2200 / VS LINK SERVICES

FUNCTIONAL SPECIFICATION

Version 1.0

June 17, 1985

doc: 0001P

1.0 INTRODUCTION

1.1 Overview

The 2200 / VS Link is a direct 928 connection between a 2200 and a VS system. A 928 slave controller, the 2258, in the 2200 is attached to a VS serial port with coax cable. This functional specification discusses three different services that will be provided over the link. These are:

- o VDISK - 2200 access to 2200 virtual disks residing on the VS
- o VS system & filing services accessible from 2200
- o VS Terminal Emulation on 2200 terminals (Data Processing Only)

1.2 Environment

o	2200:	VP,MVP,LVP (2200A-T & SVP not supported)
		2258 controller
		Terminal emulation requires DW-type keyboard
o	VS:	with serial port and coax cable attached to 2200
o	User:	2200 SW vendor & application user
o	Market:	existing 2200 sites that need expansion, VS services, or are planning to have a VS. Market is NOT new 2200 customers.

1.3 Background

It is anticipated that the 2200 / VS Link will enable Wang to sell VS products to the extensive 2200 user base. The contention is that with the link, a VS can be sold to coexist with a 2200 rather than to replace the 2200. Gradual migration to the VS would then be possible. It is expected that this offering will stem the move away from Wang equipment when the 2200 is outgrown.

2200 software vendors would like to run their existing 2200 BASIC-2 programs on the VS. They would also like to re-establish their relationship with DSO's, who are only interested in selling VS's or VS's with PC's.

1.4 Hardware impact

- The only new hardware required is the 928 slave controller for the 2200. In 1979, hardware for a 928 slave controller for the 2200 (Model 2258) was designed and prototyped. The interface, which was not fully debugged at the time, may be appropriate for the above required functions if some modifications are made. VDISK function requires changes to 2258 addressing scheme to allow the controller to be seen as a disk controller by the 2200 operating system. 2258 memory must be increased from 32K to 64K. If the 2258 is not appropriate, then a similar type of coaxial controller must be designed.

- No new VS hardware is required.

1.5 Software impact

- BASIC-2 applications need not change (other than disk selection) to use VDISK.

- BASIC-2 applications must change to make use of VS filing and system services. These changes entail replacing 2200 disk routines with access routines that communicate with the 2258 via $GIO.

- ~~Possible~~ changes must be made to VS services to recognize and service the 2258.

- The VS OS most likely must change to recognize the 2200 as a new device type.

- No changes to 2200 BASIC-2 microcode are required.

1.6 Operational impact

- 2200 keyboard layout is different than that of VS. This may impact current VS operators.

- Interaction with a VS is entirely different than with a 2200. Users running applications on both the VS and the 2200 will have to deal with two different environments.

1.7 Documentation/training

User manual describing the link services, connection, and support utilities for these functions must be developed.

2.0 VDISK

2.1 Function

o It is a design goal to ~~Provides~~ Provide up to ~~32~~ 2200 disk images of 16MB each (total=512Mb) on VS disks for 2200 access. Each disk image is a single VS file. The image is used exactly as a 2200 disk platter for data and program storage/retrieval. No changes are required of 2200 BASIC-2 programs.

2200 files and data in disk images are NOT readily accessible from VS applications (VS filing services should be used to share data between VS and 2200 applications). Furthermore, 2200 numeric representation and character sets differ from that of the VS.

o More than one 2200, each with their own 2258 and attachment to the VS, can access the VDISK with the existing 2200 disk multiplexing facilities.

o A 2200 utility, MAKEVDSK, is used to create virtual disks on the VS.

o A 2200 utility, ATTACHVS, is used to logon to the VS and establish the mapping between 2200 disk addresses and the associated VS files.

2.2 Performance

Desired performance of the VDISK is that of the conventional 2200 mass storage devices (i.e., 2280 Phoenix drive). Performance will be affected by VS loading, the number of 2200 partitions trying to access the VS disk through the 2258, the number of 2258's accessing the VDISK, and the overhead inherent in mapping the 2200 disk operations to the VS. Optimizations previously made in 2200 applications to make best use of 2200 disks may not be applicable in the VS environment.

2.3 Input-output

All access to the 2258 is through existing 2200 disk statements. A particular VDISK is specified by the 2200 disk address. It is a design goal that The 2258 controller can be set to respond to addresses:

D10-D1F & D50-D5F
or D20-D2F & D60-D6F
or D30-D3F & D70-D7F

Actual limits may change based upon final design limitations.

2.4 Operation

2200 disk statements issue disk commands to the 2258 controller using the disk address corresponding to the disk image to be addressed.

Supported commands are:

read sector
write sector
multi-sector write
compare sector
copy sectors
verify sectors
format disk (initialize disk image)
hog disk image
release disk image

The Z80 code on the 2258 converts the command to a file request message using the file handle corresponding to the specified disk image. Note, file handles for each disk image are established by the ATTACHVS program and maintained by the 2258. The 2258 then sends that message to the VS IOP that directs the message to the associated VS Access program. (This is the same program that provides VS system and filing services for the 2200.) This program performs the operation on the disk image and sends a message back to the 2258 controller. The 2258 converts the message into the appropriate 2200 disk response which is passed back to the BASIC-2 program. Note that each 2200 disk operation maintains exclusive use of the 2258, shutting out all other activity until the operation has completed.

2.5 Error handling

Standard 2200 disk errors are returned from the 2258 to the BASIC-2 programs.

2.6 Testing

VDISK testing can be done through normal 2200 disk statements and existing diagnostics. Appropriate VS configurations must be established.

2.7 User profile

- o User is a 2200 operator.
- o VS system administrator must create the virtual disks.

2.8 Operating scenario

The VS system administator first must create the virtual disks on the VS using a Wang supplied utility, MAKEVDSK. Before VDISKs are accessible, the 2200 must be logically attached to the VS by using the ATTACHVS utility; this involves logging on to the VS and establishing the mapping between 2200 disk addresses and VS files. Thereafter, operation to a 2200 application user is transparent.

2.9 User interface

Utility screens for MAKEVDSK and ATTACHVS are to be determined.

3.0 VS SYSTEM & FILING SERVICES

3.1 Function

- A set of BASIC-2 subroutines provide access to VS filing and system services. These services include DMS operations such as OPEN file, READ file, WRITE file, and CLOSE file. This enables new or modified 2200 programs to store data on the VS in VS format. VS and 2200 programs can then have common access to this data.

  ~~A specific set of BASIC-2 subroutines provide services equivalent to KFAM so that KFAM based programs can be adapted to VS DMS. This is not completely staightforward since VS DMS does not provide all the services required by KFAM and the 2200 and VS use different representations for numbers and have differing character sets.~~

  Since the VS provides its own access methods, user documentation must explain how the current KFAM ~~[illegible]~~ users how to modify their code to make use of the VS access options. No KFAM specific subroutines will be developed.

  With the services accessible, users can create VS files and queue them for printing. However, output from 2200 PRINT statements cannot be redirected to a VS printer.

3.2 Performance

Desired performance is that equivalent to native VS applications. Performance will be affected by VS loading, the number of partitions trying to access the VS through the 2258, and the number of 2258s accessing the VS.

3.3 Input-output

2200 applications access VS services through BASIC-2 subroutines.

3.4 Operation

These subroutines pass system & file requests using $GIO to the 2258 Z80 code. The Z80 sends the request message to the VS IOP, which directs the message to the VS Access program. This program satisfies the request and returns a message to the BASIC-2 subroutine via the VS IOP and 2258. The BASIC-2 subroutine has exclusive use of the 2258 until the request ~~is~~ is satisfied and the response is returned.

3.5 Error handling

VS error codes are returned to the BASIC-2 subroutines.

3.6 Testing

Since this is new functionality on the 2200, a testing plan must be developed.

3.7 User profile

User: 2200 operator, generally naive having been sheltered from traditional minicomputer operations.
Programmer: 2200 SW vendor or programmer.

3.8 Operating scenario

Once the link to the VS has been established via the ATTACHVS program, operation of an application with the 2200/VS link is transparent to the user.

4.0 VS TERMINAL EMULATION (Data Processing Only)

4.1 Function

Allows a user, with a terminal attached to the 2200, to logon to the VS and operate as a VS DP workstation. Several 2200 users can concurrently operate as VS terminals. Functionality is equivalent to a 2246S workstation; WP, Wang Office, and other such services are not accessible These unavailable services are designed to run in an intelligent WS; the 2258, on the other hand, is designed as a message passer, not as an intelligent workstation.

4.2 Performance

Desired performance of terminal emulation is that of a 2246S workstation. Performance will be affected by VS loading, the number of partitions trying to access the VS through the 2258, and 2200 loading. Note, disk requests lock out the terminal emulator until those requests are completed; long operations will make terminal emulation performance unacceptible.

4.3 Input-output

The emulator maps 2200 keys onto VS keyboard equivalents. Labelling of keys on the 2200 keyboard is not the same as on VS keyboards (e.g., function keys start at 0 on the 2200 but 1 on the VS).

CRT output is the same as on a 2246S workstation.

4.4 Operation

The 2258 code handles all terminal traffic. The code must be such that the VS cannot overrun the system with rapid, sequential screen operations. I/O is equivalent to a VS remote WS.

4.5 Error handling

Error handling is done in the same manner as the VS WS.

4.6 Testing

Since this is new functionality to the 2200, a testing plan must be developed.

4.7 User profile

Same as any VS operator.

4.8 Operating scenario

The user must load and run VS Terminal Emulation program in the 2200. Thereafter, the terminal emulation is equivalent to a VS 2246S WS, except with regards to KBD layout. Note that a user will be required to logon to the VS before other operations can be performed.

4.9 User interface

Same as VS remote Workstation except with regards to KBD layout.

5.0 SUPPORT SOFTWARE

5.1 MAKEVDSK

This utility creates 2200 disk images on the VS.

5.2 ATTACHVS

This utility establishes the mapping between 2200 disk addresses and VS files that are the 2200 disk images. In order to access the VS, this utility must request that the user logon to the VS. Consideration must be given to default on power-up, and accidental power down of the 2200.

5.3 File transfer

To facilitate migration of data, a method of file transfer must be devised to allow a shift of data from 2200 mass storage to the VS.

~~5.4 KFAM equivalent VS DMS access subroutines~~

~~To facilitate 2200 program modification to VS file system functionality, a set of subroutines should be developed that map current KFAM file access to VS type file access.~~

5.4 VS access subroutines

```
2200/VS Link - Development Tasks  ( Not including QA or Documentation )
=======================================================================          DMS message formats.

2258 Coding (1 experienced Z80 Programmer)
------------------------
Hdwr debug & diag         !-----!---->
Filing                          !-----!---->
Vdisk                                       !---->
VSTE                                              !-----!-----!-----!-[illegible]


BASIC-2 Coding (1 experienced BASIC-2 Programmer)
------------------------
Filing                    !---->
~~KFAM S/R Emulation~~          ~~!---->~~
MAKEVDSK                              !---->
ATTACHVS routine                            !---->
VSTE                                              |--->
VS Coding
------------------------
 OP                       !---->
Service Program           !-----!-----!---->


Note: !----> = 1 month
```

WANG LABORATORIES, INC. COMPANY CONFIDENTIAL
ONE INDUSTRIAL AVENUE, LOWELL, MA 01851 • TEL: 617/459-5000, TWX 710-343-6769, TELEX 94-7421

Project Engineering Plan #H0104A

2200 to VS Migration

Originator: Paul Daniel
Revision: 0
DATE: 08/05/85

APPROVAL:

[illegible] Frederick A. Wang	DATE: 9-10	[illegible] Bob Kolk	DATE: 9/6/85
[illegible] Horace Tsiang	DATE: 9/10/85	[illegible] Leo Chan	DATE: 9/5/85
[illegible] Miguel Brazao	DATE: 9/10/85	Don Dunning Don Dunning	DATE: 9/3/85
		[illegible] Dick McCusker	DATE: 8/29/85

BOM #: 191-0664-00

WANG MODEL #: 2258

PRODUCTION SITE: TBD

DESCRIPTION:

The purpose of this project is to allow the 2200 to communicate with and co-exist in a VS environment. The 2200/VS data link (2258) is one of the major elements of a total 2200 Co-existence/bridge strategy designed to protect our user and ISO base by providing a co-existence and migration/evolution to the VS product line. Three functional services will be provided over the link. These are:

VDISK - 2200 access to 2200 virtual disks residing on the VS.
VS system and filing services accessible from the 2200.
VS Terminal Emulation on 2200 terminals.

MARKETING REQUIREMENTS:

The marketing requirements are in document entitled '2200/VS Data Link Preliminary Business Plan' and/or 'FCS Requirements for the 928 2200/VS Data Link', both dated July 18, 1985 are available by contacting Gene Schulz in the Development Center.

DESIGN SPECIFICATIONS:

The design specifications are in a document entitled 'Hardware Functional Specification for the 2258 Option', dated July 12, 1985 and can be obtained by contacting Mohamed Makhlouf in the development center.

COMPANY CONFIDENTIAL

FUNCTIONAL SPECIFICATION:

The functional specifications are available in a document entitled '2200 / VS LINK SERVICES' , dated June 17, 1985, and can be obtained by contacting Paul Skibba in the Development Center.

PROJECT RESPONSIBILITY:

Hardware Design		
Dick McCusker	-	Project Manager
Paul Plouffe	-	Project Leader
Mitch Corrigan	-	Hardware Support
Mohamed Makhlouf	-	Hardware Design
Gary Cardone	-	Hardware Design
Leila Donahue	-	Mechanical Packaging
Rob Chaffee	-	Industrial Design
Software Design		
Paul Skibba	-	Project Manager
Percy Tzelnic	-	Project Manager
Jerry Sevigny	-	PC/2200 SW Project Leader
Nori Odoi	-	Software Engineer (BASIC-2)
George LaRue	-	Software Engineer (BASIC-2)
Jose Valdepenas	-	Software Engineer (BASIC-2/OMT)
Roger Kirk	-	Software Engineer (Z-80)
Isabel Tomaszewski	-	VS Support
Joe Nichols	-	Multi-Workstation Support (MWS)
Software Support		
Sheila Morgan	-	Quality Assurance/Applications
Lois Collins	-	QA Operations/SW Release
Dave Kosko	-	Performance Analysis
Jim Crawford	-	Alpha/Beta
Bill Zannini	-	Diagnostics
Ed Killeen	-	Diagnostics
Engineering Support		
Dana O'Malley	-	R & D Planning
Paul Ricker	-	Electrical Drafting
Ernie David	-	Mechanical Drafting
Gerry Gilks	-	ESD/EMS Testing
Mike Britko	-	FCC/EMI/VDE Testing
Steve Brody	-	UL/CSA/IEC Testing
Chuck Connell	-	Environmental Testing
Perry Biancavilla	-	Shipping Package Design
Peter Casey	-	Component Evaluation
Benny Shamash	-	Interconnection Design
Bob Desrosiers	-	Cable Design
Duffy Dauphinais	-	Bill of Materials
Peter Nolan	-	Human Factors
Dick Mann	-	OEM Purchasing
Ray Chang	-	Standard Interfaces
Product Planning and Management		
Gene Schulz	-	Product Planning and Management
Product Manufacturing		
Scott Stevenson	-	New Product Manufacturing
Customer Engineering		
Dave Carter	-	Maintainability Engineer
Robert Peebles	-	Maintainability Engineer
Hal Woods	-	Maintainability Engineer
Product Cost Engineering		
Fred Bodenrader	-	Product Cost Engineer
Technical Documentation		
Bill VonAchen	-	Publications Development

COMPANY CONFIDENTIAL

COST/PRICE ESTIMATES:

Acceptable Range of Manufacturing Cost: $300.00 - 500.00

Estimated Product Manufacturing Cost: TBD

Selling Price: $1000.00 - 1500.00

HARDWARE:

The 2258 Data Link consists of one Printed Circuit Board that will reside in the 2200 system and will be tied to the VS through a dual coaxial cable. The main circuitry associated with the 2258 Option board is categorized as follows:

1.0 Z80 PROCESSOR INTERFACE

A Z80A CPU running at 2.5 MHz will be used as the main data processor. It will be able to communicate with the VS master via the use of a semaphore register in Main Memory which is frequently polled by the master using the Data Link. All processing will be controlled by this Z80.

Outputs from the Z80 will be buffered and will impose no more than one TTL load upon the CPU. These outputs will be properly conditioned during any DMA type operation so as not to interfere with any external circuitry wishing to use associated control or signal lines.

On initial power up of the 2200 system the Z80 will execute the firmware in the EPROM to initialize the board and write to the main memory. This will ensure correct parity and than make the board READY for the 2200. When enabled, the Data Link will load the main memory with the required code and issue a 'Data Link Restart' command that forces Z80 to start execution at location 0000H.

2.0 MAIN MEMORY

Main memory will consist of nine 64K x 1 dynamic RAM chips which will be shared between the Z80 and Data Link. Word length of 9 bits, eight bits for program/data storage, the remaining bit will be designated for parity.

Main memory is triple ported, where two data ports co-exist between the Z80 and Data Link and a third port with the 2200/Refresh Circuit.

Data access to this memory will occur only when the Z80 wishes to exchange data with the 2200 or when a Data Link/Z80 request is received by the memory controller (the 3004 chip). The controller will grant access to the appropriate requesting device and allow memory reads or write to take place. It should be noted that memory access from the Data Link only occur during DMA type cycles, and that because of this the address, the data and control bus of the Data Link, and the Z80 are mutually exclusive to one another and can be shared.

Parity is checked on the link's memory any time that a data read occurs from the Z80, or Data Link. During normal operation, correct parity will be written on any memory write from these systems. The Z80 has the ability to select a 'Stop on Parity Error' where in the event of a parity error, circuitry will force a continuous NOP code onto the Z80 data bus, until it receives a 'RESTART' from the Data Link. This will prevent the Z80 from corrupting memory or resulting in any adverse operations to inadvertantly occur. The master will also be notified of this parity error condition through an input to the Data Link status register, where it will take appropriate action. The 'Stop on Parity Error' condition is always enabled at power up.

3.0 MEMORY ARBITRATION, SEQUENCE AND REFRESH

The memory circuitry is designed to be independent of any of the systems which can access it. The hardware circuitry operates through the following general procedures:

* The circuitry monitors requests from the Data Link, Z80, and Refresh Circuit.

* It arbitrates between requesting devices and grants an access to the first requesting device.

* It generates wait conditions to the other devices which raised a request but were not granted immediate access.

* The circuitry maps the appropriate address, data and control signals onto its' bus and generates its' own memory sequence signals to properly read or write the data from or to memory. If a read operation occurs, the data is latched onto the appropriate bus and the output is allowed to be controlled by memory read signals from the requesting device.

* Upon completing the memory access, the next device which requested the bus is granted access. The previous device will be locked out until all other requesting devices have been granted access.

The internal memory bus is eight bits wide with a ninth bit for parity. All memory refreshing is done through discrete circuitry which creates a 'Refresh Request' every 8-10 microseconds.

4.0 928 DATA LINK INTERFACE

The Data Link provides the means of local communication between the link board and an VS master. The Data Link transfers data between the board and master through use of a semaphore register located in the Main Memory of the link.

The link communicates through the use of an 11 bit serial data stream used in an asynchronous fashion. The nature of this stream consists of 1 start bit, 8 bits of data (MSB first), 1 parity bit (even) and one stop bit. This data travels at a bit rate of 4.275 MHz, therefore the total time for each byte transferred will be 2.573 uSecs. The clock frequency for the bit rate of 4.275 MHz is derived from a crystal running at 34.2 MHz which also supplies 17.1 MHz to the memory arbitration and sequencing logic.

COMPANY CONFIDENTIAL

The Data link is explained in more detail in the following specifications:

* HM-04	Hardware Device Type Switches
* HM-16	Interfacing The Universal Data Link Board
* HM-39	WL-2001 Data Link Chip Specification
* OS-2	Wang 928 DOS The OIS Workstation
* OS-6	OIS DOS Slave Processor Interface Specification

The Data Link is normally in receive mode and will enter transmit mode only on an instruction from the master which requires it to transmit. After the Data Link completes this transmission, it will go back into receive mode.

