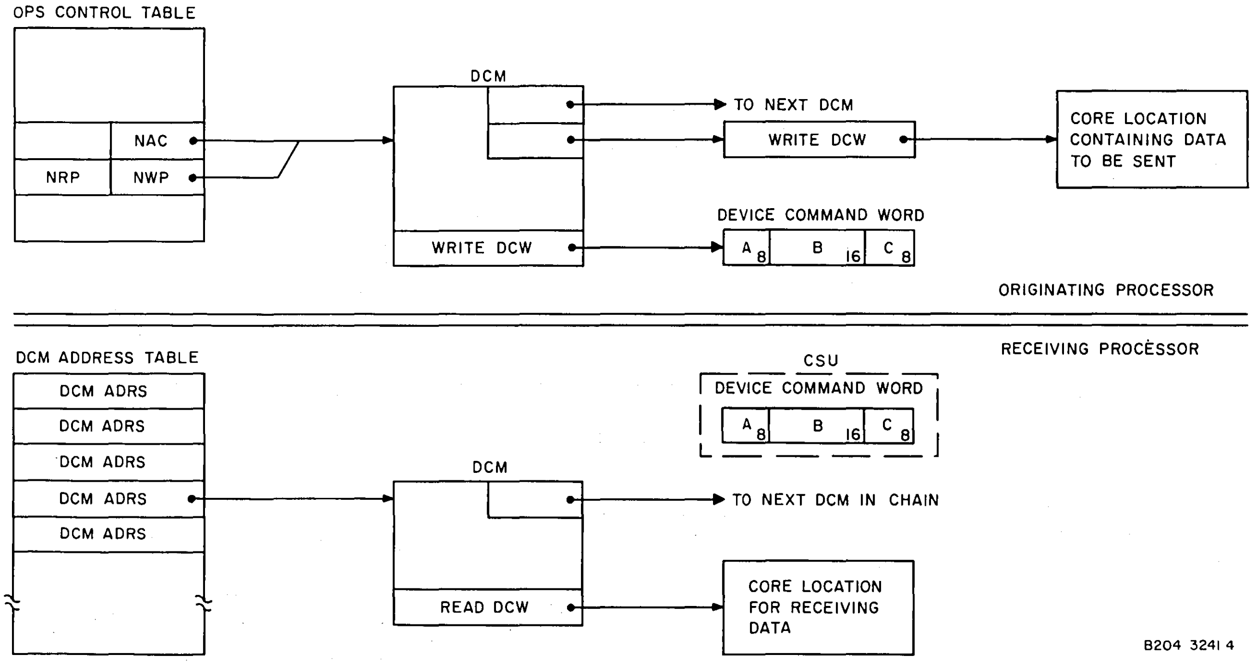


Figure 5-2. Direct Access Mechanism DCM's and DCW's to Write a Record.

To initiate the write transaction, the user builds a device command word and a Device Control Message (DCM) containing pointers to the device command word and the write Data Control Word (DCW). The write data control word contains the core location of data to be transmitted. The communications service unit of the originating processor then transmits the device command word over the orderwire 1 time division exchange channel. The receiving processor recognizes its orderwire 1 party line address and uses the device control message displacement field to enter the device control message address table to obtain the address of the required device control message. This address is used by the communications service unit to access the device control message that contains a pointer to the core location in which the received data is to be stored.

Figure 5-3 shows an example of the device control messages and data control words required to read a record from a processor via the direct access mechanism.

To initiate the read transaction, the user builds a device command word and a device control message containing a write data control word pointing to the device command word and a read data control word pointing to the core location where the received data is to be stored. The device command word is transmitted over the orderwire 1 channel. The receiving processor recognizes its orderwire 1 party line address and uses the device control message displacement field to enter the device control message address table to obtain the address of the required device control message. In this example, this device control message contains a write data control word which points to the core location of the data to be sent the originating processor.

5.4 DEVICE CONTROL MESSAGE EXECUTION

Device Control Message (DCM) chains and operations control table entries provide the software interface whereby the communications service unit controls the transfer of records to and from core storage of a processor without intervention of the arithmetic logic and control unit. Figure 5-4 shows the

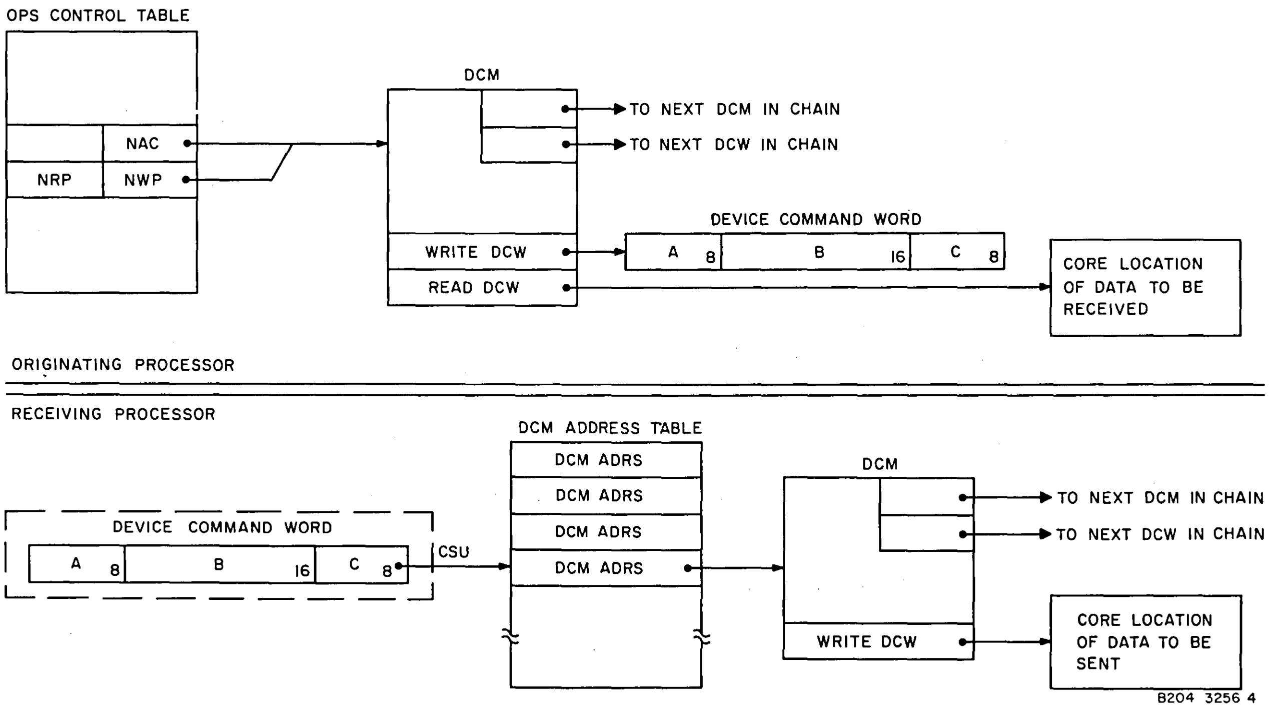


Figure 5-3. Direct Access Mechanism DCM's and DCW's to Read A Record.

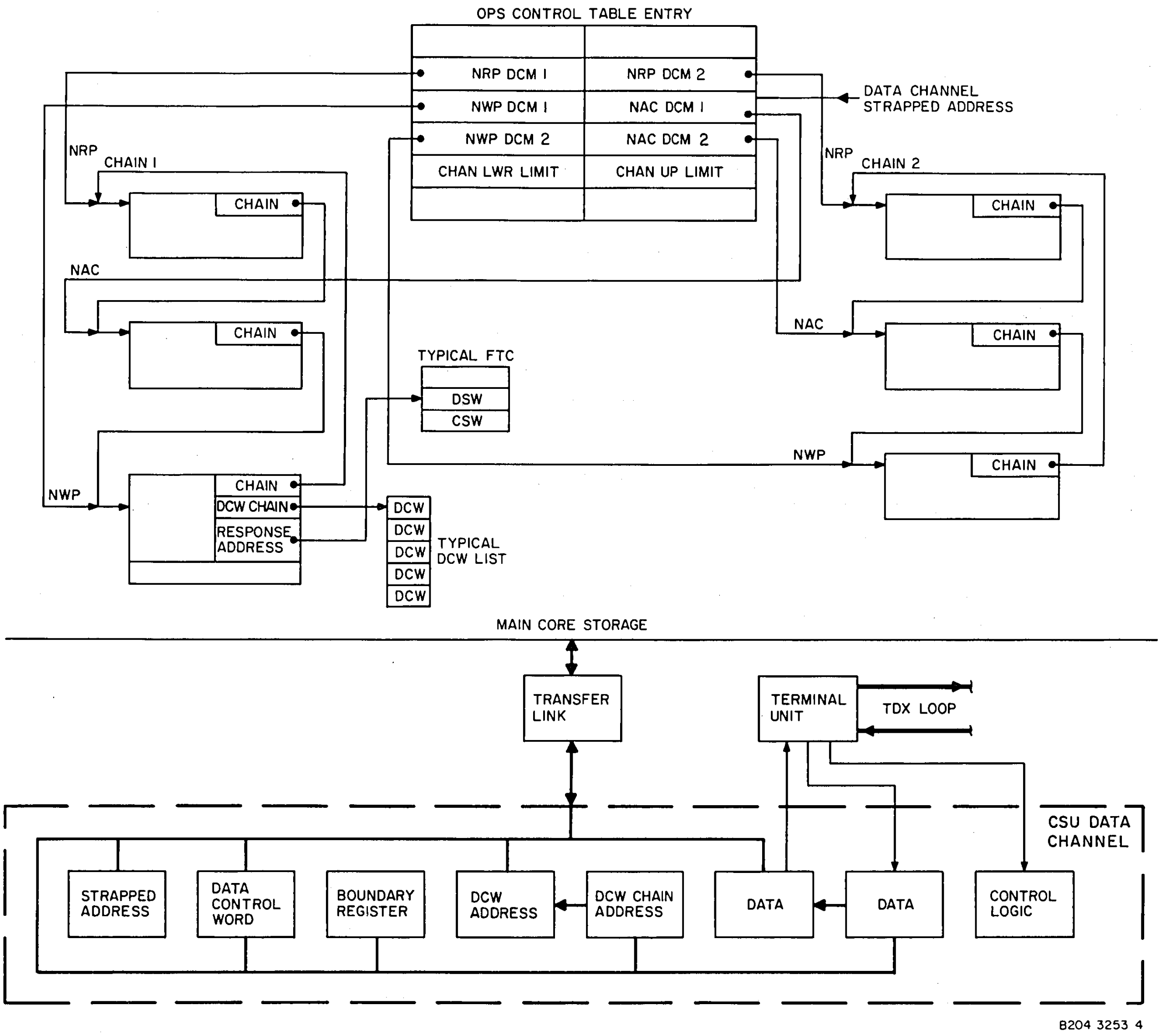


Figure 5-4. TDX I/O Hardware Software Interface Diagram.

hardware/software interfaces involved in time division exchange input/output operations. As shown, the data channel accesses core storage via the transfer link.

There are two modes of operation provided for device control message execution. One mode of operation (the data channel mode) is a self-chaining mode in which the data channel chains to the next device control message in the chain, updates the operations control table Next-Available-Cell (NAC) entry with the new device control message address, and then services a device control message in the alternate device control message chain. Data channels servicing A, B, M, and S channels operate in this mode.

The other mode (allotter mode) of operation is a single entry processing mode in which the data channel unit services one device control message per initiate command from the orderwire/absolute time clock unit. The orderwire/absolute time clock unit performs the allotter function for communications service unit data channels 3 and 4 to implement the Orderwire 1 (OW1) and Orderwire 2 (OW2) channel communication functions.

Device control message chains are constructed at processor initialization with a variable number of device control messages in a chain. This flexibility allows for different channel input/output requirements which may be encountered depending on the application.

Figure 5-5 shows a device control message chain. This circular list is a queue of device control messages.

Each channel maintains three pointers for each chain entry in its operations control table. These are the Next-Read-Position (NRP) pointer, the Next-Write-Position (NWP) pointer, and the Next-Available-Cell (NAC) pointer. The next-read-position pointer is used by operations control to locate completed device control messages. The next-available-cell is used by the communications service unit to locate the device control message to be executed and the next-write-position pointer is used by the various programs to locate the next device control message available for allocation.

In figure 5-5, all three pointers are shown pointing to the same device control message. This represents the idle condition for the device control message chain; the service queue and immediate chain bits in all device control messages are set to 1. As device control messages are allocated for input/output operations, the next-write-position pointer is moved forward (figure 5-6).

Device control messages positions 1, 2, and 3 are initialized when the processor channel is initialized. As the communications service unit completes execution of a device control message, it moves the next-available-cell pointer forward.

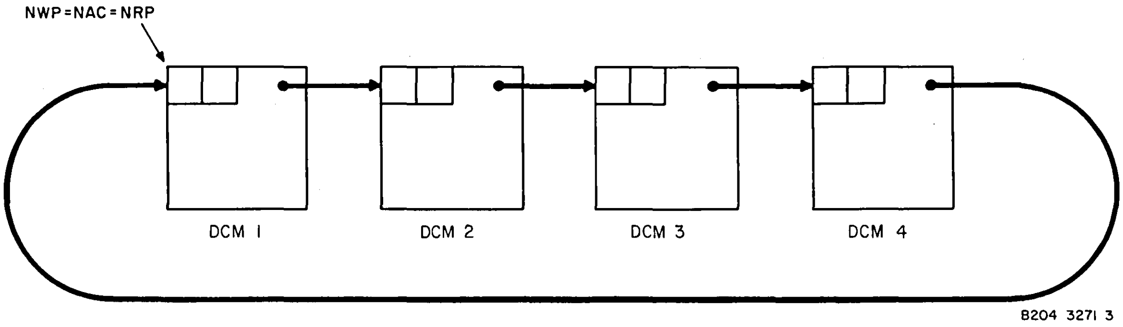


Figure 5-5. Circular Linked List.

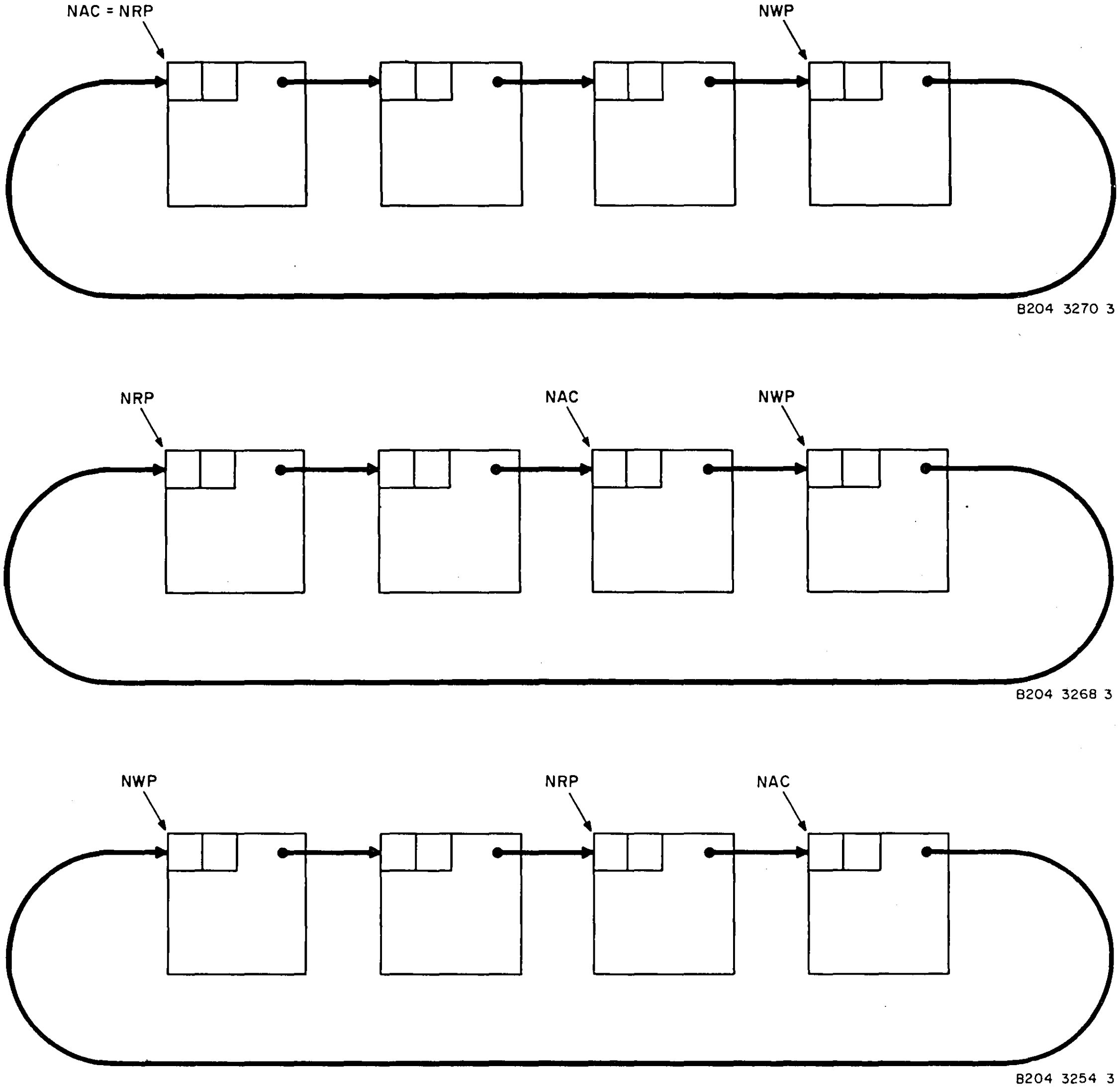


Figure 5-6. DCM Pointers.

The next time the operations sequence table points to this particular channel, operations control gives control to the device control message verification routine which performs verification of the device control message and updates the next-read-position pointer or gives control to the program pointed to by the from-program address field when the Connector Update (CU) bit is set. This program effects update of the next-read-position pointer either directly or indirectly. Completion of updating the next-read-position pointer returns control to operations control which then transfers control to the channel if time remains. A channel program may then allocate another device control message.

Assuming no further allocation of device control messages, the next-available-cell and next-read-position pointers are moved forward until eventually all three pointers again point to the same device control message position (figure 5-5).

5.4.1 Data Channel Operation

The communications service unit services processor channels A, B, M, and S in the data channel mode. Consider a device control message built to effect a data record transaction with disc file storage. The data channel is strapped to recognize the address of the next-available-cell pointer for device control message chain 1. At power on, the data channel services a device control message in chain 2 and then services the two chains on an alternate basis, one device control message at a time. The data channel contains a switching mechanism for servicing the two device control message chains on an alternate basis. The data channel is strapped to the address of the operations control table entry for device control message chain 1 next-available-cell. The data channel switching mechanism provides for switching a value between 0 and 1 which is added to the data channel strapped address to determine the address in core storage to be accessed.

The data channel accesses core storage to obtain the next-available-cell pointer. Assume that the next-available-cell pointer accessed is the device control message chain 1 pointer. The data channel then loads the contents of the word pointed to by the next-available-cell into a register. The word loaded is the first word of the device control message (device control message header) shown in figure 5-7.

SQ	IC	CU	IT	RETRY	ER	OP CODE	DCM CHAIN ADDRESS

Figure 5-7. DCM Header.

The data channel checks the Service Queue (SQ) bit. If the service queue bit is set to 1, the data channel enters an idle loop, times out, and switches to device control message chain 2. The data channel continues cycling in this loop until the service queue bit is set to 0 in one of the chains. The data channel next checks the Immediate Chain (IC) bit. When the immediate chain bit is set, the data channel updates the device control message header word by setting the service queue bit to 1 and the Error (ER) bit to 0 and storing the device control message header word back into the device control message. Next, the data channel updates the operations control table by storing the device control message chain address at the next-available-cell position. The data channel then switches to the other device control message chain and proceeds to service it before returning to service the device control message pointed to by the updated next-available-cell pointer.

Consider the conditions encountered in a device control message with the service queue bit set to 0 and the immediate chain bit set to 1. When an error occurs during the execution of a device control message, the data channel sets the service queue and error bits of the device control message header to 1, but does not update the next-available-cell pointer with the chain address. Servicing of this chain by the data channel is blocked. The procedure followed at this point varies somewhat depending on which program built the device control message. When the operations control device control message verification routine gains control and the error bit is set, control is given to the program whose address is contained in the from-program address field of the device control message. This field is the first half of the third word in the device control message (figure 5-12). Action at this point varies with the particular from-program. For example, the error handler of the direct file transfer routines retries the device control message seven times whereas the error handler of the service message transfer routine retries the device control message only three times. The retry field of the device control message header is used as a counter by the error handler programs to maintain the count of the number of times a device control message has been retried.

Retry of the device control message is accomplished by clearing the service queue and error bits of the device control message. The service queue bit controls initiation of device control message execution by the data channel. If the error cannot be cleared by the error handler, control is returned to the device control message verification routine which then sets the operations control table error bit for this device control message chain, bit 11 for chain 1 and bit 12 for chain 2, and indicates the chain complete. After verification of both chains is complete, control is given to the channel. It is then the responsibility of the channel user to take action to clear the device control message permanent error.

Action taken in clearing the device control message chain varies depending on the user. In the case where the user decides to bypass the device control message, the user can call the Direct Error Correction (DEC) routine which sets the service queue bit to 0 and the immediate chain bit to 1. This condition results in the data channel immediately chaining to the next device control message in the chain.

Execution continues with the device control message for the case in which the immediate chain bit is 0. The data channel proceeds to load a register with the channel lower and upper boundaries. The channel boundary word is contained in the operation control table entry for each channel as shown in the following diagram.

NRP DCM-1 16	NRP DCM-2 16	
NWP DCM-1 16	NAC DCM-1 16	← Q
NWP DCM-2 16	NAC DCM-2 16	
CHANNEL LOWER LIMIT 16	CHANNEL UPPER LIMIT 16	

Operations Control Table Entry.

The data channel actually retains only the seven most significant bits of both the lower and upper boundary addresses. Each address is viewed as an even address with the nine least significant bits of each address appearing as zeros. Therefore, the lower and upper limits of X'29FF and X'2BFF appear as X'2800 and X'2A00 to the data channel. Data is stored from X'2800 up to X'2A00 without detecting an error. Anything below X'2800 or equal to or greater than X'2A00 causes the operation to abort. Therefore, the channel limits selected must be even addresses and should have the nine least significant bits of each address all zeros. This field is checked only when the data channel is in the data channel mode and receiving.

Next the data channel accesses the device control message operation code field. This field is shown in figure 5-8.

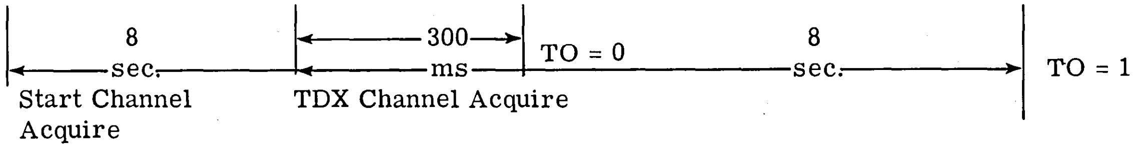
TO 1	SPARE 3	CODE 3

Figure 5-8. DCM Operations Code Field.

Where:

a. TO is the timeout indicator for the data channel operation once the desired time division exchange channel is acquired. If TO is 0, then the operation must complete within 300 ms. If TO is 1, then the operation must complete within 8 seconds. If the desired channel cannot be acquired within 8 seconds, then data channel timeout on the device control message occurs.

For either type of data channel timeout, the service queue and error bits in the device control message header are set, and device control message chaining does not occur. The Channel Status Word (CSW) is stored indicating the type of timeout. Timeout limits are shown in the following diagram:



b. CODE is a 3-bit field used with the orderwire data channels and data channels 3 and 4.

The data channel is now ready to obtain the loop 1/loop 2 address of the time division exchange channel to be acquired. The data channel increments the register pointer to the device control message so that it now points to the second word of the device control message which is shown in the following diagram.

L1 ADDR. 8	L2 ADDR. 8	DCW CHAIN ADDRESS 16

A typical disc file storage loop 1/loop 2 address is X'20/00.

The data control word chain address field normally points to the data control word list that defines the data to be transmitted or the core area in which received data is to be stored.

The data channel loads the second word of the device control message, increments the register pointer by 2, and sets up to shift the loop 1/loop 2 address field to the Terminal Unit (TU). The data control word chain field is loaded into a register and the data channel Chain Valid (CV) indicator, located in the data channel, is set if the data control word chain field is not 0. The time division exchange channel acquisition timer T0 is set at this time.

The data channel next loads the fourth wordof the device control message from core storage. This word is always the first data control word and has the following format:

E O L_1	C H_1	S K_1	R W_1	R B_1	SP 2	COUNT 9	DCW CHAIN ADDRESS OR CORE ADDRESS 16

If the chain valid indicator is 0, the data control word chain address field of the first data control word is loaded into the register which previously contained the data control word chain field of the second word in the device control message. The data channel checks the data control word Chain (CH) bit. This bit must not be set in the first data control word if the data control word chain field was originally other than a 0 as this causes the data channel to set the error bit in word 1 of the device control messages. The data channel continues to process the data control word list. As shown in figure 5-12 for the disc file storage transaction, the direct file transfer routines build the device control message with the first data control word pointing to the device command address which is always the eighth word of a device control message.

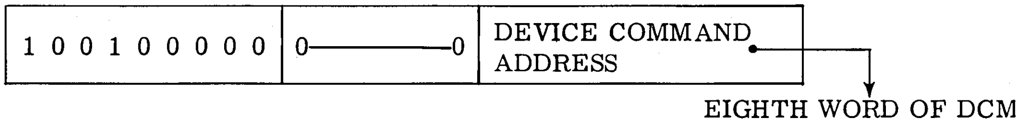
Next, the data channel sets a status read/write bit with the Read/Write indicator (RW) bit of the data control word. If this bit is set to 1, data is transmitted from the processor. If this bit is set to 0, data is received by the processor.

The data channel now raises the acquire channel line to the terminal unit and waits for the terminal unit to signal acquisition of the specified time division exchange channel by raising the channel acquired line to the data channel. When the channel is acquired, the data channel resets the timer with the value indicated in the device control message operation code field (TO), that is, 300 ms or 8 seconds.

