MOLECULAR COMPUTER

MULTIPROCESSOR COMPUTER SYSTEMS

n/STAR
Network Operating System

Programmer's Reference Manual

Version 2.240
December 14, 1981

CONTENTS

PREFACE

This manual provides the information necessary to implement custom software under the n/STAR Network Operating System.

It is recommended that new users first become familiar with CP/M and n/STAR by reading the CP/M manuals prior to reading this document. This manual is organized into the following sections:

Section I -	Describes the internal memory layout and structure of n/STAR, along with functional specifications of the Boot sequence and Sector Buffer Pooling.
Section II -	Provides details on n/STAR File and Record locking capabilities, including additional BDOS calls available for this purpose.
Section III -	Contains details on the Disk and Bus Driver interface and calling sequences.
Section IV -	Presents the hardware I/O port assignments for the Application Processor and File Processor boards with details on I/O programming.

SECTION I
n/STAR STRUCTURE

1.1 INTRODUCTION

The n/STAR Network Operating System is a "CP/M Derivative" operating system. It presents an environment to the application program that is compatible with Digital Research CP/M version 2.2.

The n/STAR Network Operating System has been designed specifically for multiprocessor computer systems manufactured by MOLECULAR Computer, and takes advantage of their unique hardware features. No method is provided to implement n/STAR on other computer systems.

In order to provide the greater efficiency available in the Z80A environment, the n/STAR Network Operating System is coded in the Z80A native instruction set, rather than the limited 8080 instruction set.

MOLECULAR Computer systems are composed of chiefly two types of single-board computers: the File Processor (FP), and one or more Application Processors (AP). The File Processor performs all of the physical disk I/O and file handling functions while the Application Processor runs the user's application program. Thus, each user on the system has a dedicated Application Processor for program execution. Communication between the Application Processors and the File Processor is handled over a unique high speed bus, the MegaBUS Interprocessor Link.

Bus transfers are initiated automatically by n/STAR for disk related BDOS calls. However, application programs may perform bus transfers directly to utilize custom system resources. These bus transfers may be made from an Application Processor to the File Processor, or directly to another Application Processor.

The Bus Driver routines utilize Z80A Interrupt Mode 2. For this reason application programs must not disable interrupts or change the processor interrupt mode. Applications may, however, use Z80A Interrupt Mode 2 and fixed vectors for the various devices on the processor board provided for this purpose.

Applications request service from n/STAR via standard CP/M BDOS calls through the vector at location 0005H. All BDOS functions present in CP/M version 2.2 are supported, except function 13, Reset Disk, which is ignored in order to preserve data integrity in the multiple user environment.

In addition to the CP/M BDOS functions, two additional calls are supported. The added functions, 42 and 43, Lock and Unlock Record, are useful for implementing shared file applications where individual records need to be protected during update without locking the entire file.

1.2 INTERNAL LAYOUT

Upon completion of the Boot Load sequence, the File Processor memory space is structured as follows:

FFFFH **F800H**	CONTENTS OF EPROM (Driver, BOOT routines, and Interrupt Vectors)
F7FFH **7800H**	DISK BUFFER POOL
77FFH **6800H**	AP BOOT IMAGE
67FFH **6500H**	BUFFERED DISK DRIVER
64FFH **4100H**	HUB MODULE
40FFH **0100H**	AP REQUEST TABLE
00FFH **0000H**	POST BOOT BLOCK

Location 0000H is the start of the Post Boot Block. It contains the Post Boot routines for both the Application Processor and the File Processor.

At location 0100H is the start of the Application Processor Request Table. Each Application Processor in the system has a unique 64 byte entry in the table. There are 256 entries in the table.

The HUB Module starts at location 4100H. All file-oriented BDOS calls generated by each Application Processor in the system are serviced by the HUB Module.

The Buffered Disk Driver at location 6500H performs the dual functions of managing the 32K buffer pool and deblocking the 1024 byte physical sectors to 128 byte blocks.

The information contained between location 6800H and 7800H is a static image of the Application Processor routines which are loaded into the Application Processor on each individual user's cold or warm start. By retaining the Application Processor boot image in RAM, the Application Processor boot operation is much faster and does not compete with disk data I/O.

The Disk Buffer Pool occupies the area above the Application Processor Boot Image from location 6800H to F800H.