The Data Link is designed around the VLSI chip; WL-2001 Data Link Chip which requires far less circuitry than the earlier discrete design. The Data Link and Z80 share the same memory bus and are mutually exclusive of one another.

The link is configured to the master as a 64K workstation through the use of three registers. Data from two of these registers are read by the Z80 where the information is transferred to the master through semaphore handling. Data from the third register is transferred to the master from a read via the Data Link controller circuitry. Data for each of these registers is generated at the 2200 side.

The two registers that are read by the Z80 are:
1. Slave Type Register #1, which is read by an "IN 07"
2. Slave Type Register #2, which is read by an "IN 08"

The third register contains DEVICE TYPE on bits D7 through D4. These bits are loaded into the DATA LINK MASTER through and "IN 07". Refer to HM-04 for the explanation of the meaning of each bit of these ports.

5.0 <u>2200 BUS INTERFACE AND I/O CONTROL CIRCUITRY</u>

The 2200 bus to Z80 bus hardware interface shall impose no more than one TTL load per signal upon the 2200 bus.

The 2200 can determine the type of I/O device located on its bus by addressing the device. The link board is designed to compare the 2200's addressing code with the addresses associated with link. If a match occurs, the board will then be enabled to exchange I/O with the 2200. The 2200 can disable the link by addressing a different device. The link board latches the address at the Address Bus Strobe (ABS) and makes all eight bits available to the Z80. If the identity of the board match the device addressed by the 2200, and if there are no constraints, the board will become enabled and will indicate this fact to the 2200 by asserting READY, R.

There are two separate and distinct 2200-type devices on the link board.

1 2200 disk device whose address is 10H (50H), 20H (60H) or 30H (70H) which has the functional equivalent of the conventional 2200 disk controller. It will be used to accomplish the Virtual Disk (VDISK) requirement. The address selected, switch settable, will be made available to Z80 to determine the READY or BUSY status.

2 General purpose device whose address, switch settable, will be determined by software and has uncommitted applications. It will be used to accomplish VS terminal emulation and the data traffic for DMS.

In conjunction with the general purpose device, the hardware will provide the Z80 with three registers; STATUS, COMMAND and DATA. Another 8 bits register which reflects the identity of the selected disk and general purpose device will be provided to the Z80. This register corresponds to a switch bank settable by the user.

<u>PRINTED CIRCUIT BOARD DESIGN</u>:

The 2258 controller board will be a multi-layered design, using the standard dimensioning for 2200 products. It will begin layout in EDD on August 26, and is scheduled to be released September 23.

<u>SOFTWARE DEVELOPMENT</u>:

The 2200/VS Link (2258) project is designed to provide the existing 2200 customer base with the following capabilities:

* Allow existing (unmodified) 2200 BASIC-2 programs to use the VS as a file server. One or more 2200 disk images are stored on the VS. The 2258 controller responds just like a phoenix disk drive.

* Allow new/modified 2200 BASIC-2 programs to share data with VS applications. The 2200 BASIC-2 programs will utilize $GIO statements to access VS DMS servies such as Open, Close, Read and Write.

* Allow the 2200 user access to the VS in order to develop or execute VS application software.

In order to accomplish these goals, the 2258 controller must be able to act as both a 2280 disk controller as well as a 928 controller.

The software development efforts are divided into two components, BASIC-2 and Z-80. The BASIC-2 development will execute on the 2200 using the 2200 Operating System. The Z-80 effort results in the 2258 controller's executable code which is downloaded from the VS at power up.

The BASIC-2 effort is responsible for:

* File Service Subroutines. The BASIC-2 $GIO statements are used to perform VS DMS service functions such as Open a File, Close a File, Read/Write a record and so forth.

COMPANY CONFIDENTIAL

* Access Method Subroutines. These BASIC-2 subroutines will provide a model, and the necessary information in order for users, VARS and software vendors to replace KFAM and other existing file access method routines in order to access 2200/VS shared data on the VS.

* Create V-Disk. A BASIC-2 utility which will result in the creation of one or more 2200 disk images on the VS.

* Attach to VS. Although V-Disk will perform an automatic logon, it is envisioned that the DMS services will require a user to have logged onto the VS prior to executing DMS applications.

* VS Terminal Emulation. Terminal emulation requires code written in both 2200 BASIC-2 and 2258 (Z-80) firmware. This software will handle all screen and keyboard activity.

The 2258 Firmware (Z-80) effort is responsible for handling power on, 2200 disk operations as well as the 928 emulation. In order to accomplish this the controller must be able to:

* Respond to the 2280 disk sequences as outlined by the 2280 DPU interface specification produced by Max Blomme on February 1980.

* Support VS DMS requests from the 2200 to the VS, handling error conditions as well as data transfers.

* Support workstation emulation from the 2258 controller.

RESOURCE REQUIREMENTS:

VS Software
Create V-Disk Utility
2258 Microcode

VS ASCII character sets and 2200 ASCII character sets are not always identical for the same language. Therefore, International requirements show a need for reconciling the two different character sets in a way that is transparent to the user.

HARDWARE DEPENDENCIES:

Jerry Sevigny's organization is in need of two 2258 controller cards, one of which is required immediately for development, the other deliverable at a later date for integration testing. Also, a ZEBUG would be very useful to facilitate software development.

DIAGNOSTICS:

Diagnostic Engineering will develop a Built In Test (BIT) PROM in support of the 2258 project. The BIT will validate the functional operation of the 2258 board by exercising the major block components such as Memory and the CPU. If an error on the 2258 board occurs the BIT will report the failure by leaving the Diagnostic LED "ON".

Diagnostic Engineering will also provide the Diagnostic Engineer with a 2200 system and a ZEBUG to facilitate this project.

COMPANY CONFIDENTIAL

<u>MECHANICAL PACKAGING</u>:

The 2258 controller board will be housed in the current 2200 system.

<u>DOCUMENTATION</u>:

Technical documentation will be created for this project. Specific information regarding the documentation and the development schedule of the documentation can be found by contacting Bill VonAchen.

<u>STANDARDS</u>:

This product must comply with the following standards for safety and electrical noise (EMI/RFI) :

DOMESTIC

1. UL Standards for Safety 114 (Office Appliance and Business Machines) or 478 (Data Processing Equipment).
2. FCC Class A requirements for interference from computing devices.
3. Wang Standard for electrostatic discharge (SPI 10-623).
4. Wang Standard for Mechanical and Environmental Testing (SP 10-708).

INTERNATIONAL

1. CSA Standard for Safety C22.2 No. 154 (Data Processing Equipment).
2. IEC 435 (Safety of Electrically Energized Office Machines).
3. VDE A Standard for Germany.

STANDARD INTERFACES:

Not Applicable.

SHIPPING PACKAGING:

Protection during storage and shipment shall be provided by packaging designed to accommodate the final product configuration, (including all auto-enclosures, cables, manuals, supplies, ect., where applicable). The packaged product shall be required to pass minimum shock and vibration levels to survive handling and transportation per Wang Standard SPI 10-521.

CUSTOMER INSTALLABILITY:

Customer Engineering will be responsible for installation of this board.

GENERAL:

Please see the attached milestone schedule.

COMPANY CONFIDENTIAL.

WANG LABORATORIES, INC.
ONE INDUSTRIAL AVENUE, LOWELL, MA 01851 • TEL: 617/459-5000, TWX 710-343-6769, TELEX 94-7421

WANG

MILESTONE SCHEDULE FOR PEP # H0104A

2200 to VS Migration

Originator: Paul Daniel
Revision: 0
DATE: 08/05/85

Task	Planned Completion Date	Revised Completion Date
Marketing Requirements	07/18/85 *	
Design Specifications	07/12/85 *	
Functional Specifications	06/17/85 *	
Golden Sample	11/15/85	
R&D Pilot Build (Qty. 10)	11/22/85	
Code Freeze	11/15/85	
To Q.A. (final version)	11/15/85	
Hardware/Software Integration	11/15/85	
Quality Assurance Certification	12/30/85	
Shipping Package Design	12/01/85	
Alpha Testing	01/01/86	
Beta Testing	01/01/86	
Manufacturing Pilot Build (Qty.)	11/20/85 **	
Final Qualification Testing	12/01/85	
Documentation	01/01/86	
Announcement	01/01/86	
First Customer Ship	01/01/86	
Volume Ship	02/01/86	

*Task has been completed.
**Build start date.

TO: Paul Plouffe, M. Mahklouhf & S.K. Ho
FROM: Roger Kirk
SUBJECT: 2258 Interface Specification
DATE: 08/26/85 - Clarified 9/18/85

Attached is the latest revision of the 2258 interface with the 2200. Several simplifications have been made and certain points have been explained in greater detail. The functionality described herein should be identical to that of the schematic provided on 08/22/85. A complete and all-encompassing description of the operation of the 2200 bus can be, at times, difficult to impart succinctly, therefore, wherever explanation is vague, ambiguous or lacking, the schematic should be consulted. All our programming efforts and expected performance will be based on this functionality. Since designing this (or any) 2200 interface is a curious blend of real-estate efficient hardware design and operating system insight, the submitted design takes both into account. If anything in either the functional specification or the proposed schematic appears incorrect, inconsistent or illegible - feel free to call me any time.

Roger Kirk
77624 or 75733

Recommended reading:

1. 2200 I/O Bus structure - Roger M. Kirk Jr - 06/15/81
2. 2200 Disk Command Sequences for the 2200 LVP - Max Blomme - 05/14/81
3. 2200 Basic-2 Reference Manual - Chapter on $GIO statement

x.c. ~~Bruce Patterson~~
~~Jerry Sevigny~~
~~Scott Tagen~~

Overview

The 2200/VS Link project should capitalize on the existing PC MWS product design and code as much as possible. MWS, provides to the PC, services similar to those wanted for the 2200. With MWS the PC can have four concurrent VS workstation sessions or three workstation sessions and an RKG session that can provide VDISK or VS filing access. The PC implementation can be adapted to provide these services for the 2200. Advantages of this approach are consistency between the PC and the 2200 for the VS and reduced development effort for the 2200. The disadvantage is that the controller would support only 3 or 4 workstation emulations. However, a second and third cards could be installed for additional workstations.

2200 Link Hardware

Use the PC LCO card as the basis for the 2200 Link card. [Engineering has pointed out that the old 2258 design is deficient is a number of ways, so the PC card may be a more reasonable starting point at any rate]. The card must be modified as follows:

- Replace the PC interface with a 2200 interface.

- Delete the screen memory.

- Provide disk addressing for VDISK access. Card should be switch settable to 10, 20 or 30. The high 2 bits of the address are ignored; however, they must be latched for the Z80 to view.

- The card must provide a 2200 disk interface, including detecting CPB being set to Ready, latching of OBS, CBS and Data, issuing IBS and controlling R/B. Also, the IOB's (AB8,AB7 & AB6) must be latched whenever a strobe is done to the card.

- Provide a 3-address set for terminal emulation and VS filing access. This set consists of a control address for passing commands and responses, a data address used for passing data streams (eg. print) [will not be used unless 2200 OS modified], and a status address used to supply ready/busy level for 2200 program polling. This 3-address set must be switch settable.

- The 2200 interface (OBS, CBS, IBS, CPB, R/B & Data Buffer) for the above addressses MUST BE INDEPENDENT of the disk interface.

  A CTC is necessary to provide O/S timer support for MWS and CBS/IBS and CPB detection for DISK and Communication interfaces.

Link Code

Use the MWS code as the basis for the link.

- The VS would view the card as PC MWS.
- The PC interface portion of the code must be replaced with a 2200 interface.
- Improved operation of terminal emulation is possible by modifying MWS to notify the BASIC-2 program of screen changes in increments smaller than 1 line.
- The ability to have more than 4 VS sessions can be explored; however, memory constraints probably preclude this.

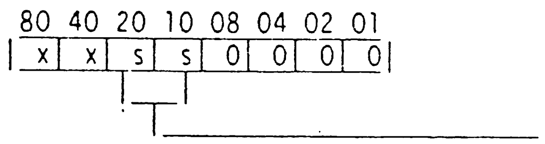
BASIC-2 Code

- The BASIC-2 terminal emulator and file access subroutines will poll the card at the status address to determine when processing needs to be done. When the status address is ready, the BASIC-2 programs will use the control address to determine if there is activity for them. If so, the appropriate action will be taken. Polling should be adequate for several BASIC-2 programs to access the VS. Polling is necessary if 2200 OS changes or a global access partition is to b avoided.
- Polling overhead can be reduced by using a single $GIO statement poll the keyboard and the link.
- Improved terminal emulation operation is possible if the BAS program does as much of the normal keystroke handling as possi rather than always passing the keystrokes on to the Z80 processing.

Controller card addressing

DISK address is used for VDISK access. COM(munication) addresses are used for access to VS Filing and Terminal Emulation.

Disk

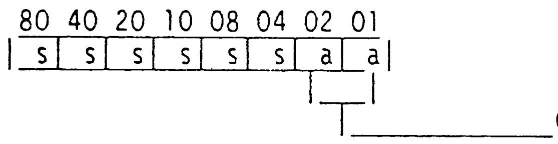


00 - NOT THIS CARD
01 - disk unit 10 (or 50)
10 - disk unit 20 (or 60)
11 - disk unit 30 (or 70)

s - compared to switches

x - ignored for enabling
80 = 1 if disk hogged
40 = 1 if 2nd drive

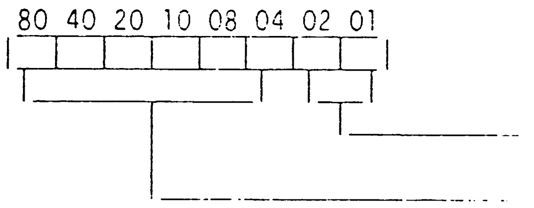
COM



00 - NOT THIS CARD
01 - Status address
10 - Command address
11 - Data address

s - compared to switches
a - Address

User settable address switch bank (8-bits)



NOTE: As is standard on all 2200 controllers, an address switch is decoded as follows:

ON = 1	Same as the AB lines, which are Low=1 / High =0.
OFF = 0	Same as the AB lines, which are Low=1 / High =0.

ABS

During each ABS strobe from the 2200, the card compares the device address on the system bus to the card addresses specified by the switches, as described above. If an address matches, the card is enabled and the following address bits are latched:

Enable latches:

80	40	02	01

If the address does not compare, the card is disabled. No address bits are latched if the card is disabled (i.e. no controller address(es) match the 2200 Address Bus at ABS.)

N.B. If the Disk Address Switches are set to 00 (an illegal address) the controller shall NOT be enabled.
If the 2200 Bus is addressed at ZZZZZZ00b (where ZZZZZZ01b, ZZZZZZ10b and ZZZZZZ11b are the 3 Communication addresses) the controller shall NOT be enabled.

READY/BUSY

The Z80 has a port through which the Z80 code can set 4 separate Ready/Busy latches, one corresponding to each of the card address types:

Disk address
Status address
Command address
Data address

Ready/Busy port
IN - - read RB latches
OUT - - set RB latches

While the card is enabled, the HW sets the Ready/Busy line to the corresponding RB latch state. Except for the Status Address (which has no buffer), Buffer Full overrides Ready and sets the card Busy.

Disk BF overrides Disk R/B

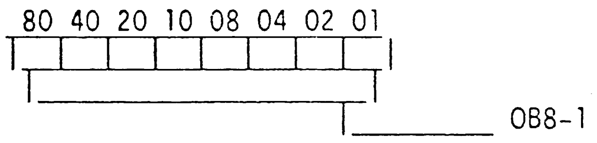
Com BF " Com R/B

OBS/CBS

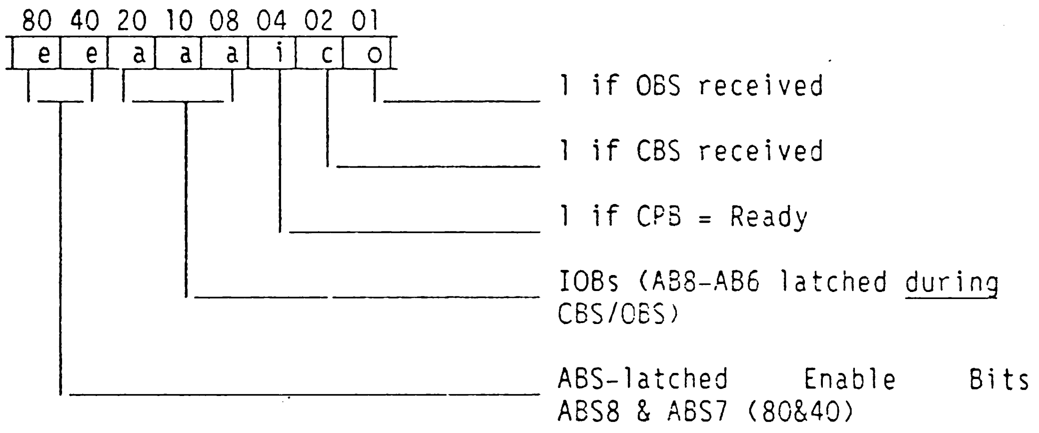
During an OBS or CBS strobe to the enabled card, the card Hardware:

1. Immediately sets the OBS/CBS Data Buffer full latch.
2. Latches the data into the Data Buffer.
3. Latches the IOB's - AB8, AB7 & AB6 at CBS/OBS time.
4. Copies the Enable latches - AB8 - AB7 at Board Enable Time.
5. Sets the OBS or CBS event latch (a.k.a. "Buffer Full Flag").
6. And interrupts the Z80.
   A single interrupt for CBS or OBS is sufficient.

OBS/CBS Data buffer



EVENT/ADDRESS LATCHES



The Z80 code has access to the above information with the following ports:

Read Data Buffer port
IN - - read buffer; "Buffer Full Flag" NOT cleared
Clear Data Buffer port
IN - - read buffer; CLEAR "Buffer Full Flag" also clear IOBs + OBS R CBS R
OUT - CLEAR "Buffer Full Flag"
Read Event/Address latches port
IN - - read latched events and AB bits

DISK ADDRESS INTERFACE

IBS

When the 2200 enables the card at the Disk address and sets CPB ready:

1. The Z80 is interrupted.

2. The Z80 sends data to the 2200 by issuing an OUT after it has recognized CPB has been set to Ready by the 2200. The OUT should place the data on the 2200 bus (IB8 - IB1) and set IBS true. IBS will remain true until the 2200 sets CPB busy which should reset IBS.

Two ports are available for IBS:

IBS port
$A_7 = 0$ OUT - - Issue IBS with ENDI = 0
IBS with ENDI port
$A_7 = 1$ OUT - - Issue IBS with ENDI = 1

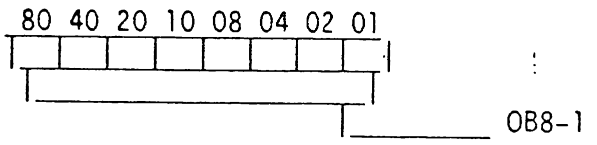
If the port addresses are not fully decoded, an undecoded address bit may be used to set ENDI = 0 or 1 and only one I/O port is needed.

COMMUNICATION ADDRESS INTERFACE

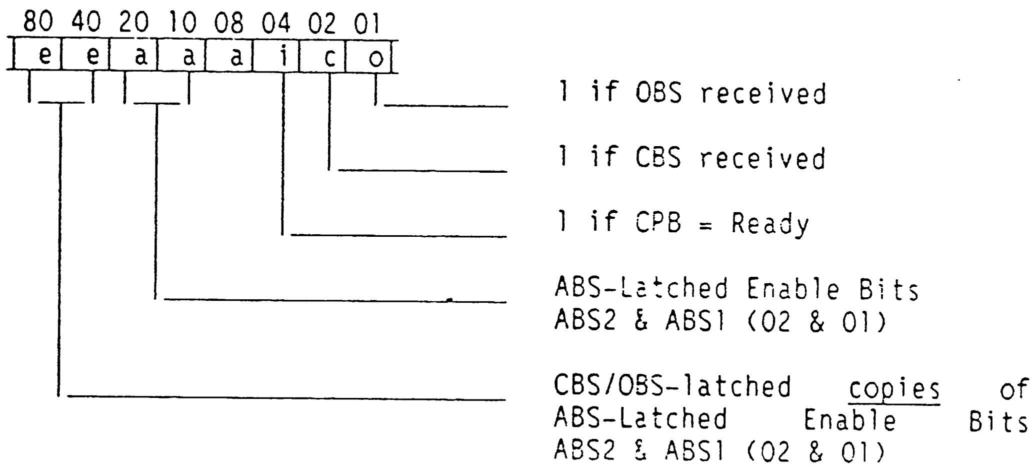
OBS/CBS

During an OBS or CBS strobe to the enabled card, the card Hardware:
1. Immediately sets the OBS/CBS Data Buffer full latch,
2. Latches the data into the Data Buffer,
3.
4. Copies the Enable latches - AB2 - AB1 AT BOARD ENABLE TIME,
5. Sets the OBS or CBS event latch, (a.k.a. "Buffer Full Flag")
6. And interrupts the Z80.
   A single maskable interrupt for any event is sufficient.