Next, the data channel checks the status read/write bit to determine if this is a transmit or receive operation. The following discussion is for the transmit operation; however, the differences for the receive operation are explained.

The Skip (SK) bit is then tested. In a transmit operation, setting this bit results in the data channel sending 32-bit words of all zeros to the device. The number of words transmitted is equal to that specified in the count field plus one. The core storage address of the data control word is not used, and no memory access is required to send the 32-bit words of all zeros. In a receive operation, the data channel discards data received from the device for the number of words specified in the count field plus one. A user could use this feature to selectively read fields of a data record or to zero fill a disc file or tape record.

If the skip bit is 0, data is read from core or written into core. This bit is always 0 in the first data control word since the first data control word points to the device command word transmitted as the first word of the input/output transaction. The data control word for transmission of the device command word is as follows:



1 0 0 1 0 0 0 0 0	0——————0	DEVICE COMMAND ADDRESS

The End-Of-List (EOL) bit set indicates this is the last data control word in this list. The read/write bit set indicates transmit and the count field set to 0 indicates one word is to be transmitted, that is, the device command word. The data channel transmits this word and then chains to the address specified in the data control word chain address field. The chain valid indicator is also set to 0.

For the disc file storage transaction shown in figure 5-12, the first data control word in the data control word list contains a data control word chain address. The data channel loads this word and sets a register with the chain address. With the chain bit set to 1 and the end-of-list bit set to 0, the data channel increments the data control word pointer, sets the chain valid indicator, and loads the next data control word in the list. This data control word is now the current data control word.

The data channel then proceeds to transmit the first word pointed to by the current data control word and tests the count field for 0. If it is not 0, it is decremented by one. Figure 5-9 shows a data control word built to effect the write of an 8-byte disc file record header.

The count field controls the number of 32-bit words which are read from core storage and transmitted or the number received and stored in core storage.

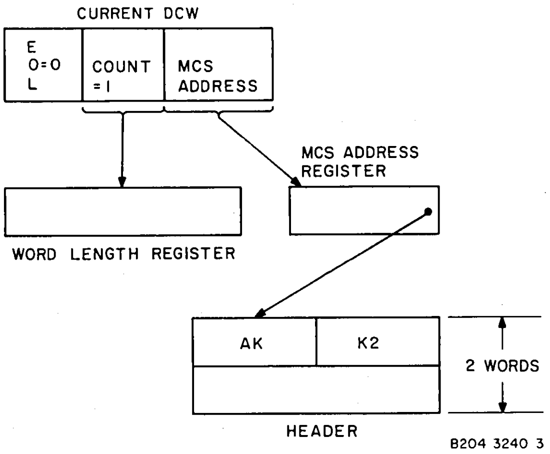


Figure 5-9. Device Control Message.

The data channel next tests the Read Backward (RB) bit. If this bit is set to 1, the core address specified in the data control word is decremented each time a word is read from or written to core. Data is thus stored in descending sequential locations. If this bit is set to 1, the core address specified in the data control word is incremented each time a word is written to or read from core.

As each data control word in a list is completed, the data channel checks the end-of-list bit. If this bit is not set, the next data control word is loaded and executed. This continues until the end-of-list is detected. The data channel then checks the chain valid indicator to determine if there are further data control words to be executed. If there are, it chains to the first data control word in the new list and continues data control word execution. Eventually, all data control words are executed and the data channel enters a wait loop until the Device Status Word (DSW) is received and buffered in the data channel.

Figure 5-10 is a data control list for a disc file storage input/output transaction.

In the left half of the first word (X'4000), only the chain bit is set. This results in the data control word chain address contained in the right half of this word being loaded into the data channel chain address register, and the chain valid indicator set. The data channel then increments to the next word in the list since the end-of-list bit is not set.

The second word of the data control word list contains X'1001 in the left half. The decoding result is that the read/write bit is set and the count field is 1. This data control word causes the data channel to transmit the 2-word header and then increment to the next word in the data control word list since the end-of-list bit is not set.

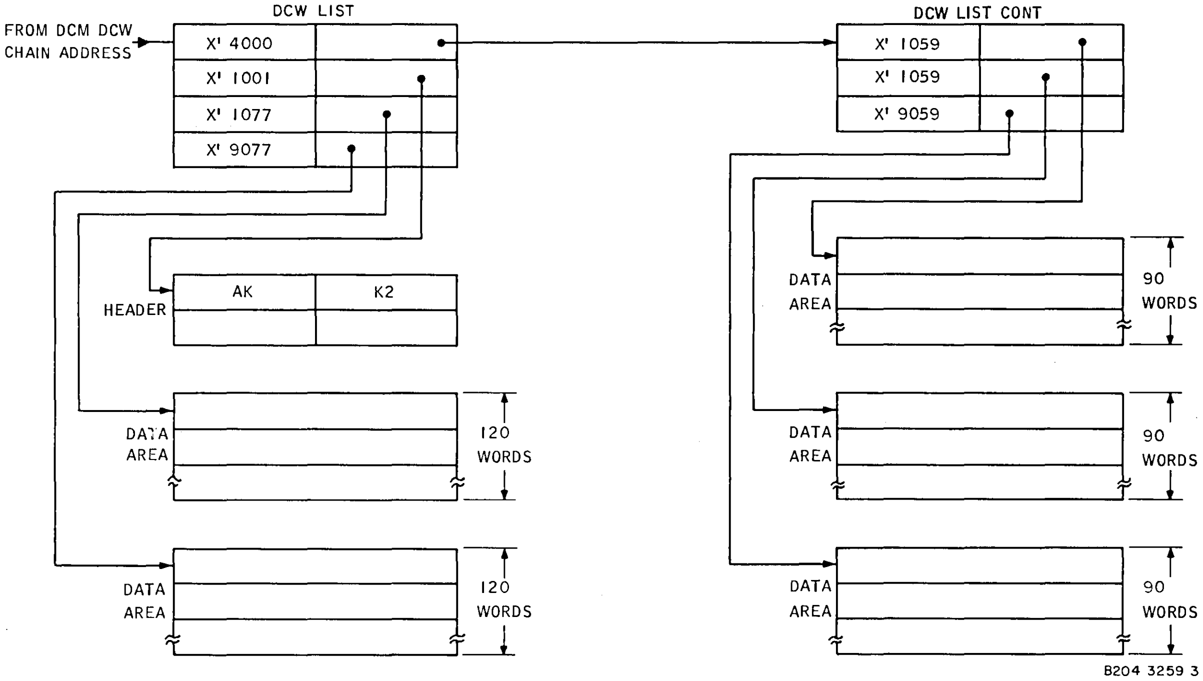


Figure 5-10. Data Control Word Lists.

The third word of the data control word list contains X'1077 in the left half. This specifies the transmission of 120 words of data from the core storage area pointed to by this data control word. Again, the data channel increments to the next word in the data control word list since the end-of-list bit is not set.

The fourth word of the data control word list contains X'9077 in the left half. This again specifies the transmission of 120 words of data from core storage. This time, however, the end-of-list bit is set, causing the data channel to load its data control word pointer from the data control word chain address register.

The first word of the new data control word list contains X'1059 which specifies the transmission of 90 words of data from core storage. The data channel again increments to the next word (X'1059) which also specifies the transmission of 90 words of data. The data channel then increments its pointer to the last word in the data control word list. This word contains X'9059. At the completion of transmitting the 90 words, the data channel determines that data transmission is complete since the end-of-list bit is set and the chain valid indicator is 0. The data channel then enters the wait loop for device status.

Several important facts concerning the use of the data control word are:

a. Data to be written or read from core storage does not have to be sequential in core storage. This allows a user to perform scatter read or write operations in a very efficient manner.
b. The maximum amount of data any one data control word can transfer is 512 words; that is, the count field is nine bits in length. This is also the maximum record length transfer allowable in the C-System.

c. Where possible, data records should be referenced by a relatively small number of data control words containing relatively large length-counts rather than by a long list of data control words with short length-counts.

d. The number of data control word lists should be kept to a minimum, that is, data control words should be in contiguous locations in core storage as much as possible.

e. The number of levels of chaining between data control word lists should be kept to a minimum.

Data control word execution continues at the point where the device status word has been received. The data channel now loads the third word of the device control message shown in the following diagram.

FROM-PROGRAM ADDRESS	RESPONSE ADDRESS (DSW)

The data channel makes use of only the response address field. The data channel stores the device status word at the location specified by this field. The general format of the device status word is:

A	B	C	D	E	F
1	1	16	1	11	2

Where:

A Program Retry — If set, device busy, service programs attempt immediate retry.

B Not Ready — If set, device is not ready and operator intervention is required to complete data transfer.

C, E — Specific status bits indicate either the nature of the error or additional information about the state of the device.

D Error Indicator — If set, indicates that an unusual condition, specified by other status bits, was detected during execution of a command. This bit is used by the communication service unit to set the device control message error bit.

F Command Status — Command status bits which are set by service routines to indicate the completion status set by the device control message verification program.

Code:

00 - The associated command has been successfully completed (a count disparity may be present in the channel status word).

01 - Direct command in error. The device control message queue involved is blocked by a device status word error. The execution of all commands in the device control message queue has been suspended and no further commands from the responsible channel are accepted until direct error correction has been invoked.

10 - Direct command in error. The device control message queue involved is blocked by a channel status word error. The execution of all commands in the device control message queue has been

suspended and no further commands from the responsible channel are accepted until direct error correction has been invoked.

11 - Indirect command. The response status is returned to the file transfer service routines. These routines build indirect device control messages which if in error do not block the user's direct device control messages. Status concerning the indirect command is contained in the channel status word. The data channel increments its pointer by one and stores the Channel Status Word (CSW). The channel status word format is:

A	B	C	D	E	F	G	H	I
1	1	1	1	1	1	1	9	16

Where:

A CC — If set, indicates command complete. This bit is not set by the data channel unit but instead it is set by the following control and service programs:

a. operations control when successful completion is found,

b. an error handler when further retry is not feasible,

c. or during the responsible channel time frame when direct error correction has been invoked to bypass a direct command in permanent error and optionally all other direct commands in the same device control message queue (channel status word bits 1-31 equal to 0 indicate the occurrence of the latter).

B SS — For indirect command - Spare.

For direct command - If 0, channel status word stored at completion of transaction. If 1, channel status word was not stored because of a channel error.

C TO — Time out of data transfer by the data channel.

D TE — Time division exchange error; loss of sync on time division exchange loop set by data channel unit.

E CE — Count error, set whenever residual count is not equal to 0 when the data channel unit receives a device status word.

F IE — Initiate error.

G ER — Error bit set by terminal unit for time division exchange error.

H Residual Count — The residual count indicate the number remaining in the word count register when the device status word was received.

I DCW Address — The data control word address, which specifies the memory location succeeding that of the data control word that was being executed when the device status word was received.

For indirect commands in error, the channel status word format is:

L	E	LI	A
1	7	8	16

Write File L = 1: The file being constructed has been aborted and subsequent commands are not accepted for the file.

Read File L = 0: The file structure is invalid and subsequent commands are not accepted for the file.

L ≠ 0: A file cell cannot be retrieved. The next read indirect command will be honored and the unretrievable cell bypassed.

FIELD

E: Error type

0 - End of File.
1 - Invalid AK/K2 connector cell identifier.
2 - Data or connector cell retrieval error.
3 - Data or connector cell write error.
4 - Lack of file linkage block space.
5 - Error occurred on the output file referenced by the close command. The file is not documented and the file structure may be invalid.
6 - Disc file storage space unavailable.
7 - System error: This error may be the result of the user channel program incorrectly modifying the file linkages block or file structure.
8 - to be defined.

LI: Level indicator - Data cells indicated by level 1 and low level connector cells at level 2.

A: Address of unretrievable cell - This may occur for an indirect write as well as in indirect read operation.

Note

The device status word for error codes 2 and 3 are that of the operation in error unless a channel status word error occurred. In this case, the device status word is replaced by the channel status word left shifted by 2.

For error-free completions, the command complete bit (bit 0) of the channel status word is set by a device control message verification routine. When control is extended to the user program, the user checks the command complete bit of the channel status word and the command status bits of device status word in his transfer command packet for successful completion of the data transfer.

When an error occurs during the file transfer sequence, appropriate error indicator bits in the device status word, channel status word, and device control message are set. The device control message verification routine extends control to the error analysis routines when the device control message Error bit (ER) is set. When the error handler routines determine that further retry is not feasible, the channel status word command complete bit is set, the command status bits (bits 30, 31) in the device status word are set to indicate the nature of the device control message error, and error bit 11 for device control message chain 1, or 12 for device control message chain 2, in the operations control table entry are set. When error bit 11 or 12 is set, the device control message queue service routine does not allocate space for issuing additional device control messages. The verification routine then terminates, and control is extended to the user program with the device control message chain idle. The user then determines what further error processing is required. The user has the option of unblocking the device control message chain by invoking the Direct Error Correction (DEC) routine to skip the device control message in error or skipping all direct command device control messages in the chain.

When direct error correction has been invoked, error bit 11 or 12 in the operations control table entry is reset. Direct error correction has no effect on indirect device control messages which are created and maintained by the file transfer service routines. The data channel checks for a device error (bit 18 of device status word set), a timeout error (timeout bit of channel status word set), a time division exchange error (time division exchange error of channel status word set), a count error (count error bit of channel status word set), an initiate error (initiate error bit of channel status word set), or an error indication set by the terminal unit for a time division exchange error (error bit of channel status word set). Any one of these error conditions causes the data channel to set an error indicator. The data channel then loads the device control message header word and sets the service queue bit to 1 and the error bit to the value contained in the error indicator, that is, 0 if no error occurred and 1 if an error occurred.

If an error has occurred, the data channel switches to the alternate device control message queue and begins servicing a device control message in this chain without updating the next-available-cell pointer. For the no error condition, the data channel first completes the update of the next-available-cell pointer by setting it to the device control message chain address and then toggles to the alternate queue.

5.4.2 Orderwire 1 Channel Operation

Orderwire 1 service messages are transmitted via the data channels servicing the A, B, M or S processor channels. The first word transmitted over the orderwire 1 channel is the device command word, which the service message transfer routine builds as the fifth word of a device control message. The format of this word is as follows:

Party Line Address 8	SPARE 16	DCM Table Displacement 8

This word contains the Party Line Address (PLA) of the called processor and a device control message table displacement that is used during direct access transactions.

Detection of channel granted on the orderwire 1 time division exchange channel by the called processor's terminal unit results in raising the channel granted line to the orderwire/absolute time clock unit. This results in the first data word (the device command word) being transferred to the orderwire/absolute time clock unit. The orderwire/absolute time clock unit compares the party line address of the called processor with its own party line address. If they compare, the orderwire/absolute time clock unit enables the terminal unit and orderwire 1 data channel for further reception of data control words.

If the device control message table displacement is 0, the orderwire/absolute time clock unit loads the address of the device control message from the operations control table entry. Figure 5-11 shows the operations control table entry for the orderwire 1 channel and the processor interface table entries used by the orderwire 1 channel.

The device control message header (word X'180) is accessed and the service queue and immediate chain bits examined. If the service queue bit is set to 1, the Orderwire/Absolute Time Clock unit (OW/ATC) responds with a busy signal to the calling data channel and appears busy until a device control message is allocated for inputting an orderwire 1 service message; that is, the service queue bit is set to 0. The busy signal transmitted by the orderwire/absolute time clock unit is 34 bits of alternate ones and zeros. This signal received by the calling data channel is interpreted as status from the called processor. The calling data channel stores the received word as the Device Status Word (DSW) for this transaction.

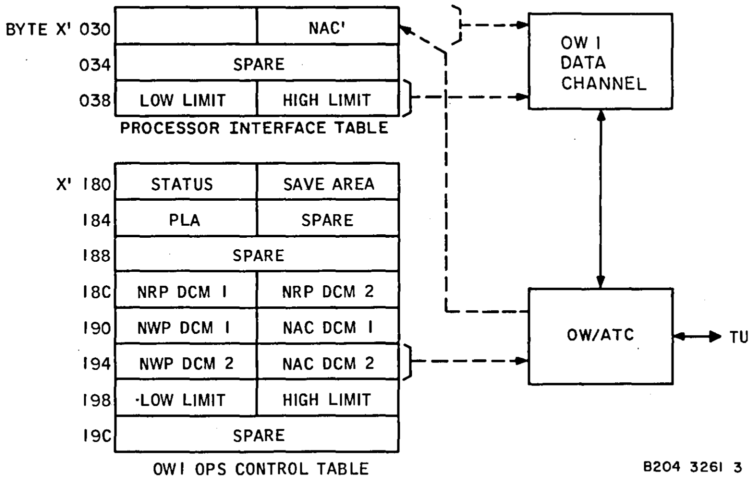


Figure 5-11. Operations Control Table Entry and Processor Interface Table Entry.

The calling data channel also finds that the device status word error bit is set (bit 18, counting 0 to 31) and, therefore, sets the device control message error bit. When the device control message verification routine gains control and finds the error bit set, it passes control to the program specified by the from program address field. In the case of an orderwire 1 transmission, this is an error handler of the service message transfer routine. This handler is set up to retry the device control message three times before declaring a permanent error condition. Thus, the orderwire one channel of the called processor being busy the first time a call is made does not indicate that the transmission will not be completed successfully.

If an immediate chain condition (service queue bit is 0, and immediate chain bit is a 1) exists in the called processor, the next-available-cell device control message chain 2 field in the operations control table is updated with the device control message chain address, and the next device control message header is accessed. If the service queue and immediate chain bits are 0, the orderwire/absolute time clock unit stores the device control message chain address in the location orderwire 1 data channel is strapped for, that is the next-available-cell (used by orderwire 1 data channel) located at core address X'32. The orderwire 1 data channel is initiated in the allotter mode; single device control message execution mode. The orderwire/absolute time clock unit clocks-out and discards the loop 1/loop 2 address from the orderwire 1 data channel since the transmission is being received on the orderwire 1 channel. This means that the loop 1/loop 2 address fields in an orderwire 1 input device control message are not used. The orderwire/absolute time clock unit now raises the channel acquired line to orderwire 1 data channel.

The orderwire 1 data channel accesses the next-available-cell pointer used by the orderwire 1 data channel to obtain the address of the device control message. Also the upper and lower channel limits stored at X'38 are used when verifying the address in which data is to be stored. The orderwire 1 data channel proceeds to execute the device control message in the same manner as for data channel operation. As shown in figure 5-12, only one data control word is used for receiving an orderwire 1 service message. The address in this data control word points to the core location in which the received service message is to be stored.

Transmission continues until the count field in the data control word is decremented to 0. The orderwire 1 data channel then transmits its channel status word to the calling data channel. If an error is detected by the data channel, it sets bit 18 of the status word. The data channel uses this bit as an error indicator on the device status word and, if this bit is set, causes the data channel to set the device control message error bit. The received channel status word is interpreted as the device status word for an orderwire 1 service message transaction. When the calling processor receives the status word, it responds by sending its channel status word to the called processor. The called processor then stores the received status in the device status word position which, for orderwire 1 input, is the fifth word of the device control message. The channel status word is stored in the sixth word of the device control message as shown for the orderwire 1 service message input transaction in figure 5-12.

If no error occurred during the transaction, orderwire 1 data channel raises the Complete (COMP) line to the orderwire/absolute time clock unit which then resets the allotter initiate signal to the orderwire 1 data channel. The orderwire/absolute time clock unit then accesses the device control message header word and sets the service queue bit to 1, indicating completion of device control message execution. The orderwire/absolute time clock unit also updates the next-available-cell pointer for device control message chain 2 with the address contained in the device control message chain address field, completing the orderwire 1 service message transaction.

If an error occurs, the orderwire 1 data channel sends an error indication to the orderwire/absolute time clock unit and then raises the complete line to the orderwire/absolute time clock unit. The orderwire/absolute time clock unit again updates the device control message header setting the service queue bit to 1 and also setting the error bit. The next-available-cell for chain 2 is not updated with the chain address but is set to the address of the device control message in error. This orderwire 1 channel then appears busy to calling processors until the error is serviced.

A nonzero value for the displacement field of the device command word indicates an orderwire 1 direct access operation. When the displacement is not 0, the device control message address is obtained from the device control message address table by using an address which is computed using a strapped base as the most significant byte and the displacement as the least significant byte. The device control message header is accessed and the service queue and immediate chain bits are examined.

If the service queue bit is a 1, the orderwire/absolute time clock unit responds busy to calling processors. This condition occurs only if an error has occurred during the execution of this direct access device control message. If an immediate chain condition (service queue bit is a 0 and immediate chain bit is a 1) exists, the device control message chain address is loaded and the next device control message header accessed. If the service queue is a 0 and the immediate chain list is a 0, the orderwire/absolute time clock unit stores the device control message address obtained from the address table at the next-available-cell position and initiates the orderwire 1 data channel.

At the completion of the operation, status is exchanged in the same manner as described for an orderwire 1 service message. When the transaction completes without error, the orderwire/absolute time clock unit does not update the device control message header nor is the next-available-cell device control message chain 2 updated. Thus the device control message is left in the initialized state ready for a subsequent operation. If an error occurs, the orderwire/absolute time clock unit updates the device control message header setting the service queue and the error bits to a 1 and stores the address of the device control message in error in the next-available-cell device control message chain 2 position.

5.4.3 Orderwire 2 Channel Operation

The orderwire 2 data channel is used only to transmit and receive orderwire 2 service messages. This data channel normally services the B channel, which replaces the orderwire 2 channel in nonorderwire 2 processors (only one orderwire 2 processor is required per processor center), in the data channel mode. When used in this manner, the orderwire/absolute time clock unit is idle and connects this data channel directly to the terminal unit.

In the orderwire mode, the orderwire/absolute time clock unit and the orderwire 2 data channel operate in conjunction to implement communication on the orderwire 2 time division exchange channel. The transfer of data and reporting of channel status during device control message execution is accomplished by the orderwire 2 data channel. The orderwire/absolute time clock unit controls the sequence of device control messages that are serviced and performs the operations control table update function and the device control message chaining, initiation, and completion functions. The orderwire/absolute time clock unit also stores working channel status words and performs call recognition. The orderwire/absolute time clock unit and the orderwire 2 data channel are strapped for the same operations control table location; however, the orderwire 2 data channel does not access the table until initiated by the orderwire/absolute time clock unit.

The orderwire 2 chain 1 is used solely to transmit orderwire 2 service messages and orderwire 2 device control message chain 2 is used solely to receive orderwire 2 service messages.

At initialization, the orderwire 2 data channel is operating in the data channel mode, that is, the allotter line is set to 0. The orderwire 2 data channel accesses the first device control message, examines the operation code, and raises the allotter line, signaling to the orderwire/absolute time clock unit that the orderwire 2 data channel has serviced a change-to-orderwire device control message. This must be the first device control message serviced at initialization since the orderwire 2 channel address is transferred to the terminal unit at this time. The following defines the device control message operations codes serviced by orderwire 2 channel.

OP CODE	DEFINITION
000	Normal process
001	Change to orderwire mode
010	Change to data channel mode

Completion of the change-to-orderwire device control message results in the orderwire 2 data channel entering a loop where it waits to be initiated by the orderwire/absolute time clock unit. The orderwire/absolute time clock unit then begins servicing the device control messages queues. It services the transmit chain only until the terminal unit signals the orderwire/absolute time clock unit that a bid has been granted on the orderwire 2 channel. The orderwire/absolute time clock unit then raises a line that signals the data channel to service the alternate chain, that is, device control message chain 2, the receive chain.

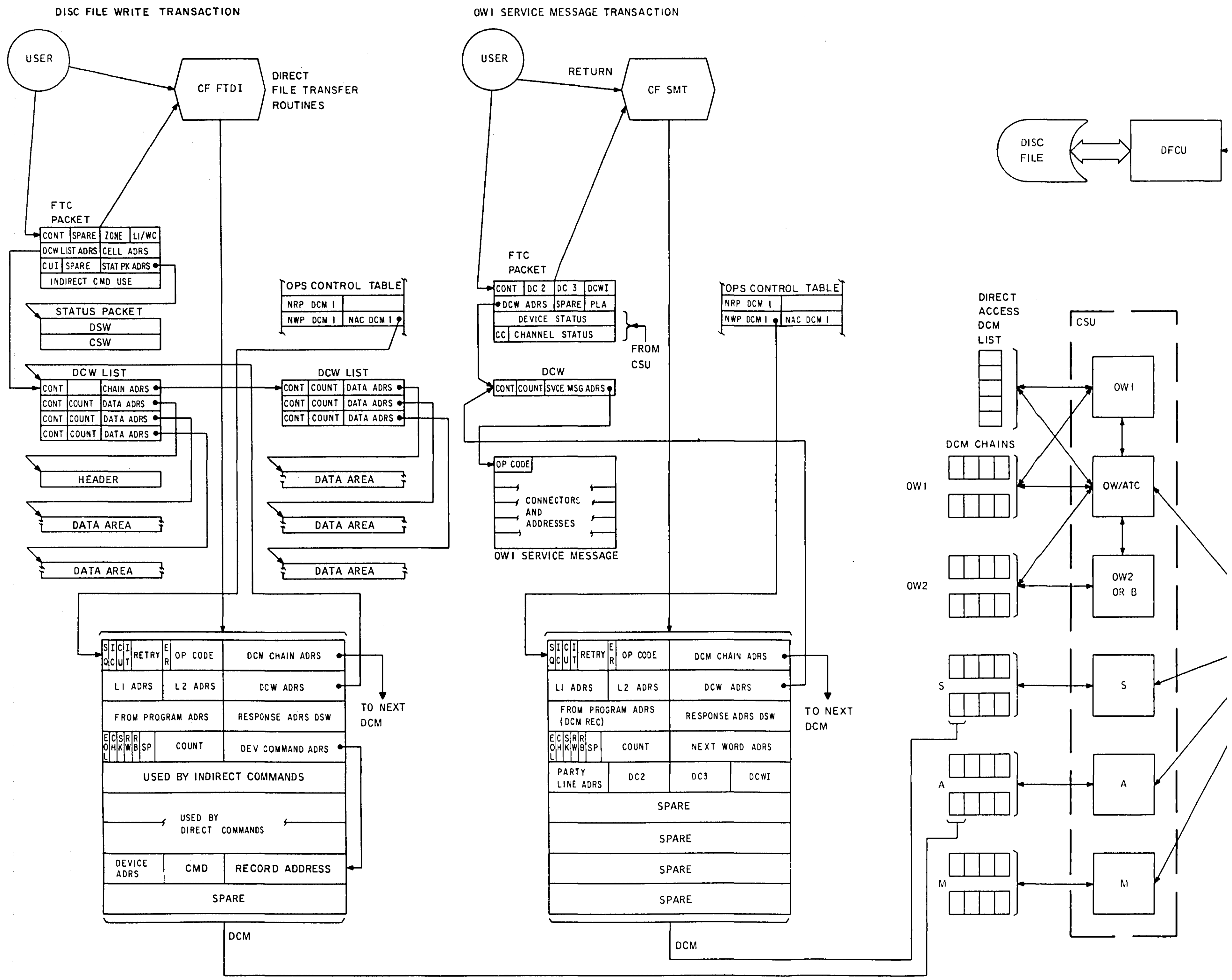
It is important to recognize that the orderwire 2 data channel executes one device control message at a time and then enters a loop to await initiation by the orderwire/absolute time clock unit. Also, upon completion of an incoming service message, the orderwire/absolute time clock unit reverts back to servicing device control message chain 1, the transmit chain.

