Critical Release Notice

Publication number: 297-1001-305
Publication release: Standard 04.05

The content of this customer NTP supports the SN06 (DMS) and ISN06 (TDM) software releases.

Bookmarks used in this NTP highlight the changes between the baseline NTP and the current release. The bookmarks provided are color-coded to identify release-specific content changes. NTP volumes that do not contain bookmarks indicate that the baseline NTP remains unchanged and is valid for the current release.

Bookmark Color Legend

Black: Applies to new or modified content for the baseline NTP that is valid through the current release.

Red: Applies to new or modified content for NA017/ISN04 (TDM) that is valid through the current release.

Blue: Applies to new or modified content for NA018 (SN05 DMS)/ISN05 (TDM) that is valid through the current release.

Green: Applies to new or modified content for SN06 (DMS)/ISN06 (TDM) that is valid through the current release.

Attention!
Adobe® Acrobat® Reader™ 5.0 is required to view bookmarks in color.

Publication History

March 2004

Standard release 04.05 for software release SN06 (DMS) and ISN06 (TDM).

Change of phone number from 1-800-684-2273 to 1-877-662-5669, Option 4 + 1.

297-1001-305

DMS-100 Family

DMS100 Family

Memory Administration Guide

NA004 and up   Standard 04.04   August 1999

NORTEL
NORTHERN TELECOM

DMS-100 Family

DMS100 Family

Memory Administration Guide

Publication number: 297-1001-305
Product release: NA004 and up
Document release: Standard 04.04
Date: August 1999

Printed in the United States of America

NORTHERN TELECOM CONFIDENTIAL: The information contained in this document is the property of Northern Telecom. Except as specifically authorized in writing by Northern Telecom, the holder of this document shall keep the information contained herein confidential and shall protect same in whole or in part from disclosure and dissemination to third parties and use same for evaluation, operation, and maintenance purposes only.

Information is subject to change without notice. Northern Telecom reserves the right to make changes in design or components as progress in engineering and manufacturing may warrant.
This equipment has been tested and found to comply with the limits for a Class A digital device pursuant to Part 15 of the FCC Rules, and the radio interference regulations of Industry Canada. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at the user’s own expense.
The SL-100 system is certified by the Canadian Standards Association (CSA) with the Nationally Recognized Testing Laboratory (NRTL).
This equipment is capable of providing users with access to interstate providers of operator services through the use of equal access codes. Modifications by aggregators to alter these capabilities is a violation of the Telephone Operator Consumer Service Improvement Act of 1990 and Part 68 of the FCC Rules.

Contents

Using operational measurements 3-1

Memory tracking work sheets 4-1

About this document

How to check the version and issue of this document

The version and issue of the document are indicated by numbers, for example, 01.01.

The first two digits indicate the version. The version number increases each time the document is updated to support a new software release. For example, the first release of a document is 01.01. In the *next* software release cycle, the first release of the same document is 02.01.

The second two digits indicate the issue. The issue number increases each time the document is revised but rereleased in the *same* software release cycle. For example, the second release of a document in the same software release cycle is 01.02.

To determine which version of this document applies to the software in your office and how documentation for your product is organized, check the release information in *Product Documentation Directory*, 297-8991-001.

References in this document

The following documents are referred to in this document:

- *Basic Administration Procedures*
- *Basic Translations Tools Guide*
- *Office Parameters Reference*
- *Operational Measurements Reference Manual*

What precautionary messages mean

The types of precautionary messages used in NT documents include attention boxes and danger, warning, and caution messages.

An attention box identifies information that is necessary for the proper performance of a procedure or task or the correct interpretation of information or data. Danger, warning, and caution messages indicate possible risks.

Examples of the precautionary messages follow.

ATTENTION Information needed to perform a task

ATTENTION

If the unused DS-3 ports are not deprovisioned before a DS-1/VT Mapper is installed, the DS-1 traffic will not be carried through the DS-1/VT Mapper, even though the DS-1/VT Mapper is properly provisioned.

DANGER Possibility of personal injury

DANGER

Risk of electrocution

Do not open the front panel of the inverter unless fuses F1, F2, and F3 have been removed. The inverter contains high-voltage lines. Until the fuses are removed, the high-voltage lines are active, and you risk being electrocuted.

WARNING Possibility of equipment damage

WARNING

Damage to the backplane connector pins

Align the card before seating it, to avoid bending the backplane connector pins. Use light thumb pressure to align the card with the connectors. Next, use the levers on the card to seat the card into the connectors.

CAUTION Possibility of service interruption or degradation

CAUTION

Possible loss of service

Before continuing, confirm that you are removing the card from the inactive unit of the peripheral module. Subscriber service will be lost if you remove a card from the active unit.

How commands, parameters, and responses are represented

Commands, parameters, and responses in this document conform to the following conventions.

Input prompt (>)

An input prompt (>) indicates that the information that follows is a command:

>BSY

Commands and fixed parameters

Commands and fixed parameters that are entered at a MAP terminal are shown in uppercase letters:

>BSY CTRL

Variables

Variables are shown in lowercase letters:

>BSY CTRL ctrl_no

The letters or numbers that the variable represents must be entered. Each variable is explained in a list that follows the command string.

Responses

Responses correspond to the MAP display and are shown in a different type:

```
FP 3 Busy CTRL 0: Command request has been submitted.
FP 3 Busy CTRL 0: Command passed.
```

The following excerpt from a procedure shows the command syntax used in this document:

1 Manually busy the CTRL on the inactive plane by typing

>BSY CTRL ctrl_no
and pressing the Enter key.

where

ctrl_no is the number of the CTRL (0 or 1)

Example of a MAP response:

```
FP 3 Busy CTRL 0: Command request has been submitted.
FP 3 Busy CTRL 0: Command passed.
```

Understanding memory

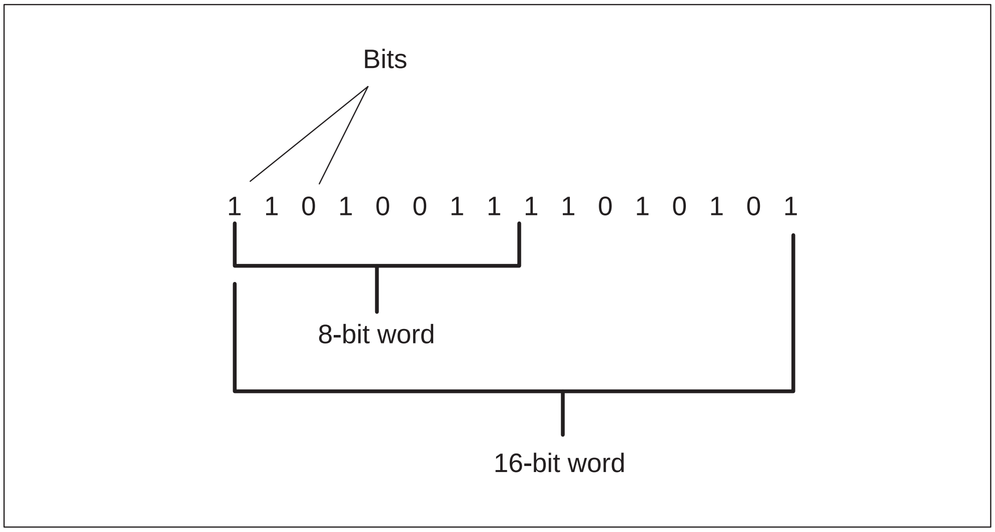
Introduction

This chapter presents a brief description of the memory devices used in the DMS-100 Family of switches. The information provided applies to the DMS-100 NT40 central processor and the DMS SuperNode.

Memory description

Memory, in this document, is a device used to store information in a form that enables the information to be easily and rapidly accessed when needed. In the DMS-100 Family, the memory device most often used is the memory circuit pack or memory card where the information is encoded and stored electronically. This electronic storage is in the form of bits of information. A bit is a binary digit of 0 or 1. Bits are grouped together to form larger units called bytes or words. Each word represents a unit of information. Words may vary in length with bits forming 8-, 16-, and 32-bit words or larger. In the SuperNode system, the memory is organized into 40-bit words: 32 data bits, 7 error-correcting code bits, and 1 parity bit. Refer to the following figure, which illustrates an 8-bit and a 16-bit word.

Figure 1-1
Example of bits and words



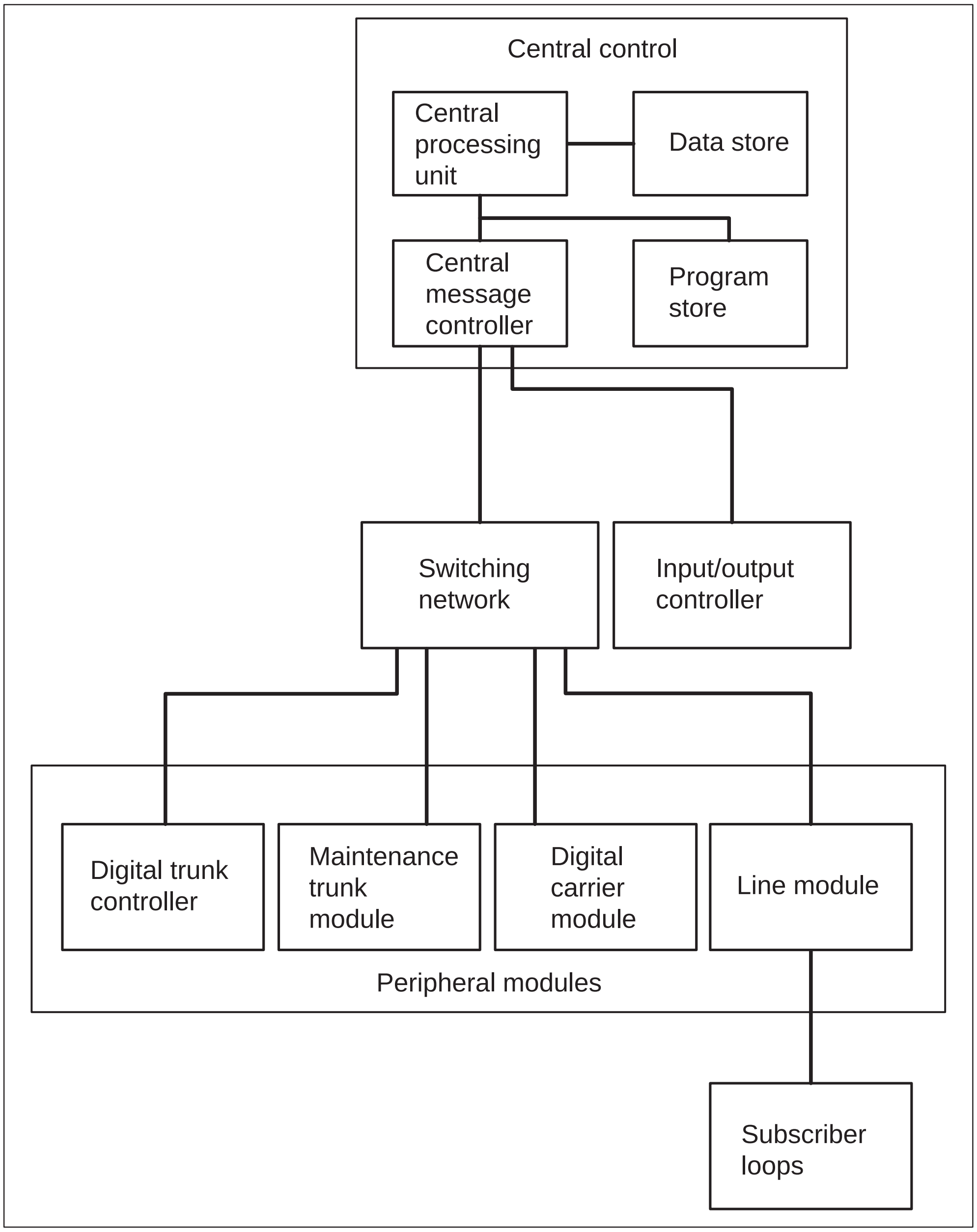
Memory is measured in terms of the number of words or bytes that the device can store. Cards currently vary in size from 64K (K=1024) words of memory to 1 M-word (1 million words) for data store and program store in the NT40 and 6 MBytes to 24 MBytes in the DMS SuperNode configuration. These sizes will probably change as the storage device technology evolves. The conversion of K-words to M-words to Mbytes is accomplished by dividing the K-words by the number 1024 to get M-words and then multiplying the M-words by 2 to get Mbytes.

The bits, bytes, and words stored in memory make up the instructions, addresses, and other necessary data to operate the switch. The DMS-100 central processing unit (CPU) controls and coordinates the operation of the peripheral modules, refer to the following figure. Multitasking in real time is based upon processes that use messages to communicate with one another. These messages can be exchanged by the processes within the CPU, or they can be exchanged by the CPU processes and those in the peripheral modules.

In addition to the software necessary to provide the basic call processing decision functions, extensive software for the administration and maintenance of the DMS-100 hardware and connecting facilities is included. There are several thousand software modules consisting of several million lines of program code. To manage the design and production of reliable software on this scale, the system must have programming in a high level language. The language used in the CPU is procedure oriented type enforcing language (PROTEL), a high level language developed by Bell-Northern Research (BNR).

NT40 central control complex

Figure 1-2
Functional view of DMS-100 first generation

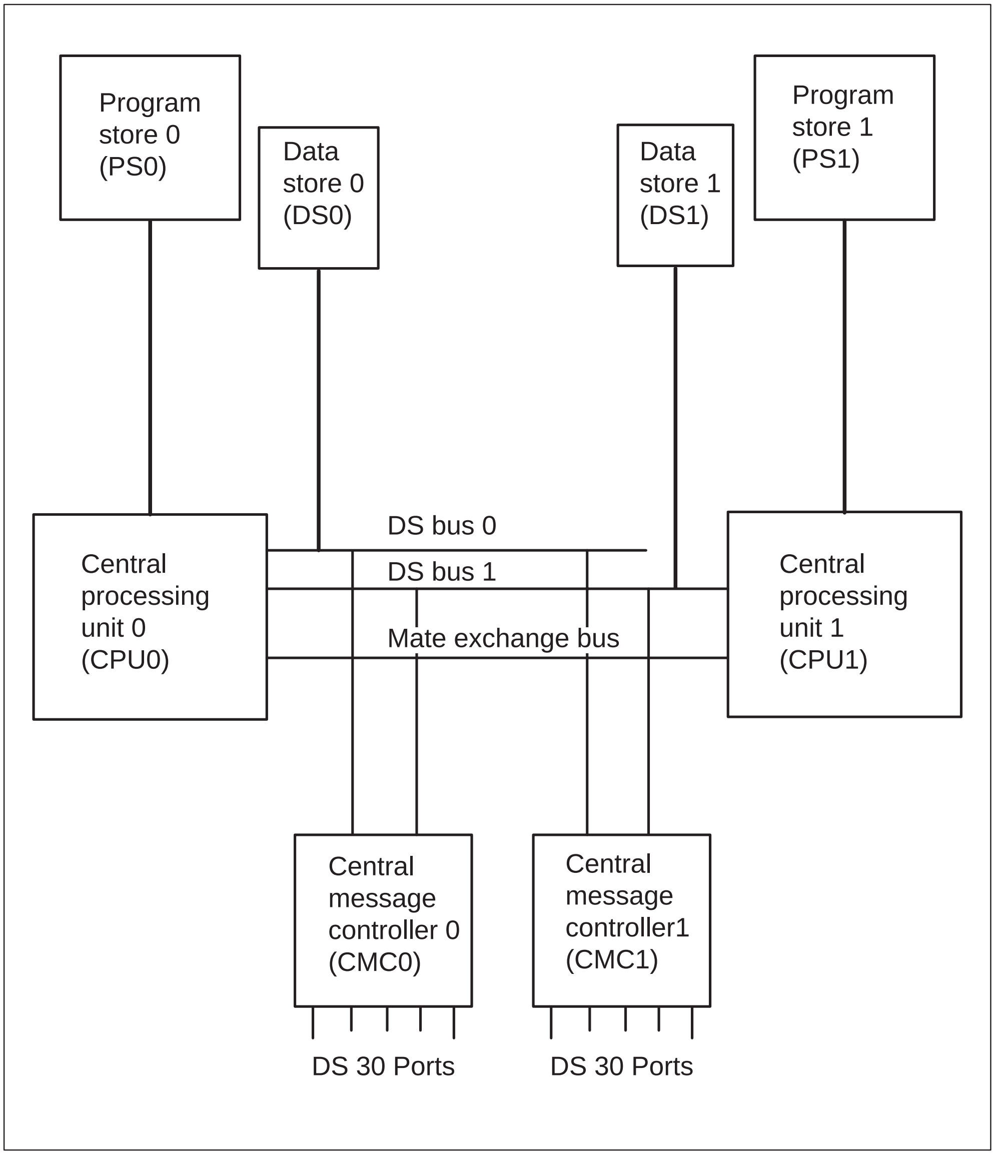


The NT40 central control complex (CCC) is a totally duplicated group of four modules which act together to evaluate incoming messages, formulate proper responses, and issue instructions to the subsidiary units, see the following figure.