OBS/CBS Data buffer



EVENT/ADDRESS LATCHES


80 40 20 10 08 04 02 01
e e a a a i c o
1 if OBS received
1 if CBS received
1 if CPB = Ready
ABS-Latched Enable Bits ABS2 & ABS1 (02 & 01)
CBS/OBS-latched copies of ABS-Latched Enable Bits ABS2 & ABS1 (02 & 01)

The Z80 code has access to the above information with the following ports:

Read Data Buffer port
    IN - - read buffer; "Buffer Full Flag" NOT cleared
Clear Data Buffer port
    IN - - read buffer; CLEAR "Buffer Full Flag"
    OUT - clear "Buffer Full Flag"
Read Event/Address latches port
    IN - - read latched events and AB bits

IBS

When the 2200 enables the card at the Com address and sets CPB ready:

1. The Z80 is interrupted.
2. The Z80 reads the Event/Address Latch to determine at which address the board is enabled.
3. The Z80 sends data to the 2200 by issuing an OUT after it has recognized CPB has been set to Ready by the 2200. The OUT should place the data on the 2200 bus (IB8 - IB1) and set IBS true. IBS will remain true until the 2200 sets CPB busy which should reset IBS.

Two ports are available for IBS:

IBS port
    OUT - - Issue IBS with ENDI = 0
IBS with ENDI port
    OUT - - Issue IBS with ENDI = 1

If the port addresses are not fully decoded, an undecoded address bit may be used to set ENDI = 0 or 1 and only one I/O port is needed.

The asymmetry of the 2258 I/O architecture (i.e   .iput buffers and one output buffer) is based on proprietary knowledge   .he 2200 MVP operating system.

There must be 2 input buffers.

Partition X may issue an OBS w/data to the command address and breakpoint, then partition Y can legally issue a CBS (no wait for ready) w/data to the disk address. With a single buffer, data from partition X would be overwritten with data from partition Y.

There need be only one output buffer.

When partition X wishes input from the 2258, it enables the card and tests *R/B. If *R/B is busy, the partition disables the card and breakpoints and will try again later. When and if *R/B is ready, the 2200 sets CPB to Ready. Partition X will now wait until input has been received or it times out (normally in the millisecond range). When CPB is set to Ready, the 2258 receives an interrupt, senses which 2200 address is requesting data and sends the appropriate byte. Since the 2258 controls *R/B for each separate address, it can dictate whether it is ready to send data at any address. The 2200 will not ask for data which is not available.

M E M O R A N D U M

TO:	Dr Wang	M/S 19C4
	Horace Tsiang	M/S 14C3
	Don Dunning	M/S 1449
	Leo Chan	M/S 1379
	Miguel Brazao	M/S 14A2

FROM: Gene Schulz

SUBJECT: 2200/VS Data Link Preliminary Business Plan

DATE: December 13, 1985

This document outlines the proposed business plan for the 2200/VS data link in order to allow the 2200 to communicate with and co-exist in a VS environment.

The 2200/VS data link is one of the major elements of a total 2200 co-existence/bridge strategy designed to protect our user and VAR base by providing a co-existence and migration/evolution to the VS product line.

Below is a summary of the attached plan:

* To develop a high speed data link from the 2200 to the VS via the 2258 controller (928).

* To develop the necessary Z80 code (firmware) for the 2258, BASIC-2 application code and VS operating system and utility programs to allow the 2200 to use the VS as a file server (VDISK), under DMS to allow VS programs to act upon 2200 data and allow the 2200 to utilize the filing services of the VS, and for the 2200 workstation user to be able to log on to the VS (VSTE) and execute VS DP applications.

GS/sma

Introduction

To date, we have sold 64,000 2200 systems worldwide. Currently and primarily through our System Houses, we are selling 100-250 new 2200 systems per month. If we were to categorize the users and our resellers by their needs, they generally would fall into one of three categories:

* The reseller who will continue to sell the product and the user who will continue to use the product as long as we continue to make it and support it.

* The reseller or user who wants to continue to use or resell the product but wants to know the life cycle and evolution/migration path and/or wants to integrate with other Wang products today.

* The reseller or user who wants to immediately migrate to another product(s).

The 2258 data link address the second category (20% + of the user market and 50% of the ISO market) and will provide the following:

* Allow current 2200 users who are either at maximum configurations, want to off-load certain tasks, or want to begin to evolve to new technology but want to be able to keep their 2200, take advantage of the facilities of the VS.

* To allow current 2200 resellers (VARs) to sell VSs to existing users and not have to trade-in their present equipment.

* In the absence of a a 2200 to VS software bridge, to provide an evolution path for both the user and reseller to migrate to the VS, to be able to utilize their existing data files, perform new DP application development on the VS and provide access to these DP applications by the 2200 workstation user.

* To protect our user base and channels of distribution with a strategy that allows the integration of the 2200 with the VS, thereby providing the reason to stay with Wang and migrate their code to a Wang VS, not competition.

Product Description

Hardware

The 2258 controller board is a modernized 928 slave controller for the 2200 that was originally designed and prototyped in 1979. The original design can provide the data link but will require modifications. Changes must be made to the 2258 such as to allow the controller to be seen by the operating systems as a disk controller to the 2200 and as a cluster of workstations to the VS . Memory must be increased to 64K. Other 2258 modifications include changing the design from a mother/daughter board to a single board and incorporating modern technology. If after the prototype is built and tested and not found to be appropriate, than the current 928 will have to be modified to include the 2200 bus.

Software

Z80 code (firmware) is required to run the 2258. BASIC-2 code which involves primarily $GIO commands is needed to map the 2200 disk I/O requests to the VS via the 2258 data link. BASIC-2 code is also needed for VS terminal emulation (VSTE). VS operating system and resident routines need to be modified with respect to how the VS will view the 2258.

On the user application side, BASIC-2 applications using VDISK need not change other then disk selection. For DMS, BASIC-2 applications require changes to the I/O routines to make use of VS filing and system services. These changes entail replacing 2200 disk routines with access routines that communicate with the 2258 via $GIO.

The combination of the above hardware and software will provide three different services:

* VDISK - 2200 access to 2200 virtual disks on the VS

* DMS - VS system and filing services accessible from the 2200

* VSTE - VS Terminal Emulation on the 2200 workstations (2336DE)

Market Analysis

The estimated 2200 user base is as follows:

	US	Intl	WW
2200 T, PCS, WCS 15, VP	20-22,000	12-14,000	32-36,000
SVP	1,600	2,500	4,100
LVP	4,700	4,200	8,900
MVP	8-10,000	8-10,000	16-20,000

Due to the memory requirements and partitioning, prospects for the 2258 may be limited to LVP and MVPs or a worldwide base of 25-30,000 potential prospects.
The 2258 project will be a major element of the Installed Base Program (refer to that document for further details of the program).

Market Requirements

* The 2258 controller must be in the price range of current controllers, - ie., $750-2,500. Ideal price range is $1,000-$1,500 (manufacturing cost $300-500).

* Due to the lack of DSO 2200 expertise and field pre and post sales support, the VAR (Value Added Reseller) channels will be the primary channel of distribution, installation and support for data linking 2200s to VSs.

* There are still approximately 120 registered 2200 software vendors that could be utilized. When the DSO does not want to work with a 2200 System Houses, a local software vendor would install and support.

Internal Requirements and Testing

* At least 10 boards by 12/30/85. Five for internal, and five for beta.

* After our own internal testing, 1-2 selected VARs should be invited in for alpha testing with their application programs.

* 1-5 beta sites should than be established with the following VARs - Redshaw, Cobe, Northeast Datacom, CYCARE and AIMs as a software vendor.

Forecasts

The purpose of the 2258 is to sell VSs to existing users. Therefore, the business opportunity will not be in selling 2258s but rather incremental VS sales.

With a potential prospect base of 25,000 qualified users, a 10% success would provide 2,500 boards sales and 2,500 new VS sales. A 1% success would generate 250 board sales and 250 new VS sales.

Assuming 500 boards at a average selling price of $1,000 and a average VS sale at $50,000, the business potential is $500,000 in 2258 and $25,000,000 in VS sales over a 5 quarter period.

2258 Booking Forecast

	Q4 FY '86	Q1 FY '87	Q2 FY '87	Q3 FY '87	Q4 FY '87	Total
U.S. Units	50	50	60	50	50	260
Int. Units	5	30	50	90	65	240
WW Units	55	80	110	140	115	500

Announcements

	U.S.	INT.
Announcement Date	2/1/86	2/1/86
FCS	4/1/86	4/1/86
Volume	5/1/86	5/1/86

2200 LOCAL COMMUNICATIONS OPTION

Version 1.0

Design Specification

This design specification describes the software developed to link the 2200 with the VS via the 2258 controller. The three services provided over the link are as follows:

- o The 2200 will be able to create and use 2200 virtual disks on the VS.
- o VS system and filing services will be accessible to the 2200.
- o The 2200 will be usable as a VS DP workstation.

Date: December 25, 1985

Document Version 1.07

File Spec #151, library nori on LINDA

Development Group: Mickey Flanagan
Roger Kirk
Nori Odoi
Jerry Sevigny
Jose Valdepenas

TABLE OF CONTENTS

1.0 INTRODUCTION

The 2200 Local Communications Option is a high speed data-link which allows the 2200 to communicate with, co-exist with and utilize the facilities of the VS. Three functional services will be provided over the link. These functional services are -

a) VDISK - 2200 access to 2200 Virtual disk images which reside on the VS.

b) VS System and Filing Services - native VS services will be available to 2200 programs via the 2258 controller and the VS File Server Facilities of the VS.

c) VS Workstation Emulation - 2200 users will be able to attach to the VS using their 2200 workstations and execute data processing functions on the VS.

The 2258 controller is based upon the WPC 928 card with the added capability of responding to the 2200 just like a disk controller. The controller is Z80 based with 82K of memory. Since the Z80 is capable of addressing only 64K bytes of memory at any time, a hardware or hardware/software scheme must be developed to allow access to the additional memory.

The software effort is based upon an existing product called MWS. MWS is used in a number of PC products and offers an interface into the VS. Some functionality will be eliminated and others modified to function within the 2200 environment.

2.0 Architecture

Figure 1 is a block diagram which is designed to represent the interrelationships of the various components (modules) making up this project. Please refer to this diagram while reading the following architectural description.

The 2200 user will access the VS either -

thru existing BASIC-2 applications which will be able to access a 2200 disk image stored on the VS,

or thru new BASIC-2 applications which will be developed to provide VS File Services to the 2200 program thru Wang developed VS Access Subroutines,

or thru VS Workstation Emulation which provides VS DP workstation emulation.

2.1 <u>VS Workstation Emulation (VSWE)</u>

A 2200 user will invoke the VSWE option using his/her standard application access method (e.g. Wang Menus). Once executing, this BASIC-2 program will communicate thru the 2258 to the VS by using one of the three MWS workstation tasks. Keystrokes and screens can then be transmitted back and forth between the 2200 and the VS. The user's 2200 workstation is acting just like a VS DP workstation (due to physical keyboard differences some operational differences may be necessary).

2.2 <u>Native VS Facilities</u>

Prior to the 2200 user being able to utilize the native VS Services facilities, he/she must "attach" to the VS. To do this, the user must select the Attach DMS/VDisk utility from the VS services menu. The user will then enter workstation emulation, but will be prompted to enter the appropriate userid selected to be used for DMS/VDisk. The userid is determined by supervisory functions on the VS which allow the user to specify the startup program.

A user-written procedure should then bring the user into the 2200-access screen and eventually into the actual attach utility. This utility requires no user input, and leaves the user in the VS Services menu.

It is also necessary for the user to "detach" from the VS prior to shutting down his/her 2200 or whenever he/she no longer wishes to use the VS File Services. This is accomplished by selecting the Detach VS File Services option.

Note that "Attach" attaches all partitions and terminals of the 2200 to the VS file serving system and that "Detach" detaches all from the VS. For this reason, only terminal 1 is permitted to attach, detach, or change VDisk configuration. The other utilities are available to all terminals.

```
2200
     _____________________________________________________________________
     ! WS     !        !        !    VDisk     ! VS ACCESS !            !
     ! Emul.  ! Attach ! Detach ! Utilities    ! Routines  !     ...    !
     ! 3.1.1  ! 3.1.2  ! 3.1.3  !3.1.4 - 3.1.7!   3.1.8   !            !
     _____________________________________________________________________
     !                                                                   !
     !                           2 2 0 0   O S                           !
     !                                                                   !
     _____________________________________________________________________
     !     BASIC-2 Disk Statements    !    $GIO Statements for -         !
     !                                !      .  VS File Services         !
     !                                !      .  VS Workstation Services  !
     _____________________________________________________________________
                                     /
                                    /_
                                      !  !
                                    2200 BUS
                                    __!  !__
                                            /
2258                                       /
     _____________________________________________________________________
     !                                                                   !
     !                2258/2200    I N T E R F A C E         (3.2.2)     !
     _____________________________________________________________________
     !     2200 Disk Interface        !   Communications Interface       !
     !       Routines (3.2.3)         !       Routines (3.2.4)           !
     _____________________________________________________________________
     !   V S   F I L E   S E R V I C E S   !   VS Workstation            !
     !       REQUEST ROUTINES (3.2.5)      !   Services (3.2.6)          !
     _____________________________________________________________________
     !                                                                   !
     !                M W S   I N T E R F A C E  (3.2.7)                 !
     _____________________________________________________________________
     ! VS File      !  Workstation   !  Workstation   !   Workstation    !
     ! Server       !                !                !                  !
     ! Transport    !      #1        !      #2        !       #3         !
     _____________________________________________________________________
                                   /
                                  /_
                                    !  !
                                SERIAL CABLE
                                  __!  !__
VS                                        /
                                         /
     _____________________________________________________________________
     !                        S E R I A L   I O P                        !
     !                            V S   O S                              !
     _____________________________________________________________________
     !   VS File Server    !  WS  !  WS  !   . . .    ! Workstation      !
     !      (3.3.1)        ! Task ! Task !   . . .    !    Task          !
     _____________________________________________________________________
```

Figure 1 - 2200 / VS Local Communications Option Software Block Diagram

New BASIC-2 applications can now make use of the Wang-developed VS Access Routines to make VS native service requests. These requests such as to Open, Read or Write a file are in the form of $GIO statements. These statements cause a block of data to be transmitted to the 2258 interface which passes the data onto the VS File Server on the VS via the VS File Server Transport of the MWS code. Information retrieved by the VS File Server is returned to the BASIC-2 program via the VS File Server Transport through the 2258 interface.

2.3 VDisk Facilities

Before the 2200 user can use the VDisk facility, the 2200 must first be attached to the VS (see description under Native VS Facilities). The Attach VS File Services utility will automatically activate the default VDisk configuration file, i.e., it will open all VDisks listed in the file, according to the parameters in the file. These parameters are the mode in which they are to be opened, and the disk platter addresses associated with them.

VDisk configuration files are created and edited through the Edit VDisk Configuration File utility. The default configuration file may be changed through the Change VDisk Configuration utility.

The VDisks are actually created on the VS through the Create VDisk utility. The 2200 disk images stored on the VS are created as a VS file consisting of fixed length consecutive records of 256 bytes. Create creates the user-named file and initializes it to the image of a "formatted" 2200 disk. The 2200 user program must perform the "SCRATCH DISK" command prior to attempting to write to the file. The reason the utility does not do it is to allow the user to choose between either the new or old hash algorithms.

Now whenever the user executes a BASIC-2 program which uses the disk address associated with the 2258, the disk sequence will be translated into the appropriate VS request to perform the specified disk operation on the appropriate 2200 disk image.

The status of the VDisks (and DMS) is given via the View DMS/VDisk Status utility.

3.0 MODULE DESCRIPTIONS

The following sections identify the modules required to accomplish the functionality defined in the 2200 LCO Specification. Due to the nature of this project, this document is a working document and reflects the current understanding of the job.

The module descriptions are broken into three logical parts:

a) Code which will execute on the 2200 (BASIC-2),
b) Code which will execute on the 2258 controller,
c) Code which will execute on the VS.

3.1 BASIC-2 MODULES

The BASIC-2 modules give the 2200 user a user-friendly method of accessing the VS for file access or workstation emulation. All modules except for the VS access subroutines are available from the VS Services menu (Figure 3). These modules are VS Workstation Emulation, Attach DMS/VDisk, Detach DMS/VDisk, Change VDisk Configuration, View DMS/VDisk Status, Edit VDisk Configuration File, Create VDisk, and Delete VDisk. The ninth module, VS Access subroutines, will provide developers with tools to create and use files via VS filing services. The modules will be written in BASIC-2 and will be executed on the 2200. The VS Services menu will be reachable via the System Utilities menu (Figure 2).

The following factors were considered important in the design of the 2258 utilities:

(1) Logging on to workstation emulation and attaching for DMS/VDisk both require the user to logon to the VS.

If the user wishes to access VS DMS/VDisk, he/she should use the appropriate userid. The user should create a procedure which will automatically start up 2200 Access. Alternatively it should be possible to start up 2200 Access directly. After 2200 Access is running, the user need only suspend VSWE and select Attach VS File Services to make VS DMS/VDisk available to all 2200 partitions. The user will then be returned to the VS File Services menu of the 2200.

The user should create a VS procedure to automatically log the 2200 off the VS when Detach is selected. The user should always detach prior to turning off the system. Detach notifies the 2258 that DMS/VDisk is being shut down, resaves VDISKMAP, and returns to the 2200 Access screen.

If a message has been sent "immediately" to 2200 Access, thereby halting the DMS/VDisk task, the user can use the View utility to check on the problem. However, it is advisable to inhibit all messages from being sent to the 2200 Access workstation task through the VS operating system.

The View utility will allow the user at terminal 1 to re-enter workstation emulation to see what the problem is. If the user sees the VS command processor screen, he/she can read the offending message and decide what to do. If it is innocuous, the user can just resume DMS/VDisk. If not, the user can detach from the File Services and log off the emulation.

(2). The 2200 is a multi-user system and users most likely to upgrade to a VS most likely will have a multi-user 2200. Therefore the utilities should be designed to support multi-user systems as conveniently yet as securely as possible.

(3) More than one 2258 board may be used by a system. The 2258 only supports three workstation emulations and disk access. To support more than three workstations, more than one 2258 board is needed. Therefore the utilities must allow the user to specify which 2258 is to be used. This may be done either through the controller address or the disk addresses associated with the controller.

(4) The 2258 use of the VS for disk access (VDisk and VS access routines) is comparable to mounting and dismounting a disk pack. Any change to the VS connection affects all users on the system. This could be quite disasterous. For example, if terminal 4 detaches (logs off) the VS while terminal 3 is in the process of using VS files, terminal 3 will suddenly run into disk errors.

For this reason, VS disk access utilities that change the active configuration are restricted to terminal 1. These are Attach, Detach, and Change VDisk Configuration. View VDisk Configuration is available to all users, but it only allows the user at terminal 1 to re-enter workstation emulation and view the message problem. Thus the central operator is responsible for maintaining the VS connection.