Figure 5-12 shows two orderwire 2 service message transactions. The receive transaction is an orderwire 2 device-initiated input message issued in response to an operator message which in this case is an instruction to mount a tape reel. The operator mounts the tape reel, presses the LOAD switch to cause the tape to position to the load point, and then presses the READY switch. Pressing the READY switch at this point causes the magnetic tape adapter to instruct its terminal unit to acquire orderwire 2. When orderwire 2 is acquired, a 3-word service message is transmitted to the processor by the device. The third word of the service message, the device status word, is stored in the device control message as shown in figure 5-12. Receipt of this service message causes the device acquisition and control service program to select an available working channel and queue up a device control message as shown in figure 5-12 to transmit the device command word to the device. The magnetic tape adapter decodes the command (assign working channel), transfers the working channel address to its terminal unit, and waits for a call on the assigned loop 1 channel.

The orderwire/absolute time clock unit, after reading the device control message header, initiates the orderwire 2 data channel and discards the channel address supplied by the orderwire 2 data channel. After the orderwire 2 data channel indicates that the device control message has been serviced without error, the orderwire/absolute time clock unit updates the device control message header and stores the chain address in the operation control table. If an error occurs, the orderwire/absolute time clock unit sets the error bit in the device control message and does not update the next-available-cell device control message chain 1.

In the receive mode, the orderwire/absolute time clock unit accesses the operations control table entry at the next-available-cell device control message chain 2. The device control message header is accessed and the service queue and immediate chain bits examined. If the service queue bit is a 1, a busy response of a 1 followed by 33 zeros is transmitted to the calling device and the orderwire/absolute time clock unit returns to the transmit mode. If the service queue bit is a 0, device control message chaining is performed if required, and the orderwire 2 data channel is initiated. The orderwire/absolute time clock unit discards the address supplied by the orderwire 2 data channel and transfers the input data from the device to the orderwire 2 data channel. If no error has occurred, orderwire 2 updates the device control message header (service queue bit is a 1) and the next-available-cell device control message chain 2 (set to device control message chain address). If an error occurs, the error bit in the device control message header is set and the next-available-cell device control message chain 2 is not updated.

When there is no activity on the orderwire 2 channel, the terminal unit transmits a status poll to gather status of all working channels. This status is 13 words long and is stored in bytes X'4B through X'7F of the processor status record. The continuing status request may be interrupted at any time by a bid for orderwire 2. The bid may originate from a device terminal unit or the orderwire 2 data channel associated with the terminal unit transmitting the status poll. After being interrupted by a bid on subsequent use of the orderwire 2 channel, the terminal unit continues requesting status at the point in the 13-word sequence at which it was cut off. The return status words are stored by the orderwire/absolute time clock unit in the processor interface table.


DISC FILE WRITE TRANSACTION

OW1 SERVICE MESSAGE TRANSACTION

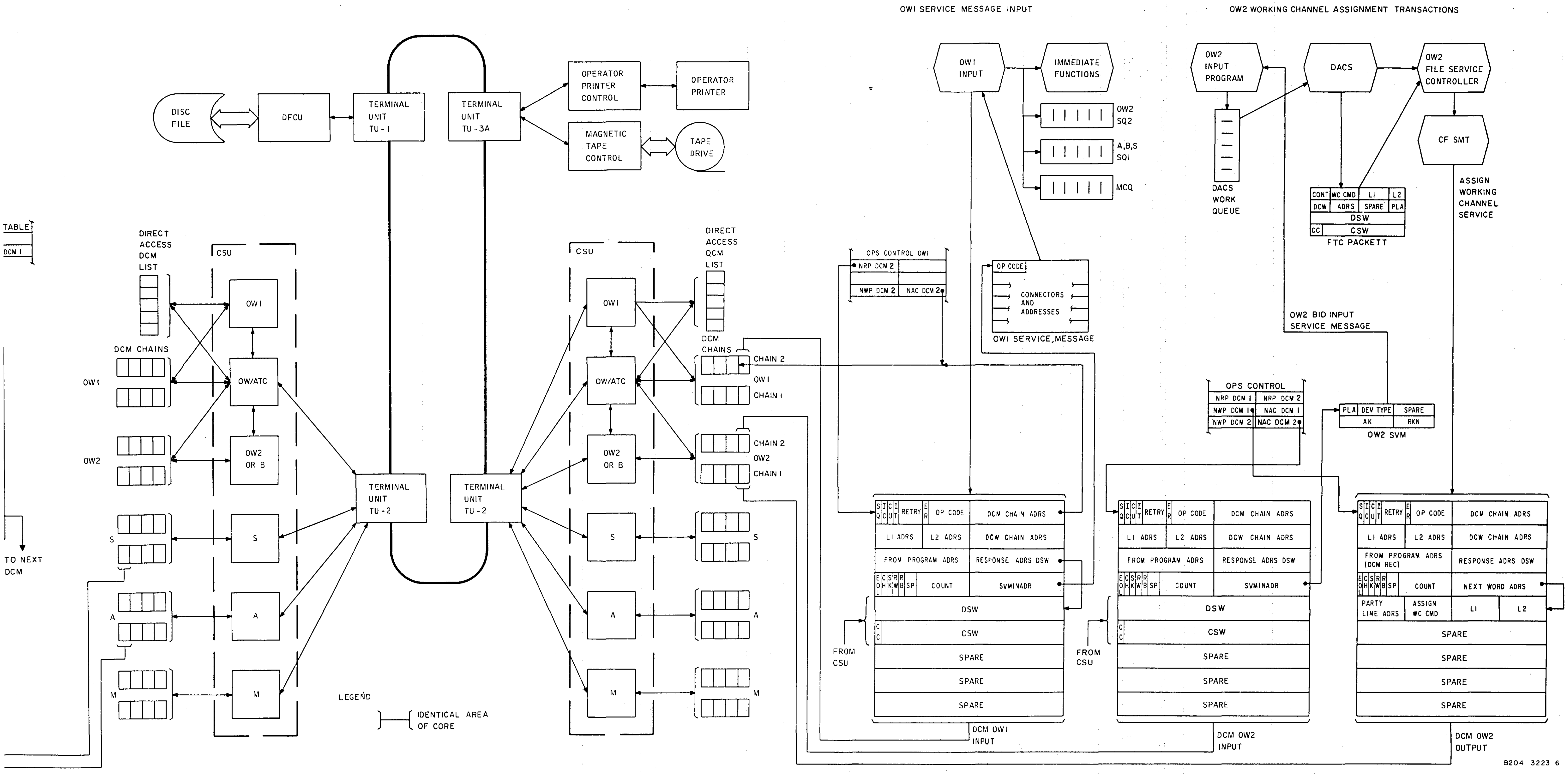


Figure 5-12. Input/Output Operation.

section 6
time division multiplex input/output

6.1 INTRODUCTION

The time division multiplex system provides the connection for a large number of medium and low-speed devices to a single computer within the C-8500 C-System. The basic structure of this system from an input/output point of view is shown in figure 6-3. The system consists of a Time Division Multiplex loop (TDM loop) to provide a serial communications facility to all devices; a Multiplex Service Unit (MSU) to connect the time division multiplex loop to the computer; and a number of hardware device couplers, such as the multiplex device coupler, to connect each device to the time division multiplex loop. It also includes a Multiplex Service Program (MSP) which is executed on the M channel time of the processor. The multiplex service program provides a device independent interface between the multiplex service unit and the multiplex channel subroutines serving the devices as well as an operating environment for these subroutines.

6.2 ENVIRONMENT

The time division multiplex loop operates at 1.2288 Mbps with 256, 36-bit (4 supervisory and 32 data) words constituting a frame; this rate produces 256,4.8 kbps individual communication channels. Strapping options are available allowing 38.4 kbps, 76.8 kbps and 153.6 kbps communication channels in addition to the basic 4.8 kbps rate.

The multiplex service unit provides the link between devices on the time division multiplex loop, the multiplex service program, and data bins in core storage via the Multiplex Status Records (MSR). The multiplex service unit operates in synchronization with the time division multiplex loop, servicing each 4.8 kbps time slot sequentially according to the received operation code from the device and the contents of the multiplex status record. Every time slot (and thus every device) on the time division multiplex loop has one multiplex status record in core with a provision for two per device for full duplex operation. The multiplex service unit has sufficient logic to allow independent movement of data between devices on the time division multiplex loop and data in core; the multiplex service program is used only to initiate and complete the transaction by obtaining the core space, setting up necessary data transfers to or from core storage, and initializing the associated multiplex status record with appropriate control information to allow the operation to take place.

Device calls to the multiplex service program are made under device control by the multiplex service unit via Multiplex Queue 1 (MQ1) and Multiplex Queue 2 (MQ2), each of which is a 256 byte table in protected memory. Multiplex queue 1 is used for multiplex service program calls relative to current transactions; multiplex queue 2 is used for calls relative to output status transactions. When a device requires multiplex service program service, the multiplex service unit places the Time Division Address Counter (TAC) corresponding to the device in multiplex queue 1 or multiplex queue 2 as appropriate. This identifies the multiplex status record corresponding to the device to the multiplex service program. The multiplex status record contains sufficient information for the multiplex service program to perform the required function. The next-read-position pointers for multiplex queue 1 and multiplex queue 2 are maintained in the M channel operations control table by the multiplex service program. The next-write-position pointers in the operations control table are maintained by the multiplex service unit.

The file transfer command (and completion) queues are the means whereby a working channel (or some other part of the multiplex service program) communicates with the time division exchange loop. File opening, reading, writing, and closing operations as well as service message transfer use these queues to gain access to the device control message chain and thus to the loop. File transfer command queue service (part of the multiplex service program) keeps all unprocessed commands in queue and keeps the device control message chain full as long as there is traffic to send or request. File transfer completion queue service monitors file and service message transactions that have been entered in the device control message chain for completion. When the transaction is completed, program control is returned to the subroutine designated in the file transfer command packet.

The multiplex channel queue is used by the multiplex service program to receive service messages from the rest of the system. Normally, these service messages contain output to device requests, but any service message to M channel is placed in this queue. For output traffic, the service message identifies the file to be output and its location. The message header is examined to determine which working channel, if any, is to output the file. After this determination, a multiplex queue tag is built which identifies the file, provides data to construct a return service message, and provides a means of keeping the output request in order. The output queue pointer record of the working channel is adjusted (or created) to reflect this new entry and the service queue bit of the multiplex status record is set to 1.

The multiplex service program is a multi-program operation. Each active device has a working channel assigned to it. This working channel (figure 6-1) consists of the multiplex status record, a channel status record, subroutine storage and working space input/output data bins, a multiplex status record operations bin, output queues, and a library of private cell subroutines. Only those parts of the channel that are needed (or likely to be needed immediately) are resident in core at any one time.

To coordinate the various channel activities, an executive program of the multiplex service program designated as Multiplex Channel Control (MCC) supervises the entire operation. This program works with five queues: multiplex queues 1 and 2, file transfer command and completion queues 1 and 2, and the multiplex channel queue. The multiplex channel control program sequentially monitors these queues for activity. Normally, a new entry in these queues requires that a working channel execute one or more subroutines. When the multiplex channel control determines that a queue entry needs service, the multiplex channel control program turns the channel over to the proper subroutine for execution and return.

The data bins are assigned and managed by the multiplex service program. These bins are available from a pool of bins that are a part of M channel core space.

All time division multiplex loop field operations take place via these data bins. In load operations (output to device), the F operand portion of the multiplex status record specifies the word address of the next word to be sent to the device and the last word address of the bin. In store operations, the F operand specifies the word address in the bin where the next word from the device is to be stored and the address of the first word of the file. The end of the bin is determined by the multiplex service unit from a strapping option that specifies the size of the input data bins for the particular time division address involved.

Data bin sizes of 64 bytes, 128 bytes, 256 bytes, 512 bytes, 1024 bytes, and 2048 bytes are available. However, only two sizes may be used in any one processor. Time division addresses 1 through 127 must use the same size bins. Time division addresses 128 through 255 must use the same size bin but the bin size may be different from that used by time division addresses 1 through 127.

In field operations, it may be desirable to have more than one bin available for use. This is the case when a long file is to be input (or output) to a device. To permit tandem bins without processor intervention, link operations have been provided. In the link mode, when one bin has been filled (or exhausted) and the second bin is ready for use, the multiplex service unit automatically switches from one bin to the other. This switch is accomplished by interchanging the R and F words of the multiplex status record. The R word in this case contains the appropriate data for the next bin including its address and status (e.g. ready or not ready).

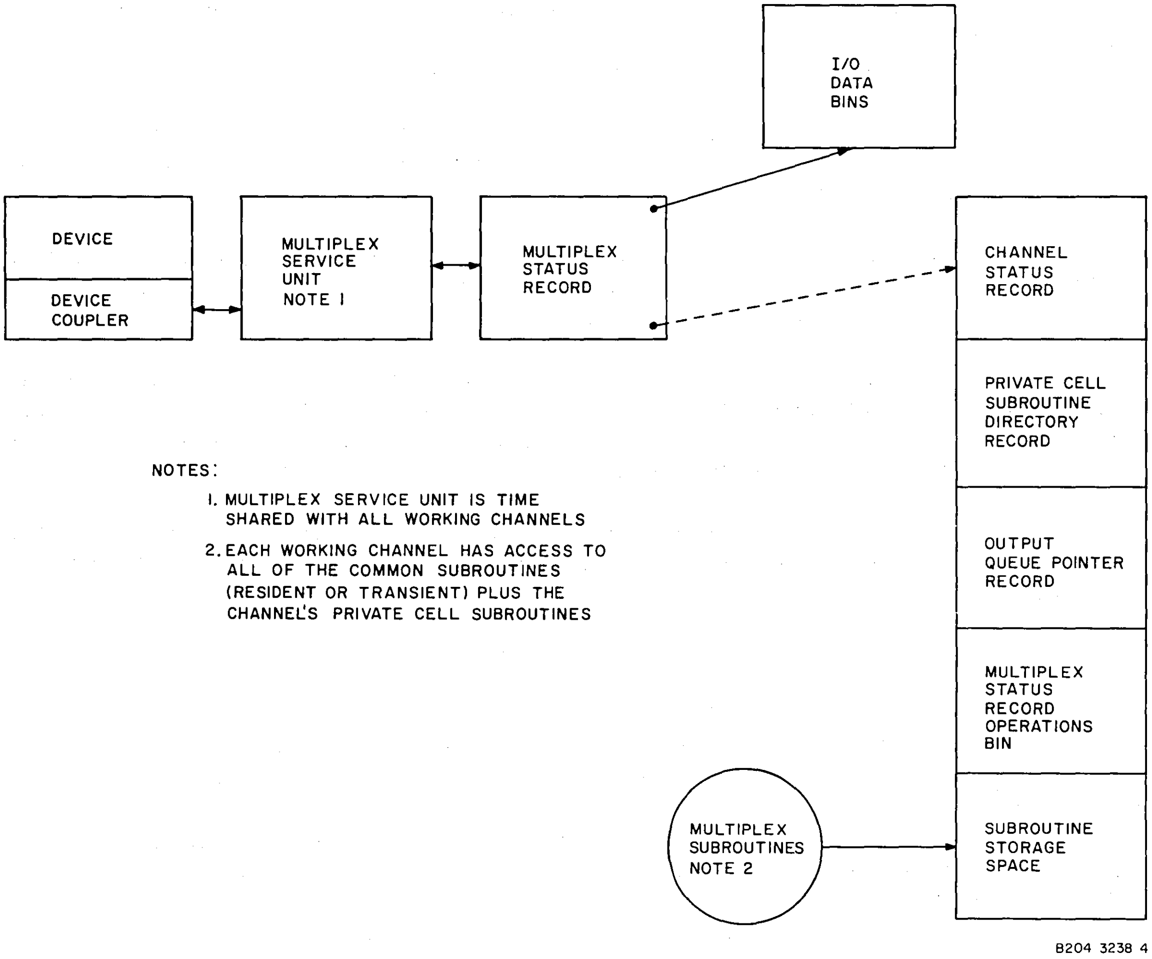


Figure 6-1. Working Channel.

6.3 MULTIPLEX SERVICE UNIT/MULTIPLEX SERVICE PROGRAM COMMUNICATIONS

The multiplex status record for each of the active time division multiplex working channels and the multiplex queues (multiplex queue 1 and multiplex queue 2) provide the interfaces between the multiplex service program and the multiplex service unit. The multiplex service program monitors multiplex queue 1 and multiplex queue 2 for requests for service from the multiplex service unit. When an entry is found in one of these queues, the associated multiplex status record is accessed to determine the required action. The multiplex service program performs the required functions modifying the multiplex status record as necessary. The multiplex service unit, in conjunction with the associated device in the time division multiplex loop, monitors the multiplex status record for completion of the multiplex service program task.

Multiplex queue 1 and multiplex queue 2 each provide 256 entries. The multiplex service unit makes an entry in multiplex queue 1 under the following conditions:

a. A subroutine call by the device (branch to subroutine signal).
b. An End-of-Message (EOM) signal is received from a device.

c. An end-of-bin condition is detected by the multiplex service unit during a block-transfer (load or store) sequence.
d. At the completion of a word-transfer (load or store) instruction.

The multiplex service unit makes an entry in multiplex queue 2 whenever an output request is received from the device and the service queue bit is set in the associated multiplex status record indicating that there is an active entry in the output queue for this device.

Figure 6-2 shows a typical multiplex status record and presents a summary of each field within the multiplex status record. Each multiplex status record contains four words which are referred to as the F word, the D word, the R word, and the P word.

The F word contains the supervisory field (FS) indicating the current controlling element in the system (the multiplex service unit, multiplex service program, or the device), operation code (FO), and an operand. The F word is basically an instruction to one of the time division multiplex devices. The operation code field is set to indicate the operation desired. A total of 11 operation codes is defined as follows:

a. Field Store (FST) - This operation code is used to direct the transfer of a field of data from the device to a bin in core storage. The operand contains the starting address of the bin plus an address counter.

START ADDRESS 10	ADDRESS COUNTER 16

The end of the bin is determined by a strapping option which denotes the size of the bin. The multiplex service unit stores received data from the device in the bin until the bin is full then places an entry in multiplex queue 1 for multiplex service program service.

b. Field Load (FLD) - This operation code is used to direct the transfer of a field of data from core storage to the device on the time division multiplex loop. The operand contains the end address of the bin plus an address counter.

END ADDRESS 10	ADDRESS COUNTER 16

The multiplex service unit successively loads the words from core into the time division multiplex loop time slots of the corresponding device until the bin is emptied then places an entry in multiplex queue 1 for multiplex service program service.

c. Field Store and Link (FSL) - This operation code is nearly identical in function and format to the Field Store operation (FST) except that it is used to direct the transfer of a field of data from the device to core in the link mode. For link mode operations, the R word has the same format and content as the F word except for the bin address. The multiplex service unit recognizes the end of the first data buffer and is able to interchange the F and R words of the multiplex status record and use the new F word to route the data to the new buffer.

d. Field Load and Link (FLL) - This operation code is nearly identical in function and format to the Field Load operation (FLD) except that it is used to direct the transfer of a field of data from core storage to the device in the link mode. The F and R words are used in a similar manner to that described for Field Store and Link (FSL).

e. Store D (STD) - This operation code is used to direct the multiplex service unit to store the word received from the device in the D word of the multiplex status record. It also directs the multiplex service unit to make an entry in multiplex queue 1.

IMPLIED OPERAND 26

FS $_2$	FO $_4$	F OPERAND $_{26}$	F WORD

D REGISTER $_{32}$	D WORD

RS $_2$	RO $_4$	R OPERAND $_{26}$	R WORD

SQ $_1$	A $_2$	DV $_2$	PR $_2$	IC $_1$	LM $_1$	DM $_4$	DT $_4$	CSR $_{16}$	P WORD

FIELD	CONTENT MNEMONIC	SET BY	READ BY	MEANING AND REMARKS
FS	PROGRAM (ALCU) MSU (STATE 1) MSU (STATE 2) DEVICE	MSU/PROGRAM	MSU/PROGRAM	DESIGNATES WHICH UNIT IS CURRENTLY IN CONTROL OF THE CHANNEL
FO/F OPERAND	NOP/ANY	PROGRAM	PROGRAM/MSU	NO OPERATION
	FST/START ADDRESS $_{10}$/ADDRESS $_{16}$	FST AND START ADDRESS BY PROGRAM	FST AND ADDRESS BY PROGRAM/MSU	FIELD STORE (NON LINK) ADDRESS IS LOCATION FOR NEXT WORD TO BE STORED
	FSL/START ADDRESS $_{10}$/ADDRESS $_{16}$	ADDRESS BY PROGRAM/MSU	START ADDRESS BY PROGRAM	FIELD STORE AND LINK ADDRESS IS LOCATION FOR NEXT WORD TO BE STORED. AT THE END OF BIN THE F WORD AND R WORD MAY BE EXCHANGED
	FLD/END ADDRESS $_{10}$/ADDRESS $_{16}$	FIELD AND END ADDRESS BY PROGRAM	PROGRAM/MSU	FIELD LOAD (NON LINK) ADDRESS IS LOCATION FOR NEXT WORD TO BE SENT TO DEVICE
	FLL/END ADDRESS $_{10}$/ADDRESS $_{16}$	ADDRESS BY PROGRAM/MSU		FIELD LOAD AND LINK ADDRESS IS LOCATION FOR NEXT WORD TO BE SENT TO DEVICE. AT THE END ADDRESS, THE F WORD AND R WORD MAY BE EXCHANGED
	STD/ANY	PROGRAM	PROGRAM/MSU	STORE WORD IN D REGISTER
	STX/ANY			STORE WORD IN D REGISTER IF WORD IS NON ZERO
	LDD/ANY			LOAD WORD FROM D REGISTER AND SEND TO DEVICE
	LDX/ANY			LOAD WORD FROM D REGISTER IF INPUT WORD IS NON ZERO AND SEND TO DEVICE
	STT/LOWER LIMIT $_{13}$/UPPER LIMIT $_{13}$			STORE WORD IN REGISTER IF INPUT WORD IS OUTSIDE LIMITS
	LDT/LOWER LIMIT $_{13}$/UPPER LIMIT $_{13}$			LOAD WORD FROM D REGISTER IF INPUT WORD IS OUTSIDE LIMITS AND SEND TO DEVICE
D REGISTER	VARIABLE	PROGRAM/MSU	PROGRAM/MSU	WORD IS 32 BIT DATA FIELD
R WORD	SAME AS F WORD	PROGRAM/MSU	MSU/PROGRAM	WORD IS USED TO REPLACE F WORD IN LINK OPERATION. IN WORD OPERATION (STD, STX, LDD) WORD CONTAINS SUBROUTINE PARAMETERS
SQ	0 OR 1	PROGRAM	MSU	INDICATES IF OUTPUT QUEUE IS FULL
A(BIT 1)	0 OR 1	PROGRAM	PROGRAM	IF SET, ONE OR MORE SEQUENTIAL TIME-OUTS HAVE OCCURRED
A(BIT 2)	0 OR 1			IF SET, THIS BIT DENOTES THAT THE WORKING CHANNEL IS ACTIVE
DV		PROGRAM AT TIME OF DEVICE INITIALIZATION	PROGRAM	00 FULLY CONTROLLED DEVICE 01 SEMI-CONTROLLED (TYPE 1) DEVICE 11 SEMI-CONTROLLED (TYPE 2) DEVICE
PR	QUANTITY OF OUTPUT QUEUES			SPECIFIES NUMBER OF OUTPUT QUEUES (1 TO 4)
IC	0 OR 1			IF DEVICE IS IN LINK MODE BIT SPECIFIES IMPLICIT OR EXPLICIT SUBROUTINE CALLS
LM	0 OR 1			SPECIFIES LINK OR NON LINK MODE DEVICE
DM				SPECIFIES MODE OF OPERATION SIMPLEX, HALF DUPLEX, FULL DUPLEX
DT				SPECIFIES MODE THAT SUBROUTINE CALLS WILL BE HANDLED
CSR	ADDRESS			ADDRESS OF CHANNEL STATUS RECORD

B204 3222 4

Figure 6-2. Multiplex Status Record Format.

f. Load D (LDD) - This operation code is used to direct the multiplex service unit to output the D word of the multiplex status record to the device. It also directs the multiplex service unit to make an entry in multiplex queue 1.

g. Store D if IB ≠ 0 (STX) - This operation code is used to direct the multiplex service unit to store the word received from the device in the D word of the multiplex status record if the word received (IB) is not equal to 0. If the multiplex service unit stores a word in D, it also makes an entry in multiplex queue 1.

h. Load D if IB ≠ 0 (LDX) - This operation code is used to direct the multiplex service unit to output the D word to the device if the word received from the device (IB) is not equal to 0. I f the multiplex service unit outputs the D word, it also makes an entry in multiplex queue 1.

i. Store D if IB is outside limits (STT) - This operation code causes the multiplex service unit to test the least significant 13 bits of the word received from the device (IB). If these bits are less than the lower limit or greater than the upper limit as specified in the F operand, then the multiplex service unit stores the word received from the device in the D word of the multiplex status record and makes an entry in multiplex queue 1.

LOWER LIMIT	UPPER LIMIT
13	13

j. Load D if IB is outside limits (LDT)- This operation code causes the multiplex service unit to test the least significant 13 bits of the word received from the device (IB). If these bits are less than the lower limit or greater than the upper limit as specified in the F operand, then the multiplex service unit outputs the D word of the multiplex status record to the device and makes an entry in multiplex queue 1.