The four modules are:

- central message controller (CMC)
- central processor unit (CPU)
- data store (DS)
- program store (PS).

Figure 1-3
Duplicated central control complex


Program store 0 (PS0)
Data store 0 (DS0)
Data store 1 (DS1)
Program store 1 (PS1)
DS bus 0
DS bus 1
Mate exchange bus
Central processing unit 0 (CPU0)
Central processing unit 1 (CPU1)
Central message controller 0 (CMC0)
Central message controller1 (CMC1)
DS 30 Ports
DS 30 Ports

Duplication of the four CCC components offers hardware fault protection, and the ability to carry out office extensions and software updates without disrupting service.

The central processing unit, with associated program store and data store, is duplicated. Under normal circumstances, the two units operate in synchronism. Both are simultaneously executing the same instruction with the same data. Each has access to certain information in the mate CPU; therefore, fault detection for example, matching for loss of synchronism, can be carried out. In addition, this access provides interprocessor communication for system-maintenance software.

The CMC is duplicated, and the two components operate in load-sharing mode. That is, messages are routed alternately through the two CMC units.

CPU memory description

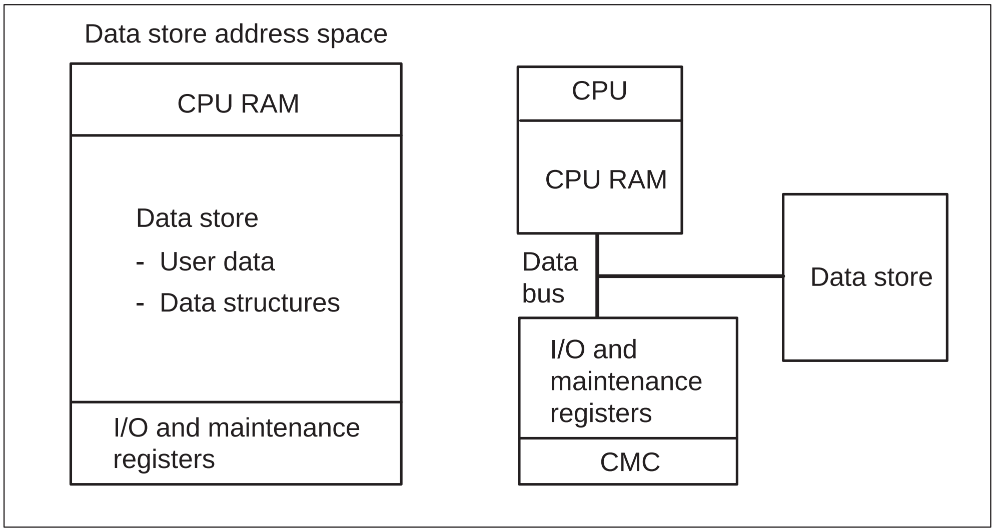
Memory within the CPU is divided into two distinct elements, each of which is independently addressable. These two elements are referred to as program store and data store.

Data store is divided into the following three regions:

- The first region is a random access memory (RAM) area known as CPU RAM. This region contains various registers (for example, base registers), and other memory locations reserved for special purposes. The CPU RAM is located on the CPU card, and, due to the type of memory device, it is faster to access than the other two regions.
- The second (and largest) region is available for data. This region is known as data store.
- The third region contains the input/output (I/O) and maintenance registers, which perform the following operations:
  - Sending control information to the peripheral modules
  - Storing control information for the CC

These three regions are physically located in different parts of the DMS-100 but are on the same data bus and are accessed in the same manner. An address space distinguishes each of the three parts. The CPU random access memory (RAM) is located on the CPU card; data store is located on separate cards; and the I/O and maintenance registers are located in the CMC and in the CPU. The three regions of the data store are illustrated in the following figure.

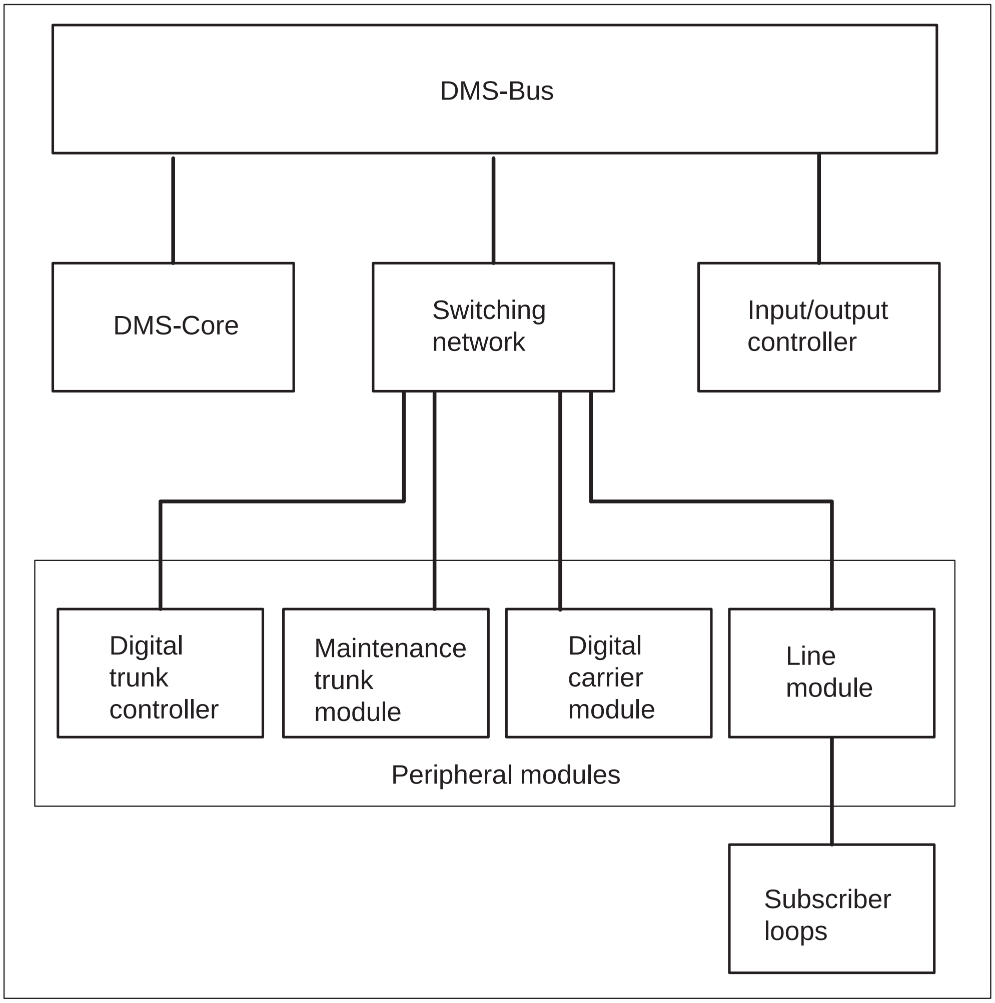
Figure 1-4
The three regions of data store



DMS SuperNode

The DMS-Core is the control component of SuperNode and consists of a computing module (CM) and a system load module (SLM). The computing module contains duplicated DMS SuperNode central processing units (DMS SuperNode CPU).

Figure 1-5
Functional view of DMS-100 second generation (DMS SuperNode)



DMS SuperNode memory

DMS SuperNode memory consists of memory circuit packs. Identical memory circuit packs are used, and memory is organized and addressed in the same manner in the computing module and in the message switch.

The memory consists of integrated program store and data store on the same bus. There are separate data and address buses, each 32 bits wide. Memory is byte addressable; thus, the logical address range is 4 gigabytes.

Data store

Data store hardware

NT40

The NT40 data store shelf has 16 card slots available for the active data store plus one card slot used as a memory controller. This controller card has 256K words of memory to be used as replacement memory for detected single memory faults in any of the memory cards. Memory cards come in different sizes. For example, the NT3X93 (256K words) and the NT4X80 (1 million words).

This shelf has a maximum capacity of 15 to 15.75 K-words of addressable storage depending upon the central control complex vintage. The rest is reserved for system use, for example, write-protected registers.

Another memory card, which may be present in existing NT40 switches, is the NTX3X40. This card contains 64K words of memory.

DMS SuperNode

Data store and program store are integrated. Memory is provided on memory circuit packs, and there are two types:

- 4 megabytes of dynamic random access memory (DRAM), divided into two 2-megabyte memory modules
- 6 megabytes of DRAM, divided into three 2-megabyte memory modules

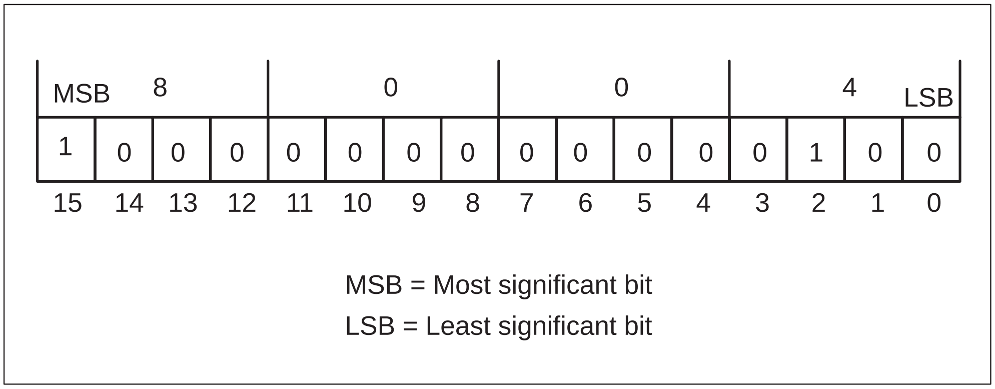
Addressing

To determine where information resides in the NT40, the DS uses an address consisting of an 8-bit page and a 16-bit offset. Each page of DS contains 64K (65,536) words. For example, #FB12A4 is an address on page #FB with an offset of #12A4 words from the beginning of the page.

The DMS SuperNode uses an address consisting of a 16-bit page and a 16-bit offset. Each page of DS contains 16M words.

Bits within a word of data store are addressed by a number from 0–15, from the least significant bit to the most significant bit. The least significant bit is the bit at the low address end of the word. For example, number 8004 would be represented by setting all of the bits within a word to zero except for bits 15 and 2, which are set to one. The numbering of bits within a word (for example, 8004) is illustrated in the following figure.

Figure 1-6
Bit numbering



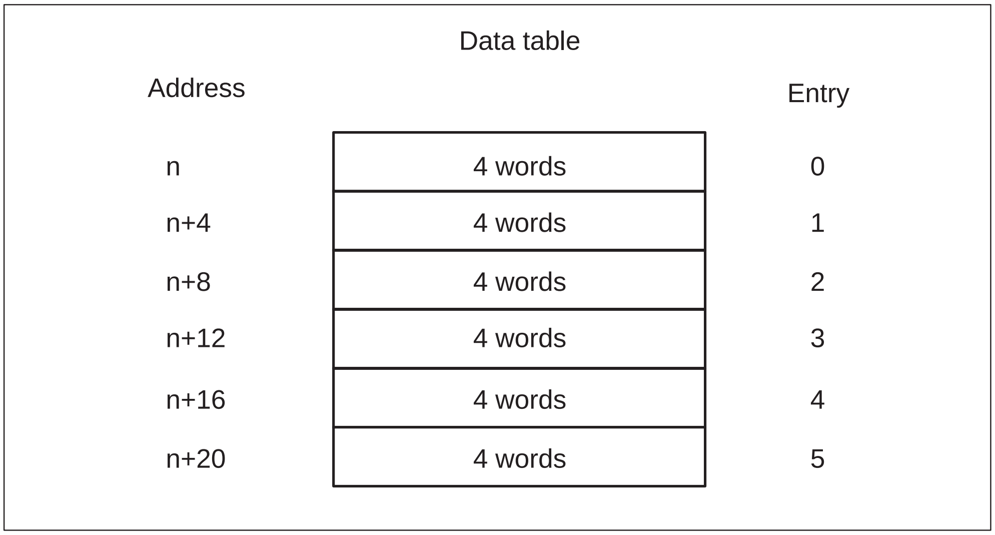
MSB 8				0				0				4			LSB
1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

MSB = Most significant bit
LSB = Least significant bit

Data store tables

The DMS-100 data store is comprised of many different data tables containing information required to complete connections between calling and called parties. A table consists of a number of objects of the same type, occupying a contiguous segment of memory in date store, so that each entry in the table may be indexed (located). An illustration of an NT40 table with six entries, four words in length, is shown in the following figure.

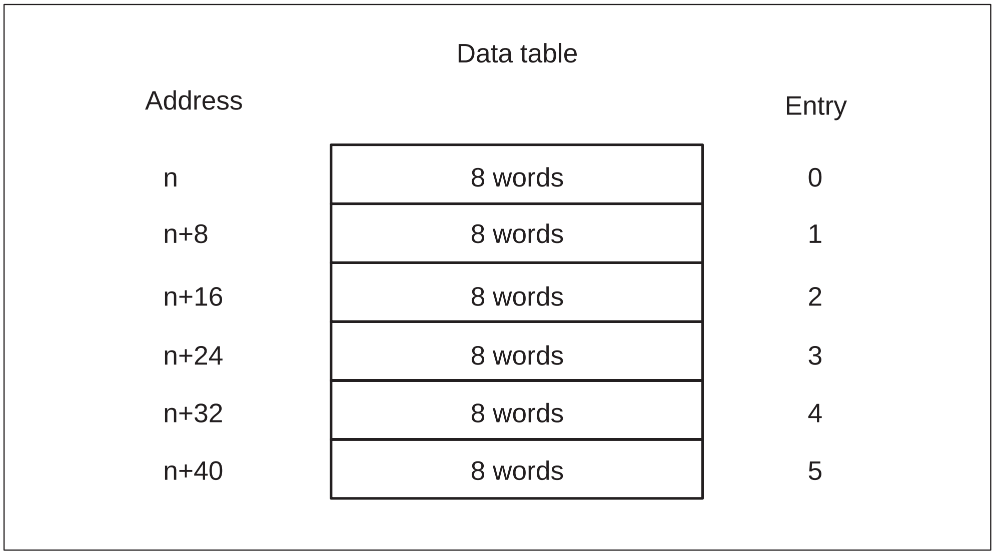
Figure 1-7
NT40 data store tables



Address	Data table	Entry
n	4 words	0
n+4	4 words	1
n+8	4 words	2
n+12	4 words	3
n+16	4 words	4
n+20	4 words	5

The following figure is an illustration of a DMS SuperNode table with six entries, eight words in length.