(5) The VDisk Configuration file specifies what VDisks are to be open. The user can create any number of configuration files and must specify which is to be used when a 2258 is attached. An internal file, VDISKMAP, keeps track of the 2258 boards that have been attached, the configuration files used in that attach, and the VDisks that have been successfully opened. This file must be on the utility disk in order for the utilities to run correctly. If unauthorized access is made to the 2258 -- opening and closing VDisks without updating this file, the utilities will work incorrectly.

All VDisks in the configuration file are opened when the VS file system is attached, and all VDisks are closed on Detach. Change VDisk Configuration is the only other way to open and close new files after the session begins.

If the user chooses to change the active configuration, the activation process opens any added VDisks and/or closes and reopens any VDisks with changed parameters. If a VDisk is deleted from the configuration, it is closed. VDisks that have identical parameters in the configuration file are neither opened nor closed.

3.1.1 VS Workstation Emulation

This subsection describes the modules which make up the VS Workstation emulation software which is invoked by a 2200 user. It is similar to the WPC product and consists of 4 main modules which are responsible for initiating a session on the VS, handling screen and keystroke updates and allowing the user to suspend, return to, restart or terminate the emulation.

It is also the gateway for activating the VS File Services, since the user uses VS WSE to first logon to the VS and then attach to the VS File Services software. This will be described in greater detail in future revisions of this spec. Also see the section on the Attach to VS File Services utility.

```
                          !--------!
                          !   WS   !
                          !  Emul. !
                          !--------!
                              !
-----------------------------------------------------------------
       !                  !                  !                  !
--------------     --------------     --------------     --------------
!   Start    !     ! Initializat !    !   Main       !   ! Termination !
!   (VSWS)   !     ! (VSWSINIT)  !    !  (VSWSMAIN)  !   !  (VSWSTERM) !
--------------     --------------     --------------!    !-------------!
```

3.1.1.1 Start Module VSWS

This is the first module executed and its purpose is to see if an emulation was previously started by this user (partition). If so, then the user is presented with a menu which allows him/her to -

return to suspended emulation,
suspend emulation
terminate emulation
restart emulation
detach DMS/VDisk
or return from DMS/VDisk.

Otherwise, execution continues with the initialization module VSWSINIT.

3.1.1.2 Initialization Module (VSWSINIT)

This module handles all emulator initialization which consists of whatever has to be done to communicate with the 2258. If the emulation request cannot be granted (i.e. 3 emulations already started), the user will be brought back to the calling menu, otherwise execution will resume with the main program (VSWSMAIN).

3.1.1.3 Main Module (VSWSMAIN)

This module does the actual VS terminal emulation by soliciting and presenting screen data on the 2200 from the 2258 and soliciting keystrokes from the 2200 and transmits them to the 2258.

Screen updates are determined by polling the 2258 status line and asking whether the 2258 has anything for this partition.

A special keystroke sequence, similar to CONTROL SHIFT CANCEL on the Wang PC, will be used to allow the user to get out of the emulation. When this keystroke is entered, VSWSMAIN will execute the termination module VSWSTERM.

3.1.1.4 Termination Module (VSWSTERM)

The termination module is very similar to the Start Module VSWS, in that the user is presented with a menu of options which include -

return to emulation
suspend emulation
terminate emulation
restart emulation
attach DMS/VDisk
or detach DMS/VDisk.

NOTE: The user should never terminate emulation before logging off the VS. If this is done, then the emulation is just left hanging. The next user to use VSWE will get this abandoned screen rather than the expected logon screen. This happens because the VS does not automatically log off a turned off workstation, as does the OIS.

This needs to be well documented since some potentially very confusing situations may result.

3.1.2 Attach DMS/VDisk

This section describes the Attach DMS/VDisk utility. To attach to the services, the user must first select the Attach DMS/VDisk utility, and log on to the VS using a userid that has been defined to automatically run 2200 Access. On reaching the 2200 Access screen, the user must interrupt emulation and select the Attach DMS/VDisk pick. This will bring the user to the Attach utility described below.

Note that the VSWE interface with the Attach utility will not be completely defined until the next revision of this spec. Please contact Nori Odoi x76867 for more specific questions.

3.1.2.1 Functionality

The Attach Utility is equivalent to a mount disk function. It specifies a task in MWS to specifically enable 2200 users to use the VS as storage medium for files, whether virtual 2200 disks (VDisks) or standard VS files (via DMS subroutines).

The utility will automatically activate the default VDisk configuration file, thereby opening the specified VDisks for usage. This file can be set or changed by using the Change VDisk Configuration utility.

Attach can be run only from terminal 1, but after it is completed, DMS/VDisk is available to all users of the attached 2200.

3.1.2.2 Inputs

There are no inputs for this utility, only a display only screen.

No keys are active in this utility.

3.1.2.3 External Files Used

This utility will use VDISKMAP to obtain the default configuration file for that controller board. It will use the specified configuration file to open the selected VDisks. The formats of these files are given in Figures 10 and 11.

3.1.2.4 Shared Routines

Screens will be created via the APEERS screen package.

3.1.2.5 Algorithm

Given below is a very simplified pseudo code representation of how this utility will operate.

Note that if the 2200 is already attached to DMS/VDisk, the screen will still be presented, but a message will advise that the 2200 is already attached to the board and user ID specified. When a key is pressed, the user will be returned to the interrupt emulation menu.

ATTACH TO VS

```
get default from VDISKMAP file
present screen
VDISK-PROBLEM-FLAG = FALSE

/* Process screen */

/* Attach VS */
if (attach for specified 2258 board = TRUE)
        print ALREADY ATTACHED

else
        notify 2258 code that this is a DMS task
        if (config file specified)
            for (each VDisk in config file)
                if (VDisk cannot be opened)
                    VDISK-PROBLEM-FLAG = TRUE
                else
                    open file
                    record in VDISKMAP
            endfor
        print ATTACH SUCCESSFUL
        if (VDISK-PROBLEM-FLAG)
            print - NOT ALL FILES OPENED
goto VS Services menu
```

3.1.2.6 Validation

No screen validation is needed.

The 2258 will be queried to see if the specified board is attached already. If so, the user will be informed and instructed to detach the other first.

If not all VDisks can be opened, the user will be advised of the problem and instructed to use View VDisk/DMS Status to learn which files are open and which are not.

3.1.3 Detach DMS/VDisk

This section describes the Detach DMS/VDisk utility. It is important that Detach be run prior to powering down the system whenever the 2200 has been attached to the VS for use with either DMS or VDisk. Detach closes all open files and notifies the 2258 that the MWS task allocated for DMS/VDisk will no longer be used for these purposes.

3.1.3.1 Functionality

Detach is equivalent to a dismount disk function. The user must specify which 2258 board is being detached. (See Figure 5.) In the process of exiting 2200 access, the VS closes all open files.

It is advisable that the user set up a procedure that will automatically logoff the VS as the 2200 access program is exited.

The 2200 must detach from each attached 2258 board before powering down. Detach is available only to terminal 1.

3.1.3.2 Inputs

The only input is the address of the controller board.

Keys active in this utility will be EXEC to detach from the VS, and CANCEL and SHIFT CANCEL to exit.

3.1.3.3 External Files Used

VDISKMAP will be used to get the default controller address.

3.1.3.4 Shared Routines

Screens will be created via the APEERS screen package.

3.1.3.5 Algorithm

Given below is a very simplified pseudo code representation of how this utility will operate.

```
DETACH FROM VS

get default controller address from VDISKMAP
present screen

/* Detach VS */
if (EXEC)
    notify 2258 board of detach
    resave VDISKMAP to show board has been detached
    exit 2200 Access
    print DETACH SUCCESSFUL

else if (CANCEL or SHIFTCANCEL)
    EXIT = TRUE
return to VSWE (Command processor)
```

3.1.3.6 Validation

The controller address should be made up of three hex digits.

3.1.4 View DMS/VDisk Status

This section describes the View DMS/VDisk Status utility. This utility allows user to view the status of DMS/VDisk on the 2200.

3.1.4.1 Functionality

View displays the status of all currently open VDisks on the specified 2258 address. It also displays the name of the configuration file used to open the VDisks. If no VDisks are open, but DMS/VDisk is available, a message to this effect is displayed.

This utility is particularly useful for users if they have attached or downloaded a configuration file and have been warned that some files could not be opened.

The utility defaults the 2258 address to the first address in VDiskmap. By entering a different address and pressing EXEC, the user can examine the status of DMS/VDisk on other controllers.

If the DMS/VDisk are disabled, the user will be given a message to this effect. If he/she is at terminal 1, the user can then re-enter workstation emulation and attempt to fix the problem.

If more than 8 VDisks are open, NEXT SCRN and PREV SCRN will allow the user to page to the other screen. CANCEL and SHIFT CANCEL provide an exit.

This utility is available to all terminals, not restricted to terminal 1.

3.1.4.2 Inputs

The only input is the 2258 address. The board will be queried as to whether files are open and how many sectors they contain. If the 2258 is not "attached" to the VS, i.e., logged on for file access purposes, a message will be displayed indicating that DMS/VDisk are not available.

Keys active in this utility are EXEC, NEXT SCRN and PREV SCRN, and CANCEL and SHIFT CANCEL to exit.

3.1.4.3 External Files Used

VDISKMAP contains the current VDisk status. See Figure 11.

3.1.4.4 Shared Routines

Screens will be created via the APEERS screen package.

3.1.4.5 Algorithm

Given below is a very simplified pseudo code representation of how this utility will operate.

```
VIEW VDISK CONFIGURATION

Default 2258 address to first address in VDISKMAP
get VDISKMAP table
for (each VDisk)
    query 2258 for # of sectors in VDisk
    if (disk error)
        mark disk as NOT READY
present screen
EXIT = FALSE

/* Process screen */
while (not EXIT)
    if (EXEC)
        if (2258 address in VDISKMAP)
            get & present screen (as above)
            if (DMS/VDisk is disabled)
                if (user is at terminal 1)
                    allow option to re-enter WSE
                else
                    warn to see system administrator
        else
            print NOT ATTACHED

    if (NEXT SCRN & there is a next screen))
        present next screen

    else if (PREV SCRN & there is a previous screen)
        present previous screen

    else if (CANCEL or SHIFT CANCEL)
        EXIT = TRUE
endwhile
return to VS Services menu
```

3.1.4.6 Validation

The specified 2258 address must be a valid address.

3.1.5 Change VDisk Configuration

This section describes the Change VDisk Configuration utility. This utility allows users to change the default VDisk configuration file and gives them the option of activating it. VDisk configuration file are created and edited via the Edit VDisk Configuration File utility.

Note that all VDisks referenced must have been created via the Create VDisk utility.

3.1.5.1 Functionality

This utility allows the user to select the default configuration file. This is stored in in VDISKMAP. The user may also choose to activate the configuration file. If so, the utility opens any added VDisks and closes those that have been deleted so that the 2258 configuration matches the configuration file.

Note that if the description of a VDisk is unchanged, i.e., it was previously opened in the same mode and associated with the same 2200 platter, it will not be affected -- i.e., it will not be closed and then re-opened.

3.1.5.2 Inputs

The controller address and the configuration file name are the only inputs. They are validated as in other utilities.

Keys active in this utility are EXEC to change the configuration and CANCEL and SHIFT CANCEL to exit the program. After the configuration has been changed, the user will be asked if he/she wishes to activate the file. If EXEC is pressed, the file will be activated.

3.1.5.3 External Files Used

VDISKMAP is used to obtain current values, and is updated when the configuration is changed.

3.1.5.4 Shared Routines

Screens will be created via the APEERS screen package.

3.1.5.5 Algorithm

Given below is a very simplified pseudo code representation of how this utility will operate.

CHANGE VDISK CONFIGURATION

```
get VDISKMAP - default to 1st controller parameters
present screen
EXIT = FALSE

/* Process screen */
while (not EXIT)

    if (EXEC)

        if (valid controller address & config file name)
            update VDISKMAP
            present activate option
                if (EXEC)
                    for (first VDisk to last VDisk)

                    /* activate */
                    if (2258 address not attached)
                        print CONTROLLER NOT ATTACHED
                    else
                        if (VDisk parameters changed)
                            close & reopen VDisk
                        if (VDisk no longer defined)
                            close VDisk
                        if (VDisk cannot be opened)
                            print COULD NOT OPEN
                    endfor
                    print ACTIVATION COMPLETED

    else if (CANCEL or SHIFT CANCEL)
        EXIT = TRUE
endwhile
return to VS Services menu
```

3.1.5.6 Validation

The only validation necessary is that the controller address be the correct address of a controller board that is attached and that the configuration file be a valid 2200 file name.

To simply change the default configuration file, a 2258 need not be attached; however, to activate a configuration file, the specified 2258 must already be attached.

If a VDisk file in the configuration file cannot be accessed, the user will be referred to the View VDisk Utility to check if a VDisk file was actually opened or not.

3.1.6 Edit VDisk Configuration

This section describes the Edit VDisk Configuration utility. This utility allows users to edit or create a VDisk configuration file. A VDisk configuration file specifies what VDisks should be opened, which 2200 platter address should be associated with which VDisk, and whether the VDisk should be opened in Exclusive or Shared mode.

Note that all VDisks referenced should have been created via the Create VDisk utility. The 2200 need not be attached to DMS/VDisk.

3.1.6.1 Functionality

This utility allows the user to select the 2200 platter address, VDisk name, and/or the VDisk file mode. The configuration files to load from and save to will default to the first file given in VDISKMAP. If the configuration file selected already exists, its contents are displayed as a default. The user can refill the screen from another configuration file by entering a new configuration file name in the load from field and pressing return.

3.1.6.2 Inputs

The configuration file to load from is the file used to prefill from; the configuration file to save to is the file the edited configuration file is save in.

The 2200 address, the VDisk name, and the pick between shared and exclusive are needed to specify the VDisk. The configuration is saved to the file given by the configuration file name to save to.

Keys active in this utility are EXEC to save the new configuration, NEXT SCRN and PREV SCRN, and CANCEL and SHIFT CANCEL to exit the program.

3.1.6.3 External Files Used

VDISKMAP is used to obtain default values.

3.1.6.4 Shared Routines

Screens will be created via the APEERS screen package.

3.1.6.5 Algorithm

Given below is a very simplified pseudo code representation of how this utility will operate.

EDIT VDISK CONFIGURATION

```
get VDISKMAP - default to first configuration file
present screen
EXIT = FALSE

/* Process screen */
while (not EXIT)

    if (RETURN from configuration file to load from)
        refill screen from config file

    if (EXEC)

        if (valid config file name)
            /* Validation */
            if (valid 2200 address used more than once)
                print ERROR - ASSIGN ADDRESS ONLY ONCE

            else
                sort by 2200 addresses
                save file

    else if (NEXT SCRN & next screen exists)
        present next screen

    else if (PREV SCRN & previous screen exists)
        present previous screen

    else if (CANCEL or SHIFT CANCEL)
        EXIT = TRUE
endwhile
return to VS Services menu
```

3.1.6.6 Validation

The VDisk configuration is valid only if each 2200 address is assigned to only one VDisk, i.e., each address can only be used once.

3.1.7 Create VDisk

This section describes the Create Disk utility. This utility is required to create VDisks on the VS.

3.1.7.1 Functionality

Create VDisk creates user-named files equivalent to formatted but unscratched 2200 disks of specified size on the VS. The specified 2258 controller must be attached to the VS at the time Create VDisk is run.

The VDisk files consist of fixed length consecutive records of 256 bytes. The scratch operation can be done through the BASIC-2 SCRATCH DISK command at the user's convenience.

When the user presses EXEC, the VDisks are created. As each VDisk is created or fails to be created, the screen is changed to indicate this. In addition to letting the user know the status of the create, this also serves as an indicator that the program is still active.

3.1.7.2 Inputs

The user must specify what 2258 address is to be used, but the utility will default to the first address in VDISKMAP.

For each VDisk, the user must enter the complete VS file name and the platter size in sectors.

Keys active in this utility are NEXT SCRN and PREV SCRN, EXEC to make VDisks, and CANCEL and SHIFT CANCEL to exit.

3.1.7.3 External Files Used

VDISKMAP is provides the default value for the controller address.

3.1.7.4 Shared Routines

Screens will be created via the APEERS screen package.

3.1.7.5 Algorithm

Given below is a very simplified pseudo code representation of how this utility will operate.

<u>CREATE VDISK</u>

```
get VDISKMAP - default to first 2258 address
if (2258 not attached)
    print ERROR-2258 NOT ATTACHED
    allow exit
present screen
EXIT = FALSE

/* Process screen */
while (not EXIT)

    /* make new files */
    if (EXEC)

        for (first new VDisk to last new VDisk)

            if (platter size over 64,000 sectors)
                print ERROR - PLATTER SIZE TOO LARGE
            if (invalid VDisk name)
                print ERROR - INVALID NAME
            else
                /* Create VDisk */
                use DMS to create VDisk file on VS
                if (VS file error)
                    print ERROR - NOT CREATED
                else
                    update VDISKMAP
                    print FILE CREATED
        endfor
        print CREATE COMPLETED

    /* Process other keys */
    else if (NEXT SCRN and next screen exists)
            present next screen
    else if (PREV SCRN and previous screen exists)
            present previous screen
    else if (CANCEL or SHIFT CANCEL)
            EXIT = TRUE

endwhile
return to VS Services menu
```

3.1.7.6 <u>Validation</u>

The first validation checks to see if the 2258 controller is attached. This is done at the beginning and whenever the 2258 address is changed.

The platter size must not exceed 64,000 sectors -- (16K * 1000)/256 bytes per sectors -- since the maximum disk size is 16MB.

If errors are received from the VS as the VDisks are being created, an error message is printed, and the program proceeds to the next VDisk.

3.1.8 Delete VDisk

This section describes the Delete VDisk utility. This utility allows users to delete a VDisk from the VS.

3.1.8.1 Functionality

Delete VDisk allows the user to delete a VDisk from the VS. The user must specify a controller address to be used during the delete process. The address defaults to the first one in VDISKMAP.

3.1.8.2 Inputs

The user must input the VDisk name, and the controller address to be used.

Keys active in this utility are EXEC to remove the VDISK, and CANCEL and SHIFT CANCEL to exit the utility.

3.1.8.3 External Files Used

VDISKMAP provides the default for the controller address.

3.1.8.4 Shared Routines

Screens will be created via the APEERS screen package.

3.1.8.5 Algorithm

Given below is a very simplified pseudo code representation of how this utility will operate.

<u>DELETE VDISK</u>

```
get VDISKMAP - default controller address to first entry
present screen
EXIT = FALSE

/* Process screen */
while (not EXIT)

    /* Delete disk image */
    if (EXEC)

        if (2258 not attached)
            print ERROR - CONTROLLER NOT ATTACHED

        else
            print "ARE YOU SURE"
            if (yes)
                delete VDisk from VS
                if (unsuccessful)
                    print ERROR - COULDN'T DELETE VDISK

    else if (CANCEL or SHIFT CANCEL)
        EXIT = TRUE
endwhile
return to VS Services menu
```

3.1.8.6 <u>Validation</u>

The controller address must be attached to the VS. If the VDisk name does not exist on the VS, an error will be printed. Note that the user should be notified of this error, so that he/she can make sure that the correct volume has been mounted.