LOWER LIMIT	UPPER LIMIT
13	13

k. No Operation (NOP) - This operation code denotes an idle channel status to the multiplex service unit. The eight least significant bits of the F operand may contain the processor Party Line Address (PLA).

Spare	PLA
18	8

The D word is a 32-bit data register. Word mode operations use the D word as the input/output data field. Field mode operations use the D word as the subroutine calling parameter field. Explicit as well as implicit subroutine calls reside in this field. Accepted branch to subroutine and branch on service queue calling parameters are placed in this field regardless of word or field mode operations.

In field mode operations using link operations, the R word defines the next operation to take place after the F word specified bin has been used up.

In word mode operations, the R word contains the current subroutine calling parameter data.

The P word data is used primarily by the multiplex service program during channel initialization to provide basic program operating parameters for the device being serviced. The various fields are defined as follows:

a. SQ (service queue) bit - This bit denotes that there is output data for the device.

b. A (bit 1) - If set, one or more sequential timeouts has occurred. A (bit 2) - If set, this bit denotes that the working channel is active.

c. DV - This field denotes the class of control the device employs. The values are:

00 Fully controlled

01 Semicontrolled (Message header edit can be performed as message segments complete.)

10 Not used

11 Semicontrolled (Message header edit must be deferred until after message is delivered to the file.)

d. PR field - This field specifies the maximum number of output queues (priority levels) the work channel may have. The value ranges from 1 to 4.

e. IC bit - This bit specifies whether or not the device uses implicit subroutine calls.

f. LM bit - This bit specifies whether or not the device operates in the link mode during field operations.

g. DM field - This field specifies the modes the device is capable of operating in. The modes are as follows:

000 Not used

001 Simplex load (eg, output only)

010 Simplex store (eg, input only)

011 Half duplex (eg, output and input but only one at a time)

100 Means the associated channel is the next greater

101 Means the associated channel is the previous

110 Means the associated channel is this channel number plus 256 (Full Duplex)

111 Not used

h. DT field - This field specifies the decode logic to be used to interpret subroutine call parameters. The field is defined as follows:

0000 Binary decode logic only. The input field must be:

SPARE 14	B 2	SPARE 8	SUBROUTINE IDENTIFICATION 8

B field 00 Common subroutine

10 Private cell subroutine

1000 ASCII decode logic only: The input field must be:

SUBROUTINE TYPE 16	TWO ASCII CHARACTER SUBROUTINE IDENTIFICATION FIELD 16

Subroutine types are limited to:

CM for a common subroutine

PC for a private cell subroutine

Subroutine identification characters are limited to the letters A-F and numbers 0-9.

0100 ASCII or binary decode logic. These use the same field as specified above but the first bit must be 1 for ASCII input or 0 for binary input.

1100 Direct linkage mode: In this mode, a private cell subroutine decodes the calling parameters. Address of this private cell is contained within the working channel. The format is specified by the private cell subroutine.

All other patterns are not used.

CSR field - This field contains the word address of the channel status record for the working channel.

6.4 MULTIPLEX SERVICE UNIT/DEVICE COMMUNICATIONS

As shown in figure 6-3, the device and its adapter communicate with the multiplex service unit using a 4-bit supervisory field and a 32-bit data word. The supervisory bits define the contents of the data word and the desired action (or in some cases the reaction to an action request) to the device and multiplex service unit. See table 6-1.

6.4.1 Device to Multiplex Service Unit Supervisory Signals

a. No Operation (NOP) - This command is used by the device to create a controlled idle situation. For one reason or another, the device does not want the processor to send it any data. The multiplex service unit always responds to a no operation command with a no operation signal.

b. Branch to Subroutine (BSR) - This command is used by the device to input subroutine and channel initialization calls. The 32-bit data field contains the subroutine calling parameters. If the multiplex service unit accepts the command, it stores the input data field in the D word, and sends an end-of-message signal to the device. When the multiplex service unit does not accept the command, it sends a read instruction signal plus the contents of the F word, or it sends a no operation signal to the device.

c. Branch on Service Queue (BSQ) - This command is used by the device to test for entries in the processor output queues for the device. This normally occurs when the multiplex status record F word is in device control, no operation mode. The data field of the loop word containing the branch on service queue contains subroutine calling parameters that start processor preparation of the output file. The multiplex service unit tests the service queue bit of the multiplex status record. If this bit is set to 1, the command is accepted, the input data field is stored in the D word, and an end-of-message signal is sent to the device. If the command is rejected, the multiplex service unit sends a read instruction signal plus the contents of the F word.

d. Load From Field (LFF) - This command is used by the device to request delivery of the multiplex status record F word. In all cases the multiplex service unit responds with a read instruction signal plus the contents of the F word.

e. Execute Field Store (EFS) - This command is used by the device to input data into the processor. The data field of the loop word is the data to be input. If the F word of the multiplex status record is in the device control mode and in any of the store modes, the command may be accepted. If the command is rejected because the F word is not in the correct status, the multiplex service unit returns a read instruction signal plus the contents of the F word. If the command is rejected because the conditional store conditions were not satisfied, the multiplex service unit sends a no operation signal.

Table 6-1. Device to Multiplex Service Unit Supervisory Signals Summary.

MNEMONIC	FUNCTION	WORD CONTENT	MSU RESPONSE
No Operation (NOP)	To create a controlled idle situation.	None	An NOP command.
Branch to Subroutine (BSR)	To input subroutine and channel initialization calls.	Control parameters	If BSR is accepted, the MSU stores input word in the MSR D word and sends an EOM. If BSR is not accepted, the MSU sends a Read Instruction (RDI) plus the contents of the MSR F word, or it sends an NOP.
Branch on Service Queue (BSQ)	To test for entries in output queues.	Control parameters	If BSQ is accepted, the MSU stores the input word in the MSR D word and sends an EOM. If BSQ is rejected, the MSU sends an RDI and the contents of the MSR F word.
Load From Field (LFF)	To request delivery of the MSR's F word	None	In all cases, THE MSU returns an RDI and the contents of the F word.
Execute Field Store (EFS)	To input data to the processor.	Data word	If rejected, the MSU returns an RDI and the contents of the MSR F word. If accepted, the MSU returns an EOM or an NOP.
Execute Field Load (EFL)	To request data from the processor.	Test parameters or none	If rejected, the MSU returns an RDI and the contents of the F word. If accepted, the MSU sends an EOM and the requested word or a Read Data RDD and the requested word.
End-of-Message (EOM)	To signal that this word is the last entry to be placed in the file.	Data	If rejected, the MSU sends an RDI and the MSR F word; if accepted, the MSU sends an EOM.

If the command is accepted and the F word is in a word mode, the multiplex service unit responds with an end-of-message signal. If the command is accepted and the F word is in a field mode, the multiplex service unit responds with a no operation signal except at the end of bin when it responds with an end-of-message signal.

f. Execute Field Load (EFL) - This command is used by the device to request data from the processor. The data field is normally 0 but may contain test parameters. If the F word of the multiplex status record is in the device control mode and any load mode of operation, the command may be accepted. If the command is rejected because the F word is not in the correct status, the multiplex service unit sends a read instruction signal plus the contents of the F word. If the command is rejected because the conditional load conditions were not satisfied, the multiplex service unit sends a no operation signal.

   If the command is accepted and the F word is in a word mode, the multiplex service unit sends an end-of-message signal plus the requested word. If the command is accepted and the F word is in field mode, the multiplex service unit sends a read data signal plus the requested data word except at the end of the bin when the unit sends an end-of-message signal plus the last word of the file.

g. End-of-Message - Device Generated (EOM) - This command is used by the device to signal that this word is the last entry to be placed in the file. The data word contains valid data. This command is permitted only with field mode store operations. All others are rejected with the multiplex service unit sending a read instruction signal plus the F word. If the command is accepted, the multiplex service unit sends an end-of-message signal.

6.4.2 Multiplex Service Unit to Device Supervisory Signals

These signals are basically responses to device commands and have been explained in the previous section. See table 6-2.

a. No Operation - Multiplex Service Unit Generated (NOP) - This response is always followed by a 0 data word. It occurs in response to a device issued no operation, an unsuccessful conditional word mode operation, a successful field store operation, or any nondevice generated command appearing in the input to the multiplex service unit.

Table 6-2. Multiplex Service Unit to Device Supervisory Signals Summary.

MNEMONIC	FUNCTION	WORD CONTENT
No Operation (NOP)	A response to an NOP or any nondevice generated command input to the MSU.	Zero data
Read Instruction (RDI)	A response to an LFF, or when a device command is rejected, or when a BSQ is issued and MSR SQ bit is 0.	MSR F word
Read Data (RDD)	A response to field mode load operations.	Data
End-of-Message (EOM)	(a) To signal a successful word mode store. (b) To signal a successful word mode load or an end of the bin field mode load.	(a) Zero data (b) Data

b. Read Instruction (RDI) - This response is always followed by the contents of the F word. It is used in response to a load from F from the device, and whenever the multiplex service unit rejects an unacceptable command from the device because the F word of the multiplex status record is not in agreement with the command received. It is also sent when the device has issued a branch on service queue and the service queue bit of the multiplex status record is 0.

c. Read Data (RDD) - This response is always followed by valid data in the data field. This response only occurs in field mode load operations.

d. End-of-Message - Multiplex Service Unit Generated (EOM) - This response has many uses. With a 0 in the data field it is used to signal a successful word mode store, acceptance of a branch to subroutine or branch on service queue, and end of the bin in field store operations. With valid data in the data field it denotes a successful word mode load operation or an end of the bin field mode load operation.

6.5 M CHANNEL SUBROUTINES

Every active device calls a subroutine. A subroutine call causes a channel to prepare and/or exchange data with the device, to prepare and/or exchange data with the time division exchange loop, to terminate operations, or to perform a combination of the above.

6.5.1 Subroutine Classification

The multiplex service program contains the framework to call and execute the various subroutines within the M channel on M channel time. The actual subroutines present vary with the system being implemented and may be classified as common or private cell. Common subroutines are those subroutines that are available to all devices on the loop. While these routines are normally general purpose, device-independent routines capable of servicing any device in the system, there is no absolute criterion that makes this mandatory. Also the common subroutines may be resident or transient. Resident common subroutines are permanently resident in M channel core space. Transient common subroutines are resident on disc but, when called, are loaded into core for execution, unless already present.

Private cell subroutines are always transient. Each working channel has the capability to call and execute its private cell subroutines but cannot call those from another channel. Once again, if the subroutine is resident in core, the program can branch directly to it. Otherwise, it must first load the subroutine and then branch to it.

6.5.2 Typical Common Subroutines

The common subroutines available will vary from system to system. Nevertheless, certain subroutines are very likely to be present.

6.5.2.1 Open Write File (OWF)

The open write file subroutine establishes the linkage between a device and a disc file storage unit for the purpose of creating a file on disc. A buffer is obtained and linked to the channel status record. A file linkage block is constructed in the buffer for the creation of a data file. A bin(s) is obtained from the multiplex service program and assigned to the multiplex status record, which is then initialized to accept input from the device.

6.5.2.2 Put Segment (PTS)

The put segment subroutine prepares a data control word to transfer the data bin just filled by the device to disc file storage. The F operand field of the completed multiplex status record instruction word is used to compute the length field of the disc file storage cell. The put segment routine then prepares a file transfer command packet to call the indirect file transfer routines. The file transfer command packet is placed in file transfer command queue 1 in order that completion processing may be performed.

On completion of the transfer to disc file storage, the bin is returned to the multiplex status record, and the multiplex status record instruction word is set to device control. If the device operates in the nonlink mode, there is no continuation of data input while the previous segment is being written to disc file storage. With link mode devices, overlap is accomplished by use of the R word. The R word initially contains the field data pointer to the input segment. Upon completion of the file transfer, the R word contains the address of the new bin to be used.

6.5.2.3 Close Write File (CWF)

The close write file routine examines the buffer to determine if a file has been opened. If a file has been opened, close write file closes the file and calls message header edit, a part of the multiplex service program, which determines to whom and how the file is to be delivered. If a service message is required, a call is issued to the service message transfer routine. The time division multiplex channel is then reset to the inactive state; i.e., auxiliary bins are released to the multiplex service program. The channel status record is retained associated with the multiplex status record in the inactive state.

6.5.2.4 Open Read File (ORF)

The open read file routine obtains the next service message from the highest priority work queue that is active for this channel and establishes the linkage between the device and disc file storage for reading the file. The service message is placed in a buffer to be used on completion of reading the file. Open read file determines if the file being read is multiplexed by examining the operation code of the service message. If multiplexed the input list is obtained from the invoking program control instruction and placed in the buffer input list. The input list contains a list of selection keys of the subfiles to be delivered to the device. After this initialization, the open read file subroutine opens the read file and examines the header of the highest level connector for a subroutine call. If a subroutine is required, the subroutine is called. Reading the file is the responsibility of the called subroutine. If a subroutine has not been called, the open read file subroutine calls upon the get segment subroutine to retrieve the data segments.

6.5.2.5 Get Segment (GTS)

This subroutine reads the next segment of a read file from disc file storage and examines the header to determine if a subroutine call is to be issued; once a subroutine is called, initialization of the multiplex status record and reading of the file are the responsibility of the subroutine. If a subroutine call is not to be issued, the multiplex status record is initialized to send the segment to the device. If the device operates in the link mode, and this is the first segment obtained for the device, a second file transfer command packet is prepared and issued to the indirect file transfer routine. On completion of this request, the second instruction word of the multiplex status record is initialized for the device. If a multiplexed file is being read, the subroutine uses the program instruction selection keys in the buffer input list to select the proper segments of the file. The entire file is sequentially searched for each program instruction selection key in the list before the next program instruction is used.

If the cell size on disc file storage is larger than the bin size for the channel, deblocking is required. The subroutine reads the cell in a logical record mode to obtain each segment. If get segment is called before an open read file, it is not accepted and the multiplex status record is set to no operation and control is returned to the device.

6.5.2.6 Release File (RLF)

This subroutine aborts the current transaction if requested by the device and returns the working channel to the inactive state. If the device is reading an output file when the abort occurs, an error file consisting of the working channel address, the buffer, and any core status, e.g., file connector cells, subroutine storage bins, etc., is created. The abnormal program return orderwire 1 service message is built and sent. The second operand field of the message contains the identifier of the error file. If there is no control program status record for return status, the error file is directed to data collection service and placed into a collection file. The collection file is examined by a recovery program.

If the file being aborted is input from the device, the disc file storage space used in the creation of the file is released.

6.6 TIME DIVISION MULTIPLEX INPUT/OUTPUT

The details of multiplex service program/multiplex service unit and multiplex service unit/device communications were presented in the previous sections along with a discussion of the common subroutines. Table 6-3 brings these concepts together by showing the various affected elements (time division multiplex loop words, multiplex status records, and multiplex queues) before and after multiplex service unit action.

To provide the reader with a better concept of how the various elements of the multiplex operations fit together, an example of an input and output operation is described. In this example, the device calls for the processing system to input a file and return a file to the device. While no particular device is specified, a typical example might be a teletypewriter sending and receiving messages in the system.

In each case, the total operation may be logically divided into three phases: preparation, file or message delivery, and completion. Each of these phases is discussed for both the input operation and the output operation. To further aid in understanding these paragraphs the reader should refer to figure 6-3 and tables 6-1, 6-2, and 6-3.

6.6.1 Typical Input Operation

In this example, the device sends a message to the processor.

6.6.1.1 Preparation

The device starts the operation by sending a Branch to Subroutine signal (BSR) plus an Open Write File signal (OWF) to the multiplex service unit. If the processor is ready to accept a branch to subroutine signal, it stores the open write file command in the D word of the multiplex record, sets the F word of the multiplex status record to program control mode, enters the time division address counter value in the next-write-position pointer of multiplex queue 1, increments the next-write-position pointer by one, and sends an End-of-Message (EOM) signal plus a 0 word to the device. The device now recognizes that its command has been accepted by the processor. It therefore sends an Execute Field Store (EFS) signal plus the first word of the message to the processor.

However, if the processor still has its multiplex status record in program control mode, it rejects the execute field store by sending a Read Instruction signal (RDI) plus the F word. Upon receipt of the F word, the device recognizes that its command was not accepted and continues to repeat this transaction until the word is accepted.

While the device is waiting, the multiplex service program finds the open command, opens the write file, and assigns a data bin. When this is complete, the processor is ready for the file transfer to take place. To signify this, the processor sets the F word of the multiplex status record to device control mode with a Field Store (FST) operation code. Now the file or message delivery action may take place.

6.6.1.2 File or Message Delivery

During this phase, the program is using the field mode. Therefore, the arithmetic logic and control unit is not required except at the end of bin or the end of the transaction. When the device sends an execute field store signal plus a 32-bit word of the message and the multiplex status record is in the device control field store mode, the multiplex service unit stores the data word in the address specified by the F word of the multiplex status record and tests this address for an end-of-bin condition. If the end of the bin has not been reached, it sends a No Operation signal (NOP) to the device and increments the F word address.

The device recognizes the no operation as a store accepted and sends an execute field store signal plus the next word. Thus, the cycle repeats until the file has been delivered. If, at any time, the device is not ready to send a word to the processor, it may send a no operation signal to the processor. When the multiplex service unit receives this signal, it does nothing except return a no operation signal to the device, thus creating a device controlled idle situation.

When the end-of-bin is reached, the processor sends an end-of-message signal to the device instead of sending a no operation signal. The device recognizes this end-of-message signal as an end-of-bin. If the system is operating in the link mode, the multiplex service unit verifies that the R word is ready (e.g., in device control) and, if so, interchanges the R word with the F word. It sets the new R word to multiplex service unit state 2 control and makes an entry in multiplex queue 1. If the R word is not ready or the system is not operating in the link mode, the multiplex service unit sets the F word to the program control mode and makes an entry in multiplex queue 1.

When the device receives an end-of-message signal, it should issue a branch to subroutine signal plus a Put Segment command (PTS), or, if it is at the end of the input message, send a branch to subroutine signal plus a Close Write File command (CWF). The put segment command causes the processor to store the data bin on disc and may cause the processor to assign another bin to the working channel. If the bin assignment is not automatic, the device must issue a branch to subroutine signal plus a Get Segment command (GTS) later.

When the end of the input file is reached, the device sends the last word preceded by an end-of-message signal. The processor responds to this end-of-message signal with an end-of-message signal. It then sets the F word to the program control mode and makes an entry in multiplex queue 1. At this point, the file or message delivery action by the device is considered completed.

6.6.1.3 Completion

The completion action may be initiated by either an implied close command or a device-initiated close write file command. If the implied close command routine is not used, the processor rejects the multiplex queue 1 entry created by the end-of-message signal by setting the F word to device control, no operation mode. In this case, the device must send a branch to subroutine signal plus a close write file command.

Upon receipt of the close write file command, either implied or explicitly sent by the device, the processor sends the last data file to disc and sends a command to the indirect file routines to close the write file. The processor then sets the multiplex status record to device control, no operation mode, to permit new transactions to take place.

The processor, upon file closing, examines the message header, determines the routing of the message, and issues a service message to cause the system to take the action directed by the input message.

6.6.2 Output Operations

While the processor is taking action on the input message, the device may either issue new inputs to the system in the manner previously discussed or make inquiries to the processor for output actions. This inquiry is accomplished by sending a Branch on Service Queue signal (BSQ) plus Open Read File command (ORF).

When the processing system has an output for the multiplex loop, it sends a service message to the M channel queue. The multiplex service program reads this service message and the header of the message to be output to determine what channel is to perform the output function. After this determination is made, a multiplex queue tag is built to identify the message and deliver it to the specified channel. The output queue pointer record of the channel is modified to show this new entry, and the service queue bit of the multiplex status record of the channel is set to 1.

6.6.2.1 Preparation

When the device is ready to receive an output from the processor, it sends a branch on service queue signal plus an open read file command. If the multiplex status record is in device control, no operation mode, the multiplex service unit examines the service queue bit. If the bit is set to 0, the channel has no output for the device. Therefore, the multiplex service unit sends a read instruction signal plus the F word to the device indicating the branch on service queue signal was rejected.

If the service queue bit was set to 1, the channel has a file to be output. In this case, the open read file command is entered in the D word, the F word is set to program control mode, an entry is made in multiplex queue 2 and an end-of-message signal is sent to the device.

The device recognizes the end-of-message signal as an accepted command and may now send an Execute Field Load signal (EFL) to the processor requesting the first word of the file. If the processor is still in program control mode, the command is rejected and the device receives a read instruction signal plus the F word.

While the device is waiting, the processor finds the file to be output and loads the first segment of the message into a data bin. After this file transaction is complete, the F word is set to device control, field load mode (and link in some cases). Now the file delivery is ready to take place.

6.6.2.2 File or Message Delivery

When the device sends an execute field load signal to the processor, the multiplex service unit checks if the end of the bin has been reached or not. If not, the multiplex service unit sends a Read Data signal (RDD) plus the data word in the file at the address indicated in the F word. The multiplex service unit also increments the F word address.

If the end of the bin has been reached, the multiplex service unit sends an end-of-message signal plus the last word of the file. If link operations are taking place and the R word is not in device control mode, or nonlink operations are taking place, the F word is set to program control mode and the multiplex service unit makes an entry in multiplex queue 1. If in link operation mode and the R word is in device control mode, the F word and R words are interchanged and the new R word is set to multiplex service unit state 2 control.

Upon receipt of the end-of-message signal the device recognizes that the data segment is complete. In load and link operations, a request for the next segment may be deferred, but in nonlink operations the request for the next segment must be made immediately. This request, which is sent by the device, consists of a branch to subroutine signal plus a get segment command. The multiplex service unit, upon receipt of this command, enters the get segment command to the D register. It sets the F word (or R word in link operations) to program control mode, makes an entry in multiplex queue 1, and sends an end-of-message signal to the device. Receipt of this signal tells the device that the processor has received the branch to subroutine signal. If the F word is still in device control mode, the device can continue receiving data by sending an execute field load signal and receiving a read data signal plus data. If the F word is not in device control mode, the response to an execute field load signal from the device is a read instruction signal plus the F word.

Upon servicing multiplex queue 1, the processor detects the get segment command and requests the next data cell from the file. If the file has been exhausted, file closing action commences in the nonlink mode. In the link mode, file closing action is readied but not commenced until the current data bin is exhausted.

Once again the device can adjust the loop to its speed by using no operation signals between requests for data.

6.6.2.3 Completion

When all of the data has been sent, file closing action is initiated by the processor. In this case it sends a return service message to the calling program indicating that the transaction has been completed. It releases the disc space that held the multiplex queue tag. It adjusts the output queue pointer record and, if necessary, resets the service queue bit to 0.

When this is completed, it resets the F word of the multiplex status record to device control, no operation mode. The system is now ready to perform the next device-requested operation.

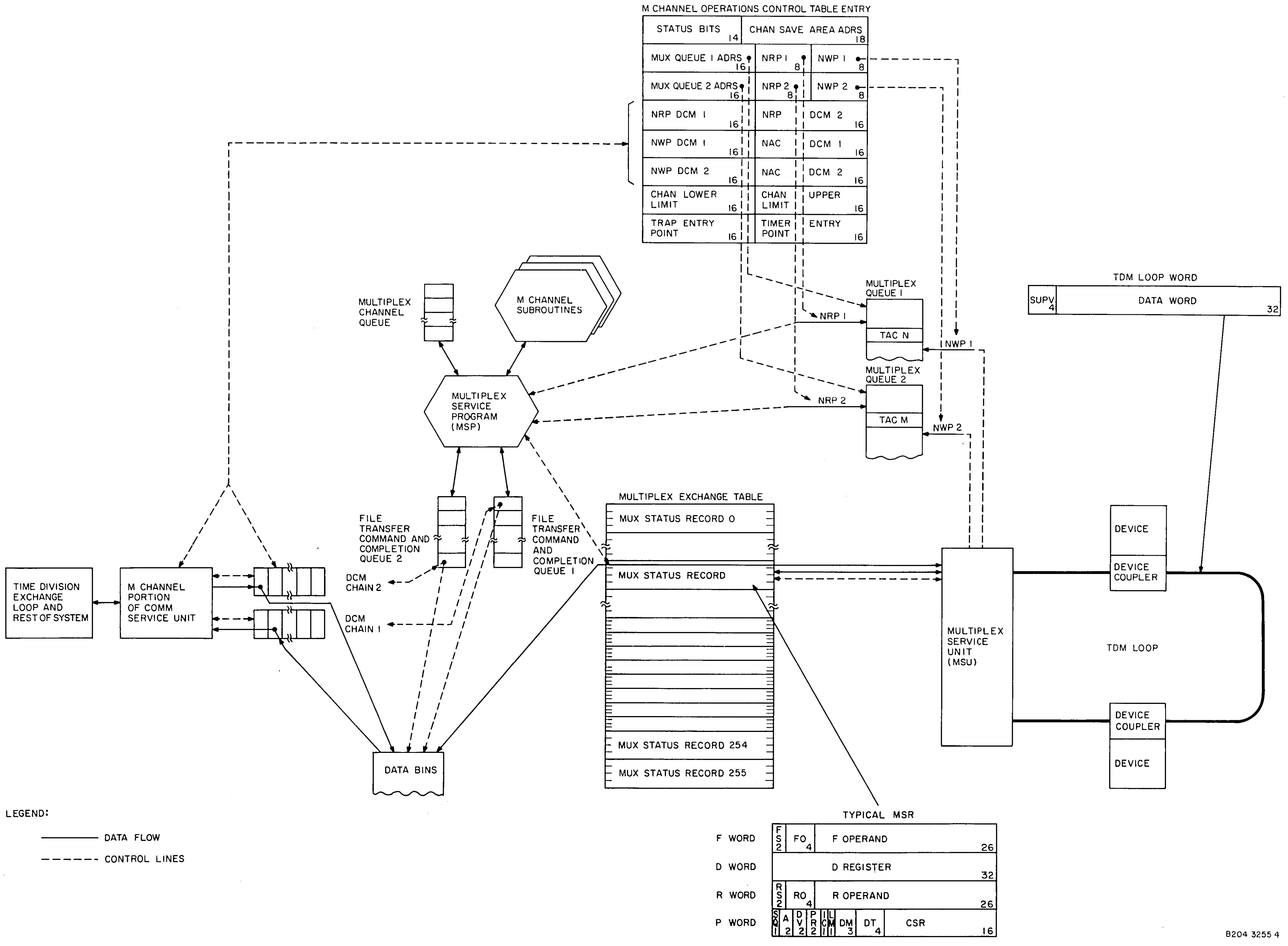


Figure 6-3. TDM Hardware/Software Interface.