Location F800H to FFFFH is reserved in both the File Processor and the Application Processor for the EPROM-Resident driver routines and Interrupt Vectors. In addition to the parity error handler and bus drivers, this area in the file Processor also contains the disk drivers.

The Application Processor memory is structured as follows:

FFFFH **F800H**	CONTENTS OF EPROM (Driver Routines, BIOS and Interrupt Vectors)
F7FFH **EC06H**	STAR MODULE
EC05H **E400H**	COMMAND PROCESSOR (part of Star Module)
E3FFH **0100H**	APPLICATION AREA
00FFH **0000H**	SYSTEM PARAMETERS

The system parameter area begins at address 0000H. This is equivalent to the system parameter area in CP/M. It contains the warm start and BDOS entry vectors as well as the default sector buffer and FCB.

The next section is the Application Program Area. This is the portion of RAM available to application programs which are loaded by n/STAR.

Directly above, at location E400H, is the Command Processor. This program interprets the n/STAR commands. Application programs may overlay this module to obtain more RAM space.

Next is the STAR module. This module processes all BDOS calls from the application programs. Only the console I/O operations are actually performed by this module. All disk- related functions are transferred by this module to the Hub module in the File Processor via a bus request.

At the top of memory is the EPROM module containing the Bus Driver, Memory Parity Error handling, Bootstrap routines and Interrupt Vectors. Space is also reserved within the EPROM area for a simulated CBIOS Jump-Vector table. This permits programs to perform direct console I/O via the Jump-Vector Table. Any disk-related Jump-Vector calls result in an error trap since there is no physical I/O link from the Application Processor to the disk.

1.3 INTERRUPT VECTORS

Fixed Interrupt Vectors are provided within the EPROM space and may be used by application programs requiring interrupt-driven I/O. As part of the initialization sequence, the EPROM routine loads the Z80A "I" register with F8H and sets the individual vector registers within the various on-board devices to point to their respective fixed vectors.

Since the EPROM routines reside in RAM, the vectors may be loaded with the addresses of user-supplied service routines. The application program need only issue the Enable Interrupts command to the specific device to utilize the interrupt feature.

Note: The DMA and CTC vectors on the File Processor and Application Processor as well as one of the File Processor CTC vectors are not available for use by applications since they perform part of the Bus Driver function.

The Fixed Interrupt Vectors for the File Processor are as follows:

ADDRESS	VECTOR
F80C	PIO Channel A
F80E	PIO Channel B
F810	SIO Vector 0
F812	SIO Vector 1
F814	SIO Vector 2
F816	SIO Vector 3
F818	SIO Vector 4
F81A	SIO Vector 5
F81C	SIO Vector 6
F81E	SIO Vector 7
F820	CTC A, Channel A
F822	CTC A, Channel B
F824	CTC A, Channel C
F826	CTC A, Channel D
F828 - F82E	CTC B (Do Not Use)
F830	DMA (Do Not Use)

The Fixed Interrupt Vectors for the Application Processor are as follows:

ADDRESS	VECTOR
F80C	PIO Channel A
F80E	PIO Channel B
F810	DMA (Do Not Use)
F818 - F81E	CTC (Do Not Use)
F820	SIO Vector 0
F822	SIO Vector 1
F824	SIO Vector 2
F826	SIO Vector 3
F828	SIO Vector 4
F82A	SIO Vector 5
F82C	SIO Vector 6
F82E	SIO Vector 7

1.4 BOOT SEQUENCE

The boot sequence has been designed to allow loading custom software without requiring a change to the system EPROM programs. This is accomplished using a modular bootstrap loading sequence.

The bootstrap loading sequence for both the File Processor and the Application Processor begins in the EPROM. After the processor is reset, or the system is powered on, the EPROM is physically mapped into all addressable memory space repeating at every 2K byte boundary. The first operation performed by the EPROM is to copy its contents into RAM at location F800H, jump to the next location relative to that address, disable the EPROM and continue running in RAM.

The next operation performed is processor initialization. This involves setting up all of the programmable peripheral controllers on the processor board (SIO, CTC, etc.) and initializing the parity RAM with the correct pattern. Following processor initialization, the bootstrap load operation is performed. The bootstrap load operation differs between the File Processor and Application Processor.