BASIC-2 UTILITY SCREENS

Figure 2. System Utilities Menu with VS Services added

```
_____________________________________________________________________
!                      ***** SYSTEM UTILITIES *****                  !
!                                                                    !
!Select item with SPACE & BACKSPACE.                  Partition x, xxK!
!Key RUN to execute, CLEAR or PREV SCRN for previous screen.  Terminal x!
!                                                                    !
!                                                                    !
!                                                                    !
!                 - Partition generator                              !
!                 - Partition status                                 !
!                 - Format disk platter                              !
!                 - Move file                                        !
!                 - Backup platter                                   !
!                 - Recover from backup                              !
!                 - System install                                   !
!                 - Initialize date & time                           !
!                 - Vertical format control                          !
!                 x VS services                                      !
!                                                                    !
!                                                                    !
!____________________________________________________________________!
```

Figure 3. VS Services Menu

```
_____________________________________________________________________
!                      ***** VS SERVICES *****                       !
!                                                                    !
!Select item with SPACE & BACKSPACE.                  Partition x, xxK!
!Key RUN to execute, CLEAR or PREV SCRN for previous screen.  Terminal x!
!                                                                    !
!                                                                    !
!                 - VS workstation emulation                         !
!                 - Attach DMS/VDisk                                 !
!                 - Detach DMS/VDisk                                 !
!                 - Change VDisk configuration                       !
!                 - View DMS/VDisk status                            !
!                 - Edit VDisk configuration file                    !
!                 - Create VDisk                                     !
!                 - Delete VDisk                                     !
!                                                                    !
!                                                                    !
!                                                                    !
!                                                                    !
!                                                                    !
!                                                                    !
!                                                                    !
!____________________________________________________________________!
```

Figure 4. Attach VS File Services Screen

```
                        ***** ATTACH DMS/VDISK *****









              Attaching to VS through controller: HHH
                   Primary VDisk platter address: HHH
                        VDisk configuration file: AAAAAAAA












```

Figure 5. Detach from VS Screen

```
                        ***** DETACH DMS/VDISK *****











                        Enter controller address: ___












                                                    EXEC - Detach from VS
                                                  CANCEL - Exit

```

Figure 6. Change VDisk Configuration

```
 _______________________________________________________________________
!                 ***** CHANGE VDISK CONFIGURATION *****                 !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                       Enter controller address: ___                    !
!                   Enter configuration file name: ________              !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                                           EXEC - Change configuration!
!                                         CANCEL - Exit                  !
!________________________________________________________________________!
```

Figure 7. View DMS/VDisk Status

```
 _______________________________________________________________________
!                 ***** VIEW DMS/VDISK STATUS *****                      !
!                                                                        !
!                 Enter controller address: ___                          !
!                 Enter configuration file: ________                     !
!                                                                        !
!                                                                        !
! 2200   VS file to be VDisk          Platter          Exclusive         !
! add.   (Volume:Library.FileName)    size (sectors)   or Shared?        !
! ___    __________________________   ______             Exclusive       !
! ___    __________________________   ______             Shared          !
! ___    __________________________   ______             Exclusive       !
! ___    __________________________   ______             Exclusive       !
! ___    __________________________   ______             Not Ready       !
! ___    __________________________   ______             Exclusive       !
! ___    __________________________   ______             Exclusive       !
! ___    __________________________   ______             Exclusive       !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!                                      PREV SCRN - Previous screen       !
!                                      NEXT SCRN - Next screen           !
!                                           EXEC - New controller        !
!                                         CANCEL - Exit                  !
!________________________________________________________________________!
```

Figure 8. Edit VDisk Configuration

```
!                  ***** EDIT VDISK CONFIGURATION FILE *****                       !
!                                                                                  !
!                        Load from configuration file: ________                    !
!                          Save to configuration file: ________                    !
!                                                                                  !
!                                                                                  !
! 2200     VS file to be VDisk              Exclusive                              !
! add.     (Volume:Library.FileName)        or Shared?                             !
! ____     __________________________        x E  _ S                              !
! ____     __________________________        x E  _ S                              !
! ____     __________________________        x E  _ S                              !
! ____     __________________________        x E  _ S                              !
! ____     __________________________        x E  _ S                              !
! ____     __________________________        x E  _ S                              !
! ____     __________________________        x E  _ S                              !
! ____     __________________________        x E  _ S                              !
!                                                                                  !
!                                                                                  !
!                                              PREV SCRN - Previous screen         !
!                                              NEXT SCRN - Next screen             !
!                                                   EXEC - Save file               !
!                                                 CANCEL - Exit                    !
!                                                                                  !
```

Figure 9. Create VDisk

```
!                          ***** CREATE VDISK *****                                !
!                                                                                  !
!                  Enter controller address: ___                                   !
!                                                                                  !
! VS file to be VDisk                Platter                                       !
! (Volume:Library.FileName)          size (sectors)                                !
! __________________________         _____                                         !
! __________________________         _____                                         !
! __________________________         _____                                         !
! __________________________         _____                                         !
! __________________________         _____                                         !
! __________________________         _____                                         !
! __________________________         _____                                         !
! __________________________         _____                                         !
!                                                                                  !
!                                                                                  !
!                                                                                  !
!                                                                                  !
!                                                                                  !
!                                              PREV SCRN - Previous screen         !
!                                              NEXT SCRN - Next screen             !
!                                                   EXEC - Create VDisk(s)         !
!                                                 CANCEL - Exit                    !
!                                                                                  !
```

Figure 10. Delete VDisk

```
 ___________________________________________________________________________
!                         ***** DELETE VDISK *****                          !
!                                                                           !
!                                                                           !
!                                                                           !
!                                                                           !
!                                                                           !
!                                                                           !
!                    Enter controller address: ___                          !
!                          Enter VDisk name: _________________________      !
!                                            (Volume:Library.FileName)      !
!                                                                           !
!                                                                           !
!                                                                           !
!                                                                           !
!                                                                           !
!                                                                           !
!                                                                           !
!                                                                           !
!                                                                           !
!                                                                           !
!                                                     EXEC - Delete VDisk   !
!                                                   CANCEL - Exit           !
!___________________________________________________________________________!
```

Figure 10. VDisk Configuration File Format

2200 Disk Address (3 bytes)	VS File Name (26 bytes) (Volume-Library-File Name)	Exclusive/Shared (1 byte)

Up to 32 entries.

Figure 11. VDISKMAP File Format

2258 Address (3 bytes)	Current VDisk Configuration File Name (8 bytes)		
2200 Disk Address (3 bytes)	VS File Name (26 bytes) (Volume-Library-File Name)	File Handle (2 bytes)	Exclusive/Shared (1 byte)

Up to 68 entries.

3.1.8 VS Access Subroutines

The VS Access Subroutines are responsible for formatting the required VS File Services Request message to be sent over the 2258 to the VS; they will perform all the necessary housekeeping.

For details concerning these routines please consult the 11x Object Management/Access Functional Specifications document as well as 11x Object Management/Access Message Definition Specification.

In general, the following requests are supported -

a) VS CATALOG OPERATIONS

OPEN, CLOSE, CREATE, READ, DELETE, RENAME
GET ATTRIBUTES

b) VS FILE OPERATIONS*

OPEN, CLOSE, READ, WRITE, SEEK, FLUSH,
LOCK, UNLOCK, GET ATTRIBUTES, SKIP,
REWRITE, DELETE, FIND

* Not all functions apply to all file types.

C) VS QUEUE OPERATIONS

OPEN, CLOSE, INSERT

3.2 2258 Microcode

The modules described in this section are executed within the 2258 controller itself. When power is applied, the 2258 Loader is loaded into Z80 memory and execution is passed to it. The loader is then responsible for loading the executable code which contains the the modules which are described in this section.

```
                                 !--------!
                                 !  2258  !
                                 !  CODE  !
                                 !--------!
                                     !
      -------------------------------------------------------------
      !          !          !           !          !           !
  ---------- ---------- ----------- ---------- ---------- ----------
  !2258/2200! !  Disk  ! ! Comm     ! ! File  ! !   WS   ! !  MWS    !
  !Interface! !Interface! !Interface! !Services! !Services! !Interface!
  ---------- ---------- ----------- ---------- ---------- ----------
```

3.2.1 2258/2200 Interface

The 2258/2200 Interface module exists to control the handshaking that goes on between the 2200 CPU and any of its peripheral controllers. All communications are accomplished a byte at a time, sometimes with acknowledgment and other times just by using READY/BUSY signals.

By design, the 2200 initiates all activities, the controllers generally are "polled" by the 2200. Thus, whenever the 2258 is enabled, the interface task must determine how it has been enabled. Under our current design, the 2258 can be enabled as a disk, enabled for status, or enabled to receive or request data.

```
                          !--------------!
                          !  2258/2200   !
                          !   Interface  !
                          !--------------!
                                 !
        -------------------------------------------------------
        !                        !                        !
  ---------------         ---------------        -----------------
  ! Disk        !         ! Status      !        ! Communications !
  !  Interface  !         ! Enabled     !        ! Interface      !
  ---------------         ---------------        !----------------!
```

Communication between the 2200 and the 2258 will make use of two methods; one being a byte by byte protocol and the second using DMA (if available) where appropriate.

The disk and communication interface modules are covered in the next section. Therefore, a little discussion about Status would be appropriate here. The Z80 microcode can set 3 READY/BUSY status bytes. One for being enabled as a disk, one for a communications controller and one for status. The ready/busy indicates whether the controller is willing to communicate with the 2200 as a disk, a communication controller or has some data for a 2200 partition.

An executing 2200 application will on some regular basis "poll" the 2258 to see if the board has any data for it, by enabling the 2258 at the status address. If the 2200 program sees a READY, it will then enable the 2258 at a command address, and determine whether the available data is for its particular partition.

For a more detailed discussion concerning how the 2258 is addressed, communicates with the 2200 and controls READY/BUSY, please consult a memo from Bruce Patterson and Dave Angel entitled 2200 / VS LINK -- DESIGN RECOMMENDATIONS or its current replacement.

3.2.2 Disk Interface Routines

Whenever the 2258/2200 Interface routine recognizes that the controller has been accessed at a disk address of 10, 20 or 30, control will be passed to the Disk Interface Module.

This module manages the interface between the 2200 and the Virtual Disks (VDisks) on the VS. It handles the disk handshaking which the 2200 Operating System would expect from a 2280-like disk device. This is accomplished by handling a byte-by-byte protocol consisting of data being sent to the 2258 and the 2258 performing certain acknowledge functions. A very detailed explanation of this protocol may be found in a 1980 document written by Max Blomme entitled "2200 Disk Command Sequences for the 2280 Drives".

The protocol is handled a single byte at a time, no DMA is available.

The disk interface routine will be capable of supporting the following functions which are transparent to the user in that the 2200 OS breaks down the various BASIC-2 disk statements into one or more the following functions.

Each of the requests are transformed into a VS File Service Request which is passed to the VS using the VS File Server transport task of MWS. Our current design defines a 2200 disk image as a VS File consisting of fixed length consecutive records of 256 Bytes. Therefore, the requests are translated into operations valid for consecutive record files only.

a) <u>Hog A Disk</u>

When the user sets the "hog" address bit (hex(80)) or executes a $OPEN at the controller address, all files defined in the configuration file will be set to Open Exclusive (versus Shared).

b) <u>Read A Sector</u>
c) <u>Write A Sector</u>
d) <u>Compare Sector (Read After Write)</u>
e) <u>Format Platter</u>
f) <u>Copy Sectors</u>
g) <u>Start Multi-Sector Write</u>
h) <u>End Multi Sector Write</u>
i) <u>Verify Sector</u>

3.2.3 Communications Interface Routines

Whenever the 2258/2200 Interface routine recognizes that the controller has been accessed at a non-disk address or a "so called" communications address, control will be passed to the Communications Interface Module.

The purpose of this module is to handle the transfer of a VS Native Services Request or a Workstation Service Request between the 2258 and the 2200. In very simple terms, this module handles all functionality except for VDISK.

The current design calls for the burden to be put on the 2200 program to transmit the request in a format which provides identification information to the controller and requires little or no change prior to being passed to the VS.

The requests which will be supported are discussed under either the VS File Services Request Routines or VS Workstation Services Request Routines with the exception of the following 2258 internal commands -

a) Disk Hog (2200 $OPEN statement)
b) Disk Unhog (2200 $CLOSE statement)
c) Download VDISK Configuration File
d) Any input for partition xx?
e Has anyone attached?

3.2.4 VS File Services Request Routines

The VS File Services Request module will extract the information which it needs to identify what type of response is expected and what to do with the response.

3.2.5 VS Workstation Request Services

Due to the development group's general lack of understanding about MWS, this section is undefined. In general, this section receives IOCWs and IOSWs from the VS and routes them to the appropriate partition in the 2200 and vica versa.

3.2.6 MWS Interface Routines

The purpose of this module is to accept requests from either the VS File or Workstation Services modules and forward them to either the VS File Server Transport or the appropriate Workstation task of MWS.

The module will also return any response data received from the VS to the appropriate service module for return to the appropriate 2200 partition.

3.3 VS Modules

The modules listed in this section are executed on the VS.

3.3.1 VS File Server

The VS File Server task is started on the VS whenever a 2200 user "attaches" to the VS. Its responsibility is to respond to File requests.

3.3.2 VS Workstation Task

The VS Workstation task is started on the VS whenever a 2200 user logs onto the VS via the VS Workstation Emulation program. Its responsibility is to respond to screen and keyboard requests.

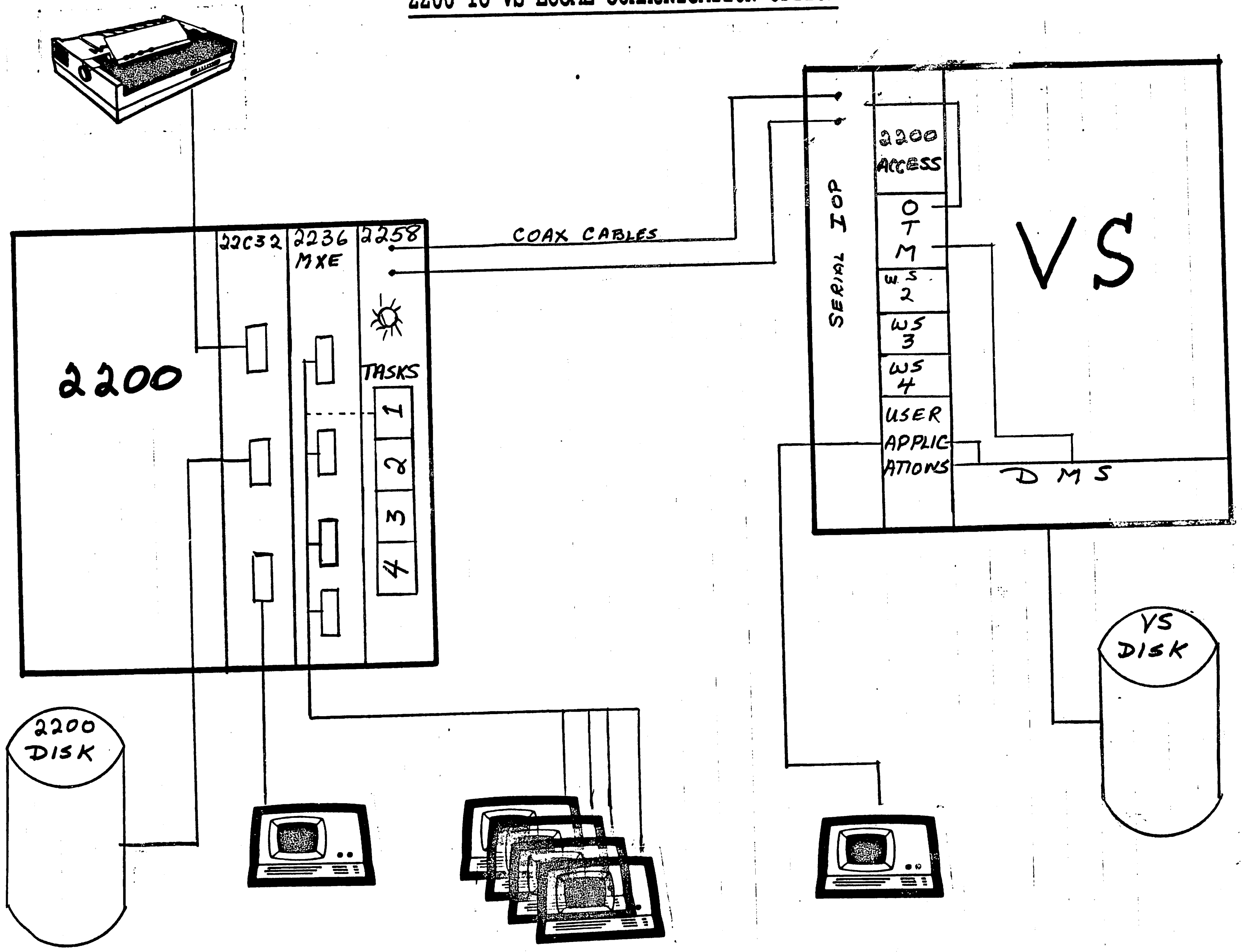
4.0 Maintenance Section

4.1 Assembly/Compilation Procedures
4.2 Link Procedures
4.3 Debug Aids

5.0 Performance Data

5.1 Code Size
5.2 Data Size
5.3 Interrupts Disabled
5.4 Timing

2200 TO VS LOCAL COMMUNICATION OPTION



HRG 1-03-86

TECHNICAL REVIEWER'S COMMENT FORM

Document Title 2200/VS Local Comm. Option D/S Pub. # 715-0721

Writer Blaise A. Corcoran M/S 1225 Tel. Ext. 7-2210

Date Submitted 24 February 1986 Return Due Date 3 March 1986

Return to Blaise A. Corcoran M/S 1225

ATTENTION REVIEWER:

Please review the attached copy, keeping the following points in mind.

1. As a technical reviewer, your responsibility is to check the accuracy of the document. A literary review usually occurs after the writer has responded to the technical reviewers' comments.

2. Make your comments as specific as possible, writing them directly on the document where they apply. If you have general questions or comments, do not hesitate to contact the writer.

3. Note the date you should return your review. If you cannot meet this date, please contact the writer. Comments received thereafter may be too late to affect revision of the text or artwork.

Please verify that you have reviewed the attached document by checking one of the following:

Accept Document ________

Accept Document with Changes ________

General Comments: (use back of sheet if necessary)

Reviewer's Signature ____________________ Date ____________

Review List:

Gene Schultz	M/S 13A9C	Bob Rainville	M/S 14A3A
Gerry Sevigny	M/S 1489	Nori Odoi	M/S 1489
Paul Plouffe	M/S 1439	Mike Riley	M/S 1469
Steve McGarry	M/S 14A3A		

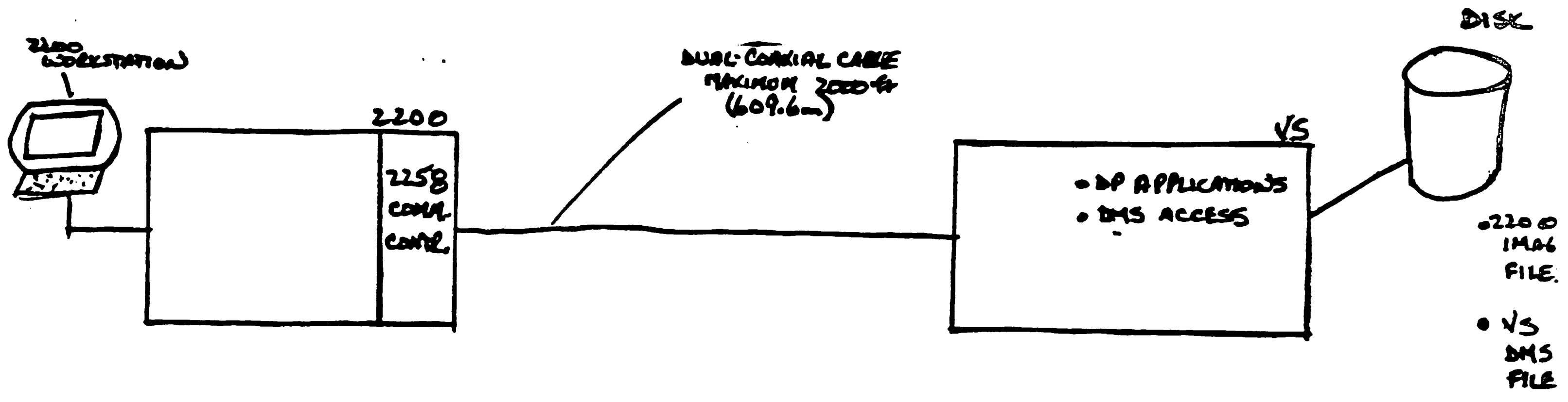
FYI

Hal Woods	M/S 0122	Bob Johnson	M/S 1225
Roger Lynn	M/S 0115	Ruth Ellis	M/S 0121A
F. Medeiros	M/S 2522	Sue Cawley	M/S 13A4
Mike Calvi	M/S 1511B		

OVERVIEW

The 2200/VS Local Communications Option (LCO) is a data communications hardware and software option that enables a Wang 2200MVP, -LVP, or Micro-VP system to communicate with a Wang VS computer system. Communications between the 2200 system and the VS system occur at speeds of 4.27 megabits per second (Mbps) over dual coaxial cable facilities.