INPUT	LOOP WORD		SUPV	EFS	EFS	EFS	EFS	EFS	EFS	EFS	EFS	EOM	EOM	EOM	EOM
			OPERAND	DEVICE DATA	DEVICE DATA (NON ZERO)	0	DEVICE DATA > UPPER R LIMIT OR < LOWER R LIMIT	DEVICE DATA WITHIN R LIMITS	DEVICE DATA	DEVICE DATA	DEVICE DATA	DEVICE DATA	DEVICE DATA	DEVICE DATA	DEVICE
	MSR	F WORD	FS	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV
			OP CODE	FSL	FSL	FSL	FSL	FSL	FSL	FSL	ANY EXCEPT STX, STT, STD, FSL OR FST	FST	FSL	FSL	FSL
			OPERAND	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ANY	ADDRESS	ADDRESS	ADDRESS	ADDRE
		R WORD	RS	DEV	DEV	DEV	DEV	DEV	DEV	DEV	ANY	ANY	ALU OR MSU	DEV	DEV
			OP CODE	LDD	LDX	LDX	LDT	LDT	FLD	FLL	ANY	ANY	ANY	FSL	ANY E
			OPERAND	ANY	ANY	ANY	UPPER AND LOWER LIMITS	UPPER AND LOWER LIMITS	ADDRESS AND END ADDRESS	ADDRESS AND END ADDRESS	ANY	ANY	ANY	ADDRESS	ANY
		D	WORD	PROGRAM DATA	PROGRAM DATA	PROGRAM DATA	PROGRAM DATA	PROGRAM DATA	ANY	ANY	ANY	ANY	ANY	ANY	ANY
		SQ	BIT	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY
	MCS		WORD	ANY	ANY	ANY	ANY	ANY	ANY AT F ADDRESS PROGRAM DATA AT R ADDRESS	ANY AT F ADDRESS PROGRAM DATA AT R ADDRESS	ANY	ANY	ANY	ANY	ANY
OUTPUT	MSR	F WORD	FS	MSP	MSP	DEV	MSP	DEV	DEV	DEV	DEV	MSP	MSP	DEV	MSP
			OP CODE	LDD	LDX	LDX	LDT	LDT	FLD	FLL	NC	NC	NC	FSL	FSL
			OPERAND	ANY	ANY	ANY	UPPER AND LOWER LIMITS	UPPER AND LOWER LIMITS	OLD R WORD ADRS WITH CURRENT ADRS = TO R ADDRESS+I	OLD R WORD ADRS WITH CURRENT ADRS = TO R ADDRESS+I	NC	NC	NC	OLD R WORD ADDRESS	OLD R ADDRES
		R WORD	RS	MSU (STATE 2)	MSU (STATE 2)	MSU (STATE 2)	MSU (STATE 2)	MSU (STATE 2)	MSU (STATE 2)	MSU (STATE 2)	NC	NC	NC	MSP	MSP
			OP CODE	FSL	FSL	FSL	FSL	FSL	FSL	FSL	NC	NC	NC	FSL	FSL
			OPERAND	OLD F WORD ADDRESS	OLD F WORD ADDRESS	OLD F WORD ADDRESS	OLD F WORD ADDRESS	OLD F WORD ADDRESS	OLD F WORD ADDRESS	OLD F WORD ADDRESS	NC	NC	NC	OLD F WORD ADDRESS	OLD F ADDRES
		D	WORD	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
	MCS		WORD	INPUT DATA AT OLD F WORD ADDRESS	INPUT DATA AT OLD F WORD ADDRESS	INPUT DATA AT OLD F WORD ADDRESS	INPUT DATA AT OLD F WORD ADDRESS	INPUT DATA AT OLD F WORD ADDRESS	DEVICE DATA AT OLD F ADDRESS NC AT OLD R ADDRESS	DEVICE DATA AT OLD F ADDRESS NC AT OLD R ADDRESS	NC	DEVICE DATA AT F WORD ADDRESS	DEVICE DATA AT F WORD ADDRESS	DEVICE DATA AT OLD F WORD ADDRESS	DEVICE OLD F ADDRE
	MULTIPLEX QUEUE I		CONTENTS	TAC_N AT NWP	TAC_N AT NWP	NC	TAC_N AT NWP	NC	NC	NC	NC	TAC_N AT NWP	TAC_N AT NWP	TAC_N AT NWP	TAC_N
			NWP	NWP+I	NWP+I	NC	NWP+I	NC	NC	NC	NC	NWP+I	NWP+I	NWP+I	NWP+I
	MULTIPLEX QUEUE 2		CONTENTS	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
			NWP	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
	LOOP WORD		SUPV	EOM	EOM	NOP	EOM	NOP	RDD	RDD	RDI	EOM	EOM	EOM	EOM
			OPERAND	WORD FROM D	WORD FROM D	0	WORD FROM D	0	DATA FROM OLD R ADDRESS	DATA FROM OLD R ADDRESS	F WORD	0	0	0	0

	EFS	EFS	EFS	EFS	EFS	EFS	EFS	EOM	EOM	EOM	EOM	EFL	EFL	EFL	EFL	EFL	EFL	EFL	EFL
	DEVICE DATA (NON ZERO)	0	DEVICE DATA > UPPER R LIMIT OR < LOWER R LIMIT	DEVICE DATA WITHIN R LIMITS	DEVICE DATA	DEVICE DATA	DEVICE DATA	DEVICE DATA	DEVICE DATA	DEVICE DATA	DEVICE DATA	0	ANY EXCEPT 0	WORD > UPPER LIMIT OR < LOWER LIMIT	WORD = TO OR WITHIN LIMITS	ANY	ANY PROBABLY 0	ANY PROBABLY 0	ANY PR
	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV
	FSL	FSL	FSL	FSL	FSL	FSL	ANY EXCEPT STX, STT, STD, FSL OR FST	FST	FSL	FSL	FSL	LDX	LDX	LDT	LDT	LDD	FSL OR FLD	FLD	FLL
	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ANY	ADDRESS	ADDRESS	ADDRESS	ADDRESS	ANY	ANY	UPPER AND LOWER LIMITS	UPPER AND LOWER LIMITS	ANY	ADDRESS AND END ADDRESS ADRS DIFFERENT	ADDRESS AND END ADDRESS ADRS IDENTICAL	ADDRES ADDRES ADRS ID
	DEV	DEV	DEV	DEV	DEV	DEV	ANY	ANY	ALU OR MSU	DEV	DEV	ANY	ANY	ANY	ANY	ANY	ANY	ANY	MSP OR
	LDX	LDX	LDT	LDT	FLD	FLL	ANY	ANY	ANY	FSL	ANY EXCEPT FSL	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY
	ANY	ANY	UPPER AND LOWER LIMITS	UPPER AND LOWER LIMITS	ADDRESS AND END ADDRESS	ADDRESS AND END ADDRESS	ANY	ANY	ANY	ADDRESS	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY
	PROGRAM DATA	PROGRAM DATA	PROGRAM DATA	PROGRAM DATA	ANY	ANY	ANY	ANY	ANY	ANY	ANY	PROGRAM DATA	PROGRAM DATA	PROGRAM DATA	PROGRAM DATA	PROGRAM DATA	ANY	ANY	ANY
	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY
	ANY	ANY	ANY	ANY	ANY AT F ADDRESS PROGRAM DATA AT R ADDRESS	ANY AT F ADDRESS PROGRAM DATA AT R ADDRESS	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	PROGRAM DATA AT ADDRESS	PROGRAM DATA AT ADDRESS	PROGRA AT F AI
	MSP	DEV	MSP	DEV	DEV	DEV	DEV	MSP	MSP	DEV	MSP	NC	MSP	MSP	NC	MSP	NC	*MSP	*MSP
	LDX	LDX	LDT	LDT	FLD	FLL	NC	NC	NC	FSL	FSL	NC	NC	NC	NC	NC	NC	NC	NC
	ANY	ANY	UPPER AND LOWER LIMITS	UPPER AND LOWER LIMITS	OLD R WORD ADRS WITH CURRENT ADRS = TO R ADDRESS+1	OLD R WORD ADRS WITH CURRENT ADRS = TO R ADDRESS+1	NC	NC	NC	OLD R WORD ADDRESS	OLD R WORD ADDRESSES	NC	NC	NC	NC	NC	ADDRESS = OLD ADDRESS+1	NC	NC
	MSU (STATE 2)	MSU (STATE 2)	MSU (STATE 2)	MSU (STATE 2)	MSU (STATE 2)	MSU (STATE 2)	NC	NC	NC	MSP	MSP	NC	NC	NC	NC	NC	NC	NC	NC
	FSL	FSL	FSL	FSL	FSL	FSL	NC	NC	NC	FSL	FSL	NC	NC	NC	NC	NC	NC	NC	NC
	OLD F WORD ADDRESS	OLD F WORD ADDRESS	OLD F WORD ADDRESS	OLD F WORD ADDRESS	OLD F WORD ADDRESS	OLD F WORD ADDRESS	NC	NC	NC	OLD F WORD ADDRESS	OLD F WORD ADDRESS	NC	NC	NC	NC	NC	NC	NC	NC
	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
OLD SS	INPUT DATA AT OLD F WORD ADDRESS	INPUT DATA AT OLD F WORD ADDRESS	INPUT DATA AT OLD F WORD ADDRESS	INPUT DATA AT OLD F WORD ADDRESS	DEVICE DATA AT OLD F ADDRESS NC AT OLD R ADDRESS	DEVICE DATA AT OLD F ADDRESS NC AT OLD R ADDRESS	NC	DEVICE DATA AT F WORD ADDRESS	DEVICE DATA AT F WORD ADDRESS	DEVICE DATA AT OLD F WORD ADDRESS	DEVICE DATA AT OLD F WORD ADDRESS	NC	NC	NC	NC	NC	NC	NC	NC
	TAC_N AT NWP	NC	TAC_N AT NWP	NC	NC	NC	NC	TAC_N AT NWP	TAC_N AT NWP	TAC_N AT NWP	TAC_N AT NWP	NC	TAC_N AT NWP	TAC_N AT NWP	NC	TAC_N AT NWP	NC	TAC_N AT NWP	TAC_N A
	NWP+1	NC	NWP+1	NC	NC	NC	NC	NWP+1	NWP+1	NWP+1	NWP+1	NC	NWP+1	NWP+1	NC	NWP+1	NC	NWP+1	NWP+1
	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
	EOM	NOP	EOM	NOP	RDD	RDD	RDI	EOM	EOM	EOM	EOM	NOP	EOM	EOM	NOP	EOM	RDD	EOM	EOM
	WORD FROM D	0	WORD FROM D	0	DATA FROM OLD R ADDRESS	DATA FROM OLD R ADDRESS	F WORD	0	0	0	0	0	D WORD	D WORD	0	D WORD	DATA FROM OLD F ADDRESS	DATA FROM F ADDRESS	DATA F F ADDR

Table 6-3. MSU Op

BSQ	BSQ	BSR	BSR	BSR	BSR	BSR	BSR	BSR	LFF	EFS	EFS	EFS	EFS	EFS	EFS	EFS	EFS	EFS	EFS
PARAMETER	PARAMETER	PARAMETER	PARAMETER	PARAMETER	PARAMETER	PARAMETER	PARAMETER	PARAMETER	ANY PROBABLY 0	ANY EXCEPT 0	0	VALUE > UPPER LIMIT OR < LOWER LIMIT VALUE	EQUAL TO OR WITHIN LIMITS	DEVICE DATA	DEVICE DATA	DEVICE DATA	DEVICE DATA	DEVICE DATA	DEVICE DATA
DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV
NOP	NOP	ANY EXCEPT FLL, FSL	FLL OR FSL	FLL	FLL	FSL	FSL	FSL	ANY	STX	STX	STT	STT	STD	FSL OR FST	FST	FSL	FSL	FSL
ANY	ANY	ANY	ADDRESS	ADDRESS	ADDRESS	ADDRESS	ADDRESS	ADDRESS	ANY	ANY	ANY	UPPER AND LOWER LIMITS	UPPER AND LOWER LIMITS	ANY	ADDRESS ADDRESS ≠ BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY
ANY	ANY	ANY	MSP OR MSU (STATE I)	DEV	MSU (STATE 2)	DEV	DEV	MSU (STATE 2)	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	MSP OR MSU	DEV	DEV
ANY	ANY	ANY	ANY	ANY	ANY	ANY EXCEPT FSL	FSL	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	FSL	STX OR STT OR STD OR FST
ANY	ANY	ANY	ANY	ANY	ANY	ANY	ADDRESS	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	NEW ADDRESS	ANY OR LIMIT OR ANY OR ADDRESS
ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY
0	I	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY
ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY
DEV	MSP	MSP	NC	MSP	NC	MSP	DEV	NC	NC	MSP	NC	MSP	NC	MSP	NC	MSP	MSP	DEV	DEV
NC	NC	NC	NC	NC	NC	R OP CODE	FSL	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	FSL	OLD R WORD OP CODE
NC	NC	NC	NC	NC	NC	R OPERAND	R ADDRESS	NC	NC	NC	NC	NC	NC	NC	ADDRESS+I	NC	NC	NEW ADDRESS	OLD R WORD OPERAND
NC	NC	NC	NC	NC	MSP	MSP	MSP	MSP	NC	NC	NC	NC	NC	NC	NC	NC	NC	MSU (STATE 2)	MSU (STATE 2)
NC	NC	NC	NC	NC	NC	FSL	FSL	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	FSL	FSL
NC	NC	NC	NC	NC	NC	F ADDRESS	F ADDRESS	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	OLD F WORD ADDRESS	OLD F WORD ADDRESS
NC	INPUT PARAMETER	INPUT PARAMETER	NC	INPUT PARAMETER	INPUT PARAMETER	INPUT PARAMETER	INPUT PARAMETER	INPUT PARAMETER	NC	INPUT LOOP WORD	NC	INPUT LOOP WORD	NC	INPUT LOOP WORD	NC	NC	NC	NC	NC
NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	INPUT DATA AT ADDRESS	INPUT DATA AT ADDRESS	INPUT DATA AT ADDRESS	INPUT DATA AT OLD F WORD ADDRESS	INPUT DATA AT OLD F WORD ADDRESS
NC	NC	TAC_N AT NWP	NC	TAC_N AT NWP	TAC_N AT NWP	TAC_N AT NWP	TAC_N AT NWP	TAC_N AT NWP	NC	TAC_N AT NWP	NC	TAC_N AT NWP	NC	TAC_N AT NWP	NC	TAC_N AT NWP	TAC_N AT NWP	NC	NC
NC	NC	NWP+I	NC	NWP+I	NWP+I	NWP+I	NWP+I	NWP+I	NC	NWP+I	NC	NWP+I	NC	NWP+I	NC	NWP+I	NWP+I	NC	NC
NC	TAC_N AT NWP	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
NC	NWP+I	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
RDI	EOM	EOM	NOP	EOM	EOM	EOM	EOM	EOM	RDI	EOM	NOP	EOM	NOP	EOM	NOP	EOM	EOM	EOM	RDI
F WORD	0	0	0	0	0	0	0	0	F WORD	0	0	0	0	0	0	0	0	0	NEW F WORD

Table 6-3. MSU Operation.

	BSR	LFF	EFS	EFS	EFS	EFS	EFS	EFS	EFS	EFS	EFS	EFS
	PARAMETER	ANY PROBABLY 0	ANY EXCEPT 0	0	VALUE > UPPER LIMIT OR < LOWER LIMIT VALUE	EQUAL TO OR WITHIN LIMITS	DEVICE DATA	DEVICE DATA	DEVICL DATA	DEVICE DATA	DEVICE DATA	DEVICE DATA
	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV	DEV
	FSL	ANY	STX	STX	STT	STT	STD	FSL OR FST	FST	FSL	FSL	FSL
	ADDRESS	ANY	ANY	ANY	UPPER AND LOWER LIMITS	UPPER AND LOWER LIMITS	ANY	ADDRESS ADDRESS ≠ BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY	ADDRESS ADDRESS = BIN BOUNDARY
	MSU (STATE 2)	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	MSP OR MSU	DEV	DEV
	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	FSL	STX OR STT OR STD OR FST
	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	NEW ADDRESS	ANY OR LIMIT OR ANY OR ADDRESS
	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY
	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY
	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY	ANY
	NC	NC	MSP	NC	MSP	NC	MSP	NC	MSP	MSP	DEV	DEV
	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	FSL	OLD R WORD OP CODE
	NC	NC	NC	NC	NC	NC	NC	ADDRESS+1	NC	NC	NEW ADDRESS	OLD R WORD OPERAND
	MSP	NC	NC	NC	NC	NC	NC	NC	NC	NC	MSU (STATE 2)	MSU (STATE 2)
	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	FSL	FSL
	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	OLD F WORD ADDRESS	OLD F WORD ADDRESS
METER	INPUT PARAMETER	NC	INPUT LOOP WORD	NC	INPUT LOOP WORD	NC	INPUT LOOP WORD	NC	NC	NC	NC	NC
	NC	NC	NC	NC	NC	NC	NC	INPUT DATA AT ADDRESS	INPUT DATA AT ADDRESS	INPUT DATA AT ADDRESS	INPUT DATA AT OLD F WORD ADDRESS	INPUT DATA AT OLD F WORD ADDRESS
NP	TAC_N AT NWP	NC	TAC_N AT NWP	NC	TAC_N AT NWP	NC	TAC_N AT NWP	NC	TAC_N AT NWP	TAC_N AT NWP	NC	NC
	NWP+1	NC	NWP+1	NC	NWP+1	NC	NWP+1	NC	NWP+1	NWP+1	NC	NC
	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC
	EOM	RDI	EOM	NOP	EOM	NOP	EOM	NOP	EOM	EOM	EOM	RDI
	0	F WORD	0	0	0	0	0	0	0	0	0	NEW F WORD

NOTE:

1. *TWO OPERATIONS ARE REQUIRED TO REACH THIS CONDITION, BUT THE DEVICE WILL EFFECTIVELY SEE IT AS ONE
2. MNEMONIC DEFINITIONS:
   MSP = MULTIPLEX SERVICE PROGRAM CONTROL
   DEV = DEVICE CONTROL
   MSU = MULTIPLEX SERVICE UNIT CONTROL
   TAC_N = TIME ADDRESS COUNTER VALUE FOR MSR_N
   MSR = MULTIPLEX STATUS RECORD
   MCS = MAIN CORE STORAGE
   NWP = NEXT WRITE POSITION
   NC = NO CHANGE
   ADRS = ADDRESSES

B204 3219 6A

section 7

file structures

7.1 INTRODUCTION

Mass storage in the C-System is provided by disc file units and magnetic tape units. Disc files provide fast direct access storage. Magnetic tape units provide sequential storage for large files. In general, disc file storage is used as temporary storage whereas tape file storage is used for permanent files.

In the C-System, many user programs have common access to data files required during execution. During the execution of the program control instructions of a control program, files are created and stored on disc or tape to be used by subsequent programs. The address locations of these files are maintained in the control programs to enable the required files to be located and accessed. Having multiple users of files requires that a commonphysical file structure be implemented. In the C-System, this structure is provided through the indirect file transfer routines which automatically read, write, search, and structure the information files for the user programs. The files may be located on either tape or disc units, and the capability exists in the system to derive tape files from disc files and vice versa. Both the disc and tape file structures are discussed in this section.

7.2 DISC FILE STRUCTURE

The basic file structure implemented in the C-System through the indirect file transfer routines is shown in figure 7-3 and is a tree structure. The file consists of a file identifier, one or more levels of connector cells, and data cells containing the user data. The data cells are the lowest level elements in the structure. The lowest level connector cells are referred to as Low Level Connectors (LLC). The highest level connector cell is referred to as the High Level Connector (HLC) and is pointed to by the file identifier. All other connector cells are referred to as Intermediate Level Connector (ILC). Each connector cell points to lower level cells with the lowest level cells in the structure pointing to the data cells.

Disc file storage in the C-System is partitioned into zones. Space within a zone is partitioned into cells with all cells in a given zone being the same size. Cell sizes of 128, 256, 512, or 2048 bytes are available. Zone sizes are arbitrary and a given zone may cover only a portion of a disc file unit. The zone size is defined for a given environment. A given file is always maintained within the boundaries of one zone and a zone never exceeds the storage capacity of a physical device. There may be many files in a zone. A file may contain any number of connector cell levels, as long as the zone capacity is not exceeded. It is also possible to place data in connector cells if these cells are properly formatted by the user. Using this technique, a single cell file may be created.

When a file is opened, the low order 16 bits of the system absolute time clock are accessed to be used as part of a file identifier. The first 16 bits of the first word of every cell in the file, referred to as the AK field, contain this value. In addition, a 16-bit counter is maintained for every zone. When a cell within the zone is allocated, the value of this counter is stored in the last 16 bits of the first word of the cell, referred to as the K2 field, and the counter is incremented by one. Therefore every cell in a zone has a unique AK, K2 identifier and every cell in a given file has the same AK identifier.

7.2.1 File Identifier

As shown in figure 7-3, a file is addressed by a 2-word file identifier which consists of the AK, K2 protection key for the highest level cell in the file plus the physical file address of this cell which allows retrieval of the file. The Z field of the file identifier contains 8 bits and identifies the disc zone in which the file resides. The 8-bit L1 field is the 8-bit time division exchange address of the disc on which the file resides. The 16-bit CA field is the disc cell address for the highest level cell.

7.2.2 Cell Headers

In addition to the AK, K2 identifier in every cell header, a second word consisting of the A, IL, L, P1 and P2 fields is used for both connector and data cells.

The A field is 1 bit in length and contains a 0 if the cell is a low level connector cell. Otherwise, the field contains a 1.

The 3-bit IL field contains the item length specification for connector cell items. The field is not used for data cells. The connector cell item length is encoded as follows:

IL Field Value	Item Length (Bytes)
0	2
1	4
2	8
3	16
4	32
5	64
6	128
7	256

Each item is used to identify a single lower level cell.

The L field is 12 bits long and indicates the number of bytes of useful data. The count is taken from the start of the cell and includes the header space. For connector cells, the count is taken from the start of the cell to the last byte of the last connector cell item. For data cells, this field can be specified by the user. If the L field is used as an address modifier, i.e., added to the cell address, the resultant byte address would point to the next available space in the cell. For example, if a low level connector cell is being prepared in core with an item length specification of 1, and if the connector cell currently contains one data cell address, the L field would contain the value 12. The cell address plus 12 would point to the word in which the next data cell address is to be placed.

The P1 field is 8 bits long and is used for data directing codes. For files that interface with system functions, this field may be used by the applied system to contain detailed information concerning data formats, editing required, compression, expansion, etc.

The 8-bit P2 field is used in conjunction with the P1 field. For files that interface with system functions, this field contains parameters that are required by the program specified in P1. For files that are used strictly within applied systems, this field may be used by the applied system in the same manner as described for P1.

7.2.3 Connector Cells

Connector cells for files maintained on disc contain the addresses of lower level cells. Information about the lower level cell: e.g., a protection key can also be contained in a connector cell. The address of a lower level cell or a cell sequence number and any related information is referred to as a connector cell item. Connector cell items within a cell are the same length as specified by the IL field in the header. If the IL field specifies a length of 2 bytes, then the connector cell item only consists of an address or block sequence number.

The high level and intermediate level connector cell items contain an address field and a K3 field. The K3 field is equal to the K2 field for the next level connector cell.

Items for the lower level connector cells contain the cell addresses for data cells plus a PI field. The PI field can be from 2 to 254 bytes long. The first 2 bytes of the PI field match the 2 bytes following the AK field of the referenced data cell. The PI field may be specified by the user and may, for example, contain sort keys which relate to the data pointed to by the first word of the connector cell item.

7.2.4 Data Cell

Data cells on disc contain the 2-word header described previously and the user supplied data.

7.3 TAPE FILE STRUCTURE

The C-System tape file structure is very similar to disc file storage. There are, however, some significant differences which are discussed in the following paragraphs.

Within the C-System, the user has several options in the organization of files on tape. The user can choose to build the tape file without building the full connector structure. In this case, only a reels list is built and retained on disc and last tape. A reels list identifies the tape reels within a tape file. The user can build the tape file with all levels of connector cells retained on disc and tape or the user can build the tape file with all data cells and connector cells on tape and the reels list on disc and tape. This provides for efficient use of disc file space and still allows for rebuilding the entire connector structure on disc at a later time.

Figure 7-4 shows the general structure of tape file organization for each of the three options. The figure shows a single reel file which was built without the full connector structure, a 2-reel file with the full connector structure retained on disc, and a multiple reel file with only the reels list retained on disc.

7.3.1 Full Connector Structure on Disc

To explain file connector structure on disc, the double reel file of figure 7-4 is used as an example. The file is addressed by the 2-word file identifier as shown in figure 7-4. The AK and K2 fields of the file identifier are identical to those of a disc file. The Z field identifies the disc zone in which the tape connector structure or reels list resides. Bit 0 of the Z field is set to a 1 to indicate that this is a tape file. The L1 field contains the time division exchange loop 1 address of the disc on which the tape connector structure resides. The CA field contains the disc cell address of the high level connector cell.

As shown in figure 7-4, the file identifier points to the high level connector cell for the file. The high level connector cell for tape files contains the 8-byte cell header and a connector cell item for each tape reel in the file. In this case, there are two connector cell items; one for each of the two tape reels. The cell header fields are identical to those for a disc file.

The item entries in the high level connector are 8 bytes in length. The RKN field contains the reel key number of the tape which together with the AK field of the item entry is the unique identifier for this tape. This 4-byte word is written as the first word on a tape when allocated via device acquisition and control service. A high level connector list may be converted into a reels list for a tape file. It is because of this that each item entry in the high level connector cell must reference a unique RKN. This means that each item entry in the high level connector cell must point to a structure which completely describes the data on a single reel.

The high level connector item entry C field indicates whether or not connectors are contained in this reel of tape. In this case, the bit would be set to 0 since the connector structure is not on the tape.

The IL field of the high level connector item entry is the item length indicator for low level connectors when written to tape.

The L field of the high level connector item entry when set to a 0 indicates that no data cells are on this particular tape reel.