Figure 1-8
DMS SuperNode data store tables



Data table

Address		Entry
n	8 words	0
n+8	8 words	1
n+16	8 words	2
n+24	8 words	3
n+32	8 words	4
n+40	8 words	5

Pointers and descriptors

A descriptor is a three-word (for the NT40) or a four-word (for the DMS SuperNode) code that describes a table or a part (slice) of a table. The individual entries in a table are addressed by indexing the descriptor as if it were a table. A descriptor contains the following information for the NT40:

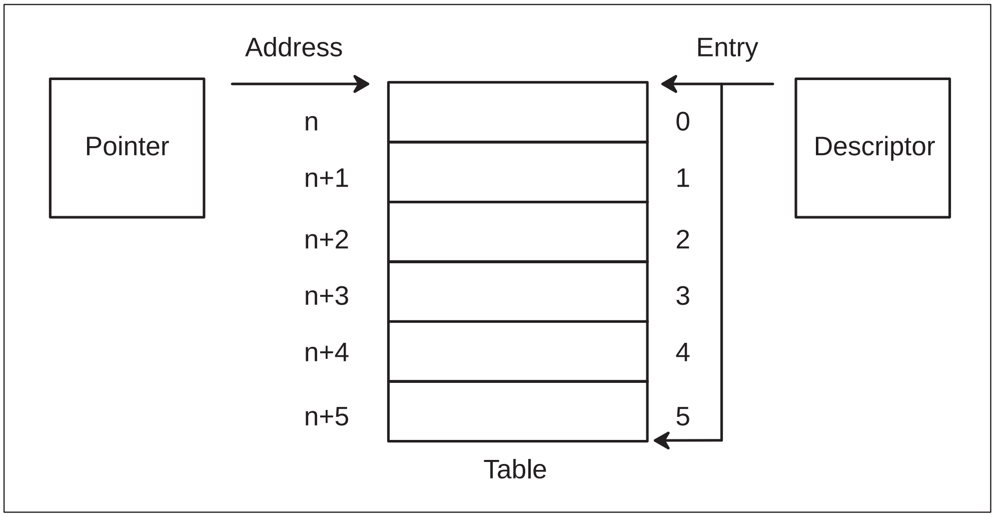
- Address: the absolute address of the base of the table or it can be the base of the slice of the table to which the descriptor is pointing.
- Stride: indicates the length of each entry in the table. Strides may be expressed in words or bits.
- Size: indicates the number of entries in the table.

A descriptor for the DMS SuperNode contains the following information:

- Address: the absolute address of the base of the table or it can be the base of the slice of the table to which the descriptor is pointing.
- Bit or byte offset: If the first entry in the table is not aligned on a word boundary, this field contains the bit or byte offset of the first entry in the table. Otherwise, if the first word is on a word boundary, this field is zero.
- Upper bound: the maximum index value of the table pointed to by the descriptor.

In normal usage it is not necessary to know the internal layout of a descriptor, since these parts are not separately manipulated. See the following figure for an illustration of a pointer, a descriptor, and a table.

Figure 1-9
Example of a pointer, a descriptor, and a table



Dynamic allocation of data store

The DMS-100 operating system allows data store to be managed as a common resource pool. This pool makes it possible for executing programs to allocate (request) store from the unused parts of data store. When this allocated storage is no longer required, it is deallocated (returned) to the common pool for use by other executing programs. Programs that allocate and deallocate tables in the data store use descriptors to store the location and size of these dynamic tables. Pointers can also be used to allocate tables of a fixed size. The fact that PROTEL can dynamically allocate data store has a very important implication: the size of data tables does not have to be decided at design time or at compile time.

Types of data store

The following types of data store are in the DMS-100 switch:

- protected data store
- permanent data store
- temporary data store
- DSSAVE data store
- DSRAM data store

It is the design requirement of a program that determines what store types are used. The low end of memory (CPU RAM) and the high end of memory which is reserved for I/O and maintenance registers are not assigned a data store type and are not allocated by the store allocator.

Protected data store

Protected data store (DSPROT) is used to define such elements as office configuration, what directory numbers are attached to what lines, and routing tables. Typically, DSPROT changes only in response to a user interface command and is used for constant data.

The DSPROT has hardware write protection and must have the protection removed prior to any write operations. The DSPROT remains allocated, and its contents unchanged over warm and cold restarts. Descriptions of restarts are located at the end of this chapter.

The CPU RAM store (DSRAM) is cleared and deallocated on reload restarts. The DSSAVE, which includes information about what caused the previous restart, is not cleared on any type of restart, even when a different software load is installed.

Permanent data store

Permanent data store (DSPERM) is a memory area that does not have hardware write protection. The DSPERM remains allocated over all restarts, but the contents must be reinitialized after a restart reload.

This area is used to hold control information and long-term statistical information. Examples of control information include shared variables, call information, and the state (free or busy) of the different input/output devices. Examples of long-term statistical information include operational measurements, call statistics, and log report data.

Although the contents of DSPERM survive both cold and warm restarts, the area that is used for control information is normally reinitialized after all restarts except warm.

Temporary data store

Temporary data store (DSTEMP) does not have hardware write protection. The DSTEMP is deallocated and the contents cleared after every restart.

The DSTEMP area is used for short-term, transient data (for example, data necessary to execute a program and local variables for a procedure call).

DSSAVE data store

The DSSAVE data store area’s page one remains allocated and unchanged over all restarts, even when a new software load is installed. Therefore, it can be used to store information over reload restarts. There is no hardware write protection on DSSAVE.

DSRAM data store

The DSRAM data store area remains allocated and unchanged over warm and cold restarts. On reload restarts, it is deallocated by the system. It is used by selected applications that need rapid memory capabilities.

Program store

Program store hardware

NT40

The program store contains the instructions required by the associated CPU for call processing, maintenance, and administrative tasks. The NT40 central processor and memory shelf has 8 card slots available for active program store plus one card slot used as a memory controller. This controller card has 256K of memory to be used as replacement memory when a detected single fault occurs in any one of the memory cards. The memory cards come in different sizes, for example, the NT3X93 (256K words) and the NT4X80 (1 M-words.)

DMS SuperNode

Data store and program store are integrated. Memory is provided on two types of memory circuit packs:

- 4 megabytes of DRAM, divided into two 2-megabyte memory modules
- 6 megabytes of DRAM, divided into three 2-megabyte memory modules.

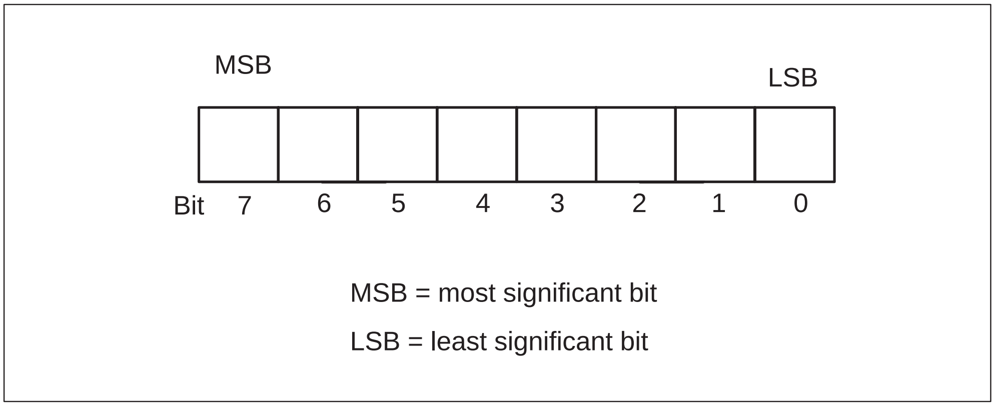
Program store organization

The NT40 program store address consists of an 8-bit page and a 16-bit offset. Each page of program store contains 64K (65,536) bytes. For example, 1FA2B4 is an address on page 1F, with an offset of A2B4 bytes from the beginning of the page.

The DMS SuperNode uses an address consisting of a 16-bit page and a 16-bit offset. Each page of DS contains 16M words.

Bits within a byte of program store are addressed by number from 7–0, from the most significant bit to the least significant bit. The numbering of bits within a byte is illustrated in the following figure.

Figure 1-10
Program store bit numbering



Pointers and descriptors

Program store procedure descriptors are located in the protected data segment of the module. They are used to point to procedures stored in the program store.

Dynamic allocation of program store

Only the loader and some debugging tools allocate program store. The same utilities and methods used to allocate data store are used to allocate program store.

Protected program store

Protected program store has hardware write protection. All permanent programs are loaded into PSPROT. Program store survives all restarts.

Store allocation

The support operating system (SOS) allocator manages the allocation and deallocation of all memory in the CC. It handles both data store and program store. When the store allocator undergoes initial program loading initialization, all of the data store and program store are divided up into areas of memory known as vast areas. For all types of data store except DSRAM and DSSAVE, vast areas range in size from 16K words to 32K words. For DSRAM and DSSAVE, vast areas can be up to 32K words, with no minimum size. Program store vast areas have 32K bytes.

Initially, vast areas do not have a type associated with them. As allocation requests are made, vast areas are set to the type required, which can be one of the following: DSTEMP, DSPERM, DSPROT, DSSAVE, DSRAM, PSTEMP, and PSPROT.

Vast areas also have an associated status that indicates if the area is available to be allocated as any store type (designated as available) or if it is already in use as a specific type (designated as in use).

Store is allocated in contiguous segments, known as store blocks, which may be any size up to the size of a vast area. Header tables are maintained to keep track of the allocated vast areas and of the store blocks that are allocated within the vast areas.

Store allocation lists

The store allocator checks to ensure that all allocations of store remain within the limits of one vast area. When store is deallocated, it is entered into the list of free store for that vast area. Adjacent free store blocks are concatenated (linked together). Blocks of store cannot be partially deallocated; therefore, half of a block of store cannot be deallocated. On restarts, vast areas of type DSTEMP are set to a status of available.

Segmented store

The segmented store utility is provided to allow the store allocation of variable-size tables. Use of the segmented store allows large tables to be extended without the need to copy or change the store already allocated. Segmented store also allows the allocation of tables larger than a vast area.

Segmented store allows a conceptually contiguous table of store to be physically distributed in noncontiguous blocks of store. All the blocks are the same size. It uses a two-level table and is implemented in the module SEGSTOR. The table can have up to 64K items, the addressing range of a 16-bit word.

SuperNode

The DMS SuperNode is part of a second generation of the DMS-100 switch in which the DMS SuperNode replaces the NT40 central processor. There are some DMS SuperNode and the NT40 similarities, for example, both of the control component CPU memory areas consist of two distinct elements, program store and data store. Program store contains variable-length program instructions. Data store contains data. At the same time, one of the most significant differences in the two control components is memory. A comparison of the memory attributes of both switch generations is summarized in the following table.

Table 1-1
Comparison of memory – SuperNode and NT40

Memory attribute	Central control CPU	DMS SuperNode CPU
Addressing unit		
program store	Byte-addressable	Byte-addressable
data store	Word-addressable	Byte-addressable
Address bus width	24 bits	24 bits
Address range		Combined program store and data store: 4 gigabytes
program store	16 megabytes	
data store	16 megawords0.	
Data bus width	16 bits	32 bits
Page size		
program store	64 kilobytes	64 kilobytes
data store	64 kilowords	64 kilobytes
Vast area size		
DSRAM	Up to 32 kilowords	12 kilobytes
DSSAVE	Up to 32 kilowords	32 or 64 kilobytes
OTHER data store	16 to 32 kilowords	32 or 64 kilobytes
program store	32 kilobytes	32 or 64 kilobytes
RAM68DT	NA	20 kilobytes
RAM68PP	NA	32 kilobytes

A DMS SuperNode has ten card slots available for memory on the DMS-Core shelf. The program store and data store memory in a SuperNode configuration is pooled in that the memory cards can be used for either of the two stores, and there is no physical separation of the two stores as there is in the NT40. However, once the memory has been designated as one store or the other, it can only be used for that store until the designation has been changed.

There are two different memory cards available.

- 6 MByte - NT9X14DB
- 24 MByte - NT9X14DB

There are recommended mixed memory card configurations when the initial DMS-SuperNode installation provides only 6 MByte cards and a memory extension job is required. Refer to the following table.

Table 1-2
Recommended mixed memory card configurations

Total MBytes active plus required administrative spare	6 MByte cards	24 MByte cards	6 MByte spare cards	24 MByte spare cards
0–42	7	0	1	0
42–48	8	0	1	0
48–54	9	0	1	0
54–60	7	1	1	1
60–80	6	2	1	1
84–100	5	3	1	1
10–116	4	4	1	1
11–132	3	5	1	1
12–156	2	6	1	1
16–168	1	7	1	1
168–212	0	9	0	0

Vast areas

Vast area sizes for each store type are shown in the previous figure. For SuperNode DSRAM, the vast area size is 12 kilobytes. For RAM68DT, it is 20 kilobytes; for RAM68PP, it is 32 kilobytes. For all other types of data store and program store, the vast area size is 32 kilobytes or 64 kilobytes. Vast areas are initially allocated with a size of 64 kilobytes. If less than half of a vast area is used at initial program load time, the store allocator splits the 64-kilobyte vast area into two 32-kilobyte vast areas. Both of these 32-kilobyte areas are of the same store type since they reside on the same 64-kilobyte page of memory. The store allocator reserves the unused 32-kilobyte vast area until some of its assigned type is required.

The store allocator cannot allocate blocks of store larger than the size of a vast area. Thus, for all types of data store, except DSRAM, data structures cannot have more than 64 kilobytes of contiguous store. Larger structures are accommodated by segmenting the structure into blocks of 64 kilobytes or less. However, procedures cannot be segmented; therefore, they cannot be larger than 64 kilobytes in size (32 kilobytes in RAM68PP).

Fragmented memory

Fragmented memory is the deallocated memory within INUSE vast areas that has no data to be assigned to its free blocks; that is, the memory was deallocated, but no additional requests were made that would fit into its space. Therefore, there are holes in the used memory areas. The available blocks of memory that make up the holes may be recovered by either a cold restart or by the dump and restart process when the allocator finds a block of data that fits into the hole. In the engineering process, the provisioning formula provides an additional eight percent of memory in data store to compensate for fragmentation.

The DSTEMP memory can be made available by a warm restart while DSPROT memory can be overwritten by a dump and restore (new software load). The DSRAM memory can be made available by a restart reload.

An accurate status of the switch memory areas may be obtained through the MAP command interpreter (CI) utility, STORE AREAS, which displays a complete listing of all vast area memory allocation. Memory addresses are presented in hexadecimal form, and a total free hexadecimal representation of all free data store or program store is displayed at the bottom of the listing.

Determining fragmented memory (in bytes) for the NT40 may be accomplished using the MAP CI utility, STORE AREAS, and can be calculated by following the steps in the following procedure.

1. Obtain the start address of the first vast area. Convert the hexadecimal address to decimal. This number is the Start In Use value in the percent fragmentation calculation.

2. Obtain the start address of the first vast area with a status of Available after the last vast area with a status of In Use. Convert the hexadecimal address to decimal. This number is the Start Avail value in the percent fragmentation calculation.

3. Obtain the start address of the last vast area. Convert the hexadecimal address to decimal. This number is the End Avail value in the percent fragmentation calculation.

4. Obtain the hexadecimal value directly below the last vast area. This is the total free space for all memory associated with the vast areas. Convert this number to decimal. This number is the Total Free value in the percent fragmentation calculation.