In the File Processor, the EPROM first sets the bus Busy indicator, which will prevent any Application Processor from attempting to boot before the File Processor has completed its boot sequence. Following the busy indication, processor initialization is completed and the first sector of the disk is read into location 0000H in the File Processor RAM. The first 128 bytes of this sector contain the Post-Boot block which is structured as follows:

007FH **007DH**	FP POST BOOT VECTOR
007CH variable	FP POST BOOT ROUTINE
variable **0003H**	AP POST BOOT TABLE
0002H **0000H**	AP ENTRY VECTOR

The File Processor then jumps to the Post Boot Vector at location 007DH which transfers control to the beginning of the FP Post Boot Routine. This routine reads the File Processor image into RAM at the proper address and transfers control to that address.

The n/STAR system loads at location 4100H and, on entry, initializes the request table and logs in all logical drives. When this is done, it clears the bus Busy indicator, and waits for Application Processor service requests.

The Application Processor bootstrap operation does not involve disk I/O. After processor initialization, the bootstrap routine in the Application Processor EPROM performs a bus request to obtain the post boot block from the File Processor, and places it at location 0000H in the Application Processor. This operation does not occur until the bus Busy indicator has cleared.

When the initial bus request has completed, another request is issued using the table now at location 0003H. This table contains the address of the Application Processor image in File Processor RAM, and the intended address in the Application Processor RAM, at which the image is to be placed. Upon completion, the Application Processor jumps to the Entry Vector at location 0000H. This transfers control to the newly loaded Application Processor image.

This procedure enables the single Post Boot Block to contain all the variable information needed to load the entire system.

1.5 DISK BUFFER POOLING

All hard disk I/O is performed by the Buffered Disk Driver. This routine manages a 32K byte sector buffer pool which serves to eliminate redundant disk I/O, thereby greatly improving the performance of the system, without detracting from the user's 64K memory space. The driver is also responsible for converting logical 128 byte sectors to physical 1024 byte sectors.

The Hub module makes disk requests to read or write logical 128 byte sectors. The Buffered Disk Driver first determines which physical 1024 byte sector contains the requested block, then checks to see whether that block is already in the 32K buffer. If it is, the data is transferred to or from the buffer with no disk I/O.

If the sector is not in the buffer, a check is made to see if there is an unused block available in the buffer. If not, the least recently used block is taken and its contents are written to the disk if there is a pending write request. The requested sector is then read in from the disk and that block is marked as the most recently used.

If the requested operation is to write to an unused portion of the disk, the pre-read operation is not performed. Also, each request to update the directory flushes all outstanding write requests to ensure that the disk is properly updated.

SECTION II
n/STAR FILE AND RECORD LOCKING

2.1 INTRODUCTION

The n/STAR multiple user environment allows different users to access data simultaneously in three different ways. These are termed "File Locking", "CP/M File Sharing", and "Enhanced File Sharing".

2.2 FILE LOCKING

The default option is called "File Locking" mode. Multiple users may read from the same file, but as soon as any user writes to the file, no other user may write to that file until the first user closes it. No special programming is required at this level.

2.3 CP/M FILE SHARING

The next option is called "CP/M File Sharing". This is provided for standard CP/M programs. It is invoked by the "SHARE" command, which sets F5′ in the FCB. Subsequent reads to the file cause the record which is read to be locked automatically, so that it cannot be read by another user. The record is unlocked when the first user writes the record (updates it), reads another record, or closes the file. The "SHARE" command should be used only for files which are to be processed randomly, and not sequentially. Also, in the case of indexed files, large portions of the file may become locked due to the locking of an index record when it is read. No special programming is required at this level.

2.4 ENHANCED FILE SHARING

The third option is for users who wish to tailor their application programs to take full advantage of the n/STAR record locking facilities, and is called "Enhanced File Sharing". This level is invoked by the "Unlock" command, which sets F6′ in the FCB. Alternatively, the application program may set F6′ in the FCB by calling BDOS function 30. Once the file has been unlocked in this way, it is the responsibility of the application program to control the locking and unlocking of records through the use of BDOS calls described below. This level provides for the most efficient file sharing implementation, and is recommended for applications which use indexed files and for any new application programs which are written, to share files under n/STAR.