The high level connector item cell address points to the next level in the connector structure which for this example is an intermediate level connector cell. This first intermediate level connector cell is the high level connector cell for a particular tape reel. This connector cell may contain item entries which in turn point to intermediate level connector cells or item entries which point to low level connector cells. An example of the first intermediate level connector cell pointing to another level of intermediate connector cells is shown in figure 7-1.

As shown in figure 7-1, several levels of intermediate level connectors may be required to completely define the structure of data on one tape reel.

A small file on tape may require only one low level connector cell. In this case, the cell address of the high level connector cell points to the low level connector and there is not an intermediate level connector in the file structure. The low level connector K2 field then contains the RKN for the tape.

The cell header K2 of an intermediate level connector contains the RKN of the tape. The remaining fields of the intermediate level connector cell header are as previously described with field A set to a 1 in all except the lowest level intermediate level connector cell.

The intermediate level connector cell item entry is a 1-word entry. As shown, the cell address may be that of a lower level intermediate level connector cell or that of a low level connector cell. For the former case, the K3 field contains the RKNof the tape. For the latter, the K3 field contains the K2 value for the low level connector cell.

As shown in figure 7-4, the low level connector contains the standard cell header fields. However, the A field is set to a 1 as opposed to a 0 in the case of a disc file. This allows cell release routines to consider the next higher level connector as the low level connector and thus perform the release on only those cells in the file structure. This is necessary since the data cells are on tape.

As shown in figure 7-4, the low level connector item entry contains the BSN field in the first two bytes. The BSN field contains a block sequence number whch specifies the displacement of the data cell on tape from the tape header.

The item entry K3 field contains the first two bytes of the item parameter which is specified by the user. The remaining PI field may contain up to 252 bytes of user supplied information. An example of its use would be to contain sort keys which relate to the data pointed to by this item entry. As shown in figure 7-4, each data cell on tape is preceded by what is called the low level connector cell prefix. This prefix consists of the cell header for the particular low level connector cell and the item entry which points to the data cell. This prefix is written to tape for checkpoint as well as sequential access purposes.

Each data cell on tape contains an 8-byte header. The first two bytes contain the Block Sequence Number (BSN) for this data cell. The next two bytes contain the first two bytes of the item parameter field. The remaining fields of the data cell header are used as previously discussed.

As shown in figure 7-4, data cells are written to tape until the file is complete or the end of tape is reached. Four tape marks are then written.

7.3.2 Connector Structure on Tape

Figure 7-4 shows a multiple reel file with the complete connector structure on tape and the reels list on disc. The file is addressed by the 2-word identifier which points to the reels list.

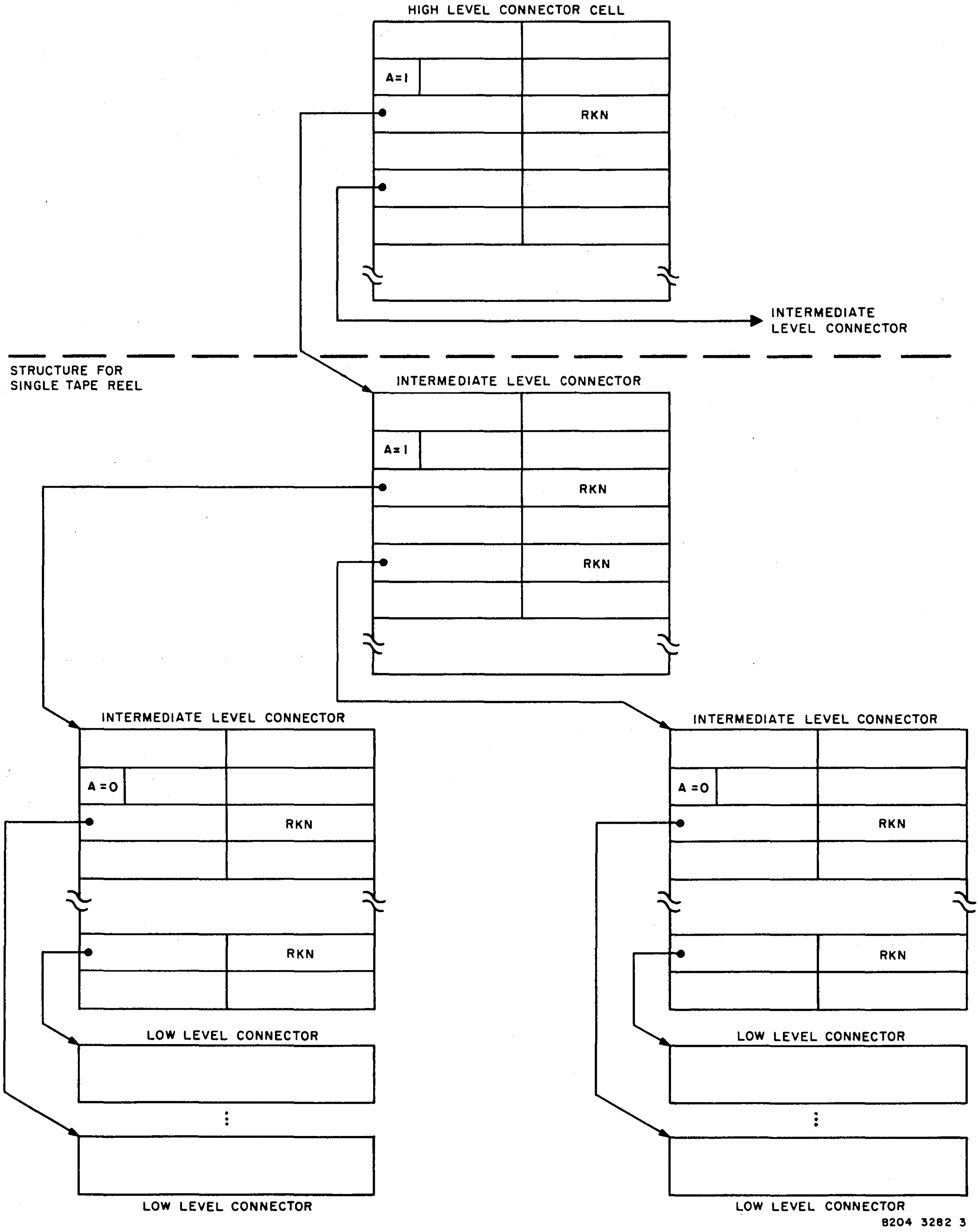

HIGH LEVEL CONNECTOR CELL
A=1
RKN
INTERMEDIATE LEVEL CONNECTOR
STRUCTURE FOR SINGLE TAPE REEL
INTERMEDIATE LEVEL CONNECTOR
A=1
RKN
RKN
INTERMEDIATE LEVEL CONNECTOR
A=0
RKN
RKN
LOW LEVEL CONNECTOR
LOW LEVEL CONNECTOR
INTERMEDIATE LEVEL CONNECTOR
A=0
RKN
RKN
LOW LEVEL CONNECTOR
LOW LEVEL CONNECTOR
B204 3282 3

Figure 7-1. Multiple Level of Intermediate Level Connector Cells.

A reels list is the vehicle by which pointers to reels of tape within a tape file are retained on disc without keeping the complete connector cell structure on disc. The high level connector cell is identical in format to the reels list with one exception. The left half of the first word in each connector cell item normally contains a disc cell address; in creating the reels list, this halfword is set to 0. Each connector cell item of a high level connector points to a connector cell substructure which in turn points to the information on one reel of tape. In a reels list, the high level connector cell items identify only the tape reels within a file and provide information as to which reels have a second file containing connector cells. The C field set to a 1 indicates connectors are contained on this reel of type. From this information and the connector level indicator in each connector cell on tape, the connector cell structure can be reconstructed on disc.

Note that a 0 address within a connector cell structure denotes a null entry.

When the connector cell structure or reels list is written to tape, the AK field of the cell header is changed to a Connector Level Indicator (CLI). An example of this structure is shown in figure 7-2.

The connector cell structure on tape is contained in a separate file from the data referenced by this structure and is separated from the data file by a single tape mark. For the case shown in figure 7-2, the connector cell structure is shown on tape reel M.

Data cells are written to tape until the file is complete or the end of tape is reached. If the file is complete, a tape mark is written and the complete connector cell structure is written as a second file to tape followed by four tape marks. If the end of tape is reached prior to the file being complete, four tape marks are written and the remaining data cells are written to the next tape. In either case, the cell address of the intermediate level connector cell and the RKN number are stored in a high level connector cell. As new tapes are required, the intermediate level connector cell addresses and the associated RKN numbers are added to the high level connector until the file is complete.

Following the tape mark of the last tape, the connector cell structure is written to tape followed by four tape marks.

In the event that the end of reel is reached while writing the second file, the tape is backspaced to the first tape mark and three more tape marks are written. A new tape is obtained for the connector cells. A new entry is made in the high level connector cell. This entry contains the RKN of the new tape. A 0 cell address is used in the address portion of the connector item since no lower level connector cells are involved in the new tape.

7.3.3 Reels List on Disc

As discussed earlier, the user has the option of building a file on tape without building the full connector structure on disc. In this case, a reels list is built and written to tape as a second file. This reels list is also maintained on disc as shown in figure 7-4.

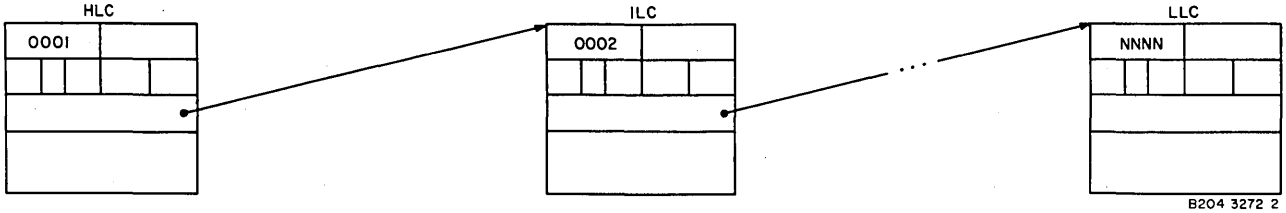
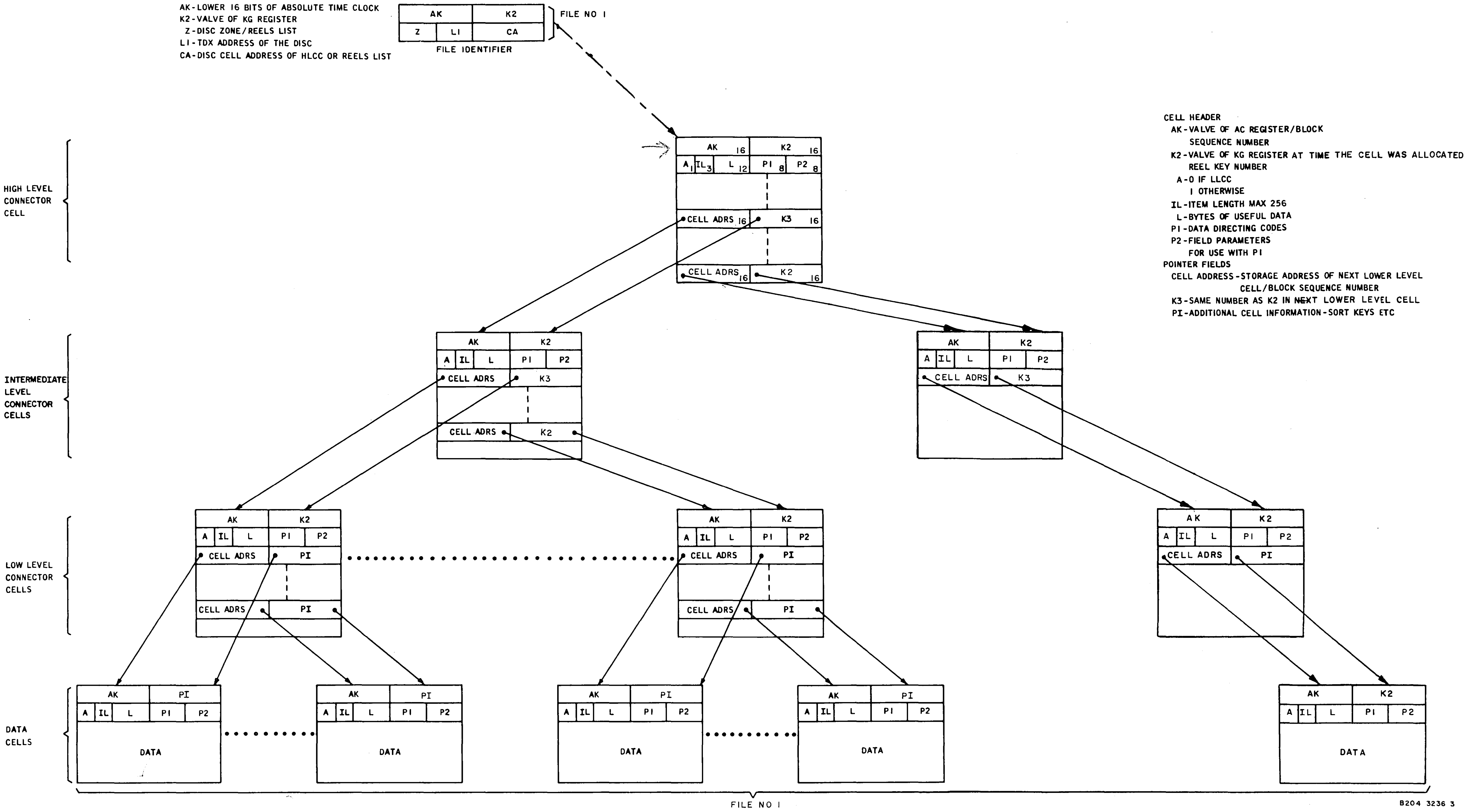


Figure 7-2. Connector Cell Structure on Tape.



AK-LOWER 16 BITS OF ABSOLUTE TIME CLOCK
K2-VALVE OF KG REGISTER
Z-DISC ZONE/REELS LIST
LI-TDX ADDRESS OF THE DISC
CA-DISC CELL ADDRESS OF HLCC OR REELS LIST

CELL HEADER
AK-VALVE OF AC REGISTER/BLOCK SEQUENCE NUMBER
K2-VALVE OF KG REGISTER AT TIME THE CELL WAS ALLOCATED REEL KEY NUMBER
A-0 IF LLCC
1 OTHERWISE
IL-ITEM LENGTH MAX 256
L-BYTES OF USEFUL DATA
PI-DATA DIRECTING CODES
P2-FIELD PARAMETERS FOR USE WITH PI
POINTER FIELDS
CELL ADDRESS-STORAGE ADDRESS OF NEXT LOWER LEVEL CELL/BLOCK SEQUENCE NUMBER
K3-SAME NUMBER AS K2 IN ~~NEXT~~ NEXT LOWER LEVEL CELL
PI-ADDITIONAL CELL INFORMATION-SORT KEYS ETC

Figure 7-3. Basic File Structure.

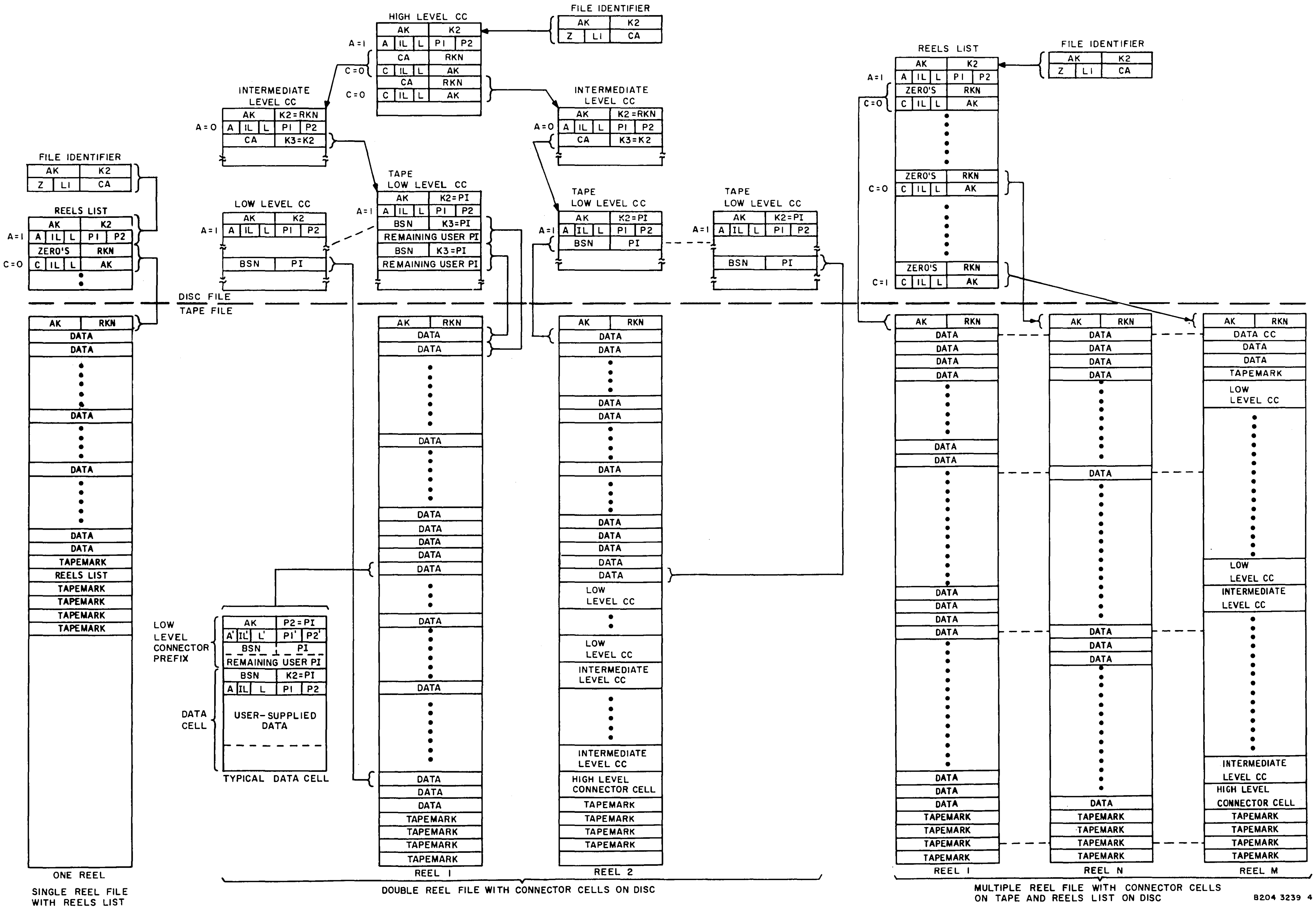


Figure 7-4. Multiple Reel Tape File Structure.

appendix

1. ORDERWIRE 1 SERVICE MESSAGE FORMATS

1.1 Immediate Operations Formats

1.1.1 Replace Output Tape Return

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td>F
8</td><td>G
8</td></tr>
<tr><td colspan="2">H
16</td><td colspan="2">I
16</td></tr>
<tr><td>J
8</td><td>K
8</td><td>L
8</td><td>M
8</td></tr>
<tr><td colspan="4">N
32</td></tr>
<tr><td colspan="4">O
32</td></tr>
</table>

A - Replace output tape return, op code - X'01
B - Spare
C - OW1 address of calling processor
D - Calling processor channel, 000 - not used, 010 - S channel, 101 - M channel, 110 - B channel/ OW2 channel, 111 - A channel
E - Address of DSW of the FTC requesting a replace output tape
F - L1 working channel of tape unit
G - Spare
H, I - AK and RKN if J = X'CO; DACS status if J = X'EO
J - X'CO or X'EO as supplied by DACS
K - Party line address of tape unit
L - First byte of tape control transfer command packet which initiated the replace output tape
M - Stack-zone into which tape reel is to be placed
N - Spare
O - Spare

Receiving Processor Function - When received by the OW1 immediate service function, the service function verifies that the address specified in field E falls within the boundaries defined by field D. If the channel boundaries are correct, words 2 and 3 (H, I, J, K, L, M) are stored in the location specified by field E.

1.1.2 Initialize Ops Control

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="4">E
64</td></tr>
<tr><td>F
8</td><td>G
8</td><td>F
8</td><td>G
8</td></tr>
<tr><td>F
8</td><td>G
8</td><td colspan="2">H
16</td></tr>
<tr><td colspan="4">I
32</td></tr>
</table>

A - Initialize ops control, op code - X'02
B - Spare
C - OW1 address of originating processor
D - Calling processor channel, X'00 - not used, X'02 - S channel, X'05 - M channel, X'06 - B channel/OW2 channel, X'07 - A channel
E - Regulator identification key. Used for added protection against illegal modifications to the table.
F - Defines which channel position of ops control is to be modified
0 - A channel
1 - B channel
2 - M channel
G - Vector that defines the required state of the regulatory status indicators of ops control. The vector corresponds to the first eight bits of word 0 of the ops control table. Only bits 1, 3, and 4 are under regulator control. These bits of the ops control table should be set to the state of the vector. The remaining bits of the ops control table are left unchanged.
H, I - Spare

Receiving Processor Function - Since direct access service messages cannot be used to update the ops control tables because of the possibility of changing data not related to load regulation, an immediate operation service message is used. The immediate operation service message is intercepted and processed by an immediate action service function within the OW1 channel of the receiving processor. The action taken is to modify the status indicators of the ops control table as defined by the service message.

1.2 S Channel Formats

1.2.1 CPSR Call

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="4">J
32</td></tr>
<tr><td colspan="4">K
32</td></tr>
<tr><td colspan="4">L
32</td></tr>
</table>

A - CPSR call op code - X'40
B, C - Spare
D - Double word increment to referenced PCI
E, F - Protection key (AK, K2)
G - Zone identifier of the CPSR
H - L1 address of the CSPR
I - Cell address of the CPSR
J - Spare
K - Spare
L - Spare

Receiving Processor Function - On receiving this message, CPS must:

a. Load the CPSR.
b. Isolate the input list in the PCI.
c. Isolate the control program address in the PCI.
d. Load and initiate the CP.

1.2.2 Program Return

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="4">O
32</td></tr>
</table>

A - Program return, op code - X'42
B, C - Spare
D - Double word increment to referenced PCI
E, F - Protection keys (AK, K2) of the CPSR
G - Zone identifier of the CPSR
H - L1 address of the CPSR
I - Cell address of the CPSR
J, K - Protection keys (AK, K2) of highest level connector cell of the output file or literal data.
L - Zone identifier of the highest level connector cell of the output file or zero if literal data.
M - L1 address of the highest level connector cell of the output file or zero if literal data.
N - Cell address of the highest level connector cell of the output file or zero if literal data.
O - Spare

Receiving Processor Function - On receiving this message, CPS must:

a. Load the CP status record.
b. Isolate the PCI and store the second operand field (J, K, L, M, N) of the return message in the O field of the PCI.
c. Build space release messages if required.
d. Decrement N fields of PCIs referenced by the E field of this PCI.
e. Build call messages for those PCIs whose N field went to zero.
f. Store the updated CPSR back on disc storage.

1.2.3 *CP Abnormal Return*

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="4">O
32</td></tr>
</table>

A - CP abnormal return, op code - X'43
B - Spare
C - Spare
D - Double word increment to the referenced PCI
E, F - Protection keys (AK, K2) of the CPSR
G - Zone identifier of the PCI
H - L1 address of the CPSR
I - Cell address of the CPSR
J, K - Protection keys (AK, K2) of CPSR completing abnormally
L - Zone identifier of the CPSR completing abnormally
M - L1 address of the CPSR completing abnormally
N - Cell address of the CPSR completing abnormally
O - Spare

Receiving Processor Function - On receiving this message, CPS will:

a. Freeze the initiating CPSR.
b. When all outstanding PCIs have completed, initiate the recovery PCI if provided. If a recovery PCI is not provided, CPS abnormally completes the calling CPSR.

1.2.4 AP Abnormal

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="4">O
32</td></tr>
</table>

A - Abnormal return, op code - X'44
B - Spare
C - Spare
D - Double word increment to the referenced PCI
E, F - Protection keys (AK, K2) of the CPSR
G - Zone identifier of the CPSR
H - L1 address of the CPSR
I - Cell address of the CPSR
J, K - Protection keys (AK, K2) of abnormal status
L - Zone identifier of abnormal status
M - L1 address of abnormal status
N - Cell address of abnormal status
O - Spare

Receiving Processor Function -

a. Freeze the initiating CPSR.
b. When all outstanding PCIs have completed, initiate the recovery PCI if provided. If a recovery PCI is not provided, CPS abnormally completes the calling CPSR.

1.2.5 Space Release

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="4">O
32</td></tr>
</table>

A - Space release, op code - X'45
B - Spare
C - Spare
D - Double word increment to referenced PCI
E, F - Protection keys (AK, K2) of CPSR
G - Zone identifier of CPSR
H - L1 address of CPSR
I - Cell address of CPSR
J, K - Protection keys (AK, K2) of high level connector of file to be released
L - Zone identifier of high level connector of file to be released
M - L1 address of high level connector of file to be released
N - Cell address of high level connector of file to be released
O - Spare

Receiving Processor Function - When the disc storage allocator receives this message, the file specified by J, K, L, M, N is placed in the released state.

1.2.6 CPSR Generator Return

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="4">O
32</td></tr>
</table>

A - CPSR generator return, op code - X'47
B - Spare
C - Spare
D - Double word increment to referenced PCI
E, F - Protection keys (AK, K2) of calling CPSR
G - Zone identifier of calling CPSR
H - L1 address of calling CPSR
I - Cell address of calling CPSR
J, K - Protection keys (AK, K2) of generated CPSR
L - Zone identifier of generated CPSR
M - L1 address of generated CPSR
N - Cell address of generated CPSR
O - Spare

Receiving Processor Function - On receipt of this message, CPS updates the calling PCI with the address of the generated CPSR and issues a CPSR call service message.