5. Use the following formula to calculate the percentage of fragmentation:

$$\% \text{ Fragmentation } = \frac{\text{Total Free} - (\text{End Avail} - \text{Start Avail})}{(\text{End Avail} - \text{Start In Use})} \text{ X } 100$$

Program store memory fragments very little. Sufficient program store is provided for approximately one year of source code, in accordance with current NT software administration policy.

Restarts

A restart is an ordered initialization of every module in the system. Restarts serve two important purposes.

- initialization of the system, when it is installed or when new system components (either hardware or software) are installed
- last-resort error recovery while the system is operational

There are four types of restart: warm, base, cold, and reload.

Warm restart

A warm restart is the least severe type of restart. Temporary data store (DSTEMP) is cleared and deallocated. All other types of store survive a warm restart.

Base restart

A variation of a warm restart, the base restart is used to debug initialization code during the software development process. A base restart is a drastic measure. It disables the system's call processing capability and, therefore, should never be performed in a system that is operational.

Cold restart

A cold restart is more severe than a warm restart. DSPERM survives a cold restart, but its contents are suspect since some programs cause automatic cold restarts if their DSPERM appears to be corrupted. For this reason, many modules reinitialize DSPERM on cold restarts. A cold restart has all of the effects of a warm restart, plus additional effects due to the reinitialization of data in DSPERM.

Call processing is affected by the reinitialization of the data in DSPERM in which the following events occur:

- The switching network connection maps in the CC are cleared, allowing existing switching network paths to be reused by new calls after the restart.
- The state of each line and trunk in the CC is set to idle.

- Data associated with each call is reinitialized to the idle state.

Calls in progress that have reached the talking state retain their switching network connections during the restart, but they may be disconnected if their switching network paths are reused by new calls after the restart. There is no record of the call’s existence in the CC; the only call processing function that can be performed is call take-down. No billing data are recorded for these calls.

Reload restarts

A reload restart is even more severe than a cold restart. It has all the effects of a cold restart; in addition, DSPERM is cleared but remains allocated. The CPU RAM store (DSRAM) is cleared and deallocated on reload restarts. The DSSAVE, which includes information about what caused the previous restart, is not cleared on any type of restart, even when a different software load is installed.

Using MEMCALC

Introduction

As more features were added to the DMS-100 switch, a mechanized method of engineering memory and forecasting the impact of these new features was needed. This chapter addresses MEMCALC, Northern Telecom's memory forecasting tool, which was designed to answer this need.

Looking at MEMCALC

The Northern Telecom MEMCALC engineering tool is designed to provision memory and in some applications to measure the impact of new features on memory usage of a DMS-100 switch. The MEMCALC tool (program) is available in two versions: stand-alone or integrated.

The stand-alone version provides the user with the ability to look at memory in word values and equivalent memory cards (for example, 64K, 256K, 6 megabytes, and 24 megabytes). In this version, only items relating to memory are required for input into the program.

Integrated MEMCALC is found in Northern Telecom's NT-ACCESS, an automated engineering and planning tool. It requires the input of a list of all software packages that will reside in the switch's in-service BCS load as well as data based upon end of design (EOD) quantities. This tool is an integral part of memory provisioning and is useful in monitoring the accuracy of the projected memory requirements. Telephone company engineers project usage, which is part of the process of estimating the number of lines, trunks, and types of features that are required in an exchange through the engineered period. This period is usually two to three years following the in-service date. These projections are based upon growth forecasts provided by the customer.

The MEMCALC tool is available either in a line or full screen mode. It functions in a similar manner in either mode but with increased speed and flexibility in full screen.

The operating company and Northern Telecom (NT) engineers provision memory based upon operating company projections. In MEMCALC, the NT tool future BCS requirements are based upon the NT forecast of the memory required for these BCSs. Therefore, the memory provisioned is based upon two forecasts: customer EOD parameters and the NT memory program MEMCALC.

After the cutover of a new switch or a major addition to an existing switch, MEMCALC should be run using the actual parameters. The MEMCALC questionnaire refers to the switch data necessary to run the tool.

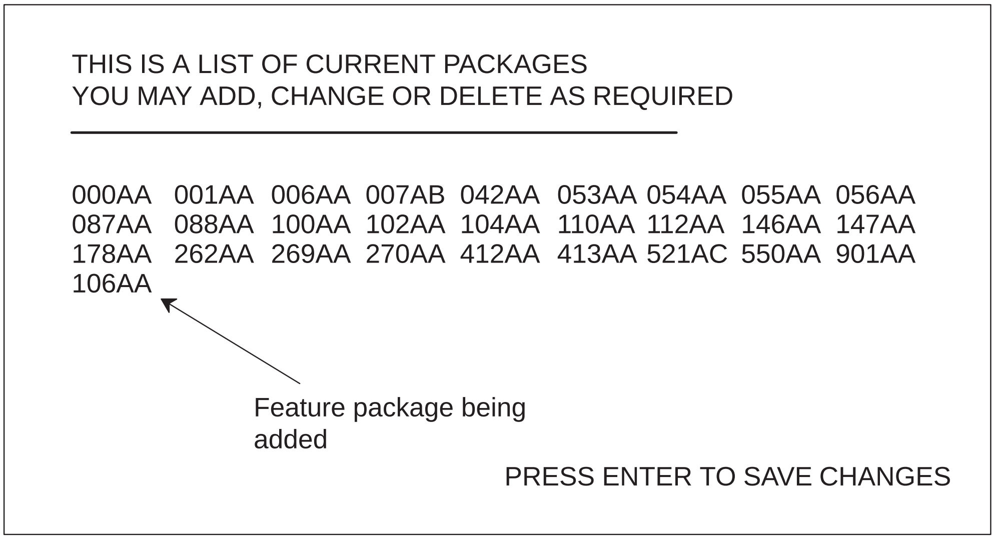
The MEMCALC tool calculates the memory required for the switch application only and provides required allocation rounding rules as part of its output. Administrative spare must also be calculated. Administrative spare memory is memory required in addition to that required to process calls.

Using MEMCALC to determine feature impact

The MEMCALC tool can also be used to predict the impact of placing new customer accounts into the switch, such as, a large centrex complex with its many features, area transfers, or the introduction of new features. The following is an example of how to use MEMCALC for this purpose. In this example, a feature package is being added to the switch (NTX106AA, IBN Proprietary Business Set). It is assumed that the reader is familiar with the use of MEMCALC and the example is only part of the complete procedure.

1 After accessing MEMCALC and displaying the panel that lists the current feature packages loaded in the switch, add the new feature package (in this example, it is NTX106AA), refer to the following figure.

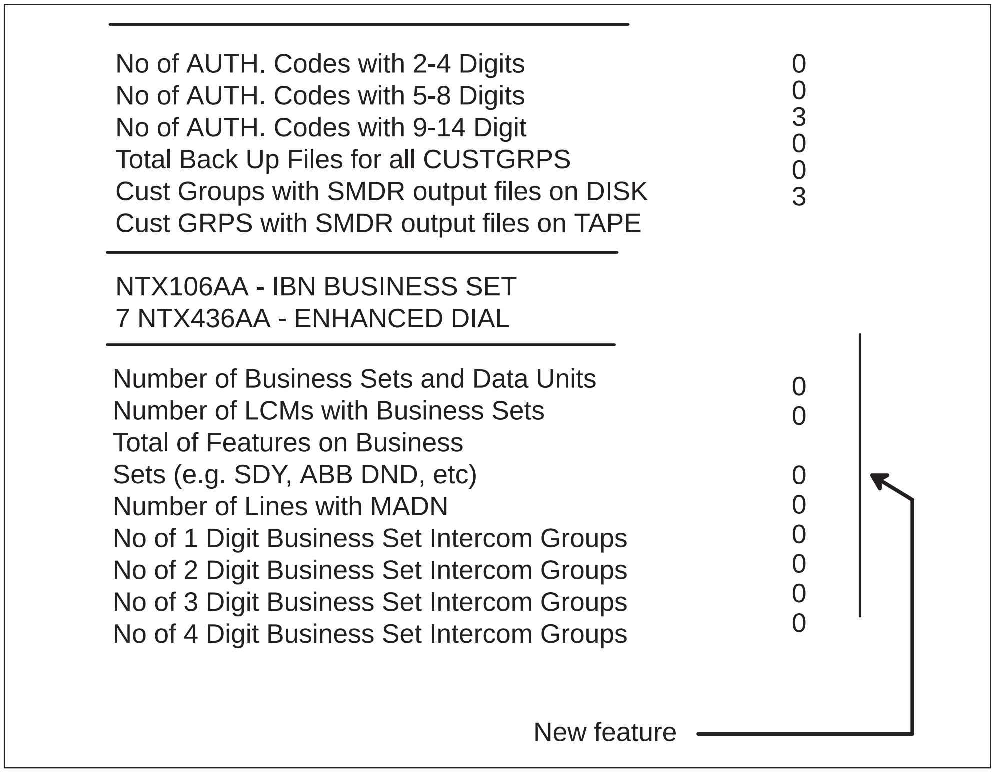
Figure 2-1
Example of list of feature packages in switch – MEMCALC



```
THIS IS A LIST OF CURRENT PACKAGES
YOU MAY ADD, CHANGE OR DELETE AS REQUIRED
_________________________________________

000AA  001AA  006AA  007AB  042AA  053AA 054AA  055AA  056AA
087AA  088AA  100AA  102AA  104AA  110AA 112AA  146AA  147AA
178AA  262AA  269AA  270AA  412AA  413AA 521AC  550AA  901AA
106AA

                              PRESS ENTER TO SAVE CHANGES
```

2 The next panel will display the new feature package added to the list of features. The feature package being added will show all zeros for the quantities assigned, see the following figure.

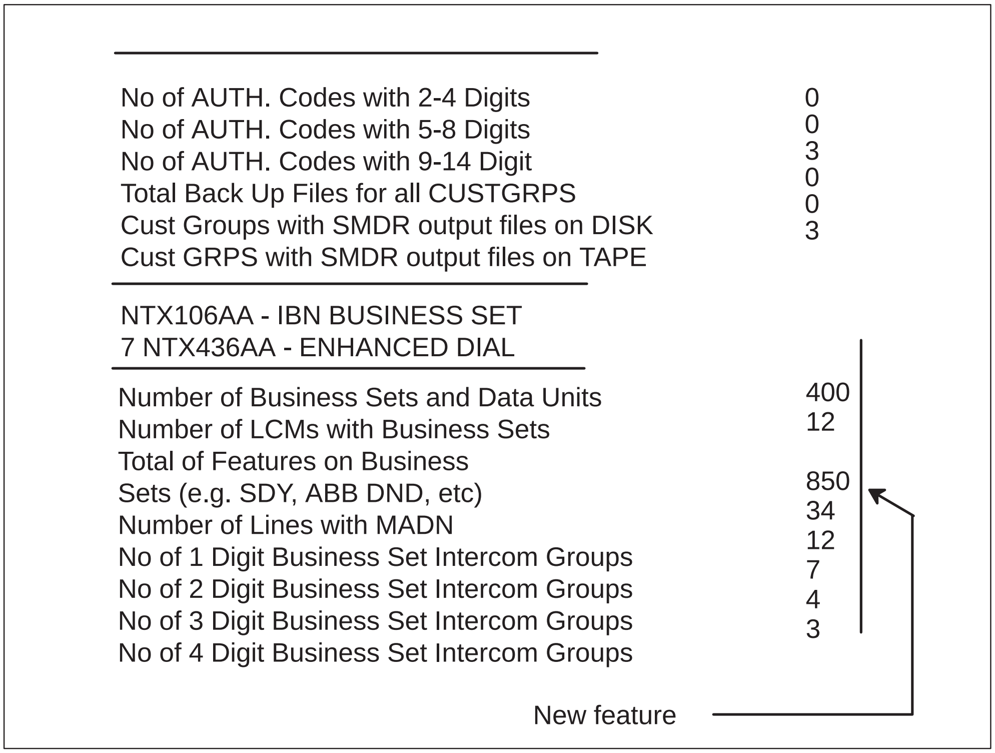
Figure 2-2
Example of list of feature packages in switch – MEMCALC



```
No of AUTH. Codes with 2-4 Digits                0
No of AUTH. Codes with 5-8 Digits                0
No of AUTH. Codes with 9-14 Digit                3
Total Back Up Files for all CUSTGRPS             0
Cust Groups with SMDR output files on DISK       0
Cust GRPS with SMDR output files on TAPE         3

NTX106AA - IBN BUSINESS SET
7 NTX436AA - ENHANCED DIAL

Number of Business Sets and Data Units           0
Number of LCMs with Business Sets                0
Total of Features on Business
Sets (e.g. SDY, ABB DND, etc)                    0
Number of Lines with MADN                        0
No of 1 Digit Business Set Intercom Groups       0
No of 2 Digit Business Set Intercom Groups       0
No of 3 Digit Business Set Intercom Groups       0
No of 4 Digit Business Set Intercom Groups       0
```

3 Enter the expected number of assignments to the individual features that will be working in the new feature package, see the following figure.

Figure 2-3
Estimated feature usage numbers – MEMCALC

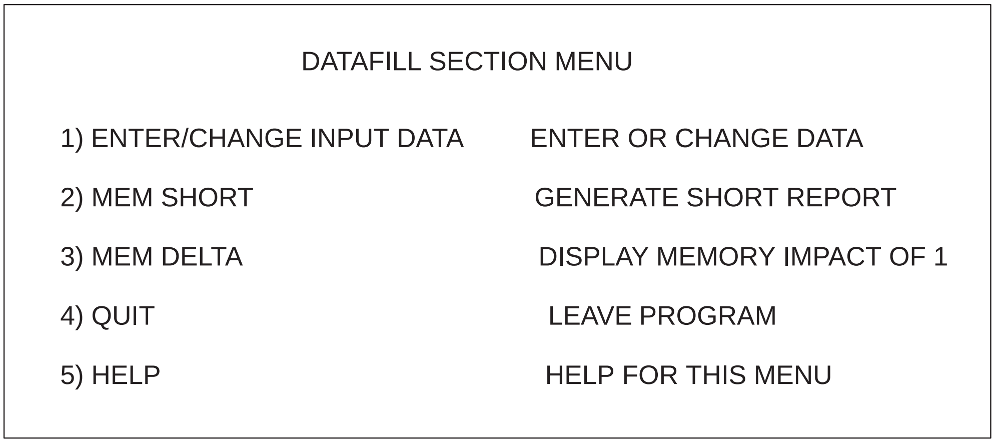


```
No of AUTH. Codes with 2-4 Digits                0
No of AUTH. Codes with 5-8 Digits                0
No of AUTH. Codes with 9-14 Digit                3
Total Back Up Files for all CUSTGRPS             0
Cust Groups with SMDR output files on DISK       0
Cust GRPS with SMDR output files on TAPE         3

NTX106AA - IBN BUSINESS SET
7 NTX436AA - ENHANCED DIAL

Number of Business Sets and Data Units           400
Number of LCMs with Business Sets                12
Total of Features on Business
Sets (e.g. SDY, ABB DND, etc)                    850
Number of Lines with MADN                        34
No of 1 Digit Business Set Intercom Groups       12
No of 2 Digit Business Set Intercom Groups       7
No of 3 Digit Business Set Intercom Groups       4
No of 4 Digit Business Set Intercom Groups       3
```

New feature

4 The next panel will display the datafill section menu. Select option 3 (display memory impact of 1). This will display the memory impact of previous input of data panel, see the following figure.