The 2200/VS LCO is intended for those 2200 environments that have a local VS system (up to 2000 ft from the 2200) with an available serial port for a coaxial cable connection. While maintaining the efficient, stable and familiar 2200 systems and applications software environment, the 2200/VS LCO provides 2200 users with convenient access to additional VS disk storage (for 2200 file storage and VS file storage) and to the complementary VS applications.



With the 2200/VS LCO hardware and software installed on a 2200, a coaxial connection to the VS, and an appropriately configured VS system, a 2200 user can perform the following:

- Log on to the VS and run VS application programs that do not require the downloading of microcode to a VS workstation. Among the available applications are

  - All DP applications
  - Wang OFFICE
  - Professional Application Creation Environment (PACE)
  - VS Batch Communications

- Run 2200 application programs that store data to and retrieve data from the 2200 disk-image files on the VS system's disk storage, without modifying the existing 2200 application programs. The 2200 disk-image files on the VS are created by utilities included with the 2200/VS LCO package.

- Run 2200 application programs that store data to and retrieve from native VS files, using subroutines provided with the 2200/VS LCO package. For example, the 2200 user can run one 2200 program to update inventory and run another 2200 program to take the results and add them to a PACE database file on the VS.

PRODUCT FEATURES

As a hardware and software package, the 2200/VS LCO provides a new communications controller. The 2258 Communications Controller is an internal microprocessor-based controller mounted in an empty slot in the 2200 chassis. It supports up to four concurrent sessions between 2200 users and the VS.

When the 2258 controller is installed in the 2200 chassis, the Wang Customer Engineer sets two addresses on the board: a 2200 disk address (for VDISK requests) and a Communications address (for workstation emulation and VS Filing requests). After the 2258 controller is installed and the 2200 is powered on, the controller receives microcode from the VS and initializes itself.

After the 2258 controller is initialized, a 2200 user can run one of the three LCO components:

- Workstation Emulation
- VDISK
- Native VS Filing Services

Workstation Emulation

The workstation emulation component of the 2200/VS LCO package allows a 2200 user to log on to a VS system and access VS DP applications from any workstation on the 2200. It is also easy to use. You invoke workstation emulation from a menu similar to those provided with other Wang 2200 software packages.

Once you choose workstation emulation, you then provide the Communications address of the 2258 controller to use and select one of the session controls. You can start an workstation emulation session, terminate a session, suspend a session, return to a suspended session, or restart the session.

While in a workstation emulation session with the VS, a simple keystroke sequence enables you to exit temporarily from workstation emulation, suspend the emulation, and run a 2200 BASIC-2 application. The application can use disks connected to the 2200, VDISK files, or native VS files.

The 2200/VS LCO workstation emulation has the following features:

- Available from any workstation on the 2200

- Supports up to four workstation emulation sessions concurrently through a single 2258 controller. If your 2200 system use demands additional sessions, additional 2258 controllers must be installed on the 2200.

- Emulation of a VS 2256C workstation (DP only)

- A function strip that indicates how the 2200 workstation special function keys correspond to VS workstation keys

VDISK

The VDISK component of the 2200/VS LCO package allows a 2200 user to create 2200 disk-image files stored on the VS system using the VDISK Utilities. A 2200 disk-image file is a VS Data Management System (DMS) file that consists of 256-byte, fixed-length consecutive records. The VS DMS manages all disk file space and services all file I/O requests on the VS system.

Once a 2200 disk-image file is created on the VS, 2200 BASIC-2 application programs can write data to and read data from the VDISK file. The VDISK files can be shared with other 2200 systems that are equipped with the 2200/VS LCO package. However, the files cannot be shared with any applications on the VS.

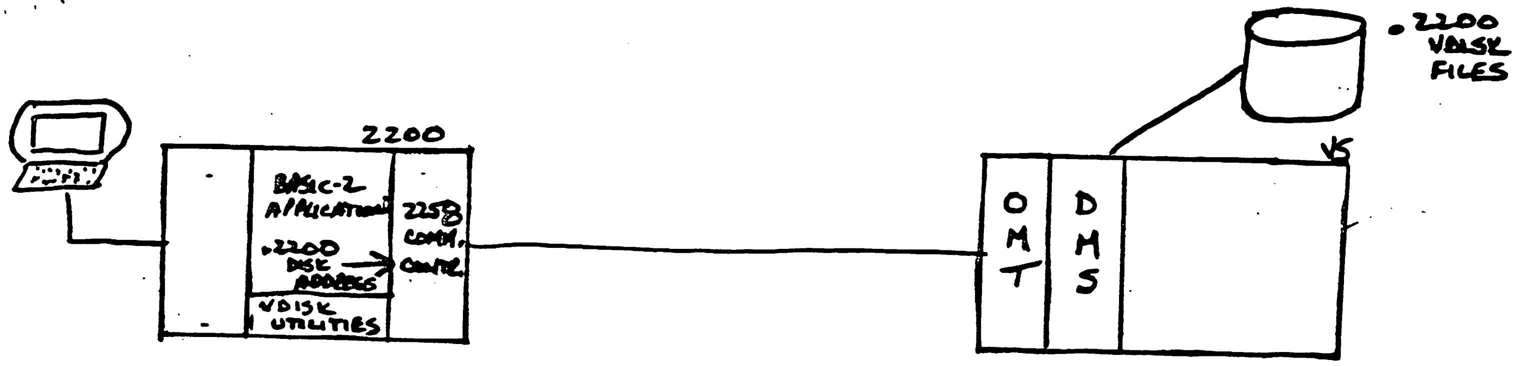
To run a VDISK session, a 2200 Workstation 1 operator starts a workstation emulation session, logs on to the VS, starts up the Object Management task (OMT) to handle filing requests, and suspends the emulation. The operator then attaches to VS filing services, opening up all VDISKs in the configuration file and making them available to all 2200 partitions.

When the VDISKs are opened, any 2200 user can run an application program and access the disks. Since the application program specifies a 2200 disk address in its disk statements, the 2258 controller recognizes the address and passes filing requests to the OMT program on the VS.

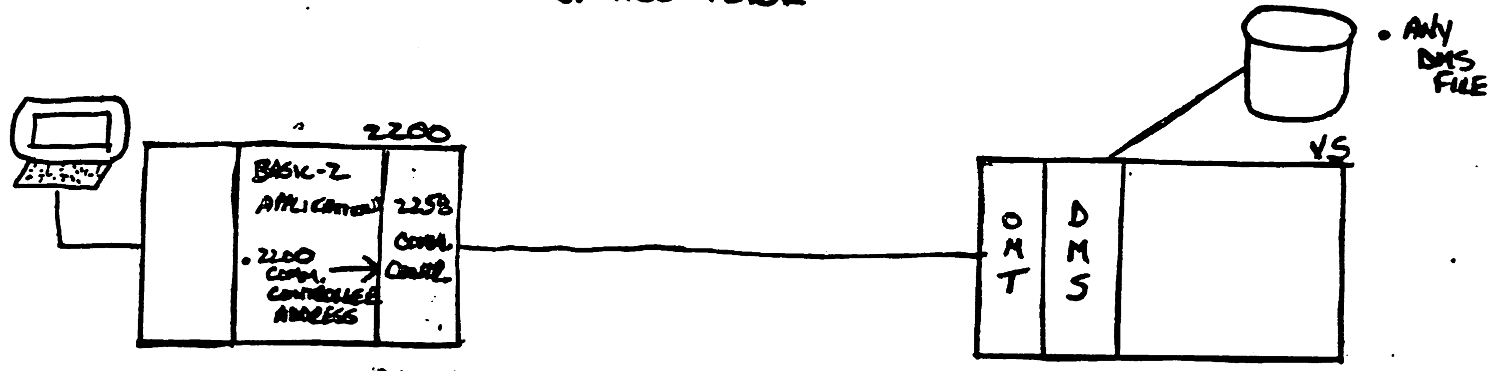
When the application program is finished processing, the user can then run another 2200 application. Only the Workstation 1 operator can close the VDISK files, change the VDISK configuration file, and terminate the VS filing services session.

The 2200/VS LCO VDISK has the following features:

- Supports 1 VS Filing Services session (both VDISK and native VS filing services) per 2258 controller
- Supports up to 32 VDISKs for each 2258 controller. If additional VDISKs are required, additional 2258 controllers must be installed on the 2200 system.
- Supports a VDISK maximum platter size of 64,000 sectors.
- Maintains a table containing each open VDISK. Each disk is assigned a 2200 disk address, VS file name, platter size in sectors, and mode (shared or exclusive).
- Requires no changes to the BASIC-2 application program unless the disk address in the program differs from the disk address assigned to the 2258 controller.
- Provides VDISK utilities to create, edit, view, or delete VDISKs in the configuration file. The utilities can only be run from Workstation 1 on the 2200. However, any 2200 user can view VDISK status from any workstation on the 2200.
- Supports the sharing of VDISK files with similarly equipped 2200 systems connected to a VS.
- Enables a 2200 user to copy a VDISK file to a 2200 disk.



a. LCO VDISK



b. LCO VS FILING SERVICES

Native VS Filing Services

The Native VS Filing Services component of the 2200/VS LCO package allows a 2200 BASIC-2 application program full access to native VS DMS files on the VS system. An application can open native DMS files, read, write, and delete records from DMS files, and close DMS files.

In order to gain access rights to DMS files, the 2200 BASIC-2 application program must address the 2258 controller as a communications device using the communications address assigned. All DMS file requests (e.g., open, read, write, and close) are handled by VS DMS File Access Subroutines provided with the LCO package. The DMS files can be shared with other VS systems and applications.

Starting a Native VS Filing Services session is no different from starting a VDISK session. A 2200 Workstation 1 operator starts a workstation emulation session, logs on to the VS, starts up the Object Management task (OMT) to handle filing requests, and suspends the emulation. The operator then attaches to VS filing services.

A 2200 user from any partition can run an application program. Since the application program specifies a 2200 communications address in its I/O requests, the 2258 controller recognizes the address and passes filing requests to the OMT program on the VS.

When the application program is finished processing, the user can then run another 2200 application. Only the Workstation 1 operator can terminate the VS filing services session.

The 2200/VS LCO Native VS Filing Services component has the following features:

- Supports 1 VS Filing Services session (both VDISK and native VS filing services) per 2258 controller
- Requires the addition of File Access Subroutines to the BASIC-2 application program. The File Access Subroutines are provided with the 2200/VS LCO package.
- Allows access to any currently defined native VS DMS file type. For example, consecutive, relative, and multiple-indexed ISAM (Indexed Sequential Access Method) files are supported.
- Supports the sharing of native DMS files with other VS applications and systems and with similarly equipped 2200 systems connected to a VS.

SPECIFICATIONS

Number of Sessions
4 workstation emulation sessions or 3 workstation emulation session and 1 VS Filing Services (both VDISK and native DMS file access) session per 2258 controller.

2200 to VS Connection

Operating Mode
Asynchronous, polled

Data Format
Serial

Data Rate
4.27 Mbps

Transmission Medium
Dual-coaxial baseband cable

2200 System Requirements

MVP OS Release 2.6 or greater

Requires a 28K memory partition to run the VDISK Utilities and Workstation Emulation.

Adding File Access Subroutines to a BASIC-2 application increases the program size by 5-7K.

add 2436 DW

Any 2336DE, 2336DW, 2226DW, 2236DE/DW, or 2326DW workstation can be used to run Workstation Emulation.

VS System Requirements

Operating System, Version 7.20 or greater.

One unused serial port. The serial port is configured for a 2258MWS device using the VS GENEDIT program.

A procedure the Workstation 1 operator can use to start up the VS Object Management task (OMT) software for processing the 2200 file requests (both VDISK and native DMS).

ORDERING INFORMATION

Model	Model Number
2200/VS Local Communications Option	2258-3 2258-5 2258-9

All hardware, software, and cable to connect the 2200 to the VS are included with the 2200 Local Communications Option product. The software is available in 5 1/4-inch double-sided double-density (DSDD) diskettes, 8-inch single-sided single-density (SSSD) or double-sided double-density (DSDD) diskettes.

The Wang dual coaxial cable supplied is 25 feet in length. Additional cable, to meet other length requirements, is available from Wang Direct in lengths up to 2000 ft (609.6 m). A workstation function strip is also included.

Standard Warranty Applies

TO:	Roger Kirk, Jerry Sevigny
FROM:	Nori Odoi
DATE:	May 15, 1986
SUBJECT:	2258 Command Changes Rev. 1.05.08

2258 Command Structure:

The first byte of all commands gives command type and partition number. The second byte of all commands except the Send Keystroke and the Query commands contains the actual command number. This makes processing by the Z-80 microcode easier. The general formats of the two bytes are given below:

Byte 1 (all commands):

```
 _______________________________
! Command type ' partition-1 !
! high nibble     low nibble !
 ‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾
```

Byte 2 (all commands except Send Keystroke and Query command):

```
 ____________
! Command # !
! 1 byte    !
 ‾‾‾‾‾‾‾‾‾‾‾‾
```

Note that partition numbers on the 2200 are numbered from 1 to 16. In the 2258 commands, they will be considered to range from 0 to 15, which allows them to be stored in one nibble. Thus the 2258 partition number is always equal to 2200 partition number - 1.

The Query command only needs byte 1 to ask the 2258 "Is it me?" when the 2258 status address is READY. The 2258 will respond "0" if the answer is "No". If the 2258's answer is "Yes, it's you", the 2258 will respond as described in the command summary description of the Query command.

The Send Keystroke command is two bytes long. The first byte will be as described above, and the second byte will be the VS scan code of the keystroke. See command summary for more detail.

Generally a 2258 response to a command will be "0" if no and non-zero if yes. However, in the case of commands with more than one flavor of no, "FF" will be yes and less than "FF" will be no.

2200/2258 Communications Procedures

2200/2258 Link: To simplify and streamline communications between the 2200 and the VS, both the 2200 and MWS maintain the current screen image. The 2200 accepts each keystroke, updates the screen, then sends the keystroke to the Z-80 code. The Z-80 does a minimum of bookkeeping and then passes it on to MWS. MWS updates the screen image, sends back error messages, and/or constructs and sends IOSW's to the VS.

The advantages of this design are:

(1) This eliminates sending entire screens along the 2200 bus to the 2258. Only one-byte keystrokes need be sent.

(2) MWS uses code already written and designed to handle screens, making the 2200 Work Station Emulation Code easier to write and maintain.

(3) The command structure needed for communication between the 2200 and the 2258 is simpler -- commands like update screen image and send IOSW can be eliminated altogether.

Receiving from the VS. When waiting for an IOCW/data from the VS, the following sequence should be followed:

```
While (not READY & not DONE))
   check for READY at the 2258 status address (using $IF ON)
   if (2258 is ready at the status address)
         send Query command
         if (2258 responds non-zero)
               send command & partition number, to request input
               if (2258 responds non-zero)
                     send remainder of command/data
               else
                     try again
               set DONE
end while
```

SUMMARY OF 2258 COMMANDS

The following is a summary of all suggested commands with a brief discussion of what they do. Command numbers are in hexadecimal. The upper nibble of the command byte identifies the class of command.

Note that commands are by definition sent from the 2200 to the 2258 card. The sending program must then get the 2258's response. The possible responses are also described below.

QUERY COMMAND - Upper nibble = 0

COMMAND NUMBER	COMMAND NAME	COMMAND DESCRIPTION
Op (1 byte)	What do you have for me?	This command will always contain 0 in its upper nibble. The lower nibble will contain the partition number - 1.

```
 ______________________________
 ! 0            ' partition-1 !
 ! high nibble    low nibble  !
 ______________________________
```

The 2258 will respond:

Not for you:

```
 ____________
 ! 00       !
 ! (1 byte) !
 ____________
```

Something for you:

```
 ____________
 ! non-zero !
 ! (1 byte) !
 ____________
```

The non-zero response will have at least one of the first four bits set. These bits (beginning with the least significant) signify as follow:

Bit 0 (D0):	Ready to send a screen
Bit 1 (D1):	Ready to receive VSS (VS Services) command, i.e., your turn to send command
Bit 2 (D2):	Ready to send Ack/Nack, i.e., a response to VSS command

SEND KEYSTROKE COMMAND - Upper nibble = 1

The 2258 response to the Send Keystroke command is 0 if error and non-zero if accepted. The lower nibble of the non-zero response will have one bit set which will indicate which MWS task is involved. The upper nibble will contain other information. The bits (beginning with least significant bit) mean as follow:

Bit 0 (D0):	Task 0 assigned
Bit 1 (D1):	Task 1 assigned
Bit 2 (D2):	Task 2 assigned
Bit 3 (D3):	Task 3 assigned
Bits 4-6 (D4-D6):	Reserved
Bit 7 (D7):	WS Emulation active if set

COMMAND NUMBER	COMMAND NAME	COMMAND DESCRIPTION
1p	Send Keystroke to VS	This command sends a 2200 keystroke to the VS. The first byte identifies the command and the partition number (lower nibble = partition number in hex - 1). If the 2258 responds with non-zero, the 2200 can then send the VS scan code of the keystroke pressed.

2200 command (1 byte):

```
_________________________________
! 1           ' partition-1 !
! high nibble    low nibble !
_________________________________
```

The 2258 response:

```
  error:        ____________
                ! 00       !
                ! (1 byte) !
                ____________

  accepted:     ____________
                ! non-zero !
                ! (1 byte) !
                ____________
```

The 2200 will then send the keystroke:

```
_____________
! scan code !
! (1 byte)  !
_____________
```

WORKSTATION EMULATION COMMANDS - Upper nibble = 3

Generally the 2258 response to a workstation emulation command is 0 if negative and non-zero for a positive response. The lower nibble of the non-zero response will have one bit set which will indicate which MWS task is involved. The upper nibble will contain other information. The bits (beginning with least significant bit) mean as follow:

Bit 0 (D0):	Task 0 assigned
Bit 1 (D1):	Task 1 assigned
Bit 2 (D2):	Task 2 assigned
Bit 3 (D3):	Task 3 assigned
Bits 4-6 (D4-D6):	Reserved
Bit 7 (D7):	WS Emulation active if set

COMMAND NUMBER	COMMAND NAME	COMMAND DESCRIPTION
30	Start a WS Emulation	This is used to link the partition with the MWS workstation. Upon receiving this command the 2258 code assigns a workstation task to the specified partition. If successful, MWS constructs an IOCW and data string to send the partition the log on screen. Emulation then starts.

```
 ______________________________________________
! 3          ' partition-1 ! 30         !
! high nibble   low nibble ! (1 byte)   !
 ______________________________________________
```

The 2258 will respond as follows:

if (error)

```
 ____________
! 00        !
! (1 byte)  !
 ____________
```

else

```
 ________________________________________________________________________
! non-zero ! message length ! IOCW       !order area !data . . .
! (1 byte) ! (2 bytes)      ! (3 bytes) !(4 bytes)  !(1920 bytes)
 ________________________________________________________________________

           ! tabs       !
           ! (10 bytes)!
 ______________________
```

where message length = IOCW + order area + data + tabs. Note that MWS will construct the IOCW, order area, data and tabs from screen buffer.

WORKSTATION EMULATION COMMANDS - Upper nibble = 3 (Cont.)

COMMAND NUMBER	COMMAND NAME	COMMAND DESCRIPTION
31	Receive IOCW/Data from VS	This command is used to signal ready for input from the VS.

```
 ____________________________________________________
! 3            ' partition-1 ! 31        !
! high nibble    low nibble ! (1 byte) !
 ____________________________________________________
```

The 2258 will respond as follows:

if (no data)

```
 ___________
! 00        !
! (1 byte) !
 ___________
```

else

```
 ______________________________________________________________________________
! non-zero ! message length ! IOCW      ! order area ! data . . !
! (1 byte) ! (2 bytes)      ! (3 bytes) ! (4 bytes)  !          !
 ______________________________________________________________________________
```

where message length = IOCW + order area + data

COMMAND NUMBER	COMMAND NAME	COMMAND DESCRIPTION
32	Status of Emulation	Upon receiving this command the 2258 code, checks to see if a workstation task exists for the specified partition.

```
 ____________________________________________________
! 3            ' partition-1 ! 32        !
! high nibble    low nibble ! (1 byte) !
 ____________________________________________________
```

The 2258 will respond:

```
does not exist:   ___________
                 ! 00        !
                 ! (1 byte) !
                  ___________

suspended:        ___________
                 ! 01        !
                 ! (1 byte) !
                  ___________

active:           ___________
                 ! ff        !
                 ! (1 byte) !
                  ___________
```

WORKSTATION EMULATION COMMANDS - Upper nibble = 3 (Cont.)