1.2.7 Initial CPSR Call

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="4">O
32</td></tr>
</table>

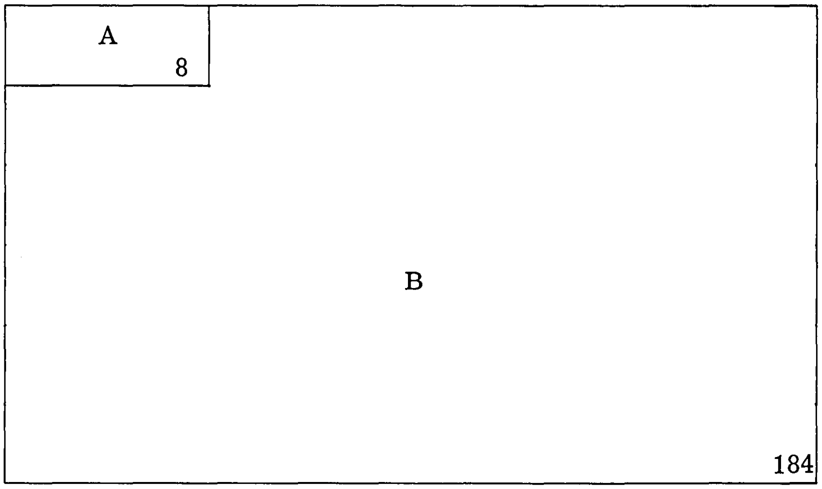
A - Initial CPSR call, op code - X'4D
B - Spare
C - Spare
D - Double word increment to referenced PCI
E, F - Protection keys (AK, K2) of calling CPSR
G - Zone identifier of calling CPSR
H - L1 address of calling CPSR
I - Cell address of calling CPSR
J, K - Protection keys (AK, K2) of generated CPSR
L - Zone identifier of generated CPSR
M - L1 address of generated CPSR
N - Cell address of generated CPSR
O - Spare

Note

The calling CPSR identifier (D, E, F, G, H, I) in this message is not required by CPS. It has been included in case of abnormal. conditions.

Receiving Processor Function - On receiving this message, CPS must load and initiate the CPSR.

1.2.8 Redirect



A - Redirect, op code X'4E

B - Remaining structure of service message is dependent upon the service message being redirected.

The following types of service message are possible for redirect or rerouting:

a. CPSR call
b. AP call
c. CPSR generator call
d. All service function calls

Receiving Processor Function - CPS reads the CPSR, determines the type of service message, and reissues the service message to another processor.

1.3 Orderwire 2 Channel Formats

1.3.1 DACS Functions

1.3.1.1 Tape Release

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="4">O
32</td></tr>
</table>

A - Tape release, op code - X'80
B - Spare
C - Spare
D - Double word increment to referenced PCI
E, F - Protection keys (AK, K2) of CPSR
G - Zone identifier of CPSR
H - L1 address of CPSR
I - Cell address of CPSR
J, K - Protection keys (AK, K2) of the reels list or the high level connector of the file to be released.
L - Zone identifier of the reels list or the high level connector of the file to be released.
If LO = 0, disc file zone and no tapes are to be released.
If LO = 1, tape file and tapes to be released are contained in the referenced reels list.
M - L1 address of the reels list or the high level connector of the file to be released.
N - Cell address of the reels list or the high level connector of the file to be released.
O - Spare

Note

The CPSR identifier (D, E, F, G, H, I) in this message is not required for tape release. It is included for recovery if tape release errors occur.

Receiving Processor Function - DACS places the indicated tape reels in a released state and then generates a space release message to release the disc space.

1.3.1.2 DACS Request

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="2">O
16</td><td colspan="2">P
16</td></tr>
</table>

A - DACS request, op code - X'83
B - Number of inputs in PCI input list
C - OW1 address of the originating processor
D - Double word increment to calling PCI
E, F - Protection keys (AK, K2) for calling PCI
G - Zone identifier of the CPSR
H - L1 address of the CPSR
I - Cell address of the CPSR
J, K, L, M, N, O - Variable dependent upon the type of DACS request. These fields are always obtained from the first input and the first 16 bits of the second input of the calling PCI. The contents of these fields are as follows.

a. Literal DACS Request

Field	Description	
J, K	For input tape-	J = AK, K = RKN
	For scratch tape-	J = O, K = Library Zone X'1 thru X'F
	For card reader-	forms identifier
	For CRT-	forms identifier
	For line printer-	forms identifier
L	Device type-	7 channel tape - X'50
		9 channel tape - X'51
		Card reader - X'10
		CRT - X'40
		Line printer - X'30 thru X'3E
M	Party line address	
N	Zero	
O	L1/L2 address or zero	

1.3.1.2 (Cont)

b. Tape File Request

Field	Description
J, K, L, M, N	Disc file identifier of the high level connector cell or reels list.
O	Spare

c. Indirect Request

Field	Description
J, K, L, M, N	File identifier of either a single cell or a multiplex data file which may contain multiple DACS requests
O	Zero/PI selection key for retrieval of a data cell from a multiplexed file.
P	Spare

Receiving Processor Function - DACS processes the request and on completion builds a program return message that references the output of DACS.

1.3.1.3 Replace Output Tape

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td>F
8</td><td>G
8</td></tr>
<tr><td colspan="2">H
16</td><td colspan="2">I
16</td></tr>
<tr><td>J
8</td><td>K
8</td><td>L
8</td><td>M
8</td></tr>
<tr><td colspan="4">N
32</td></tr>
<tr><td colspan="4">O
32</td></tr>
</table>

A - Replace output tape, op code - X'84
B - Spare
C - OW1 address of calling processor
D - Calling processor channel, 000 - not used, 010 - S channel, 101 - M channel, 110 - B channel/ OW2 channel, 111 - A channel
E - Core address where the return status is to be stored
F - L1 working channel of tape unit
G - Spare
H - AK to be used as AK for tape identification
I - Zeros in replace output tape message
J - Spare
K - Party line address of tape unit
L - First byte of tape control transfer command packet which initiated the replace output tape
M - Library zone into which tape reel is to be placed

Receiving Processor Function - DACS assigns an available tape to the requestor, writes a 1-word header on the tape and then generates a replace output tape return service message.

1.3.1.4 DACS Routing Request

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="2">O
16</td><td>P
8</td><td>Q
8</td></tr>
</table>

A - DACS routing request. op code - X'8D
B - Count the number of inputs in the calling PCI; zero if from a program.
C - OW1 address of the originating processor
D - Zero or double word increment of calling PCI
E, F, G, H, I - File identifier of calling CPSR; zero if from a program
J, K, L, M, N - File identifier of message
O - If D=0, C-Number to substitute in program callservice message or zero indicating request for general service program.
P - Zero or party line address of particular device
Q - Output device type

Receiving Processor Function - DACS either assigns a TDX device or selects the specific device requested and issues a program call service message for the required I/O program.

1.3.2 DCS Functions

1.3.2.1 DCS File Update from CPS

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td><td></td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td><td></td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td><td></td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td><td rowspan="2">Input list entry 1 of calling PCI</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="2">O
16</td><td colspan="2">P
16</td><td></td></tr>
</table>

A - File update from CPS, op code - X'81
B - Count of the number of inputs contained in the referenced PCI
C - OW1 address of the originating processor
D - Double word increment to calling PCI
E, F - Protection keys (AK, K2) of calling CPSR
G - Zone identifier of calling CPSR
H - L1 address of calling CPSR
I - Cell address of calling CPSR
J, K - Protection keys (AK, K2) of file to be added to the DCS file.
L - Zone identifier of file to be added to the DCS file
M - L1 address of the file to be added to the DCS file
N - Cell address of the file to be added to the DCS file
O - DCS code in binary - first 16 bits of input list entry 2 of calling PCI
P - Zeros/RTS parameters. Not to be used by DCS

Note

J, K, L, M, N may constitute a tape file identifier.

Receiving Processor Function - DCS adds the input file to the specified DCS collection file. A program return service message is issued with a null output unless the DCS code is unassigned, in which case an abnormal program return is sent.

1.3.2.2 DCS File Update from Program

<table>
<tr><td>A
8</td><td colspan="2">NOT USED
16</td><td>B
8</td></tr>
<tr><td colspan="2">C
16</td><td colspan="2">D
16</td></tr>
<tr><td>E
8</td><td>F
8</td><td colspan="2">G
16</td></tr>
<tr><td colspan="2">H
16</td><td colspan="2">I
16</td></tr>
<tr><td>J
8</td><td>K
8</td><td colspan="2">L
16</td></tr>
<tr><td colspan="2">M
16</td><td colspan="2">N
16</td></tr>
</table>

A - File update from program, op code - X'81
B, C, D, E, F, G - Zeros
H, I - Protection keys (AK, K2) of file to be added to DCS file.
J - Zone identifier of the file to be added to DCS file
K - L1 address of disc containing the file to be added to DCS file
L - Cell address of the file to be added to DCS file
M, N - DCS code expressed as 4 ASCII characters.

Note

H, I, J, K, L may constitute a tape file identifier.

Receiving Processor Function - The file identifier is added to either the tape entry or disc entry of the data collection file, and thus is available for the next extract performed on its DCS code.

1.3.2.3 DCS Extract and Delete

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="2">O
16</td><td colspan="2">P
16</td></tr>
</table>

A - DCS extract and delete, op code - X'82
B - Count of the number of input list entries in the calling PCI
C - OW1 address of the originating processor
D - Double word increment of calling PCI
E, F - Protection keys (AK, K2) of the calling CPSR
G - Zone identifier for calling CPSR
H - L1 address of calling CPSR
I - Cell address of calling CPSR
J - DCS code in binary form - first 16 bits of input list entry 1 of the calling PCI
K, L, M, N - Zeros
O - Spare
P - Spare

Note

The output field of the calling PCI contains the address of the collection file. or zero.

Receiving Processor Function - This service message causes DCS return via a program return service message the address of the collection file (may be null).

If the DCS code is inactive (has not been assigned), an abnormal program return service message is issued.

1.3.2.4 DCS Code Delete

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td><td></td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td><td></td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td><td></td></tr>
<tr><td colspan="4">J
32</td><td rowspan="2">Input list entry 1 of calling PCI if from control program</td></tr>
<tr><td colspan="4">K
32</td></tr>
<tr><td colspan="2">L L
16</td><td colspan="2">M
16</td><td></td></tr>
</table>

A - DCS code delete, op code - X'87
B - Count of the number of inputs in the calling PCI; zero if from a program
C - OW1 address of originating processor
D - Double word increment of calling PCI; zero if from program
E, F - Protection keys (AK, K2); zero if from program
G - Zone identifier of calling CPSR; zero if from program
H - L1 address of calling CPSR; zero if from program
I - Cell address of calling CPSR; zero if from program
J - Spare
K - 4 ASCII characters representing responsible area or person
L - DCS code to delete in binary form - first 16 bits of input list entry 2 of calling PCI if from control program
M - Spare. This field may have residue RTS parameters if call originating from CPS

Note

The contents of field K must be identical to that given at the time the DCS code was assigned.

Receiving Processor Function - This service message causes a DCS code to be made inactive and data in the file to be placed under DCS code 1. This message may be issued by a program (with no return) or a control program. If the four ASCII characters in field K do not correspond with those given when the code was assigned, no action is taken by DCS. If the call originated from a CP, an abnormal return is issued; otherwise, an intercept message is delivered to the intercept station for the DCS code.

1.3.2.5 DCS Code Request (Unload Allowed)

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="4">J
32</td></tr>
<tr><td colspan="4">K
32</td></tr>
<tr><td colspan="2">L
16</td><td colspan="2">M
16</td></tr>
</table>

Input list
entry 1 of
calling PCI

A - DCS code request (unload allowed), op code - X'88
B - Count of the number of inputs in the calling PCI; zero if from a program
C - OW1 address of the calling processor
D - Double word increment of calling PCI; zero if from a program
E, F - Protection keys (AK, K2) of the calling CPSR; zero if from a program
G - Zone identifier of calling CPSR; zero if from a program
H - L1 address of calling CPSR; zero if from a program
I - Cell address of calling CPSR; zero if from a program
J - Truncated binary C-Number of intercept station consisting of EEE-NNN-PP
K - 4 ASCII characters representing responsible area or person at intercept station
L - Spare
M - Spare

Receiving Processor Function - This service message causes DCS to issue a DCS code. The response to this message is a program return service message. If there are no DCS codes available, an abnormal program service message is issued.

The DCS code assigned is returned in the first 16 bits of the output field of the calling PCI, the remainder of the field being zero.

1.3.2.6 DCS File Return

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="2">O
16</td><td colspan="2">P
16</td></tr>
</table>

A - DCS file return, op code - X'8B

B - Count of the number of inputs in the calling PCI; zero if from a program

C - OW1 address of the originating processor

D - Double word increment to the calling PCI; zero if from a program

E, F, G, H, I - File identifier of calling CPSR; zero if from a program; unique protection value if from the systems checkpoint program

J, K, L, M, N - File identifier of a collection file(s) that is to be merged with that in existence for the DCS code in field O. If field O is zero, this is the address of an updated DCS file that is to replace that currently in use.

O - Binary DCS code or zero. If zero, this is a return of a DCS service file and field E, F, G, H, I must contain a protection value.

P - Spare

Receiving Processor Function - This service message is used by the DCS control program to return the address of the new DCS file or a DCS code collection file after the disc to tape unload has taken place. If this file is not on a disc attached to the OW2 processor, it will be copied into such a disc.

1.3.2.7 DCS Collection Inhibit

<table>
<tr><td>A
8</td><td colspan="2">B
24</td></tr>
<tr><td colspan="3">C
64</td></tr>
<tr><td colspan="3">D
64</td></tr>
<tr><td colspan="2">E
16</td><td>F
16</td></tr>
</table>

A - DCS collection inhibit, op code - X'8C
B - Spare
C - Unique protection value of system update and checkpoint program
D - Spare
E - 0 for inhibit action on all DCS codes, DCS code to inhibit action by queueing all requests for service
F - Spare

Note

DCS inhibit service messages are used only by the system.

1.3.2.8 DCS Extract and Save

A 8	B 8	C 8	D 8
E 16		F 16	
G 8	H 8	I 16	
J 16		K 16	
L 8	M 8	N 16	
O 16		P 16	

A - DCS extract and save, op code - X'90
B - Count of the number of inputs in the calling PCI.
C - OW1 address of originating processor
D - Double word increment of calling PCI
E, F - Protection keys (AK, K2) for the calling CPSR
G - Zone identifier of calling CPSR
H - L1 address of calling CPSR
I - Cell address of calling CPSR
J - DCS code in binary form - first 16 bits of input list entry 1 of the calling PCI
K, L, M, N - Zeros
O - Spare
P - Spare

Receiving Processor Function - DCS extracts and saves data for those DCS codes which have the attribute 'no unload' and were assigned by the system. A return message is generated causing the output field of the calling PCI to contain either the address of the single cell collection file or zeros if a file does not exist.

1.3.3 Scheduler Functions

1.3.3.1 Scheduler Service Call

<table>
<tr><td>A
8</td><td>B
8</td><td colspan="2">C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="3">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="3">I
16</td></tr>
<tr><td>J
8</td><td>K
8</td><td>L
3</td><td>M
5</td><td>N
8</td></tr>
<tr><td>O
8</td><td>P
8</td><td colspan="3">Q
16</td></tr>
<tr><td colspan="5">R
32</td></tr>
</table>

A - Scheduler request, op code - X-85
B - Number of inputs in PCI input list
C - OW1 address of originating processor
D - Double word increment to calling PCI (schedule PCI)
E, F - Protection keys (AK, K2) of calling CPSR
G - Zone identifier of calling CPSR
H - L1 address of calling CSPR
I - Cell address of calling CPSR
J, K, L, M, N - Schedule start, fiscal data/time for current activation J - year, K - week, L - day, M - hour, N - minute
O - Schedule priority code
P - Cycle lapse code
Q - Spare
R - Spare

Receiving Processor Function - The scheduler program places the service message in a schedule queue if the activation time is equal to or later than the next table generator call time. If the activation time is earlier than the next table generator call time, then the service message is placed in an advanced activation cell.

1.3.3.2 Scheduler Table Generator Return

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="4">O
32</td></tr>
</table>

A - Scheduler table generator return, op code - X'86
B - Spare
C - OW1 address of originating processor
D - Double word increment to calling PCI
E, F - Protection keys (AK, K2)
G - Zone identifier of CPSR
H - L1 address of CPSR
I - Cell address of CPSR
J, K - Protection keys (AK, K2) of new scheduler file
L - Zone identifier of new scheduler file
M - L1 address of new scheduler file
N - Cell address of new scheduler file
O - Spare

Receiving Processor Function - Upon receiving the message, the scheduler:

a. Replaces the current schedule table files with the new schedule table files.
b. Establishes the current priority process pointers for the new priority sub files.

1.3.4 Load Regulator Functions

1.3.4.1 Processor Initialization

<table>
<tr><td>A
8</td><td colspan="2">B
8</td><td>C
8</td><td colspan="2">D
8</td></tr>
<tr><td colspan="6">E
64</td></tr>
<tr><td>F
8</td><td colspan="2">G
8</td><td>H
8</td><td>I
3</td><td>J
5</td></tr>
<tr><td colspan="2">K
10</td><td colspan="4">L
22</td></tr>
<tr><td colspan="6">M
32</td></tr>
</table>

A - Processor initialization, op code - X'91
B - Type of processor
C - OW1 address of the calling processor
D - Specifies channels assigned to calling processor
E - Protection key for load regulator call service message
F - Number of discs assigned to calling processor
G - Increment to an entry in the queue threshold table
H - L1 address of the primary disc for the calling processor
I - Party line address of the primary disc for the calling processor
J - Logical disc format code of the calling processor
K - Exchange number assigned to the processor
L - Spare
M - Spare

Receiving Processor Function - On receipt of this message, the load regulator updates the center configuration tables, sets zone and queue threshold values for the calling processor, and updates the routing tables in the center.

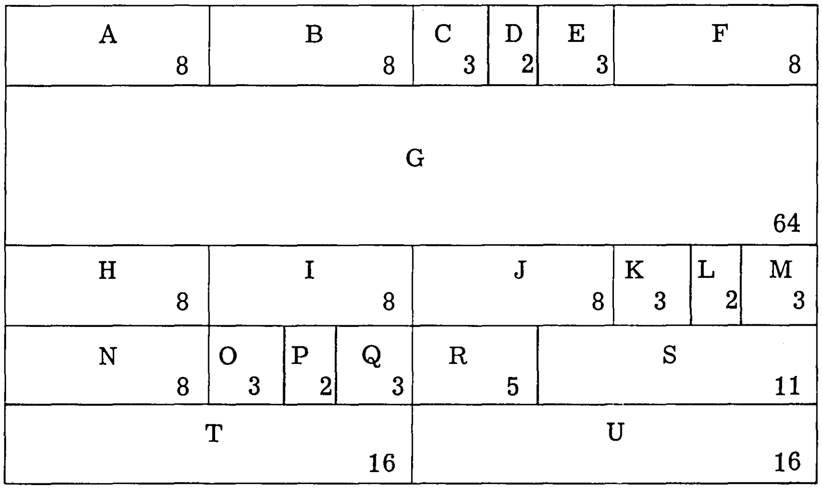
1.3.4.2 *Processor Removal*

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="4">E
64</td></tr>
<tr><td colspan="4">F
32</td></tr>
<tr><td colspan="4">G
32</td></tr>
<tr><td colspan="4">H
32</td></tr>
</table>

A - Processor removal, op code - X'92
B - OW1 address of processor that failed
C - OW1 address of processor (calling processor) that assumed the failed processor's duties
D - Spare
E - Protection key for load regulator call service message
F - Spare
G - Spare
H - Spare

Receiving Processor Function - On receipt of this message, the load regulator updates the center configuration tables and routing tables.

1.3.4.3 Disc Initialization



A - Disc initialization, op code - X'93
B - L1 address of disc being initialized
C - Party line address of disc being initialized
D - Type of disc
E - Software zone status
F - OW1 address of calling processor
G - Protection key for load regulator call service message
H - OW1 address of processor assigned to this disc
I - OW1 address of backup processor assigned to this disc
J - L1 address of the disc containing the backup software zone for this disc
K - Party line address of the disc containing the backup software zone for this disc
L - Backup zone use indicator
M - Spare
N - L1 address of the disc containing the backup data zone for this disc
O - Party line address of the disc containing the backup data zone for this disc
P - Backup data zone indicator
Q - Spare
R - The physical format code for this disc
S - Disc bad-spot threshold, used to initiate unscheduled disc maintenance
T - Integer specifying which day of the year the disc is scheduled for maintenance
U - Spare

Receiving Processor Function - On receipt of this message, the load regulator updates the center configuration tables.

1.3.4.4 Disc Removal

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
3</td><td>E
5</td></tr>
<tr><td colspan="5">F
64</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="3">I
16</td></tr>
<tr><td colspan="5">J
32</td></tr>
<tr><td colspan="5">K
32</td></tr>
</table>

A - Disc removal, op code - X'94
B - L1 address of the disc being removed from the center
C - OW1 address of the calling processor
D - Party line address of the disc being removed from the center
E - Spare
F - Protection key for load regulator service message
G - OW1 address of the processor that the failed disc belonged to. This address may or may not be the same as field C.
H - Indicates status of disc
I, J, K, L - Spare

Receiving Processor Function - On receipt of this message, the load regulator updates the center configuration tables.

1.3.4.5 *Overload*

<table>
<tr><td>A
8</td><td colspan="2">B
8</td><td>C
8</td><td>D
5</td><td>E
2</td><td>F
1</td></tr>
<tr><td colspan="7">G
64</td></tr>
<tr><td>H
8</td><td colspan="2">I
8</td><td colspan="4">J
16</td></tr>
<tr><td>K
8</td><td>L
6</td><td colspan="5">M
18</td></tr>
<tr><td colspan="7">O
32</td></tr>
</table>

A - Overload, op code - X'95
B - Spare
C - OW1 address of calling processor
D - Pointer to an entry in the queue threshold table or disc threshold table depending on the type of overload
E - Indicator specifying type of overload
B'00 = Disc overload
B'01 = Queue overload (number)
B'10 = Queue overload (time)
B'11 = Queue overload (number and time)
F - Indicator specifying type of threshold reached
B'0 = Minimum threshold reached
B'1 = Maximum threshold reached
G - Protection key for load regulator call service message
H - Zone identifier for the overloaded zone
I - L1 address of disc containing the overloaded zone
J - Spare
K - Zone type overloaded
L - Indicate channel and queue overload
B'100000 A Channel on load
B'010000 A Channel off load
B'001000 B1 Channel on load
B'000100 B1 Channel off load
B'000010 B2 Channel on load
B'000001 B2 Channel off load
M - Spare
O - Spare

1.3.4.6 Regulator Tables Update Return

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="4">O
32</td></tr>
</table>

A - Regulator tables update return, op code - X'96
B - Spare
C - OW1 address of originating processor
D - Double word increment to calling PCI
E, F - Protection keys (AK, K2) of CPSR
G - Zone identifier of CPSR
H - L1 address of CPSR
I - Cell address of CPSR
J, K - Protection key (AK, K2) of new tables file
L - Zone identifier of new tables file
M - L1 address of new tables file
N - Cell address of new tables file
O - Spare

Receiving Processor Function - On receipt of this message, the load regulator replaces the old tables (queue threshold disc threshold, switch, etc.) with the new tables.

1.4 Multiplex Channel Formats

1.4.1 I/O Call 1 Service Message

A 8	B 8	C 8	D 8
E 16		F 16	
G 8	H 8	I 16	
J 16		K 16	
L 8	M 8	N 16	
O 16		P 16	

A - I/O Call 1 service message, op code - X'A0
B - Count of the number of inputs in the calling PCI; zero if from a program
C - OW1 address of originating processor
D - Double word increment to calling PCI; zero if from a program
E, F - Protection keys (AK, K2) of calling CPSR, zero if from a program
G - Zone identifier of CPSR; zero if from a program
H - L1 address of CPSR; zero if from a program
I - Cell address of CPSR; zero if from a program
J, K - Protection key (AK, K2) of message file
L - Zone of message file
M - L1 address of message file
N - Cell address of message file
O - Selection key of subfile if multiplexed file; zero if nonmultiplexed
P - Spare

Receiving Processor Function - The receiving multiplex channel reads the header from the specified message file, validates the header, and attempts routing to all of the addressees contained in the header.

1.4.2 I/O Call 2 Service Message

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td>L
8</td><td>M
8</td><td colspan="2">N
16</td></tr>
<tr><td colspan="2">O
16</td><td colspan="2">P
16</td></tr>
</table>

A - I/O call 2 service message, op code - X'A1
B - Count of the number of inputs in the calling PCI
C - OW1 address of originating processor
D - Double word increment to calling PCI
E, F - Protection key (AK, K2) of CPSR
G - Zone of CPSR
H - L1 address of CPSR
I - Cell address of the CPSR
J, K - Protection keys (AK, K2) of message file
L - Zone of message file
M - L1 address of message file
N - Cell address of message file
O - Selection key of subfile if multiplexed file
P - Spare

Receiving Processor Function - The receiving multiplex channel routes on only those addressees local to its exchange. This type of service message is normally issued by MSP. An example of its use in the case where MSP has received a message, either from an external source or CPS, which indicates multiple address routing. After the receiving processor has actioned his local addressees, and built a dynamic CPSR, I/O call 2 service message may be sent to other exchanges. The other exchanges action only the addressees local to their exchange.

1.4.3 TDM Device Initialization Response

<table>
<tr><td>A
8</td><td colspan="2">B
16</td><td>C
8</td></tr>
<tr><td colspan="2">D
16</td><td colspan="2">E
16</td></tr>
<tr><td>F
8</td><td>G
8</td><td colspan="2">H
16</td></tr>
<tr><td colspan="2">I
16</td><td colspan="2">J
16</td></tr>
<tr><td>K
8</td><td>L
8</td><td colspan="2">M
16</td></tr>
<tr><td colspan="2">N
16</td><td colspan="2">O
16</td></tr>
</table>

A - TDM device initialization, op code - X'A4
B - Spare
C - Double word increment to calling PCI
D, E - Protection key (AK, K2) of CPSR
F - Zone of CPSR
G - L1 address of CPSR
H - Cell address of CPSR
I, J - Protection key (AK, K2) of the output parameter file
K - Zone of output parameter file
L - L1 address of output parameter file
M - Cell address of output parameter file
N, O - Spare

Receiving Processor Function - If the double word increment is not equal to zero, MSP tries to open a working channel. If the opening of the channel is successful, the MSP sends a normal return service message to the CPSR. If the double word increment is equal to zero, the MSP does not respond.