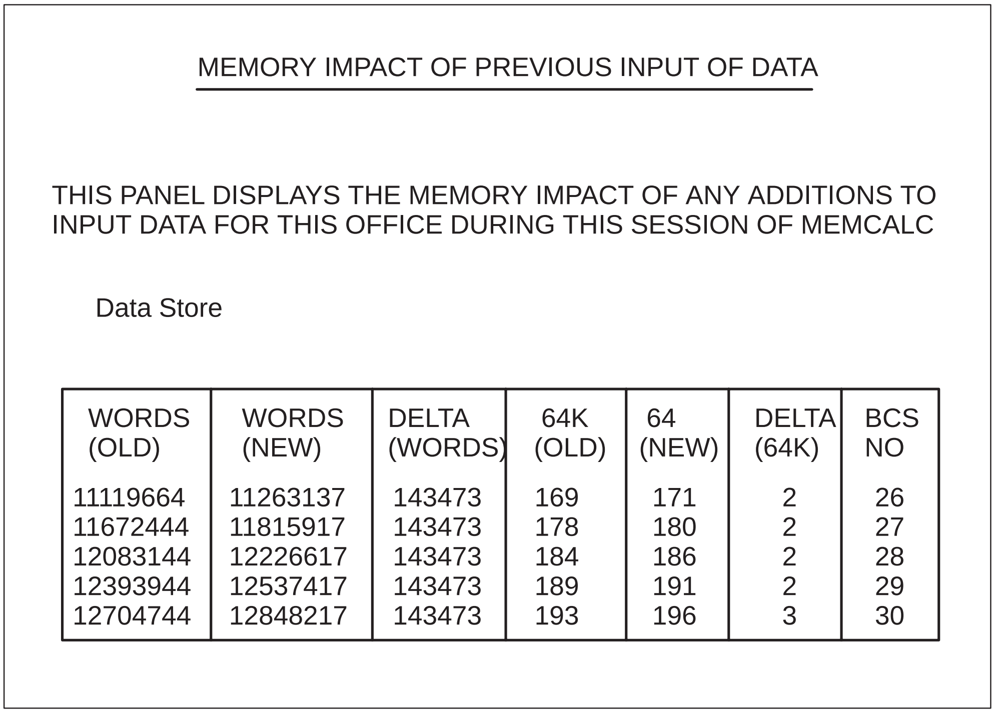
Figure 2-4
Example of datafilled menu - MEMCALC



```
                  DATAFILL SECTION MENU

1) ENTER/CHANGE INPUT DATA       ENTER OR CHANGE DATA

2) MEM SHORT                     GENERATE SHORT REPORT

3) MEM DELTA                     DISPLAY MEMORY IMPACT OF 1

4) QUIT                          LEAVE PROGRAM

5) HELP                          HELP FOR THIS MENU
```

5 Move to the next panel, which indicates the impact to memory in words. These figures do not include overhead requirements, see the following figure.

Figure 2-5
Example of impact panel - MEMCALC



MEMORY IMPACT OF PREVIOUS INPUT OF DATA

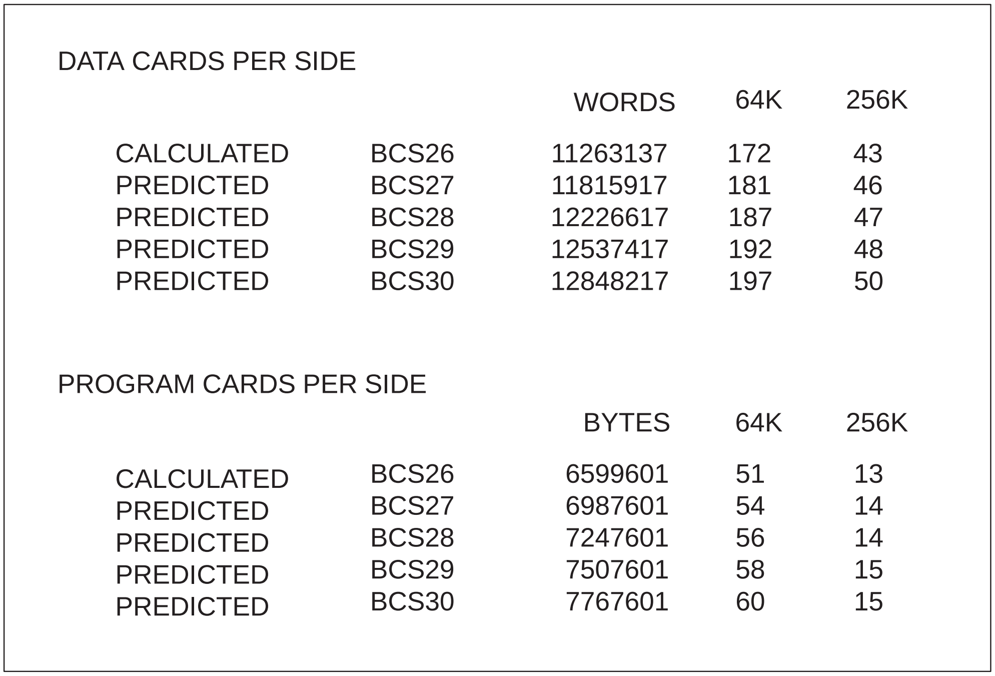
THIS PANEL DISPLAYS THE MEMORY IMPACT OF ANY ADDITIONS TO INPUT DATA FOR THIS OFFICE DURING THIS SESSION OF MEMCALC

Data Store

WORDS (OLD)	WORDS (NEW)	DELTA (WORDS)	64K (OLD)	64 (NEW)	DELTA (64K)	BCS NO
11119664	11263137	143473	169	171	2	26
11672444	11815917	143473	178	180	2	27
12083144	12226617	143473	184	186	2	28
12393944	12537417	143473	189	191	2	29
12704744	12848217	143473	193	196	3	30

6 The next panel displays the data cards and program cards per side predicted for the current BCS and the four succeeding BCS releases. These figures include overhead requirements and increase the calculated data store cards by one for each BCS over that shown in the previous panel, see the following figure.

Figure 2-6
Example of required memory cards - MEMCALC



```
DATA CARDS PER SIDE
                                              WORDS     64K     256K

    CALCULATED          BCS26          11263137     172        43
    PREDICTED           BCS27          11815917     181        46
    PREDICTED           BCS28          12226617     187        47
    PREDICTED           BCS29          12537417     192        48
    PREDICTED           BCS30          12848217     197        50


PROGRAM CARDS PER SIDE
                                              BYTES     64K     256K

    CALCULATED          BCS26           6599601      51        13
    PREDICTED           BCS27           6987601      54        14
    PREDICTED           BCS28           7247601      56        14
    PREDICTED           BCS29           7507601      58        15
    PREDICTED           BCS30           7767601      60        15
```

> ***Note:*** The number of 64K cards are one more in this figure than those shown in the previous figure. The larger numbers contain allowances for memory used for overhead while the smaller numbers reflect only the memory needed for the feature package itself.

In a similar manner, the memory impact of large customer accounts being introduced to the switch, such as a large centrex complex, can be estimated.

Using operational measurements

Collecting memory data

Each time a new feature is added to an existing switch or a new customer needs a large number of services and features, the question of the adequacy of available memory has to be addressed. This chapter discusses the impact on memory of individual features and the use of operational measurements (OM) to determine that impact. The primary OM group is STORE. The STORE registers that are available are listed in the following table. The STORE register values may be displayed using the MAPCI level command OMSHOW STORE. Refer to *Basic Administration Procedures*, 297-1001-300, for detailed instructions on the OMSHOW command.

Using OM group STORE

The OM group STORE provides information similar to that provided through the MAP commands. One difference is the way fragmented memory is treated. With the STORE OM, when measurements of memory use are made, fragmented memory is counted as memory in use. In addition, there are registers available for SuperNode installations that indicate the spare memory that is available on the memory cards but is not accessible to the memory allocator. The registers are described in detail in *Operational Measurements Reference Manual*. The following table lists the registers in this OM group:

The STORE OM group is suited for a mechanized data collection system and thresholds can be established to alert the user when designated levels of memory usage have been reached.

Table 3-1
STORE OM

Group	Register	Information
STORE	ALL	Description: STORE provides information on the use of data store, program store, and the amount of memory available in the NT40 CC or DMS SuperNode. For the DMS SuperNode, the amount of spare memory is also monitored. Separate registers provide information in megabytes and kilobytes for: data store used and available program store used and available total available memory total addressable physical memory available to the SOS store allocator spare memory for SuperNode. BCS history: STORE was created in BCS27.

—continued—

Table 3-1
STORE OM (continued)

Group	Register	Information
	DSAVAILK	Description: data store available in kilobytes The value in this register represents the number of kilobytes of memory in use in addition to the megabytes available in DSAVILM for data store. The total amount of memory available for a large memory extension block, such as that required to add a large amount of datafill, equals the sum of DSAVAILK and DSAVAILM. This does not include small, fragmented blocks of memory. BCS history: DSAVAILK was created in BCS27. Associated registers: DSAVAILK, DSAVAILM, FREEMB, FREEKB, PSAVAILM, and PSAVAILKB Register validation: Total DS available (in kilobytes) = DSAVAILK + (DSAVAILM x 1024) FREEMB + FREEKB = DSAVAILM + DSAVAILK + PSAVAILM + PSAVAILKB ***Note:*** Values for FREEMB + FREEKB should not drop below 192 kilobytes (3 vast areas), the minimum free memory required for the proper DMS operation.

—continued—

Table 3-1
STORE OM (continued)

Group	Register	Information
	DSAVAILM	Description: data store available in megabytes The value in this register represents the number of megabytes of memory in use in addition to the kilobytes available in DSAVAILK for data store. The total amount of memory available for a large memory extension block, such as that required to add a large amount of datafill, equals the sum of DSAVAILK and DSAVAILM. This does not include small, fragmented blocks of memory. BCS history: DSAVAILM was created in BCS27. Associated registers: DSAVAILK, DSAVAILM, FREEMB, FREEKB, PSAVAILM, and PSAVAILKB Register validation: Total DS available (in megabytes) = DSAVAILM + (DSAVAILK / 1024) FREEMB + FREEKB = DSAVAILM + DSAVAILK + PSAVAILM + PSAVAILKB ***Note:*** Values for FREEMB + FREEKB should not drop below 192 kilobytes (3 vast areas), the minimum free memory required for the proper DMS operation.
	DSUSEDK	Description: data store used in kilobytes. The value in DSUSEDK represents the number of kilobytes of memory in use in addition to the megabytes in use in DSUSEDM for data store. The total amount of memory in use that is allocated or reserved for special use equals the sum of DSUSEDK and DSUSEDM. These registers should always be viewed together to determine the total amount of data store memory in use. BCS history: DSUSEDK was created in BCS27. Associated registers: DSUSEDM Register validation: Total DS used (in kilobytes) = DSUSEDK + (DSUSEDM x 1024)

—continued—

Table 3-1
STORE OM (continued)

Group	Register	Information
	DSUSEDM	Description: data store used in megabytes. The value in DSUSEDM represents the number of megabytes of memory in use in addition to the kilobytes in use in DSUSEDK for data store. The total amount of memory in use that is allocated or reserved for special use equals the sum of DSUSEDK and DSUSEDM. These registers should always be viewed together to determine the total amount of data store memory in use. BCS history: DSUSEDM was created in BCS27. Associated registers: (DSUSEDK Register validation: Total DS used (in megabytes) = DSUSEDM + (DSUSEDK / 1024)
	FREEKB	Description: free memory in kilobytes The value of FREEKB represents the number of kilobytes of memory available in vast areas in addition to the megabytes available in FREEMB for use as program store and data store. FREEKB should always be viewed in conjunction with FREEMB to determine the total amount of memory available in vast areas for use in program store and data store. BCS history: FREEKB was created in BCS27. Associated registers: FREEKB, DSAVAILM, DSAVAILK, PSAVAILM, and PSAVAILKB Register validation: Total free memory (in kilobytes) = FREEKB + (FREEMB x 1024) FREEMB + FREEKB = DSAVAILM + DSAVAILK + PSAVAILM + PSAVAILKB ***Note:*** Values for FREEMB + FREEKB should not drop below 192 kilobytes (3 vast areas), the minimum free memory required for the proper DMS operation.

—continued—

Table 3-1
STORE OM (continued)

Group	Register	Information
	FREEMB	Description: free memory in megabytes. The value of FREEMB represents the number of megabytes of memory available in vast areas in addition to the kilobytes available in FREEKB for use as program store and data store. FREEMB should always be viewed in conjunction with FREEKB to determine the total amount of memory available in vast areas for use in program store and data store. BCS history: FREEMB was created in BCS27. Associated registers: FREEKB, DSAVAILM DSAVAILK, PSAVAILM, and PSAVAILKB Register validation: Total free memory (in megabytes) = FREEMB + (FREEKB / 1024) FREEMB + FREEKB = DSAVAILM + DSAVAILK + PSAVAILM + PSAVAILKB ***Note:*** Values for FREEMB + FREEKB should not drop below 192 kilobytes (3 vast areas), the minimum free memory required for the proper DMS operation.

—continued—

Table 3-1
STORE OM (continued)

Group	Register	Information
	PSAVAILK	Description: program store available in kilobytes The value in PSAVAILK represents the number of kilobytes of memory available for a large memory extension in addition to the megabytes available in PSAVAILM for large memory extension. Large memory extensions are required to load new feature packages. The total amount of memory available for large memory extensions for program store equals the sum of PSAVAILK and PSAVAILM. PSAVAILK and PSAVAILM do not include fragmented memory. PSAVAILK should always be viewed in conjunction with PSAVAILM to determine the amount of memory available for program store. BCS history: PSAVAILK was created in BCS27. Associated registers: PSAVAILK, PSAVAILM, FREEMB, FREEKB, DSAVAILM, DSAVAILK, and PSAVAILKB Register validation: Total PS available (in kilobytes) = PSAVAILK + (PSAVAILM x 1024) FREEMB + FREEKB = DSAVAILM + DSAVAILK + PSAVAILM + PSAVAILKB ***Note:*** Values for FREEMB +FREEKB should not drop below 192 kilobytes (3 vast areas), the minimum free memory required for the proper DMS operation.

—continued—

Table 3-1
STORE OM (continued)

Group	Register	Information
	PSAVAILM	Description: program store available in megabytes The value in PSAVAILM represents the number of megabytes of memory available for a large memory extension in addition to the kilobytes available in PSAVAILM for large memory extension. Large memory extensions are required to load new feature packages. The total amount of memory available for large memory extensions for program store equals the sum of PSAVAILK and PSAVAILM. PSAVAILK and PSAVAILM do not include fragmented memory. PSAVAILM should always be viewed in conjunction with PSAVAILK to determine the amount of memory available for program store. BCS history: PSAVAILM was created in BCS27. Associated registers: PSAVAILK, FREEMB, FREEKB, DSAVAILM, DSAVAILK, PSAVAILM, and PSAVAILKB Register validation: Total PS available (in megabytes) = PSAVAILM + (PSAVAILK / 1024) FREEMB + FREEKB = DSAVAILM + DSAVAILK + PSAVAILM + PSAVAILKB ***Note:*** Values for FREEMB + FREEKB should not drop below 192 kilobytes (3 vast areas), the minimum free memory required for the proper DMS operation.