COMMAND NUMBER	COMMAND NAME	COMMAND DESCRIPTION
33	Suspend Emulation	This command signals the 2258 to check if a workstation task exists for the specified partition. If so, emulation is be suspended, otherwise an error is returned. Note: Screens will continue to be updated internally while emulation is suspended.

```
 ______________________________________________
! 3            ' partition-1 ! 33          !
! high nibble    low nibble  ! (1 byte)    !
 ______________________________________________
```

The 2258 will respond:

error:

```
 __________
! 00       !
! (1 byte) !
 __________
```

suspended:

```
 __________
! non-zero !
! (1 byte) !
 __________
```

WORKSTATION EMULATION COMMANDS - Upper nibble = 3 (Cont.)

COMMAND NUMBER	COMMAND NAME	COMMAND DESCRIPTION
34	Resume Emulation	Upon receiving this command the 2258 code checks to see if a workstation task exists for the specified partition. If so, MWS constructs an IOCW and data string to send the partition the current screen. Emulation then resumes.

```
 ________________________________________________
! 3             ' partition-1 ! 34        !
! high nibble     low nibble ! (1 byte) !
 ________________________________________________
```

The 2258 will respond as follows:

```
if (error)

        ! 00       !
        ! (1 byte) !

else

        ! non-zero ! message length ! IOCW      ! order area ! data . . .
        ! (1 byte) ! (2 bytes)      ! (3 bytes) ! (4 bytes)  !(1920 bytes)

                   ! tabs       !
                   ! (10 bytes)!
```

where message length = IOCW + order area + data + tabs. Note that MWS will construct the IOCW, order area, data and tabs from screen buffer.

WORKSTATION EMULATION COMMANDS - Upper nibble = 3 (Cont.)

COMMAND NUMBER	COMMAND NAME	COMMAND DESCRIPTION
35	Get MWS screen buffer	This command signals the 2258 to send the screen image maintained by MWS. It is not needed when resuming emulation as it is implied within that command.

```
 _______________________________________________
 ! 3            ' partition-1 ! 35          !
 ! high nibble    low nibble ! (1 byte)    !
 _______________________________________________
```

The 2258 will respond as follows:

if (error)

```
 ____________
 ! 00        !
 ! (1 byte)  !
 ____________
```

else

```
 ________________________________________________________________________________
 ! non-zero ! message length ! IOCW       ! order area ! data . . .
 ! (1 byte) ! (2 bytes)      ! (3 bytes)  ! (4 bytes)  !(1920 bytes)
 ________________________________________________________________________________

 _________________
       ! tabs       !
       ! (10 bytes)!
 _________________
```

where message length = IOCW + order area + data + tabs. Note that MWS will construct the IOCW, order area, data and tabs from screen buffer.

WORKSTATION EMULATION COMMANDS - Upper nibble = 3 (Cont.)

COMMAND NUMBER	COMMAND NAME	COMMAND DESCRIPTION
36	Terminate Emulation	Upon receiving this command the 2258 code, shall check to see if a workstation task exists for the specified partition. If so, emulation shall be terminated, otherwise an error is returned.

```
 ___________________________________________
! 3          ' partition-1 ! 36          !
! high nibble   low nibble ! (1 byte)    !
 -------------------------------------------
```

The 2258 will respond:

error:

```
 __________
! 00       !
! (1 byte) !
 ----------
```

terminated:

```
 __________
! non-zero !
! (1 byte) !
 ----------
```

FILE SERVER COMMANDS - Upper nibble = 5

The file server commands use the data address rather than the command address. The first three commands are used to maintain the DMS/VDisk link. They notify the 2258 when a DMS task is being started or ended and get general status when needed.

The next command, Send OMT Request, is used to communicate with the VS and is used by the DMS routines to communicate with OMT. The DMS routines will pack the OMT commands and pass them through the 2258 with a minimum of processing required.

COMMAND NUMBER	COMMAND NAME	COMMAND DESCRIPTION
50	Assign DMS task	Set this MWS task to be a DMS task. This will be used by the Attach utility. Prior to this, the user must have logged on using terminal emulation and entered OMT ACCESS. There can only be one DMS task per 2258 board. The suggested procedure for assigning a DMS task is as follows: (1) Enter WSE and logon. (2) Run OMT Access. (3) Assign DMS.

```
! 5           ' partition-1 ! 50       !
! high nibble   low nibble  ! (1 byte) !
```

The 2258 will respond as follows:

if (error)

```
! 00       !
! (1 byte) !
```

else (done)

```
! non-zero !
! (1 byte) !
```

51	Deassign DMS task	Complete any pending DMS task and do not accept any new ones. Clear VDisk tables. The 2258 will set a soft bit that indicates that DMS has been deassigned. Using the Query command (Op) will cause this bit to be reset. The suggested logoff procedure for DMS is as follows: (1) Send a $OPEN. (2) Set VDISK controller busy (63). (3) Deassign DMS (51). (4) Wait for ACK in status bit (soft bit). (5) Resume or restart emulation. (6) Exit OMT and logoff.

```
! 5          ' partition-1 ! 51        !
! high nibble   low nibble ! (1 byte) !
```

The 2258 will respond as follows:

if (error)

```
! 00       !
! (1 byte) !
```

else (done)

```
! non-zero !
! (1 byte) !
```

52 Get DMS status Get general status of DMS from 2258.

```
 ____________________________________________________
! 5            ' partition-1 ! 52          !
! high nibble    low nibble ! (1 byte) !
 ____________________________________________________
```

The 2258 will respond as follows:

if (error)

```
 ______________
! 00          !
! (1 byte) !
 ______________
```

else (done)

```
 ______________
! non-zero !
! (1 byte) !
 ______________
```

53	Send OMT request	Application wants to send a request to the VS. If the 2258 agrees (response 2), the application can then send request.

5	partition-1	53
high nibble	low nibble	(1 byte)

The 2258 will ack/nack as follows:

(1) There is no DMS task - can't help you.

00h
(1 byte)

(2) Go ahead and send request.

80h
(1 byte)

The 2200 application would then send the request in the following format:

OMT request	buffer count		other information required for OMT
1 byte	high byte	low byte	request (headers, data, etc.)

4000

The 2258 response to the 2200 would be as follows:

status	buffer count		response as a stream
1 byte	high byte	low byte	of data

Possible values for status include:

(1)	01h	- Not initialized
(2)	80h	- Done
(3)	83h	- Connection down
(4)	xxh	- Partial -- Continuation needed

If there is more than can be sent in one chunk, the VS will have sent a continuation code. The 2200 application must check this and let the 2258 know if it is expecting more to come and needs to keep the data channel open. This is expressed through the 2200's acknowledgement to the 2258:

if (done)

00
(1 byte)

else (more to go)

non-zero
(1 byte)

DISK COMMANDS - Upper nibble = 6

These commands add and remove VDisk entries from the 2258's tables and set the controller busy and unbusy. The latter can be used to prevent disk contention prior to changing VDisk and/or DMS status.

60	Add VDisk entry	Add VDisk entry to platter table. Information passed includes platter address, file handle, and the open mode, i.e., whether exclusive or shared. The 2258 stores this in the appropriate table(s).

```
 ___________________________________________
! 6            ' partition-1 ! 60          !
! high nibble    low nibble  ! (1 byte)    !
 -------------------------------------------
```

The 2258 will respond as follows:

(1) There is no DMS task.

```
 ____________
! 00        !
! (1 byte)  !
 ------------
```

else go ahead

```
 ____________
!non-zero   !
! (1 byte)  !
 ------------
```

The 2200 will send the data in the following format:

```
 _____________________________________________________________
! platter as a DXX address   ! file handle             ! ....
! (bottom 2 digits) (1 byte) ! high byte ! low byte!
 -------------------------------------------------------------
```

```
 __________________________________________________________________
! highest legal sector (size - 1) ! open mode *
! high byte ! low byte            ! (1 byte)
 ---------------------------------------------
   * open mode can be variations of R,RW, EXC, SH, etc.
```

The 2258 will respond:

(1) Illegal disk address.

```
 ____________
! 00h       !
! (1 byte)  !
 ------------
```

(2) This VDisk is already in table.

```
 ____________
! 01h       !
! (1 byte)  !
 ------------
```

(3) Successful.

```
 ____________
! 80h       !
! (1 byte)  !
 ------------
```

61 Delete VDisk entry — Delete VDisk entry from platter table(s). User should do a $OPEN first.

```
 _________________________________________
! 6            ' partition-1 ! 61        !
! high nibble    low nibble  ! (1 byte)  !
 -----------------------------------------
```

The 2258 will respond as follows:

(1) There is no DMS task.

```
 __________
! 00       !
! (1 byte) !
 ----------
```

(2) Go ahead.

```
 __________
!non-zero  !
! (1 byte) !
 ----------
```

The 2200 will send the data in the following format:

```
 _____________________________
! platter as a DXX address   !
! (bottom 2 digits) (1 byte)!
 -----------------------------
```

** File handle will be sent high order byte first, then low order byte.

The 2258 will respond:

(1) Illegal disk address.

```
 __________
! 00h      !
! (1 byte) !
 ----------
```

(2) Other failure.

```
 __________
! 01h      !
! (1 byte) !
 ----------
```

(3) Successfully deleted.

```
 __________
! 80h      !
! (1 byte) !
 ----------
```

62 Get VDISK status — Get general status of VDISK from 2258.

```
 _______________________________________________
! 6            ' partition-1 ! 62           !
! high nibble    low nibble ! (1 byte) !
 -----------------------------------------------
```

The 2258 will respond as follows:

if (error)

```
 ___________
! 00        !
! (1 byte) !
 -----------
```

else (done)

```
 ___________
! non-zero !
! (1 byte) !
 -----------
```

63 Set VDISK controller busy — Set controller busy so no access is possible. This is used to eliminate disk contention, especially prior to changing VDisk status on the 2258. User should do a $OPEN first.

```
 _______________________________________________
! 6            ' partition-1 ! 63           !
! high nibble    low nibble ! (1 byte) !
 -----------------------------------------------
```

The 2258 will respond as follows:

(1) Yes, immediately.
(2) Can't do it.

64 Set controller unbusy — Reset the controller so it is no longer busy.

```
 _______________________________________________
! 6            ' partition-1 ! 64           !
! high nibble    low nibble ! (1 byte) !
 -----------------------------------------------
```

The 2258 will respond as follows:

if (error)

```
 ___________
! 00        !
! (1 byte) !
 -----------
```

else (done)

```
 ___________
! non-zero !
! (1 byte) !
 -----------
```

2200/VS Local Communications Option (2258 OPTION)

PP& M: G. Schulz	Mfg Prog Mgr. Larry McGonagle
STATUS: Pilot	Date: 5/30/86
FCS:7/31/86	Total Revenue Impact @ ISV:
Product Ann. 7/01/86	Q1 Revenue impact: $252,000.
Vol Ship: 8/31/86	FY 87 Revenue impact: $426,000.

Program Description:

The purpose of this project is to allow the 2200 to communicate with and co-exist in a VS enviroment. The 2200/VS data link (2258) is one of the major elements of a total 2200 co-existence/bridge strategy designed to protect our user and ISO base by providing a co-existence and migration/evolution to the VS product line. Three functional services will be provided over the link. These are:VDISK - 2200 access to 2200 virtual disks residing on the VS,VS system and filing services accessible from the 2200.,VS terminal emulation on 2200 terminals. PRODUCT LIFE: est. 5yrs.

Assuming 500 boards at an average selling price of $1,200 and an average VS sale at $50,000, the business potential is $3,000,000 in FY 87.

Major Program Changes: Software will not be Ready on Schedle New FCS will be 7/31/86 Formerly 6/01/86.__________________ 05/30

Significant Accomplishments

5/30	1. 50 BDS complete 200 in peices part stock.
2/24	2. Pilot will be in Parkwood.
5/30	4. Spares/Repair Plan. (H. Wood)
5/15	5. Insystem Test Bed (M. Calvi).

Planned Activities:	Planned Date:	Date of Entry:
Documentation Completion	6/30	05/01
Forcast wil be Revised for loading.in MPP Rev 10.	6/15	05/01

Significant Issues/Concerns

Software Availability Need VS 7.10

ECO will ad switch at Rev 1. ECO in process.

Options:1. REV 0 Boards will be used until stock is diminished. ECO will phase in on new bare board purchase.
2. Scrap present stock (200 BDS).Initiate Build at Rev 1.

Status: ______________________________ Yellow

Misc.

Cost: ISV: $1200		MFG COST: $281 est.	GPM: 76%	
CEI #:	2258-3	200-1213-3	PEP #:	H0104A
	2258-5	200-1213-5	BOM #:	191-0664-00
	2258-9	200-1213-9	MEI #	289-0611

BUILD PLAN:

	Q1	Q2	Q3	Q4
BOOK FORECAST:	150	55	80	110
CE DEMAND	60	0	0	0
TOTAL	210	55	80	110

	Q1	Q2	Q3	Q4
PIL(PARK.)	210	0	0	0
PB	0	55	80	110
TOTAL	210	55	80	110

INSTALLATION:Customer Engineering will be responsible for installation of this Option .

2200/VS Local Communications Option (2258 OPTION)

PP& M: G. Schulz
STATUS: Pilot
FCS: 7/31/86
Product Ann. 7/01/86
Vol Ship: 8/31/86

Mfg Prog Mgr. Larry McGonagle
Date: 5/30/86
Total Revenue Impact @ ISV:
Q1 Revenue impact: $252,000.
FY 87 Revenue impact: $426,000.

Program Description:

The purpose of this project is to allow the 2200 to communicate with and co-exist in a VS enviroment. The 2200/VS data link (2258) is one of the major elements of a total 2200 co-existence/bridge strategy designed to protect our user and ISO base by providing a co-existence and migration/evolution to the VS product line. Three functional services will be provided over the link. These are:

VDISK - 2200 access to 2200 virtual disks residing on the VS.
VS system and filing services accessible from the 2200.
VS terminal emulation on 2200 terminals.
PRODUCT LIFE: est. 5yrs.

Assuming 500 boards at an average selling price of $1,200 and an average VS sale at $50,000, the business potential is $3,000,000 in FY 87.

Major Program Changes: Software will not be Ready on Schedle New FCS will be 7/31/86 Formerly 6/01/86. 05/30

Significant Accomplishments

5/30	1. 50 BDS complete 200 in peices part stock.
2/24	2. Pilot will be in Parkwood.
1/23	3. Technician has been trained.(J. Beauregard)
5/30	4. Spares/Repair Plan. (H. Wood)
5/15	5. Insystem Test Bed (M. Calvi).

Planned Activities:

	Planned Date:	Date of Entry:
Documentation Completion	6/30	05/01
Forcast wil be Revised for loading.in MPP Rev 10.	6/15	05/01

Significant Issues/Concerns -

Software Availability Need VS 7.10

ECO will ad switch at Rev 1. ECO in process.REV 0 Boards will be used until stock is diminished. ECO will phase in on new bare board purchase.

Status: Yellow

Misc.
Cost: ISV: $1200 MFG COST: $281 est. GPM: 76%
CEI #: 2258-3 200-1213-3 PEP #: H0104A
2258-5 200-1213-5 BOM #: 191-0664-00
2258-9 200-1213-9 MEI # 289-0611

BUILD PLAN:

	Q1	Q2	Q3	Q4
BOOK FORECAST:	150	55	80	110
CE DEMAND	60	0	0	0
TOTAL	210	55	80	110

	Q1	Q2	Q3	Q4
PIL(PARK.)	210	0	0	0
PB	0	55	80	110
TOTAL	210	55	80	110

INSTALLATION: Customer Engineering will be responsible for installation of this Option .

2200/VS Local Communications Option (2258 OPTION)

PP& M: G. Schulz
STATUS: Pilot
FCS:6/01/86
Product Ann. 5/15/86
Vol Ship:8/01/86

Mfg Prog Mgr. Larry McGonagle

Total Revenue Impact @ ISV:
Q4 Revenue impact: $252,000.
FY 87 Revenue impact: $426,000.

Program Description:
The purpose of this project is to allow the 2200 to communicate with and co-exist in a VS enviroment. The 2200/VS data link (2258) is one of the major elements of a total 2200 co-existence/bridge strategy designed to protect our user and ISO base by providing a co-existence and migration/evolution to the VS product line. Three functional services will be provided over the link. These are:

VDISK - 2200 access to 2200 virtual disks residing on the VS.
VS system and filing services accessible from the 2200.
VS terminal emulation on 2200 terminals.
PRODUCT LIFE: est. 5yrs.

Assuming 500 boards at an average sel ing price of $1,200 and an average VS sale at $50,000, the business po tial is $3,000,000 in FY 87.

Planned Date : 04/11

Major Program Changes:
- None

Significant Accomplishments

2/28	1.	CEI, MEI NO,s have en Assigned
2/24	2.	Pilot will be in F wood.
2/27	3.	Forcast has been p ared for loading.in MPP Rev 7.
2/18	4.	All parts have been dered. (J. McGovern)
1/23	5	Technician has been ained.(J. Beauregard)
1/23	6.	Test & Repair Plan E. Daigneault).

Planned Activities:

Spares/Repair Plan (H. Wood)	04/30
Documentation Completion	04/30
Material Availability Review (J. Mc overn)	04/12

Significant Issues/Concerns -

Software Availability Need VS 6.4 7.10	04/30
ECO will ad switch at Rev 1. ECO in process.	04/30

Status: Green

Misc.
Cost: ISV: $1200(est) MFG COST: $281 est. GPM: 76%

CEI #:	2258-3	200-1213-3PEP #:	H0104A
	2258-5	200-1213-5BOM #:	191-0664-00
	2258-9	200-1213-9MEI #	289-0611

BUILD PLAN:

	Q4	Q1	Q2	Q3		Q4	Q1	Q2	Q3
BOOK FORECAST:	150	55	80	110	PIL(PARK.)	210	0	0	0
CE DEMAND	60	0	0	0	TEWK	0	55	80	110
TOTAL	210	55	80	110	TOTAL	210	55	80	110

INSTALLATION:Customer Engineering will be responsible for installation of this Option .

PRICING AND ORDERING INFORMATION

ORDER NUMBER	DESCRIPTION	PRICE
195-5210-9	GENIE	$99.95

Customers may place orders through WangDirect or by contacting their local Wang sales representative.

WORLDWIDE AVAILABILITY

ORDER NUMBER	FCS	VOLUME SHIP
195-5210-9	NOW	NOW

American English only. Foreign language versions of GENIE are not available.

ADDITIONAL INFORMATION
This article was prepared by Software Marketing and WangDirect.

2200 SYSTEMS: 2200/VS LOCAL COMMUNICATIONS OPTION

Wang Laboratories, emphasizing our continuing commitment to the 2200 and MicroVP product line, to our 2200 Value Added Reseller (VARs) and to our users, announces the ability to data-link a 2200 to a VS with the new 2258 Controller. Also known as the 2200/VS LCO (Local Communications Option), the 2258 provides a high-speed 928 data-link to the VS for our 2200 VARs (Value Added Resellers) and users.

The 2200/VS LCO provides the following benefits:

. VDISK - The ability to store and retrieve 2200 data in 2200 format on the VS's disk storage system.

. DMS - The ability to store and retrieve 2200 data on the VS's disk storage system in VS native file format, thereby giving VS programs the ability to act upon the 2200 data.

. VSTE (VS Terminal Emulation) - The ability for a 2200 workstation user to log on to the VS and execute any VS task that doesn't require the downloading of microcode to a VS workstation.