Record locking in "Enhanced File Sharing" mode is provided as follows:

FUNCTION 42: LOCK RECORD	
Entry Parameters:	
Register C:	**2AH**
Registers DE:	**FCB Address**
Returned Value:	
Register A:	**Return Code**

The Lock Record function allows the application program to "own" individual records (sectors) within a random-access file, thereby preventing other application programs within the system from accessing the record while it is being updated. The File Control Block (FCB) must refer to a file which has been declared "unlocked" by setting indicator F6'. The FCB must also contain the 24 bit random record number (bytes r0, rl, and r2). The indicated record must reside in an allocated block of the file.

The Lock Record function verifies that the indicated record has not been locked by another application program before proceeding to attempt a record lock. All locked records for a given application program are freed by the Close File function or upon warm start.

The Lock Record function returns zero in register A if the operation was successful or 08H if the record is already locked by another application program. If the file is not declared "unlocked", locking will not be performed, and register A will always contain zero.

FUNCTION 43: UNLOCK RECORD	
Entry Parameters:	
Register C:	**2BH**
Registers DE:	**FCB Address**
Returned Value:	
Register A:	**Return Code**

The Unlock Record function allows the application to "free" records previously locked with function 42. Entry parameters are similar to function 42. The record referred to by the random record count (bytes r0, r1, and r2) must have been previously locked with function 42. Register A always contains zero.

SECTION III
DRIVER INTERFACE GUIDE

3.1 INTRODUCTION

This section provides detailed information regarding application or custom operating system calls to the basic Disk and Bus Driver EPROM routines.

3.2 DISK DRIVERS

Applications running in the File Processor may access the EPROM- resident Disk Driver routines by simply loading register pair H&L with the address of a user-supplied request block and calling the vector at location F806H. Access to the Diskette Driver is provided using the same request block format but by calling location F809H. These routines save the contents of registers B&C and D&E. The format of the request block for both disk and diskette drivers is as follows:

NAME	LEN	FUNCTION	DISK (F806H)	DISKETTE (F809H)
OPR	1 byte	Disk OP Code	(see below)	(see below)
DRV	1 byte	Drive Address	(0-3)	(0)
CYL	2 byte	Cylinder Address	(0-554)	(0-76)
TRK	1 byte	Track (Head)	(0-2)	(0-1)
SEC	1 byte	Sector Address	(0-17)	(1-26)
CNT	1 byte	Sector Count	(1-255)	(N/A)
BUF	2 byte	Buffer Address	(0-FFFFH)	(0-FFFFH)

Disk (F806H) OP Codes are:

OPCODE	FUNCTION
A8	Format Disk and build Defect Map
42	Write Data
52	Write Data, Retry on Error
43	Read Data
53	Read Data, Retry on Error
83	Sequence Up and Return
82	Sequence Up and wait for Completion
81	Sequence Down and Return
AB	Initialize Disk with pattern in Buffer
A3	Verify Disk

Diskette (F809H) OP Codes are:

OPCODE	FUNCTION
02	Read Data, Retry on Error
03	Write Data, Retry on Error

On return from the call, a zero condition indicates successful completion: otherwise, the following status is returned in register A:

DISK (F806H):