—continued—

Table 3-1
STORE OM (continued)

Group	Register	Information
	PSUSEDK	Description: program store used in kilobytes The value in PSUSEDK represents the number of kilobytes of memory in use in addition to the megabytes in use in PSUSEDM for program store. The total amount of memory in use for program store equals the sum of PSUSEDK and PSUSEDM. PSUSEDK should always be viewed in conjunction with PSUSEDM to determine the total amount of memory in use for program store. BCS history: PSUSEDK was created in BCS27. Associated registers: PSUSEDM Register validation: Total PS used (in kilobytes) = PSUSEDK + PSUSEDM x 1024
	PSUSEDM	Description: program store used in megabytes The value in PSUSEDM represents the number of megabytes of memory in use in addition to the kilobytes in use in PSUSEDK for program store. The total amount of memory in use for program store equals the sum of PSUSEDK and PSUSEDM. PSUSEDM should always be viewed in conjunction with PSUSEDK to determine the total amount of memory in use for program store. BCS history: PSUSEDM was created in BCS27. Associated registers: PSUSEDK Register validation: Total PS used (in megabytes) = (PSUSEDK/1024) + PSUSEDM

—continued—

Table 3-1
STORE OM (continued)

Group	Register	Information
	SPAREKB	Description: spare memory in kilobytes (SuperNode) The value in SPAREKB represents the amount of memory in kilobytes in addition to the amount of memory in megabytes in SPAREMB that is available (to become allocated memory) in additional memory cards. This register is zeroed in an NT40. SPAREKB should always be viewed in conjunction with SPAREMB to determine the total amount of spare memory. BCS history: SPAREKB was created in BCS27. Associated registers: SPAREMB Register validation: Total spare memory (in kilobytes) = SPAREKB + (SPAREMB x 1024)

—continued—

Table 3-1
STORE OM (continued)

Group	Register	Information
	SPAREMB	Description: spare memory in megabytes (SuperNode) The value in SPAREMB represents the amount of memory in megabytes in addition to the amount of memory in kilobytes in SPAREKB that is available (to become allocated memory) in additional memory cards. This register is zeroed in an NT40. SPAREMB should always be viewed in conjunction with SPAREKB to determine the total amount of spare memory. BCS history: SPAREMB was created in BCS27. Associated registers: SPAREKB Register validation: Total spare memory (in megabytes) = SPAREMB + (SPAREKB / 1024) ***Note 1:*** For a SuperNode, the OM SPAREMB and SPAREKB values should be monitored to ensure that spare memory does not fall below three memory modules (a module is defined as one third of a physical CM memory pack). ***Note 2:*** AVAIL memory is not part of the SPARE memory values.

—continued—

Table 3-1
STORE OM (continued)

Group	Register	Information
	TOTALKB	Description: total memory in kilobytes The value in TOTALKB represents the number of kilobytes of addressable physical memory available to the support operating system (SOS) store allocator in addition to the megabytes available in TOTALMB. TOTALKB should always be viewed in conjunction with TOTALMB to determine the total amount of addressable physical memory available to the SOS. BCS history: TOTALKB was created in BCS27. Associated registers: TOTALMB Register validation: Total addressable physical memory available to the SOS store allocator (in kilobytes) = TOTALKB + (TOTALMB x 1024)
	TOTALMB	Description: total memory in megabytes The value in TOTALMB represents the number of megabytes of addressable physical memory available to the support operating system (SOS) store allocator in addition to the kilobytes available in TOTALKB. TOTALMB should always be viewed in conjunction with TOTALKB to determine the total amount of addressable physical memory available to the SOS. BCS history: TOTALMB was created in BCS27. Associated registers: TOTALKB Register validation: Total addressable physical memory available to the SOS store allocator (in megabytes) = TOTALMB + (TOTALKB / 1024)

—end—

Reviewing STORE data

A sample printout of the STORE operational measurements is shown in the following figure. Memory administration is best performed using the STORE OM.

Figure 3-1
Example of OM STORE printout

```
DSUSEDM          DSUSEDK          DSAVAILM         DSAVAILK
FREEMB           FREEKB           TOTALMB          TOTALKB
PSUSEDM          PSUSEDK          PSAVAILM         PSAVAILK
SPAREMB          SPAREKB

       27              957            3                 567
       17              896            62                509
       16              704            14                320
         0             0
```

The following formulas should be used with the STORE OM to determine words and 64K word card equivalents:

NT40 data store:

Memory used (words) = [((DSUSEDDM x 1024) + DSUSEDK) x 1024) / 2
Memory used (cards) = memory used (words] / 65536

NT40 program store:

Memory used (words) = [((PSUSEDM x 1024) + PSUSEDK) x 1024) / 2
Memory used (cards) = memory used (words] / 65536

SuperNode data store:

Memory used (bytes) = [(DSUSEDM x 1024) + DSUSEDK) x 1024
Memory used (megabytes) = memory used (bytes] / 1046576

SuperNode program store:

Memory used (bytes) = [(PSUSEDM x 1024) + PSUSEDK) x 1024
Memory used (megabytes) = memory used (bytes] / 1048576

The following is an example of determining the words and 64K word card equivalents for an NT40, using the values shown in the preceding figure:

NT40 data store:

Memory used (words) = [((27 x 1024) + 957) x 1024) / 2 = 14645760 words
Memory used cards = 14645760 / 65536 = 223.47 cards (64K equivalents]

NT40 program store:

Memory used (words) = [((16 x1024) + 704) x 1024) / 2 = 8749056 words
Memory used (cards) = 8749056 / 65536 = 133.50 cards (64K equivalents]

Again using the values in the preceding figure, determine the memory requirements for a SuperNode as follows:

SuperNode data store:

Memory used (bytes) = [(27 x 1024) + 957) x 1024 = 29291520 bytes
Memory used (megabytes] = 29291520 / 104856 = 27.93 megabytes

SuperNode program store:

Memory used (bytes) = [(16 x 1024) + 704) x 1024 = 17498112 bytes
Memory used (megabytes] = 17498112 / 1048576 = 16.69 megabytes
Total memory = 27.93 + 16.69 = 44.62 megabytes

Individual features

Determining memory impact

A large portion of the data store memory is defined in office parameter tables. Using this list, the total data store memory required for each feature that is defined in the parameter tables can be determined or estimated by referring to *Office Parameters Reference*. Part of this reference manual (under each parameter) is a heading: "Memory Requirements" under which is listed the minimum, maximum, and default values for each feature and how the feature is activated. For example, the following figure is an excerpt from the reference manual concerning call forwarding extension blocks (CFW_EXT_BLOCKS) that are required for the Enhanced Call Forwarding-POTS feature package.

Figure 3-2
Office parameters reference manual example

```
MEMORY REQUIREMENTS


Fourteen words of memory are required for each call
forwarding block.

Minimum      0
Maximum      32767
Default      10
Activat      cold
ion          restart
```

In this example, the number of words per call forwarding extension block is 14; the minimum and maximum number of extension blocks are 0 and 32, 767, respectively, with a default value of 10. The feature is activated through a cold restart. The total amount of memory can then be calculated by multiplying the number of extension blocks by 14 (number of words per block).

Because a software feature package is made up of a number of software modules, some of these modules may be accessed by other features, and only estimates of memory impact can be made when a new feature is introduced. For example, in table OFCENG, the parameter FTRQ2WAREAS (number of FTRQ 2 word areas) can be accessed by a number of features. Therefore, the provisioning of the FTRQ2AREAS is based upon the total requirements of all the features that access this one area of memory and not the requirement of only one feature. Each feature that accesses this area accounts for only a piece of the total memory required. An absolute value for each feature is not that critical to the effective monitoring of memory utilization.

Tracking and reporting memory status using operational measurements

Memory administration requires that methods and procedures be developed for tracking memory use and availability. Methods are also required for reporting the results to those individuals making use of the information. The following paragraphs suggest one method in which OMs are used.

A manual tracking method

Many of the memory resources may be monitored and tracked indirectly by manually recording operational measurements (OMs). This is an indirect method of tracking memory because OMs count blocks of memory performing a particular function; they do not indicate how much memory is being used.

To determine the amount of memory used or remaining, these blocks must be multiplied by the number of words per block. For example, call forwarding extension block usage and adequacy can be monitored by using the operational measurement EXTHI (OM group EXT) which counts the total number of blocks in simultaneous use during the current OM transfer period.

These registers are known as high-water registers because they record the highest level of simultaneous usage during the measurement period. By comparing the output of each register group with the value in the parameter tables, the number of blocks that are being used and the number of spare blocks can be determined.

This method may be used for any of the parameters that have an associated OM provided. Refer to *Office Parameters Reference Manual*, which contains all of the parameters and associated OMs.

To determine how many blocks are currently required and the total currently provided, follow the steps listed in the following procedure.

Manual method of tracking memory

1 Determine how many blocks are provided for the item being studied by referring to the parameter table.

2 Read the busy hour _ _ _HI OM that corresponds to the memory blocks being monitored. The _ _ _HI OM register (one provided for each block group) records the maximum number of blocks in simultaneous use during the current transfer period (examples include: EXTHI for extension blocks, CCBHI for call condense blocks, and MULTHI for the number of multiblocks).

3 Divide the number obtained from the OM by the total number of blocks provided and multiply by 100. The result is the percentage of blocks provided that are being used simultaneously during the study period. For example, using the values in figure 3-3 and a parameter value of 150, the following percentage is derived: 29 (EXTHI) / 150 (parameter value) x 100 = 19.3 % of the blocks were in simultaneous usage.

Figure 3-3
Example of EXT OM group register

```
FEATURE_CONTROL BLOCK


     EXTSEI             EXTOVFL
     Z                  EXTHI

       20                   0          2
       0                               9
```

By keeping a record of these readings on a periodic basis (for example, monthly), growth trends can be developed and used as a basis to estimate when the number of blocks will exhaust. Work sheets for this purpose have been provided in Chapter 4.

The following table contains a list of the high-water registers associated with the office parameter displayed through the use of the BCSMON DUMP OMS command.

Table 3-2
High–water OM

PARAMETER	OM
AOSS_NUM_RECORDING_UNITS	EXTHI
AVCDR_RU_COUNT	EXTHI

Table 3-2
High–water OM (continued)

PARAMETER	OM
CDIV_EXT_BLOCKS	EXTHI
CFD_EXT_BLOCKS	EXTHI
DSA_RU_COUNT	EXTHI
FTRQAGENTS	FTRQHI
FTRQ2WAREAS	FTRQHI
FTRQ4WAREAS	FTRQHI
FTRQ8WAREAS	FTRQHI
FTRQ16WAREAS	FTRQHI
NTL_FTR_EXT_BLOCKS	EXTHI
KSHUNT_EXT_BLOCKS	EXTHI
NCCBS	CCBHI
NMULTBLKS	MULTHI
NO_INTL_RECORDING_UNITS	EXTHI
NO_LOCAL_COIN_EXT_BLKS	EXTHI
NO_LOCAL_COIN_EXT_BLKS	EXTHI
NO_OF_FTR_CONTROL_BLKS	EXTHI
NO_OF_FTR_DATA_BLKS	EXTHI
NO_OF_FTR_XLA_BLKS	EXTHI
NO_OF_MCDR_REC_UNITS	EXTHI
NO_OF_SC_EXT_BLOCKS	EXTHI
N0_OF_SMDR_REC_UNITS	EXTHI
N0_OF_TWC_EXT_BLKS	EXTHI
NUM_MTR_EXT_BLOCKS	EXTHI
NUM_DCR_EXT_BLKS	EXTHI
NUM_ISUP_EXT_BLKS	EXTHI
NUM_OF_BC_AMA_UNITS	EXTHI
NUM_OF_BC_LAMA_UNITS	EXTHI
NUM_OF_CCIS_INWATS_BLOCKS	EXTHI

—continued—

Table 3-2
High–water OM (continued)

PARAMETER	OM
NUM_OF_NSC_EXT_BLK	EXTHI
NUM-OF NT-RECORDING_UNITS	EXTHI
NUMCALLPROCESSES	CPLHI
NUMCPLETTERS	CPLHI
NUMCPWAKE	WAKEHI
NUMCSDDSPERMEXT	EXTHI
NUMIBNCQEXTBLK	EXTHI
NUMOUTBUFFS (TABLE OFCSTD)	OUTHI
NUMPERMEXT	EXTHI
NUMSDPATCEXTBLK	EXTHI
OOC-NUM-RU	EXTHI
TOPS_NUM_CAMA_RU	EXTHI
TOPS_NUM_RU	EXTHI
—end—	

Using other memory usage indicators

MAPCI commands

The MAPCI level commands may be used to determine memory used and availability of memory.

The CCMNT command

Memory usage indicators vary by BCS as new measurements are developed. The amount of memory being used in an NT40 may be monitored at the MAPCI level by requesting CCMNT. An example of the CCMNT output for an NT40 is shown in the following figure.

Figure 3-4
Example of CCMNT output

```
CCMNT:


 Store (KWORDS):   DS:USED = 12208K  AVAIL = 3920K
 TOTAL = 16128K PS:USED = 3224K  AVAIL = 4712K   TOTAL =
                7936.K
```

STORE SUMMARY command

The individual areas (for example, DSTEMP, PSPROT) of data store and program store may be scanned to determine the number of allocated and free blocks of memory and the total number of vast areas through the use of the command STORE SUMMARY.

BCSMON

Another useful MAPCI level feature is the Batch Change Supplement Monitoring feature (BCSMON). The command BCSMON DUMP MEMORY will give the current count of memory cards in the switch and the total amount of memory and memory available in kilobytes. Examples of the output for the NT40 and the SuperNode are shown in the following figures.

Note: If you receive the message “please use DMSMON,” replace the BCSMON command with DMSMON MEMORY.

Figure 3-5
Example of NT40 DUMP MEMORY output

```
                    ***************
                    *********MEMORY
                    ***************
                    *********

                                _________________
                                   Number of
                                   Cards
          PS SHELF:       64K      256K      1M
          ________
             0             0         0        7

          TOTAL PS        ____      ____     ____
          CARDS:           0         0        7


          DS               64K      256K      1M
          SHELF:
             0             0         0        1
                                              5
          TOTAL DS        ____      ____     ____
          CARDS:           0         0        1
                                              5
          PS:   Total   7168K,   Avail  1287K
          DS:   Total  15360K,  Avail  3496K
```

Note: To convert Mbytes to kbytes multiply the Mbytes by 1024. For example, 7M = 7 x 1024 = 7168K. Similarly, 15M = 15 x 1024 = 15360K.

DUMP MEMORY command (SuperNode)

The SuperNode DUMP MEMORY printout shown in the following figure gives a current look at the memory cards installed and the amount of memory (in megabytes) per card. In this example, cards for CPU 0, shelf 0 are indicated. Shown are nine 6M cards and one 1M card. The memory for each card is divided into three modules (or sectors). The size of these modules varies with the type of card. In this example, card 1 is a 24M card and has 3 modules of 8M each for a total of 24M (megabytes). Cards 2 through 10 are 6M cards with each card containing 3 modules of 2M.

The SPARE MODS column indicates the number of modules per card that are designated spare. For example, card 3 indicates that two of the three modules for that card are spare while card 1 indicates one module spare.

The last column indicates the card status at the time of the printout. If the card is faulty or is indicating in-service trouble (ISTB), the column would indicate a YES.

In this example, card 0 and slot 19 have no entries because the slot is not equipped and is set aside for other processor uses.

Figure 3-6
Example of SuperNode DUMP MEMORY output

```
             CPU 0:
            ---------------------------------------------------------------------------------------------------
    CARD     | EQP | SHLF | SLOT |EQPEC | RELEASE    |   MODULES  |   MEGS    |   MEGS  |
    SPARE |     FAULTY

    NO.                          PER CARD   PER MOD          MODS       OR ISTB
             ----------------------------------------------------------------------------------
            0     NO    0             -                   -      -          -
                             19    -              -                          -
                                  20                             24              NO
            1     YES   0    16 9X14DA           3      8               1
                                  21                                6            NO
            2     YES   0    15 9X14BB           3      2               3
                                  21                                6            NO
            3     YES   0    14 9X14BB           3      2               2
                                  21                                6            NO
            4     YES   0    13 9X14BB           3      2               0
                                  21                                6            NO
            5     YES   0    12 9X14BB           3      2               0
                                  21                                6            NO
            6     YES   0    11 9X14BB           3      2               0
                                  21                                6            NO
            7     YES   0    10 9X14BB           3      2               0
                                  38                                6            NO
            8     YES   0     9 9X14BB           3      2               0
                                  3N                                6            NO
            9     YES   0     8 9X14BB           3      2               0
                                  22                                6            NO
            10    YES   0     7 9X14BB           3      2               0
            ____________________________________________________________________________

                  10                                              78     6        0
```

CMMNT command (SuperNode)

A DMS-Core can be limited by total memory. Therefore, the total memory should be tracked closely. Monitoring the amount of memory being utilized can be done at the MAPCI level by requesting CMMNT. An example of the CMMNT output is shown in the following figure.