HIGHLIGHTS
Wang Laboratories has sold, since its release, more than 67,000 2200 systems. According to the results of a recent internal Wang survey of the U.S. 2200 user base, 10% of our users intend to add another CPU (type not specified) and keep their 2200, and 10% intend to migrate to a mini like the VS. With 50% of the 2200s installed in the U.S. and the other 50% installed internationally, the study projects that 13,400 worldwide 2200 users are potential VS prospects.

The 2200/VS LCO was designed specifically to meet the needs of two types of 2200 users:

- Those users wishing to keep their 2200, but link to a VS or VSs as a method of expansion and/or systems enhancement.

- Those users wishing, either short-term or long-term, to update their system and migrate to the VS or wanting to have that option available to them if the need arises.

Therefore, the benefits and objectives of the 2258 are:

- To provide an expansion path for current 2200 users who want to keep their 2200 but are either at maximum configurations, want to off-load certain tasks better done on a VS type product, add new applications, or want to evolve to new technology.

- In the absence of a 2200 to VS software bridge, to provide a migration path for both the 2200 user or 2200 VAR to evolve to the VS and be able to utilize their existing data files, perform DP application development on the VS, and provide access to these new VS applications by the 2200 workstation user as they are developed on the VS.

- To protect our user base and channels of distribution with a strategy that provides an optional present and future migration path to the VS, thereby providing the reasons to stay with Wang and integrate and/or migrate their code to a Wang VS, not to a competitive system.

PRODUCT DESCRIPTION

The 2200/VS LCO is a data communications hardware and software option that enables a Wang 2200MVP, LVP, or MicroVP system using Release 2.7 of the Multi-user Operating System (OS) to communicate with a Wang VS computer system. Communications between the 2200 system and the VS system occurs at speeds of 4.27 megabits per second over dual coaxial cable facilities. The VS, with an available serial port for the coaxial cable connection, may be located up to 2,000 feet (609.6m) away (25 feet is standard) from the 2200.

The 2258 Communications Controller is an internal microprocessor-based controller mounted in an empty slot in the 2200 chassis. Each controller supports four concurrent sessions between the 2200 and the VS. You can either have four terminal emulation sessions (VSTE) or three VSTE and one disk session (VDISK and/or DMS) per controller. Multiple 2258 Controllers will allow combinations as 16 sessions logged on to a single VS, or one 2200 linked to four VSs, or one 2200 with eight sessions logged on to two VSs, etc.

When the 2258 Controller is installed in the 2200 chassis, two addresses are set: a 2200 disk address for VDISK requests and a communications address for workstation emulation and VS filing requests. When the 2200 is powered on, the controller receives microcode from the VS and initializes itself.

A 2200 user will now be able to perform the following:

- Under VDISK, run 2200 application programs on the 2200 that store and retrieve 2200 disk-image files in 256-byte records on the VS's filing system. These files are created by 2200 utilities included with the LCO package.

- Using the DMS services of the VS, run 2200 application programs that store and retrieve 2200 data from the VS in native VS files, using subroutines supplied with the LCO package.

- Using VSTE, log on to the VS and run VS application programs that do not require the downloading of microcode to a VS workstation. Some examples are DP applications, Wang OFFICE, PACE, and VS Batch Communications.

VDISK has the following features:

- Supports one VS filing services session per 2258 Controller (all 2200 applications have access to the VS disk through this single link).

- Supports up to 32 VDISKs for each 2258 Controller.

- Supports a VDISK maximum platter size of 65,536 sectors (a 16MB 2200 platter).

- Maintains a table containing each open VDISK

- Requires no change to the 2200 BASIC-2 program other than a device address (if different from the device address of the 2258).

- Provides from 2200 workstation number one, the VDISK utilities to create, edit, view, or delete VDISKs in the configuration file. Any other workstation can view the VDISK status.

- Supports the sharing of VDISK files with similarly equipped 2200 systems connected to the VS.

- Enables a 2200 user to copy a VDISK file to a 2200 disk.

DMS or the 2200/LCO native VS filing services has the following features:

- Supports one VS filing services session per 2258 Controller (all 2200 applications have access to the VS disk through this single link).

. Requires the addition of file access subroutines to the BASIC-2 application (provided with the package).

. Allows access to any currently defined native VS DMS file type. For example, consecutive, relative, and indexed files are supported.

. Supports the sharing of data converted from 2200 to native VS DMS files with other VS applications and systems with similarly equipped 2200 systems connected to a VS.

VS Terminal Emulation (VSTE) has the following features:

. Available from any workstation on the 2200.

. Supports up to four workstation emulation sessions concurrently per 2258 Controller.

. Emulation of a VS 2256C workstation (DP only).

<u>NOTE:</u> The statement that you can only have one VDISK or DMS disk session per 2258 means you can only log on to the VS for disk tasks once per 2258 Controller. This should be done by a workstation on the 2200 designated as Workstation 1. However, all resident applications have simultaneous access to this link and are able to store and access their files on the VS in the normal 2200 time-slice mode. In addition, some applications can be using VDISK and others DMS, all at the same time.

CONFIGURATION REQUIREMENTS

2200 System Requirements

. An LVP, MVP, or MicroVP with Release 2.7 or greater of the Multi-user Operating System (OS) and 28KB Control Memory. OS releases below 2.7 will allow VSTE only.

. A 28KB partition to run the VDISK utilities and VSTE.

. 5KB to 7KB additional needed per existing program for file access routines when using DMS.

. Any 2236/26, 2336/26, or 2436/26 DE/DW series of workstations to run VSTE.

VS System Requirements

. Operating System 7.00 series that supports VS/2200 SRV. (Server that maps 2200 disk requests into VS DMS requests.) OS releases below 7.0 will allow VSTE only.

. An unused serial port for a 2258MWS (Multi-Workstation) device using the VS GENEDIT program.

. The designated 2200 Workstation 1 operator start up the VS ACCESS Task software for processing the 2200 file requests (both VDISK and native VS).

2200 To VS Connection

. Operating Mode - Asynchronous, polled
. Data Format - Serial
. Data Rate - 4.27MPS
. Transmission Medium - Dual-coaxial baseband cable

PERFORMANCE

VDISK

The VS Serial I/O Processor (SIOP) is designed to handle one request at a time from the 2258 Controller. The Serial IOP performance has been determined to be approximately 6 I/Os per second, independent of the size of the request. VDISK creates a 2200 disk image within a user-named VS file. A single 2258 may control up to 32 of these disk images at any one time. The user's application software then addresses the disk image just as if it were a real disk platter on the 2200 system.

However, when a user makes use of the 2200 cataloging capability, the user is actually using a three-tier system Catalog index and a File Storage Area. In order to open a file or load a file, the 2200 Operating System will first read Sector 0 of the disk image to determine the size (in sectors) of the catalog index. A hashing algorithm is used to determine where the file information may be stored within the catalog index. If the information about the file is not located within that sector, the index is read and searched one sector at a time in a circular fashion until it is found. Thus, in order to locate the start of a file, two or more sectors must be read from the disk image. Since the number of requests going to the VS is equal to the number of sectors needed to complete the disk request, added to the number of partitions attempting to use the disk link, users should carefully calculate their use of VDISKs.

Another operation that may become time consuming is the $OPEN statement. Whenever the user attempts to "hog" a disk, the 2200 will not allow any other user access to the entire disk unit. This means that all VDISK images must be "locked," one file at a time. If all 32 possible disk images were defined, response time would be excessive.

The 2258 support utilities allow the user to define multiple VDISK maps, which assign a user-determined number of 2200 platter addresses to an equivalent number of 2200 disk images. This facility will allow the user to pick the VDISK map for the particular application the system is currently executing, thereby minimizing the 2258 overhead and improving performance.

In larger systems or ones where heavy use of the 2258 data-link and VDISK is expected, multiple 2258 boards (three maximum) will increase the possible throughput.

DMS
The disk task service is available to all 16 partitions at the same time. If all partitions attempt to access the VS disk simultaneously, the response time could be excessive.

Hence, if heavy use is anticipated, multiple 2258s should be considered (limited only by the number of I/O slots available).

PRODUCT STRATEGY

Target Markets
Research has shown that 10% to 20% of our 2200 user base are prospects for a VS and a large number of our 2200 software vendors and VARs want either a method of migration to or integration with the VS. Because of the architectural differences between the 2200 and the VS, a software bridge or BASIC-2 emulator on the VS is not practical at this time. We have found that an emulator on the VS can be built, but the current projected performance of the applications running on the VS is less than satisfactory. With this in mind, the 2200/VS LCO has been primarily designed for current 2200 users and our 2200 resellers or VARs as both a method of providing the integration of the 2200 with the VS and as an alternate method of migration. To understand how and which type of user or VAR could use the data-link, you must first understand the profile of our users and VARs.

Users
Our users can generally be classified as falling under one of four categories:

. A small- to medium-size business happy with the 2200 (multi-user) and will never outgrow the capabilities of the product.

. A single workstation (as the T or VP) user who wants to or should update their system and can either migrate upstream to a multi-user or multi-user capable system or downstream to another single-user type system.

. A small- to medium-size business outgrowing the capabilities of the 2200 but would rather add on to, not replace their 2200.

. The user who wants to migrate to a mini, but wants to maintain their current software.

VARs
VARs and software vendors account for 90% of the 2200s sold. ISOs (Independent Sales Organizations, referring to both software vendors and VARs) generally fall under one of three categories:

. Will continue to sell or provide software for the 2200 as long as we continue to make, enhance and support it.

. Will continue to sell and provide software for the 2200 but want to know the life cycle of the product, what is the evolution/migration path to other Wang products, and want to begin to integrate with and sell new technology. The need for an evolution/migration path can either be based on wanting to leave the 2200 or in addition to selling the 2200, going up-market to the VS or down-market to the PC in order to increase their market coverage.

. Want to leave the 2200 immediately for various reasons. This ISO wants a software bridge to another Wang product.

Hence, the 2200/VS LCO is a major part of the overall 2200 product strategy that has been formulated to address the needs of each of these defined user and VAR categories. This article deals with the integration with or migration to the VS.

External Product Positioning
Two of the most frequent questions that will be asked are "When do I need a software bridge?" and "When do I need the data-link?" By definition, a software bridge means you can move 2200 application software over to the VS and it will run as is. The data-link is a hardware bridge that allows for the sharing of data and resources between the 2200 and the VS. It is not a software bridge and should not be communicated as such.

The user or VAR who has perceived that they only want a software bridge, should be asked the following questions:

. Do you want to move your applications to the VS "as is" or is it time to re-systematize and update your software?

. Do you want an emulation or would you rather take advantage of the software features of the VS system?

. Do you want to move to the new system immediately with all applications or would you prefer a gradual transition over a period of time?

. What are your real objectives? To increase the capabilities and efficiency of your system or just to replace your 2200 system?

. Have you explored all the options of expansion, e.g., multiplexing additional CPUs, PCs/APCs as workstations and/or freestanding BASIC-2 CPUs to off-load single-user/single task applications, the 2200/VS LCO, etc.?

If the user or VAR would like to update their software, optimize their code on a VS, gradually migrate their applications or assign tasks to specific systems that do them best, chances are this user or VAR is a better prospect for the data-link than a software bridge. If a software migration to the VS is still more desirable, the data-link is an excellent migration assistance tool and will provide an evolution/migration path to the VS.

The following are two examples:

- A current 512KB MVP user with 10 workstations and a 2280 80MB Phoenix disk drive. The user will need to add a second 2280 to add new applications that will require a database management system (DBMS) and two more workstations. As the VS is a far better database manager, this user can data-link to a VS 5 for not much more than the cost of a second 2280 and using the DMS services of the VS, store their 2200 files on the VS in VS format. Through VSTE, current 2200 workstation users can log on to the VS and access the new VS applications that can utilize the VS resident 2200 data files.

- A user who wants to phase out of their 2200 and move their applications over to a VS. When the VS and the 2258 are installed, using DMS, they will move their files over to the VS in VS format and begin new application development using PACE. As the new applications are developed, VSTE allows the current 2200 workstation users to log on to the VS and execute those applications now resident on the VS.

In the first example, a user is integrating with the VS to perform tasks that are better done on the VS, while leaving tasks better done on a 2200, on the 2200. In the second example, in the absence of a software bridge, the data-link provides a migration path to the VS and gives the user the reason why their code should be ported to a Wang VS, not to a competitive system.

Over the last several years, our VARs have almost become the sole method of distribution for the 2200. As a result, unless a user is being serviced by a VAR, a perception has been developed by the user that Wang is not interested in nor is doing any new development for the 2200. In addition, those users that have made inquiries or have investigated the feasibility of migrating to the VS, have been told that their only option is to rewrite their applications on a VS. To clear up these misconceptions, an efficient method of information dissemination must be devised. Therefore, the 2200 Installed Base Marketing Program has been created. The basis of this program will be a series of 2200 user seminars designed to educate our 2200 user base on the status of the 2200 and their migration/evolution options.

COMPETITION

As the perception exists that no method of migrating to the VS exists, competition has used the approach that "If you have to do a major rewrite, why not rewrite to an IBM system? Will the same thing happen again if you need to update your VS?" This condition appears to be prevalent in the major accounts where we have lost VS business to IBM System/36s. Hence, it is very important that these accounts be identified and presentations and demonstrations be made to update them on their options. No other computer manufacturer offers a hardware link to their product from the 2200 like the 2200/VS LCO. The benefit of a phase-in/phase-out approach is that it allows an orderly transition from one system to another.

WANGCARE SERVICES

Software Services

Current 2200 WSS support policies and services will apply. Refer to the Support Services section of the current U.S. Pricing Manual, Book I, for WSS services offered for the 2200 product line.

Documentation

The 2200/LCO User's Reference Guide (715-0564) has been developed for the 2258 Controller and will be enclosed when the 2258 is shipped. The guide contains a complete overview of the capabilities of the 2258 Controller and complete systems requirements.

The 2200/LCO Programmers Reference Guide (715-0562) will provide the programmer for the 2200 and the VS systems administrator all the necessary information on both the 2200 and VS systems utilities and programming information necessary for a successful data-link. The Programmer's Reference Guide will be auto-enclosed when the 2258 is shipped.

Sales Aids

A 2200/LCO Data Sheet (715-0563) will be available from WangDirect at the end of Q3 CY'86.

HARDWARE SERVICES

Customer Warranty

The 2258 Controller is warranted to be free from defects in materials or workmanship for a period of 90 days from date of installation. Warranty is in accord with terms and conditions in effect as of the time of sale.

On-Site Maintenance Agreement

On-Site (Plan A), Wang's standard on-site maintenance agreement, provides for 12 months of on-site service.

Per-Call On-Site Service

Per-Call On-Site service is available on a time-and-material basis. Customers who wish to use this service should call the nearest Area Call Control Center toll-free number to arrange for a service appointment.

ORDERING INFORMATION

Refer to the Pricing Update article in this issue for U.S. list prices of the following products:

MODEL NUMBER	DESCRIPTION
2258-3	2258 Controller with 2200 software on 8" SSSD Diskettes, VS software on both 8" and 5 1/4" Diskettes
2258-5	2258 Controller with 2200 software on 8" DSDD Diskettes, VS software on both 8" and 5 1/4" Diskettes
2258-9	2258 Controller with 2200 software on 5 1/4" DSDD Diskettes, VS software on both 8" and 5 1/4" Diskettes

All orders must specify the media type using the appropriate model number suffix. Media type codes are defined on the inside back cover of FOCUS.

WORLDWIDE AVAILABILITY

MODEL NUMBER	FCS	VOLUME SHIP	REQUIREMENTS
2258-X	9/86	10/86	VS OS 7.00 2200 OS 2.7

QUESTIONS AND ANSWERS

Q1. Why have we developed a hardware link first and not a software bridge?

A1. For two reasons. The first is to meet the specific needs of our users. There are those users who want to expand their system by adding a VS, but keep their 2200, and there are those users who want to migrate off the 2200 and go to a mini. The second is we want to do now what we can do best. A 2200 to VS software bridge has been found to be, at this time, impractical. According to our research, it is possible to automatically port over 75% to 90% of the 2200 code to the VS. However, this code, running in emulation mode, will generally run 4 to 15 times slower on a VS than it ran on the 2200 due to the architectural differences of the two products. This is unacceptable performance to both Wang and the user. Performance for the data-link is very acceptable and also provides a method of migrating to the VS.

Q2. <u>I have been told that a software bridge from the 2200 to the VS is the only way to go. What is your response</u>?

A2. A software bridge and the 2200/VS LCO address different needs. If a user or reseller wants to move their software over to the VS "as is," then a software bridge is desirable. If a bridge were technically feasible, either Wang or a third party would have one available. The data-link serves two distinct purposes. First, in the absence of a software bridge, the data-link provides a method (not available before) of migration/evolution to the VS and the reason why a user, software vendor or VAR, should port their code to a Wang VS, not to another manufacturer. Also, not every user will want to port their code over "as is." Under those ground rules, the data-link is a better answer. Second, our recent study indicates that 90% of current 2200 users intend to keep on using their 2200s for one to four years. Therefore, the 2200/VS LCO addresses the larger segment of our user base.

Q3. <u>How does the data-link provide a migration path for 2200 users wishing to migrate to the VS</u>?

A3. The data-link provides two services that will facilitate the conversion process. The first is the ability to move the 2200 files to the VS in native VS file mode. The second is VSTE, which will allow the current 2200 workstation user to access the new VS applications as they are developed while still using the applications on the 2200 that have not been moved over to the VS. No other mini manufacturer offers these services. Hence, we have given the 2200 users logical reasons, even though they have to rewrite their code, why they should migrate to a Wang VS.

Q4. <u>Why would a user want to keep their 2200 and link to a VS</u>?

A4. For three reasons. The first is product satisfaction. Most users are very pleased with their 2200 and would prefer to expand their system, not replace it. Second, they do not want to incur the time and expense to move to another system. Third, certain applications run better on the 2200 than the VS and vice versa, and an integrated approach, depending on the application, can yield more efficient results.

Q5. <u>I thought Wang wanted to upgrade our 2200 users to other Wang products as we are discontinuing the 2200</u>?

A5. Wang is not only not discontinuing the 2200, but is in the process of enhancing and modernizing the product in the form of the MicroVP. It is the goal of Wang to provide our users the path they want to take. Hence, if keeping their 2200 and adding a VS is a better answer for a user or enhancing and/or updating their 2200 with more 2200 products is more desirable, they should have those choices.

Q6. Isn't adding a VS an expensive proposition?

A6. With the announcement of the VS 5, a 2200 user can purchase a single workstation VS 5 system, with 1MB main memory, 67MB disk and tape cartridge back-up for as low as $19,250. The current cost of a 2200 single 2280 80MB Phoenix disk drive is $16,500 including DPU. If the current user has a storage problem and is about to purchase a 2280, by using VDISK and the storage facilities of the VS, a user can get a complete small VS system for less than $3,000 more than the cost of a 2280.

Q7. Will a 2200 workstation user under VSTE be able to log on to the VS and execute VS WP?

A7. No. VSTE only supports those VS tasks that don't require the downloading of microcode to a VS workstation and the emulation of a VS 2256C workstation.

Q8. What does a user have to do to their program to use VDISK? DMS?

A8. Under VDISK, you only have to change the device address to conform to the device address of the 2258 board. DMS requires the addition of File Access Subroutines to the BASIC-2 application program. However, the 2200 program now has access to any currently defined native VS file type. For example, consecutive, relative, and indexed files are supported.

Q9. Isn't just having one VDISK or DMS session per 2258 Controller restrictive?

A9. The statement that you can only have one session is misleading. It does not mean only one workstation has access to VS disks or DMS. It means you can only log on to the VS for disk tasks once per 2258 Controller. Hence, you could have every partition doing a different application but storing and accessing their files on the VS. In addition, it can be a mix of both VDISK and DMS per application, all at the same time.

Q10. I haven't been trained on the 2200. How or where can I get help in calling on the 2200 users in my territory?

A10. Every district has 2200 VARs either located within or servicing their district who know and understand the 2200. Plans should be made to utilize these resources by coordinating through your district's DAM (district applications manager).

ADDITIONAL INFORMATION

This article was prepared by 2200 Product Planning and Management. For additional information, contact SMART.