DRIVE		TYPE		CODE					
D7	D6	D5	D4	D3	D2	D1	D0		
X	X	0	0	0	0	0	0	= 00	= SUCCESSFUL COMPLETION
X	X	0	0	0	0	0	1	= 01	= MOTION RETRY
X	X	0	0	0	0	1	0	= 02	= DATA RETRY
X	X								
X	X	0	1	0	0	0	1	= 11	= CRC ERROR
X	X	0	1	0	0	1	0	= 12	= DRIVE SEEK FAULT
X	X	0	1	0	0	1	1	= 13	= DRIVE FAULT
X	X	0	1	0	1	0	1	= 15	= CYLINDER MISMATCH
X	X	0	1	0	1	1	0	= 16	= INITIALIZATION COMPLETE
X	X	0	1	0	1	1	1	= 17	= STACK ERROR
X	X	0	1	1	0	0	0	= 18	= HARDWARE TRAP
X	X	0	1	1	0	0	1	= 19	= READ LOSS SYNCHRONIZATION
X	X	0	1	1	0	1	0	= 1A	= RAM FAILURE
X	X	0	1	1	0	1	1	= 1B	= ID BUFFER FAILURE
X	X								
X	X	1	0	0	0	0	0	= 20	= DRIVE NOT READY
X	X	1	0	0	0	0	1	= 21	= WRITE PROTECT
X	X	1	0	0	0	1	0	= 22	= DRIVE NOT PRESENT
X	X	1	0	0	0	1	1	= 23	= SECTOR SIZE INVALID
X	X	1	0	0	1	0	0	= 24	= ALTERNATE AREA OVERFLOW
X	X								
X	X	1	1	0	0	0	0	= 30	= SECTOR NOT FOUND
X	X	1	1	0	0	0	1	= 31	= SMART COMMAND REJECT
X	X	1	1	0	0	1	0	= 32	= DRIVE BUSY TIME-OUT
X	X	1	1	0	0	1	1	= 33	= DATA TRANSFER TIME-OUT
X	X	1	1	0	1	0	0	= 34	= INVALID CYLINDER/HEAD
X	X	1	1	0	1	0	1	= 35	= INVALID DRIVE NUMBER
X	X	1	1	0	1	1	0	= 36	= INVALID SECTOR NUMBER
X	X	1	1	0	1	1	1	= 37	= COMMAND ALREADY IN PROGRESS
X	X	1	1	1	0	0	0	= 38	= COMMAND DOUBLE WRITE
X	X	1	1	1	0	0	1	= 39	= DRIVE COMMAND REJECT
X	X	1	1	1	0	1	0	= 3A	= MULTISECTOR OPERATION ERROR

0	0	= DRIVE 0
0	1	= DRIVE 1
1	0	= DRIVE 2
1	1	= DRIVE 3

DISKETTE (F809H):

0 = Successful, Non Zero = Hard Failure

As an example, the assembler routine to read sector 4 from cylinder 10, head 1 on drive zero would be as follows:

```
DISK        EQU     0F806H
;
READ:       LXI     H,RQBLK
            CALL    DISK
            JNZ     ERROR

;
RQBLK:      DB      53H             ;READ SECTOR
            DB      0               ;DRIVE ZERO
            DW      10              ;CYLINDER 10
            DB      1               ;HEAD 1
            DB      4               ;SECTOR 4
            DB      1               ;READ 1 SECTOR
            DW      BUFFER

;
BUFFER:     DS      1024
```

3.3 BUS DRIVER

The processor boards within the system communicate via a high-speed local network called the MegaBUS Interprocessor Link. It employs the contention access protocol termed CSMA/CD (Carrier Sense Multiple Access with Collision Detection). This approach eliminates the overhead associated with other network architectures, such as those based on a polling scheme.

Externally, the bus communication between processors appears as a simple DMA transfer to or from the target processor's RAM. The processor requesting a transfer may either "send" a buffer to any other processor on the bus, or it may request a buffer from any other processor.

Bus transactions are performed by calling the EPROM Bus Driver routine at location F803H with registers H&L containing the address of a user-supplied request block in the following format:

NAME	LEN	FUNCTION	VALUE
OPR	1 Byte	Operation; 0 = Send, 1 = Receive	
UAD	1 Byte	Target Unit Address	(0000H - 00FFH)
SRC	2 Byte	Source Buffer Address	(0000H - FFFFH)
DST	2 Byte	Destination Buffer Address	(0000H - FFFFH)
LEN	2 Byte	Buffer Length	(0002H - 0800H)

The Operation Code is always either 1 or 0. Operation 1, Receive, transfers the specified buffer from the source address in the target processor's RAM to the destination address in the requesting processor's RAM. Operation 0, Send, transfers the specified buffer from the source address in the requesting processor's RAM to the destination address in the target processor's RAM.

Due to timing requirements, the maximum buffer size is 2048 bytes. DMA chip restrictions limit the minimum buffer size to 2 bytes.

The Unit Address is a single byte representing the Unit Address Number of the Target Processor. The File Processor is always unit 255 (FFH) and an Application Processor may be any other unit number. Specific Application Processor unit addresses may be determined by observing the "INITIALIZING..." message containing the unit number at power-up.

The unit address for any processor is stored in that processor's RAM at location F802H and may be referenced by the application program, if desired, but it must NOT be altered.

The Source, Destination, and Length are absolute values corresponding to the address of the buffers and their length in bytes respectively.