Example of CMMNT output

```
CMMNT:

MEMORY: (KBYTES):   USED = 35904   AVAIL = 5026    TO-
TAL = 40960
```

This output gives the amount of memory being used and the amount of available memory followed by the total memory currently provisioned. The output is reported in kbytes. Notice, in this example, that the USED and AVAIL values do not equal the TOTAL value. This is due to the pooling of memory in the SuperNode; that is, the spare memory is only allocated as the demand for it arises. Therefore, the TOTAL value is always greater then the sum of USED and AVAIL until all of the memory has been allocated.

By recording this switch output at regular intervals, growth trends may be developed which will aid in the accurate projection of a memory exhaust date based upon actual working office data.

Tracking memory for tables

Table description

The data associated with the hardware and software systems of the DMS-100 Family are stores in the form of two-dimensional entities called tables. A table consists of rows (horizontally) and columns (vertically). A row is called a tuple and the columns are fields within a tuple. The following figure shows the structure of a typical table, containing 1 through N tuples and 1 through N fields. The fields in a table (or subtable) have the following properties:

- Each field has a unique field name. A field name consists of a maximum eight-character string.
- Each field in a tuple has an associated field number. Fields are numbered consecutively (first field is 1, second field is 2, and so forth).
- An individual field is accessed using either of its field identifiers (field name or field number). The field name is used as a prompt for data input.
- A field is either a single element field (contains one element of data) or a multiple element field (contains several elements of data).
- Where the contents of a multiple element field may vary, each element of information is treated as a separate subfield with its own subfield identifier. This is used for:
  - lists: complete field or one or more elements within a field can be repeated to form a list (for example, the list of features associated with a particular business set). A field, element, or group of elements that can be repeated is called a vector.
  - refineable fields: where the field can include a selection of two or more groups of elements. A refineable field comprises
    - a selector subfield

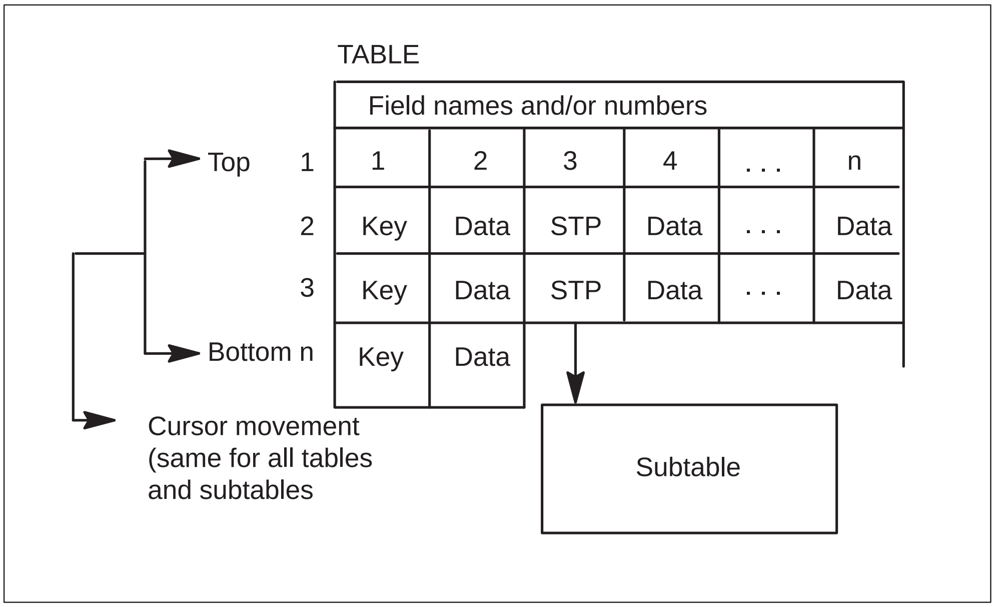
  - zero or more data subfields. The requirement for data subfields and their content is dependent on the value in the selector field.

A field or subfield may contain data expressed in the form of numerics or alpha-numeric strings.

The tuples in a table or subtable have the following properties:

- Each tuple is identified by a key that always contains the first field (field 1). For most tables, the key comprises field 1 only. Some tables use a longer key comprising field 1 plus one or more of the subsequent fields in the tuple.
- Each tuple is unique. Duplicated keys are not allowed.
- Tuples are referenced either by their key or by the table editor cursor. The cursor is an internal pointer to a tuple of a table. The cursor can be moved by commands. The tuple to which the cursor points at any given time is called the cursor tuple.
- Multi-element fields containing lists (vectors) and variable (refineable) contents cannot form part of the key.

Figure 3-7
Example of a table configuration



Some table limits are not defined in parameters; therefore, other methods must be used to track how much memory is provided and how much remains. Memory for tables can be administered through the use of the table editor (refer to *Basic Translations Tools Guide*, 297-1001-360) using the COUNT command. This command counts the total number of tuples in the table or counts tuples with specific values in designated fields. Using this command allows the user to determine the current table size (in tuples) and specific entries may be counted. The following are examples of using the COUNT command:

- command used without parameters displays the current size of the table in tuples:
  - **>count**
    - BOTTOM
    - SIZE 253
- conditional command given to count all tuples where field_2 contains a value of 4:
  - **>count (2 eq ‘4’)**
    - BOTTOM
    - SIZE 35
- conditional command given to count all tuples where field_2 contains a value of 4 or 5:
  - **>count (2 eq ‘4’) and (3 eq ‘5’)**
    - BOTTOM
    - SIZE 47

Memory alarm indicators

A low memory alarm is activated in an NT40 switch whenever total spare memory reaches five or less vast areas (160K words of data store). In a SuperNode equipped office, a low spare alarm occurs whenever spare memory reaches three 2- or 8-Mbyte blocks (the equivalent of a reliability spare card). A low memory alarm occurs when only five or less spare data store vast areas or two or less program store vast areas (64K bytes per vast area) remain.

Memory sparing

The NT40 and DMS-Core (SuperNode) sparing philosophies are different in their premise: the NT40 utilizes a 256K-word block (always provided) for both the program store and the data store (NT4X79AA memory controller card). This reliability spare can take the place of any 256K-word block of memory from a 256K-word NT3X93AA or 1 M-word NT4X80AA card that encounters a single fault.

The DMS-Core, however, must be provisioned for reliability spare. DMS-Core memory is spread evenly over all cards in the memory shelf, so there is no physical spare. When a card fails, the store allocator automatically reconfigures memory over the remaining cards without service interruption.

Note 1: For a SuperNode, the OM SPAREMB and SPAREKB values should be monitored to ensure that spare memory does not fall below three memory modules (a module is defined as one third of a physical CM memory pack).

Note 2: AVAIL memory is not part of the SPARE memory values.

Memory administration recommendations when memory use exceeds the forecast

When plotting the memory requirements on the memory tracking work sheets, the slope of the actual memory usage may suddenly increase or exceed the MEMCALC forecast. If this occurs, administrative procedures must be activated to recalibrate the forecast. The MEMCALC program should be run again with current actual switch parameter data which may produce a different memory exhaust date.

As the operative data store gets closer to its exhaust (90 percent of addressable memory), daily tracking of memory is recommended and plans must be implemented to most efficiently use the remaining memory. When memory is equal to or less than ten percent, the operating company must have a plan to establish memory relief, for example, a memory extension or an upgrade to DMS-SuperNode. Resizing key office parameters and possibly the removal of unused features may be involved to provide spare memory. Contact Northern Telecom’s Systems Software Engineering representative for assistance. If these steps are taken, MEMCALC should again be run with the new parameter settings to adjust the MEMCALC forecasts.

If the maximum NT40 capacity is reached and the switch is upgraded to SuperNode, then deloading of the parameters and features in the NT40 load may be required to insert the new SuperNode load. Contact the operating company’s Northern Telecom Systems Software Engineering representative, who can provide parameter reduction procedures.

Memory tracking work sheets

Using memory tracking work sheets

This chapter presents sample work sheets that can be used to track data store and program store memory usage and availability. Instructions for completing the work sheets are provided, and examples of typical work sheet entries are also included.

Work sheet instructions

Data store

The data store memory tracking work sheet is provided to record the usage and availability of memory through the output from the high-water operational measurements compared to the associated parameters for the feature being tracked.

Data store memory tracking (high-water) work sheet instructions

This work sheet is for recording the high-water operational measurements (OM) that are associated with particular parameters for features or options and comparing the OM values with the current values of the related parameters. The result is a percent-used value of the resource being measured. This data store work sheet (figure 4-1) should be completed per procedure 4-1. See figure 4-2 for an example.

Procedure 4-1. Completing the data store tracking (high–water) work sheets

1 In column 1, enter the feature or option that is being measured. This is a feature or option that has a high-water OM register associated with it.

2 In column 2, enter the feature or option parameter designation.

3 In column 3, enter the parameter size of the feature or option. This figure may be obtained from the parameter tables record.

4 In column 4, enter the busy hour (determined through the busy hour studied) OM count for the option or feature under study.

5 In column 5, enter the number of remaining resources by subtracting the usage count (column 4) from the parameter size of the feature or option (column 3).

6 In column 6, enter the percentage of the parameter size that is being used during the busy hour by dividing the number remaining (column 4) by the parameter size (column 3) and multiplying the result by 100.

Data store tracking work sheet (actual/installed) instructions

The data store tracking work sheet (actual/installed) is provided to track the actual memory usage in data store. It is expressed in a percentage of the total data store memory installed. This work sheet provides for the recording of the memory usage percentage on a monthly basis. However, if available memory has become critical, it may require recording at shorter intervals until more memory has been installed.

Procedure 4-2. Completing the data store tracking work sheets (actual/installed)

At the MAPCI level:

1 Request the CCMNT (NT40) output (refer to figure 4-4 for an example of the output).

2 From the CCMNT output, divide the DS:USED value by the TOTAL value and multiply the result by 100 to convert the values to a percentage.

3 On the work sheet, locate the percentage value from step 2 and mark the point on the vertical line that corresponds to the month when the value was derived.

4 Each month, repeat steps 1 through 3 and join the points to develop usage trends (see figure 4-4).

Program store tracking work sheet (actual/installed) instructions

Program store memory usage is static until additional features are added or if program patches are installed. Patches are temporary corrections to an operating program. The program store work sheet (figure 4-5) described here records the total memory installed and the impact as new features are introduced to the switch. Figure 4-6 gives an example of what this record might look like for a working office. Refer to procedure 4-3 for instructions on using this form.

Procedure 4-3. Completing the program store tracking work sheets (actual/installed)

At the MAPCI level:

1 Request the CCMNT (NT40) output. Refer to figure4- 6 for an example of the output.

2 From the CCMNT output, divide the PS:USED value by the TOTAL value and multiply the result by 100 to convert the values to a percentage.

3 On the work sheet, locate the percentage value from step 2 and mark the point on the vertical line that corresponds to the month in which the value was derived. This value will remain the same until a patch is performed or features are added or removed from the switch. Refer to figure 4-6.

Total memory tracking work sheet (actual/installed) instructions

The total memory tracking work sheet actual/installed (figure 4-7) is provided to track the combined data store and program store memory usage and availability for SuperNode equipped switches. See figure4- 8 for an example. Refer to procedure 4-4 for instructions on using this form.

Procedure 4-4. Completing the total memory tracking work sheets

1 Enter the feature or option that is being measured. This is a feature or option that has a high-water OM associated with it.

2 Add the USED and AVAIL values.

3 Divide the sum in step 2 by the TOTAL value and multiply by 100. The result is the percentage of memory used of the total memory provisioned.

4 On the work sheet, locate the percentage value from step 3 and mark the point on the vertical line that corresponds to the month in which the value was derived.

5 Each month, repeat steps 1 through 4 and join the points to develop usage trends. Refer to figure 4-8 for an example of a working office.

Figure 4-1
Data store memory tracking (high water) work sheet

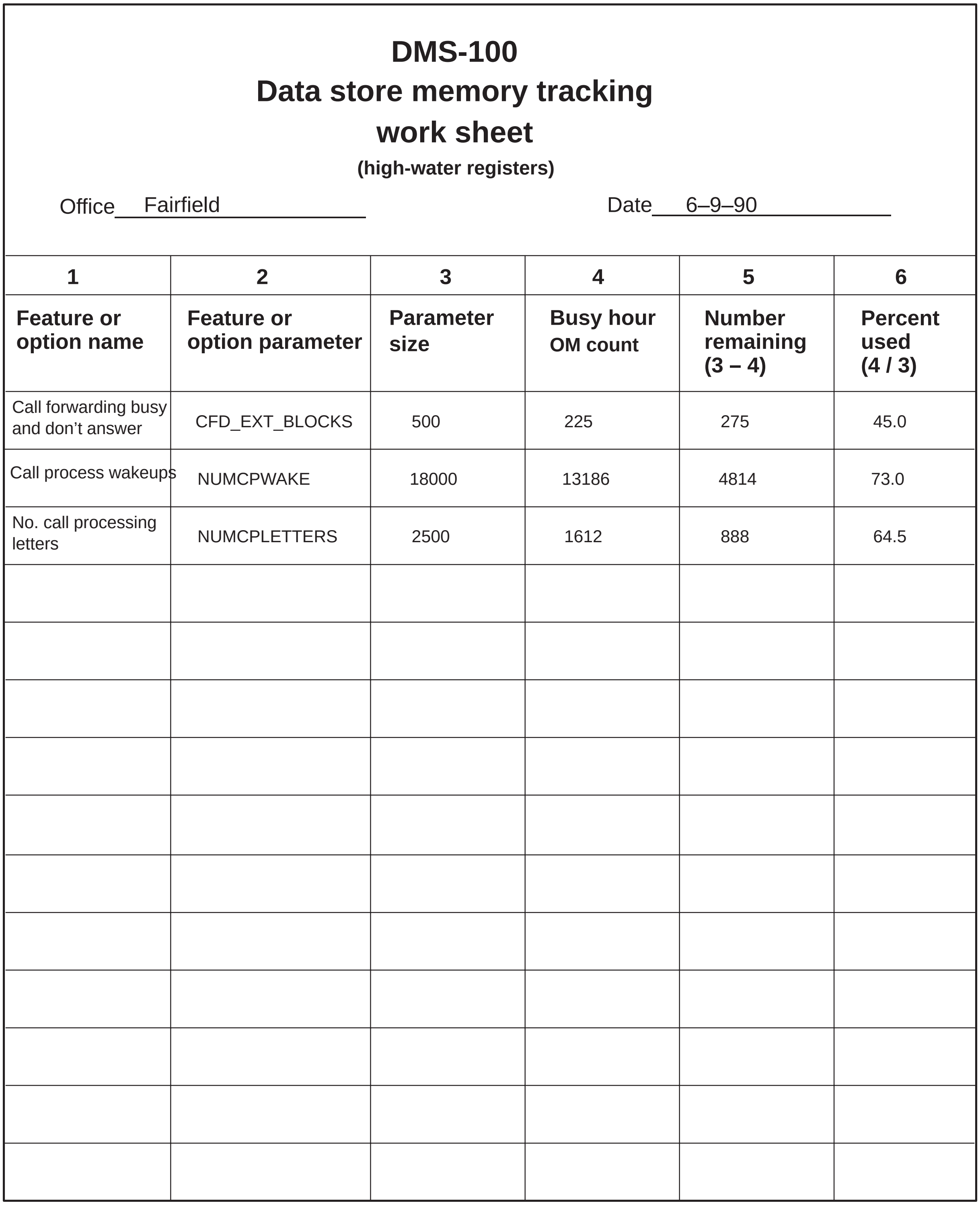
DMS-100
Data store memory tracking work sheet