1.5 B Channel Formats

1.5.1 Application Program Call

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td colspan="4">L
32</td></tr>
<tr><td colspan="4">M
32</td></tr>
</table>

A - Application program call, op code - X'CO, X'DO
B - Spare
C - Spare
D - Double word increment to calling PCI
E, F - Protection keys (AK, K2) of CPSR
G - Zone identifier of CPSR
H - L1 address of CPSR
I -Cell address of CPSR
J - Spare
K - Maximum run time in minutes
L, M - Spare

Receiving Processor Function - On receiving this message, the AP initialization function must:

a. Load the CPSR
b. Isolate the input list in the PCI and build a program status record.
c. Isolate the AP address in the PCI and load the program.
d. Initiate the program

1.5.2 Program Call

<table>
<tr><td>A
8</td><td colspan="2">B
16</td><td>C
8</td></tr>
<tr><td colspan="2">D
16</td><td colspan="2">E
16</td></tr>
<tr><td>F
8</td><td>G
8</td><td colspan="2">H
16</td></tr>
<tr><td colspan="2">I
16</td><td colspan="2">J
16</td></tr>
<tr><td colspan="4">K
32</td></tr>
<tr><td colspan="4">L
32</td></tr>
</table>

A - Program call, op code - X'C8, X'D8
B - Truncated C-Number in binary consisting of the (NNN) subscriber portion of the C-Number. All high order bits are set to 1.
C - Double word increment of calling PCI; zero if from a program
D, E - Protection keys (AK, K2) of CPSR or message file. If C = 0, message file.
F - Zone identifier of CPSR or message file. If C = 0, message file.
G - L1 address of CPSR or message file.
H - Cell address of CPSR or message file
I - Spare
J - Program run time
K, L - Spare

Receiving Processor Function - The program call service message may be issued by CPS, MSP, or DACS for the purpose of initiating a service function to be executed in an A or B channel. For CPS or MSP, it is issued as a result of the resolution of a C-Number which indicates this type of service message is required. The truncated C-Number which is placed in field B is that returned in the RTS parameter packet.

The A/B channel initialization function must call upon RTS to resolve the file location of the non-resident service function. The specified program is loaded into the channel for execution. If field C of the service message does not equal zero, no input list is provided.

1.6 A Channel

1.6.1 Application Program Call

<table>
<tr><td>A
8</td><td>B
8</td><td>C
8</td><td>D
8</td></tr>
<tr><td colspan="2">E
16</td><td colspan="2">F
16</td></tr>
<tr><td>G
8</td><td>H
8</td><td colspan="2">I
16</td></tr>
<tr><td colspan="2">J
16</td><td colspan="2">K
16</td></tr>
<tr><td colspan="4">L
32</td></tr>
<tr><td colspan="4">M
32</td></tr>
</table>

A - Application program call, op code - X'E0, X'F0
B - Spare
C - Spare
D - Double word increment to calling PCI
E, F - Protection keys (AK, K2) of CPSR
G - Zone identifier of CPSR
H - L1 address of CPSR
I - Cell address of CPSR
J - Spare
K - Maximum run time in minutes
L, M - Spare

- Receiving Processor Function - On receiving this message, the AP initialization function must:

a. Load the CPSR
b. Isolate the input list in the PCI and build a program status record.
c. Isolate the AP address in the PCI and load the program.
d. Initiate the program.

1.6.2 Program Call

<table>
<tr><td>A
8</td><td colspan="2">B
16</td><td>C
8</td></tr>
<tr><td colspan="2">D
16</td><td colspan="2">E
16</td></tr>
<tr><td>F
8</td><td>G
8</td><td colspan="2">H
16</td></tr>
<tr><td colspan="2">I
16</td><td colspan="2">J
16</td></tr>
<tr><td colspan="4">K
32</td></tr>
<tr><td colspan="4">L
32</td></tr>
</table>

A - Program call, op code - X'E8, X'F8
B - Truncated C-Number in binary consisting of the (NNN) subscriber portion of the C-Number. All high order bits are set to 1.
C - Double word increment of calling PCI; zero if from a program
D, E - Protection keys (AK, K2) of CPSR or message file. If C = 0, message file.
F - Zone identifier of CPSR or message file. If C = 0, message file.
G - L1 address of CPSR or message file.
H - Cell address of CPSR or message file
I - Spare
J - Program run time
K, L - Spare

Receiving Processor Function - The program call service message maybe issued by CPS, MSP or DACS for the purpose of initiating a service function to be executed in an A or B Channel. For CPS or MSP, it is issued as a result of the resolution of a C-Number which indicates this type of service message is required. The truncated C-Number which is placed in field B is that returned in the RTS parameter packet.

The A/B Channel initialization function must call upon RTS to resolve the file location of the non-resident service function. The specified program is loaded into the channel for execution. If the service message field C $\neq$ 0, the A/B channel initialization routines will isolate the input list in the calling PCI and build the PSR. If C=0, no input list is provided.

1.7 Direct Access Message Format

DEVICE COMMAND WORD

8	B 16	C 8

A - Receiving Processor OW1 address
B - Spare
C - DCM table displacement (nonzero for direct access service message, zero for normal service message)

Direct access to tables in a processor core is provided via OW1 for reading and updating tables of interest to the load regulator.

Each processor has in its core a set of prebuilt DCMs and DCW lists which reference the tables of interest. In general, a pair of DCMs and DCWs (one for reading, one for updating) is required for each table to which the load regulator has access. A DCM address table containing the addresses of the prebuilt DCMs is constructed in a fixed location in core and its address wired in the CSU. Any table can be read or updated by selecting the appropriate DCM address from the DCM address table.

2. DEVICE CONTROL MESSAGE DCM (DCM) FORMAT

<table>
<tr><td>A
SQ
1</td><td>B
IC
1</td><td>C
CU
1</td><td>D
IT
1</td><td>E
Retry
4</td><td>F
ER
1</td><td colspan="2">G
Op Code
7</td><td>H
DCM Chain Address
16</td></tr>
<tr><td colspan="5">I
L1 TDX Address
8</td><td colspan="3">J
L2 TDX Address
8</td><td>K
DCW Chain Address
16</td></tr>
<tr><td colspan="8">L
From Program Address
16</td><td>M
Response Address for Device Status
16</td></tr>
<tr><td>N
EOL
1</td><td>P
CH
1</td><td>R
SK
1</td><td>S
RW
1</td><td>T
RB
1</td><td>U
SP
2</td><td colspan="2">V
Count
9</td><td>W
Core Address
16</td></tr>
<tr><td colspan="9">(Additional DCW's or DCW Chain as Required)</td></tr>
</table>

Field	Name	Description
A	SQ	Complete indicator: If 0, service is required by the CSU. The CSU sets this bit to 1 upon completion of the DCM or DCM immediate chain.
B	IC	Immediate chain. The SQ and IC indicators are interpreted by the CSU as follows.

SQ	IC	
0	0	Process DCM, and chain to next DCM unless ER = 1
0	1	Chain immediately. Do not process this DCM but set SQ bit to 1.
1	0	Normal completion. The CSU interprets this condition as an idle condition.
1	1	Completion of immediate chain operation. The CSU interprets this condition as an idle condition.

Field	Name	Description
C	CU	Connector update indicator as set and read by the responsible software routines upon recognition of successful completion of DCM.
D	IT	Indirect transfer indicator as set and read by the ALCU programs responsible for the creation and recovery processes of the DCM. 0 indicates the transaction is a result of a direct transfer request and 1 indicates an indirect transfer request.
E	Retry	The retry count as set and read by the ALCU programs responsible for the creation and recovery processes of the DCM.

2. (Cont)

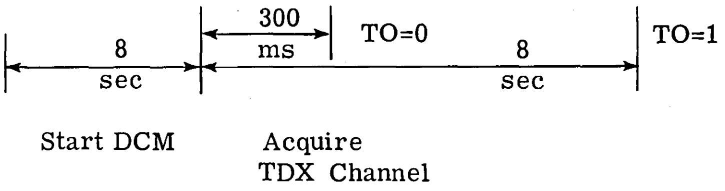
Field Name Description

F ER Error indicator. If this bit is set to 1, an error was detected during execution of this DCM. If the bit is set to 0, the DCM was processed without error. When an error occurs, the CSU inhibits chaining and alternately monitors the SQ bit of the DCM in error and the other DCM chain for the data channel. Reset of SQ by the program causes the CSU to proceed as normal on the DCM chain.

G Op Code Interpreted by the CSU data channels

TO	SPARE	CODE
1	3	3

TO Time out indicator for the CSU operation once the desire TDX channel is acquired. If 0, then the operation must complete within 300 ms. If 1, the operation must complete within 8 seconds. If the desired TDX channel cannot be acquired within 8 seconds, then CSU timeout on the DCM occurs. For either type of CSU timeout, the SQ and ER bits in the DCM header are set and DCM chaining does not occur. The CSW is stored indicating the type of timeout.



Code 000 - Normal Process DCM
001 - Set OW2 mode } B channel
010 - Set data channel mode }
011 - Invalid code
100 - Invalid code
101 - Set party line address } OW1 channel
110 - Read party line address }
111 - Invalid code

H *DCM chain address. Core address of the next DCM.

I L1 Address of TDX channel

J L2 Address of TDX channel

K *DCW chain address. A DCW chain address is contained in this field if the content of this field is not equal to zero. This DCM chain address may be used to point to the list of DCWs after the last DCW appended to the DCM (i.e., after the DCW with an EOL bit set to 1 and CH bit set to 0). See definition of CH for further definition of DCW chain.

*18 bit byte address, right justified by two = word address.

2. (Cont)

Field Name Description

L *From program address. This address specifies the location of the error program or routine to be used if the CSU detects an error during processing of the DCM. If the ER bit is set to 1, the completion routine uses this address to initiate the proper error recovery program at the beginning of the time frame of the responsible program channel.

M *Response address for device status and CSU status. Device status is stored at this core address. Channel status is stored in the word location following the device status.

N EOL End-of-list. If this bit is set to 1, this is the last DCW or DCW chain in this list. If set to 0, another DCW or DCW chain follows. This bit is further defined under the CH bit definition.

P CH Chain. This bit is used to indicate whether or not the word contains a DCW chain address. If this bit is set to 0, the word is a DCW and contains the core address of data. The EOL and CH bits are defined as shown below:

EOL	CH	
0	1	DCW chain. DCW list follows; use DCW chain address for next DCW list after last DCW in the current list.
0	0	DCW. DCW or DCW chain follows.
1	1	DCW chain. No DCW list follows; chain immediately to core address in DCW chain for next DCW or DCW chain.
1	0	DCW. Last DCW in list; if DCW chain was previously available, chain to this address for next DCW or DCW chain. If no DCW chain address is available, operation for this DCM is complete.

EOL, CH, and core address are the only fields of the DCW which are used if CH is set to 1, indicating a DCW chain.

R SK Skip indicator. If this bit is set to 0, data is read from core and sent to the device or is received from a device and written into core. If this bit is set to 1, the following occurs:

a. If the RW bit is set to 1, the CSU sends 32-bit words of all zeros to the device. The CSU sends the number of words specified in the count field plus one. The MCS address of the DCW is not used and no memory accesses are required to send the 32-bit words of all zeros.

b. If the RW bit is set to 0, the CSU discards data received from the device for the number of words specified in the count field plus one.

S RW DCW read/write indicator. This bit specifies the direction of data transfer. If this bit is set to 1, data is transmitted from core to the device. If this bit is set to 0, data is received from the device and written into core.

*18-bit byte address, right justified by two = word address.

2. (Cont)

Field	Name	Description
T	RB	Read backward. If this bit is set to 1, the core address specified in the DCW is decremented each time a word is read or written into core. Data is thus stored (loaded) in descending sequential locations. If this bit is set to 0, the core address specified in the DCW is incremented each time a word is written to core or read from core.
U		Spare
V	Count	DCW word count. This field specifies the number of sequential words to be transferred, beginning at the location specified by the core address specified in the DCW. This count specifies from 1 to 512 words to be transferred. A count of zero results in transferring one word. A count of 511 transfers 512 words.
		Normal completion of a DCW returns a residual count of 0 in the channel status with no count error set in the channel status. Failure to transfer the last word results in a count of zero and the count error bit set. Setting of only the CE (count error) bit in the channel status does not result in setting the error bit (field F) of the DCM.
W	Core Address	*Core address of data or device command. This address normally specifies the core location of the device command. On other DCWs, this field specifies the core address of data to be transferred.

*18 bit byte address, right justified by two = word addresses.

3. OPERATIONS CONTROL TABLE FORMAT

The operations control table provides an interface mechanism between the operations control program and the processor internal channels. Processor internal and external channel status information is maintained in entries of this table as shown below.

<table>
<tr><td>A1
1</td><td>A2
1</td><td>A3
1</td><td>A4
1</td><td>A5
1</td><td>A6
1</td><td>A7
2</td><td>A8
1</td><td>A9
2</td><td>A10
1</td><td>A11
1</td><td>A12
1</td><td colspan="2">B
CHANNEL SAVE AREA ADDRESS 18</td></tr>
<tr><td colspan="12">C
QUEUE 1 CELL ADDRESS 16</td><td>D
QUEUE 1 NRP 8</td><td>E
QUEUE 1 NWP 8</td></tr>
<tr><td colspan="12">F
QUEUE 2 CELL ADDRESS 16</td><td>G
QUEUE 2 NRP 8</td><td>H
QUEUE 2 NWP 8</td></tr>
<tr><td colspan="12">I
DCM 1 NRP</td><td colspan="2">J
DCM 2 NRP</td></tr>
<tr><td colspan="12">K
DCM 1 NWP</td><td colspan="2">L
DCM 1 NAC</td></tr>
<tr><td colspan="12">M
DCM 2 NWP</td><td colspan="2">N
DCM 2 NAC</td></tr>
<tr><td colspan="12">O
CHANNEL LOWER LIMIT ADDRESS</td><td colspan="2">P
CHANNEL UPPER LIMIT ADDRESS</td></tr>
<tr><td colspan="12">Q
TRAP ENTRY POINT</td><td colspan="2">R
SPARE</td></tr>
</table>

Word	Bit(s)	Field	Description
0	0	A1	I/O error indicator, where 0 = no 1 = yes
0	1	A2	Channel idle indicator, where 0 = no 1 = yes
0	2	A3	Channel busy/checkpoint indicator, where 0 = checkpoint 1 = busy
0	3	A4	Queue 1 enable indicator, where 0 = enable 1 = inhibit
0	4	A5	Queue 2 enable indicator, where 0 = enable 1 = inhibit
0	5	A6	Service queue indicator, where 0 = service queue 1 1 = service queue 2
0	6-7	A7	Reserved spares

3. (Cont)

Word	Bit(s)	Field	Description
0	8	A8	Service DCM indicator, where 0 = service DCM 2 1 = service DCM 1
0	9-10	A9	Reserved spares
0	11	A10	DCM 1 error indicator, where 0 = no 1 = yes
0	12	A11	DCM 2 error indicator, where 0 = no 1 = yes
0	14-31	B	Channel save area address
1	0-15	C	Queue 1 address
1	16-23	D	Queue 1 Next-Read-Position (NRP)
1	24-31	E	Queue 1 Next-Write-Position (NWP)
2	0-15	F	Queue 2 address
2	16-23	G	Queue 2 NRP
2	24-31	H	Queue 2 NWP

4. PROCESSOR INTERFACE TABLE FORMAT

BYTE	FUNCTION			
00-0F	Spare			
10-1F	Spare			
20-2F	Spare			
30-3F	Spare			
40-4F	A	B	C	D
50-5F	D			
60-6F	D			
70-7F	D			
80-8F	Spare			
90-9F	Spare			
A0-AF	E	F	G	H
B0-BF	I	J	K	L
C0-CF	M	N	Spare	
D0-DF	O	P	Q	R
E0-EF	Spare			
F0-FF	Spare		S	T

Field	Byte (X')	Description
A	40-43	Processor control word
B	44-47	Processor status word
C	48-4B	Absolute time clock
D	4C-7F	Working channel status
E	A0-A3	New IAC for program interrupt
F	A4-A7	Condition code, overflow indicator, and current IAC for program interrupt
G	A8-AB	New IAC for MCS parity interrupt

4. (Cont)

Field	Byte (X')	Description
H	AC-AF	Condition code, overflow indicator, and current IAC for MCS parity interrupt
I	B0-B3	New IAC for time interrupts, IPL, and INIT
J	B4-B7	Condition code, overflow indicator, and current IAC for time interrupts, IPL, and INIT
K	B8-BB	New IAC for branch return link to protected area entry
L	BC-BF	Condition code, overflow indicator, and current IAC for branch return link to protected area entry
M	C0-C3	Accumulator B
N	C0-C7	Accumulator A
O	D0-D3	Trapping mechanism new IAC
P	D4-D7	Trapping mechanism effective address
Q	D8-DB	Trapping mechanism function word
R	DC-DF	Trapping mechanism condition code, overflow indicator, and current IAC
S	F8-FB	(OPSTAC) core address of the current entry in the operations control table
T	FC-FF	(OPSLOW) starting address of the current channel's program status record

5. CHANNEL SAVE AREA FORMAT

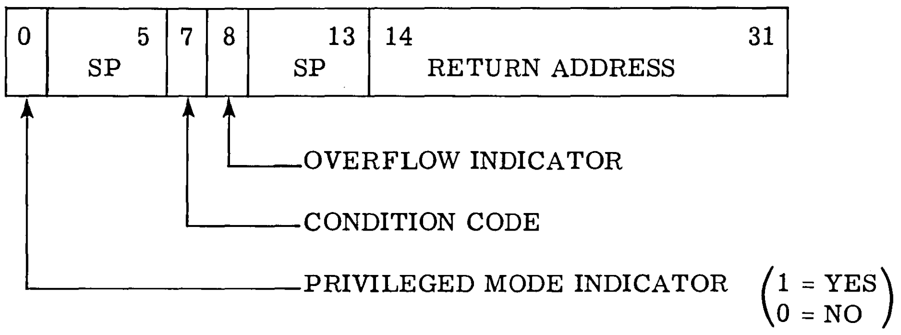
C-8561

A REGISTER	
B REGISTER	
RESERVED CORE	1
RESERVED CORE	2
RESERVED CORE	3
INDEX REGISTER	1
INDEX REGISTER	2
INDEX REGISTER	3
RESERVED CORE	4
RESERVED CORE	5
RESERVED CORE	6
RESERVED CORE	7
RESERVED CORE	8
RESERVED CORE	9
RESERVED CORE	10
RESERVED CORE	11
PROGRAM STATUS WORD	

C-8563

GENERAL REGISTER	0
GENERAL REGISTER	1
GENERAL REGISTER	2
GENERAL REGISTER	3
GENERAL REGISTER	4
GENERAL REGISTER	5
GENERAL REGISTER	6
GENERAL REGISTER	7
GENERAL REGISTER	8
GENERAL REGISTER	9
GENERAL REGISTER	10
GENERAL REGISTER	11
GENERAL REGISTER	12
GENERAL REGISTER	13
GENERAL REGISTER	14
GENERAL REGISTER	15
PROGRAM STATUS WORD	

PROGRAM STATUS WORD



program

6. PROCESSOR STATUS RECORD

Program Status Record (PSR)		Offset	Contents	Size
Program Status Record (PSR)	All Channels	X'0	REGISTER 1: CHANNEL DISCONNECT REGISTER A	16 Words Channel Status Registers
		4	REGISTER 2: CHANNEL DISCONNECT REGISTER B	
		8	REGISTER 3: BRLP LINKAGE SAVE INDEX 1	
		C	REGISTER 4: BRLP LINKAGE SAVE INDEX 2	
		10	REGISTER 5: BRLP LINKAGE SAVE IAC	
		14	REGISTER 6: CHANNEL DISCONNECT INDEX 1	
		18	REGISTER 7: CHANNEL DISCONNECT INDEX 2	
		1C	REGISTER 8: CHANNEL DISCONNECT INDEX 3	
		20	REGISTERS 9-16: CHANNEL WORK REGISTERS FOR BRLP LINKED COMMON FUNCTIONS	
		40	CHANNEL DISCONNECT AND COMPLETION STATUS	2 Words
		48	FILE TRANSFER STATUS	2 Words
	A & B Channels Only	50	AP CALL SERVICE MESSAGE	3 Words
		5C	FORMAL OUTPUT FILE IDENTIFIER	2 Words
		64	CHANNEL INITIALIZATION LOAD PACKET AS REQUIRED FOR RELOCATABLE LOADER	4 Words
		74	PCI INPUT LIST: MAXIMUM NUMBER DECLARED AT ASSEMBLY TIME	2 Words Per Input
			FILE STATUS POINTERS: MAXIMUM NUMBER DECLARED AT ASSEMBLY TIME	1 Word Per FLB

7. MULTIPLEX STATUS RECORD FORMAT

WD 0	A 2	B 4	C 26						
WD 1	D 32								
WD 2	E 2	F 4	G 26						
WD 3	H 1	I 2	J 2	K 2	L 1	M 1	N 3	O 4	P 16

Field	Word	Bits	Description
A	0	0-1	Supervisory field set and read by the MSU hardware and MSP for the purpose of establishing channel supervision as follows: 00 MSU is in control 01 Program is in control 10 Device is in control 11 MSU is in control
B	0	2-5	The content of this field specifies the current channel operation through use of one of the following operation codes.

Word Operations

0100 (STD) Store input data in D-word

0110 (STX) Store non-zero input data in D word

0101 (STT) Store input data in D word if outside limit test

1100 (LDD) Load output data from D word

1110 (LDX) Load output data from D word if input data is nonzero

1101 (LDT) Load output data from D word if input data is outside limits

Field Operations

0111 (FST) Store input data at (word) address counter location

1111 (FLD) Load output data from (word) address counter location

0011 (FSL) Same as FST but perform link operation when input buffer is filled

1011 (FLL) Same as FLD but perform link operation when output buffer is emptied.

7. (Cont)

Field Word Bits Description

Miscellaneous/Special Operations

0000 (NOP) Specifies no operation (i.e., channel idle state)

1010 (FLL) Pseudo operation code used during FLL operation when program is in control (MFS=01)

Note

The value of the B field is established by the program and specifies the program to MSU instruction. This instruction controls data transfers but may be overridden by certain device to MSU instructions (BSR, BSQ).

C 0 6-31 The contents of this field are determined by the B field value as shown below:

Word Operations

STD, STX, LDD, LDX:

IMPLIED OPERAND 26

STT, LDT:

LL 13	UL 13

LL -Unsigned value used as a lower limit test against input data word.

UL -Unsigned value used as the upper limit test against input data word.

Field Operations

FST, FSL:

START ADDRESS 10	ADDRESS COUNTER 16

START ADDRESS - Right most 10 bits of initial address counter value.

ADDRESS COUNTER - Initially set to the beginning word (16-bit) address where input data is to be stored in core. The counter is incremented by 1 each time the MSU stores a word (except for the final input buffer word).

7. (Cont)

FLD, FLL:

END ADDRESS 10	ADDRESS COUNTER 16

END ADDRESS - The right most 10 bits of the final word (16-bit) address from which data may be loaded (from this output buffer).

ADDRESS COUNTER - Initially set to the beginning (output buffer) word address from which data is to be loaded. The counter is incremented by 1 as each word is loaded from the buffer (except for the final buffer word).

Miscellaneous/Special Operations

NOP:

SPARE 18	SPARE/ OW1 8

SPARE-OW1 - Configuration required following the occurrence of the identification sequence prior to returning the channel to device control (MFS=10).

SPARE-SPARE - Allowed configuration once it is known that the device has accessed its OW1 value and may be used by the program to retain any desired operand(s).

OW1 - The OW1 address of the current (this) processor.

Field	Word	Bits	Description
D	1	0-31	Field is referred to as the D word or data register. This word is used to receive input data words or to provide output data words to the device during word operations. The D word is also utilized as a device to program instruction register for receiving the subroutine call parameters explicitly issued by the device in conjunction with device to MSU commands (BSR, BSQ). The D word content is also interpreted as an implicit subroutine call parameter during non-link field operations (FST, FLD).
E	2	0-1	During link operations (FSL, FLL) this field is exchanged with field A at the time the linkage occurs. Supervisory codes similar to those of the A field are contained here. For non-link operations see note.
F	2	2-5	During link operations this field is exchanged with field B at the time linkage occurs and should contain the same operation code. For non-link operations see note.

7. (Cont)

Field	Word	Bits	Description
G	2	6-31	During link operations this field is exchanged with field C at the time linkage occurs. The contents of this field are similar to those described for field C during link operations and pertain to the next core bin from which data is to be obtained or to which data is stored. For non-link operations see note. **Note** Fields E, F, and G are applicable to link (FSL, FLL) and to word operations (STD, STX, STT, LDD, LDX, and LDT). The field contents during link operations have been previously described. During word operations the three fields are combined to form a subroutine call parameter word.
H	3	0	This bit indicates to the MSU that the output queue associated with this working channel contains an active entry.
I	3	1-2	Bit 1 - If set to 1, one or more sequential timeouts have occurred. Bit 2 - If set to 1, working channel is active.
J	3	3-4	Denotes the class of control the device employs. Bit 3 Bit 4 0 0 Fully controlled 0 1 Semi-controlled (Message header edit can be performed as message setments complete.) 1 0 Spare 1 1 Semi-controlled (Message header edit must be deferred until after message is delivered to the file.)
K	3	5-6	This is a counter indicating the number of output queue priorities associated with this MSR.
L	3	7	This indicator specifies if implicit calls are used by the associated device.
M	3	8	This field indicates whether the device operates in link or nonlink modes during field operations.
N	3	9-11	The operating characteristics of the associated device are specified by this field as follows: 000 - Not used 001 - Simplex load 010 - Simplex store 011 - Half duplex 100 - Full duplex, the associated channel is next adjacent

7. (Cont)

Field	Word	Bits	Description
			101 - Full duplex, the associated channel is previous adjacent 110 - Full duplex, the associated channel is a secondary working channel. This state is required only when expansion to 512 TDM channels is considered 111 - Not used
O	3	12-15	This field is used by MSP to select the correct decode logic for interpretation of the subroutine call parameters associated with the MSR. Field interpretations are as follows: 0000 - Binary decode logic only. 0100 - Binary or ASCII decode logic is used as determined by the most significant bit of the subroutine call parameter. If this bit is 0 (zero) binary decode logic is used. 1000 - ASCII decode logic only 1100 - Direct linkage is required. MSP performs its initial MSR error detection functions to verify the validity of a subroutine call but MSP will not interpret the subroutine call parameter. In this mode MSP transfers control to a subroutine specified in the channel's subroutine directory record. The subroutine so specified is responsible for the decode of the MSR subroutine call parameters.
P	3	16-31	This field contains the core word address of the channel status record associated with this MSR.