An example of an assembler routine call to the bus driver to retrieve the Post Boot Block from the File Processor follows:

```
MBXFR      EQU      F803H
;
BOOT:      LXI      H,RQBLK
           CALL     MBXFR
           JNZ      ERROR
;
RQBLK      DB       1         ;RECEIVE POST-BOOT BLOCK
           DB       OFFH      ;FROM FP
           DW       0         ;FROM 0000H IN FP
           DW       0         ;TO 0000H IN OUR RAM
           DW       128       ;128BYTES
```

Upon return from the Bus Driver routine, register A contains zero and the zero flag is set if the operation was successful, otherwise a non-zero condition indicates that the target processor did not respond to the call. A non-zero return usually means the target unit address does not exist in the system.

SECTION IV
HARDWARE I/O PORT ASSIGNMENTS

4.1 INTRODUCTION

This section contains a list of the hardware Input/Output devices available on the File Processor and Application Processor boards along with their port address assignments and usage.

For detailed information on the programming of specific Z80A peripheral devices, it is recommended that the reader refer to the appropriate Z80A technical manual.

The devices available on the File Processor processor board are:

Z80A FAMILY	
1 CTC	(for applications use)
1 CTC	(used for baud rate and interrupt control)
1 SIO	(used for serial interface ports)
1 PIO	(used for bus interface)
1 DMA	(used for bus transfers)

Other
WD1793 Diskette Controller
Bi-directional 16 Bit TTL Parallel Port
TTL Hard Disk Interface

The devices available on the Application Processor board are:

Z80A FAMILY	
1 CTC	(used for baud rate and interrupt control)
1 SIO	(used for console and printer ports)
1 PIO	(used for bus interface)
1 DMA	(used for bus transfers)

The following tables list the I/O port assignments for the Application Processor and File Processor boards individually.

4.2 FILE PROCESSOR PORT ASSIGNMENTS

PORT	NAME	FUNCTION
00	DMA	DMA CONTROL CHANNEL
10	CTC-0	(spare)
11	CTC-1	(spare)
12	CTC-2	(spare)
13	CTC-3	(spare)
20	CTC-0	SERIAL A CLOCK
21	CTC-1	SERIAL B CLOCK
22	CTC-2	PARITY ERROR INTERRUPT
23	CTC-3	BUS INTERRUPT
40	SIOAD	SERIAL A DATA CHANNEL
41	SIOAC	SERIAL A CONTROL CHANNEL
42	SIOBD	SERIAL B DATA CHANNEL
43	SIOBC	SERIAL B CONTROL CHANNEL
50	PIOAD	PARALLEL A DATA
51	PIOAC	PARALLEL A CONTROL
52	PIOBD	PARALLEL B DATA
53	PIOBC	PARALLEL B CONTROL
70	PLOW	LOWER PARALLEL I/O
71	PHI	UPPER PARALLEL I/O
60	FDSTAT	DISKETTE CONTROL REG
61	FCYLDR	DISKETTE CYLINDER REG
62	FSECTP	DISKETTE SECTOR REG
63	FDDATA	DISKETTE DATA REG
80	DSTAT	HARD DISK STATUS/COMMAND
81	DDATA	HARD DISK DATA
82	R/PO	RESULT/PARAMETER REG 0
83	R/P1	RESULT/PARAMETER REG 1
84	R/P2	RESULT/PARAMETER REG 2
85	R/P3	RESULT/PARAMETER REG 3
86	R/P4	RESULT/PARAMETER REG 4
87	R/P5	RESULT/PARAMETER REG 5

4.3 APPLICATION PROCESSOR PORT ASSIGNMENTS

PORT	NAME	FUNCTION
00	PIOAD	PARALLEL A DATA
01	PIOAC	PARALLEL A CONTROL
02	PIOBD	PARALLEL B DATA
03	PIOBC	PARALLEL B CONTROL
20	DMA	DMA CONTROL CHANNEL
30	CTC-0	SERIAL A CLOCK
31	CTC-1	SERIAL B CLOCK
32	CTC-2	PARITY ERROR INTERRUPT
33	CTC-3	BUS INTERRUPT
60	SIOAD	SERIAL A DATA CHANNEL
61	SIOAC	SERIAL A CONTROL CHANNEL
62	SIOBD	SERIAL B DATA CHANNEL
63	SIOBC	SERIAL B CONTROL CHANNEL