(high-water registers)

Office______________________ Date_____________________

1	2	3	4	5	6
Feature or option name	**Feature or option parameter**	**Parameter size**	**Busy hour OM count**	**Number remaining (3 – 4)**	**Percent used (4 / 3)**

Figure 4-2
Example of data store memory tracking (high water) work sheet entries



DMS-100
Data store memory tracking work sheet

(high-water registers)

Office Fairfield

Date 6–9–90

1	2	3	4	5	6
Feature or option name	**Feature or option parameter**	**Parameter size**	**Busy hour OM count**	**Number remaining (3 – 4)**	**Percent used (4 / 3)**
Call forwarding busy and don’t answer	CFD_EXT_BLOCKS	500	225	275	45.0
Call process wakeups	NUMCPWAKE	18000	13186	4814	73.0
No. call processing letters	NUMCPLETTERS	2500	1612	888	64.5

Figure 4-3
Total data store memory tracking work sheet

DMS-100
Memory tracking—data store

(actual /installed)

%	Monthly busy hour % memory used											
	Memory installed:											
100												
95												
90												
85												
80												
75												
70												
65												
60												
55												
50												
45												
40												
35												
30												
25												
20												
	J	**F**	**M**	**A**	**M**	**J**	**J**	**A**	**S**	**O**	**N**	**D**

OFFICE_____________
—

YEAR____________
___

Figure 4-4
Example of total data store memory tracking work sheet entries

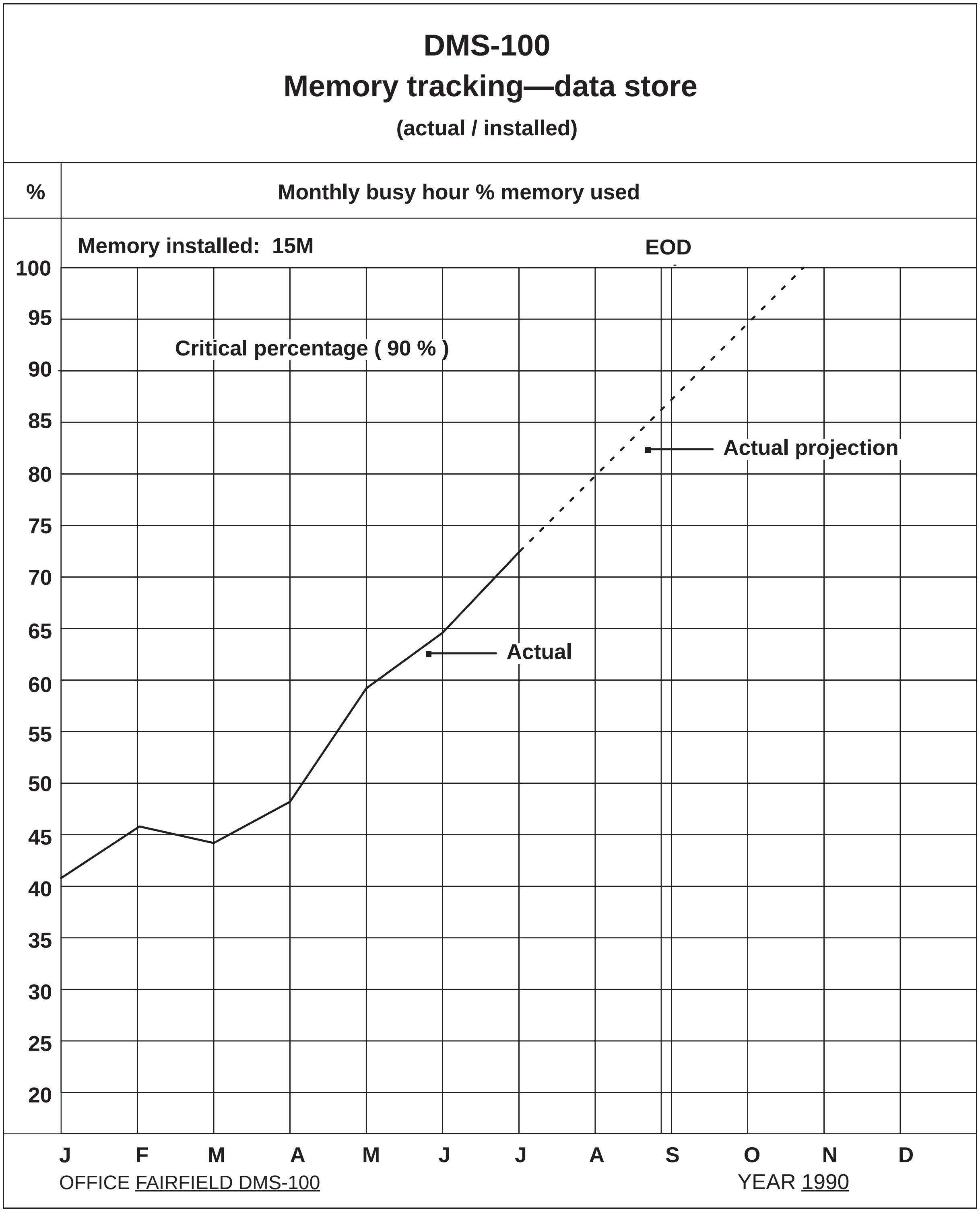


Figure 4-5
Program store memory tracking work sheet

DMS-100
Memory tracking—program store

(actual / installed)

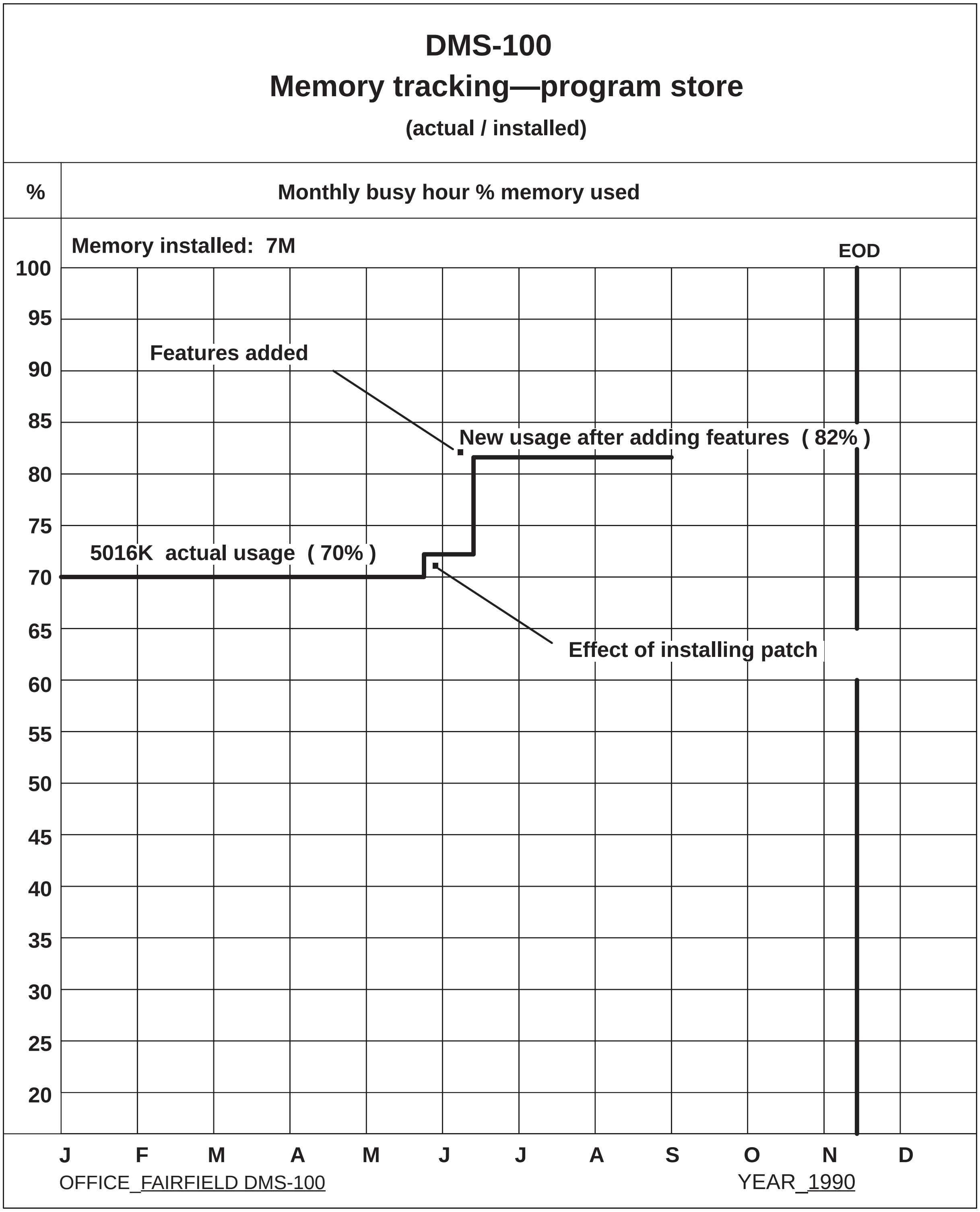
% **Monthly busy hour % memory used**

Memory installed:

100												
95												
90												
85												
80												
75												
70												
65												
60												
55												
50												
45												
40												
35												
30												
25												
20												
	J	**F**	**M**	**A**	**M**	**J**	**J**	**A**	**S**	**O**	**N**	**D**

OFFICE______________ YEAR_____________

Figure 4-6
Example of program store tracking work sheet entries



DMS-100
Memory tracking—program store
(actual / installed)

OFFICE_FAIRFIELD DMS-100 YEAR_1990

Figure 4-7
Total memory tracking work sheet

DMS-100 (SuperNode)
Memory tracking—total memory

(actual / installed)

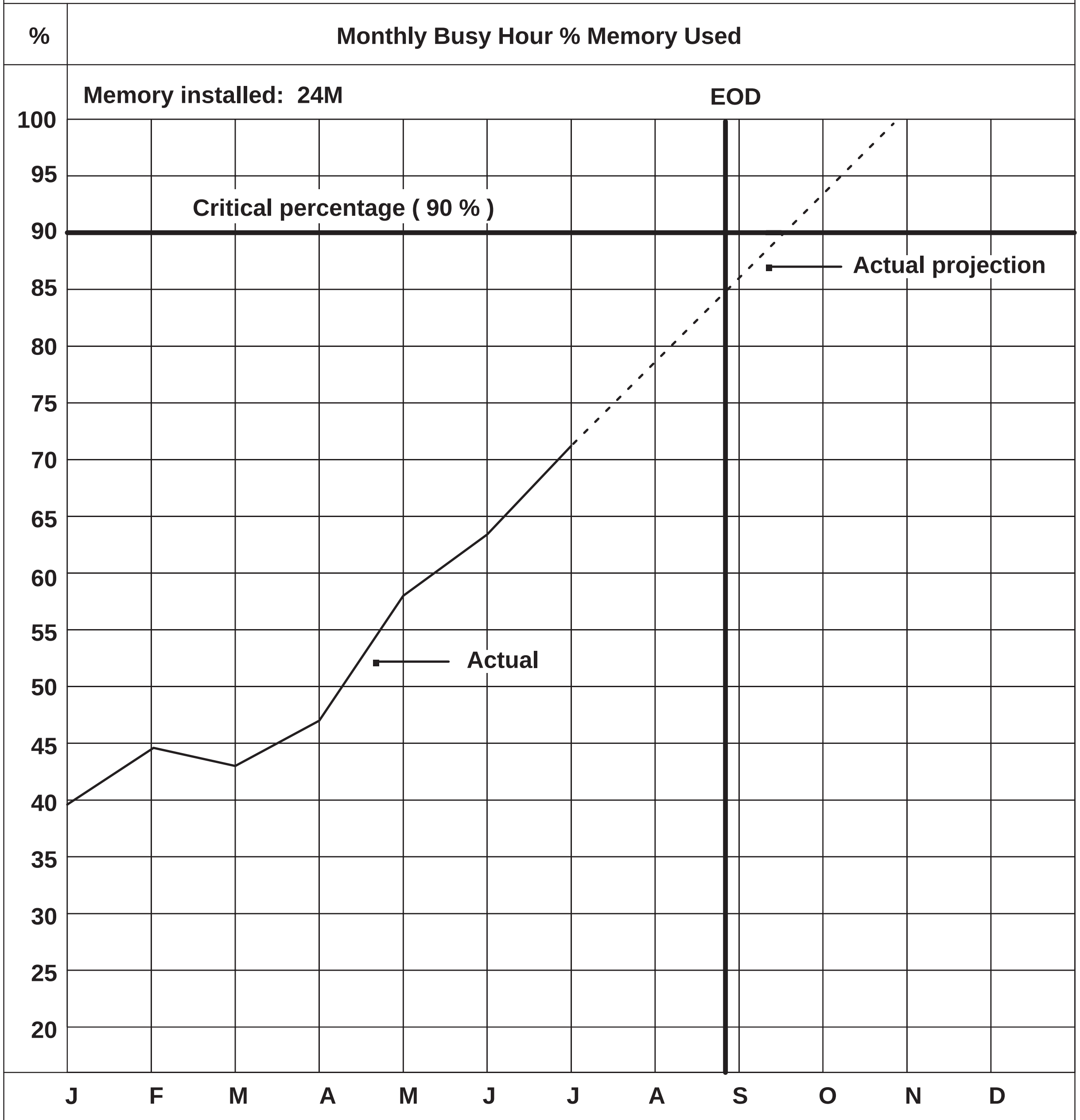
%	Monthly busy hour % memory used											
	Memory installed:											
100												
95												
90												
85												
80												
75												
70												
65												
60												
55												
50												
45												
40												
35												
30												
25												
20												
	J	**F**	**M**	**A**	**M**	**J**	**J**	**A**	**S**	**O**	**N**	**D**

OFFICE______________ YEAR____________

Figure 4-8
Example of total memory tracking work sheet entries

DMS-100 (SuperNode)
Memory tracking—total memory

(actual / installed)



OFFICE_FAIRFIELD DMS-100 YEAR_1990

Index

DMS-100 Family

DMS100 Family

Memory Administration Guide

Product Documentation–Dept 3423
Northern Telecom
P.O. Box 13010
RTP, NC 27709–3010
1-877-662-5669, Option 4 + 1

NORTHERN TELECOM CONFIDENTIAL: The information contained in this document is the property of Northern Telecom. Except as specifically authorized in writing by Northern Telecom, the holder of this document shall keep the information contained herein confidential and shall protect same in whole or in part from disclosure and dissemination to third parties and use same for evaluation, operation, and maintenance purposes only.

Information is subject to change without notice. Northern Telecom reserves the right to make changes in design or components as progress in engineering and manufacturing may warrant.This equipment has been tested and found to comply with the limits for a Class A digital device pursuant to Part 15 of the FCC Rules, and the radio interference regulations of Industry Canada. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at the user's own expense.

The SL-100 system is certified by the Canadian Standards Association (CSA) with the Nationally Recognized Testing Laboratory (NRTL).

This equipment is capable of providing users with access to interstate providers of operator services through the use of equal access codes. Modifications by aggregators to alter these capabilities is a violation of the Telephone Operator Consumer Service Improvement Act of 1990 and Part 68 of the FCC Rules.

Publication number: 297-1001-305
Product release: NA004 and up
Document release: Standard 04.04
Date: August 1999
Printed in the United States of America

NORTEL
NORTHERN TELECOM