LBL-29351

Computerized CAMAC and NIM Module Library *

G. F. Pope [a] and R.J. McDonald [b]

Accelerator and Fusion Research and Nuclear Science Divisions

Lawrence Berkeley Laboratory
University of California
Berkeley, CA 94720

LBL--29351

DE91 004129

Abstract

The Lawrence Berkeley Laboratory owns a large number of CAMAC and NIM modules which can be connected together to form data acquisition systems used in experiments. Many of these modules are contained in "pools" for common usage. This paper describes a system of storage and inventory control that allows easy check-out and check-in of the modules utilizing networked Macintosh computers, FoxBase+/Mac software, and bar-code technology. It also provides search capability for the user and tracking capability for the pool administrator. This inventory system has applications to any pool of items that are routinely loaned.

a) Present address: Department of Physics, University of California, Davis, CA 95616
b) Present address: MS 88 Lawrence Berkeley Laboratory, #1 Cyclotron Rd.,
Berkeley CA 94720 Bitnet: 88USERS@LBL.gov

* This work was supported by the Director, Office of Energy Research, Office of High Energy and Nuclear Physics, Division of Nuclear Physics, U.S. Department of Energy under contract # DE-AC03-76SF00098

1/2

1) Introduction:

There is a large pool of electronic modules available for the use of experimenters and staff at the Lawrence Berkeley Laboratory (LBL) Bevalac and 88-Inch Cyclotron accelerators, with approximately 1300 units in the Bevalac pool and 400 units in the 88 pool. It is necessary to keep close track of these items as they are borrowed and returned in order to insure their effective use and to be able to determine that sufficient numbers are available for upcoming experiments. In the past, this was accomplished through the use of a log book, wherein borrowers recorded their name, location, and phone number, along with the descriptions of the modules they borrowed and the date they expected to return them. This allowed the pool administrator some degree of knowledge about which modules were out and whom to contact if a particular module were needed. This method also required the user to spend a considerable amount of time recording module identification descriptions, as well as multiple entries of his or her name, address, etc. Considering the large numbers of modules required in some experiments, this was an onerous task. For many users, it was considered so onerous and unpleasant that they left incomplete records. With this system, searching for late modules or particular types of modules was difficult even with a well kept log book. In addition, the lack of suitable storage for the modules in the pool generated additional problems.

A better way to manage these pools lies in utilizing bar-code technology and a computer to perform the inventory and tracking functions for the pool. Under this scheme, all the module identifiers, property accounting (DOE) numbers, manufacturer's names, model numbers, etc., reside in a computer-resident database. User information, such as the borrower's name, new location, phone number, and expected return date, is entered at check-out time, and can be entered only once even when checking out a large number of items. This vastly improves the accuracy and completeness of the information obtained and is simple and fun to use. The system was implemented on a Macintosh computer to make use of the extensive Macintosh networks at LBL. Thus, the pools are accessible, at least for search functions, from any Mac on the network. An optical wand was purchased for each pool and **FoxBase+/Mac** was selected as the database software. Programs to handle check-out and check-in of modules, search, and other functions were written and the databases were constructed, electronically when possible, from old inventory lists. Figure 1 is a picture of one of the pool Macintoshes.

In addition, suitable floor space was set aside for each pool and a customized system of storage racks was fabricated to hold the major types of pool-resident items. Although less glamorous than the computerized check-out system, sufficient space and appropriate storage facilities are essential foı a well functioning pool.

2) Implementation:

A) Storage:

The first step in organizing the pools was to develop an efficient means of storage for the equipment. Three types of custom wooden storage racks were designed: CAMAC module storage, NIM module storage, and NIM/CAMAC bin storage. Wood was chosen as the material since it is relatively inexpensive and easy to work. The CAMAC module storage racks will also accept NIM modules, and would suffice for a limited pool of mixed NIM/CAMAC modules. As shown in Figure 2, these CAMAC module racks fit conveniently into 18 inch deep closed storage cabinets where they can be locked and protected from dust, etc. The NIM module racks are not as deep and can be fitted conveniently into standard book cases (see Figure 3). The larger bin and crate storage racks (Figure 4) are very convenient storage places for a wide variety of power supplies as well as NIM bins and CAMAC crates. The construction of these pieces is described in Appendix B. In addition, space in the pool is allocated for oscilloscopes, cables, connectors, and other miscellaneous items.

B) Inventory System:

i) Hardware Environment:

Most pool items are expensive enough or sensitive enough to be entered into the LBL property accounting system. Thus, they are uniquely identified by a DOE inventory number which is attached as a bar-code label. Since the bar-code labels attached by property accounting are small and hard to read, even with expensive scanners, the DOE numbers were duplicated onto larger bar-code labels. These labels were generated with PrintBar™ bar-code fonts and attached to the item. Items without DOE numbers were given numbers different enough from the DOE number series that there will be no overlap.

Bar-code reading equipment was purchased for each pool. This consists of a light-generating and light-sensing wand that the user passes over the bar code, a decoder to translate the bar code to ASCII characters, and an interface to transmit these characters to the computer. The decoder and interface are built into what is called a wedge, because it is inserted or "wedged" in between the keyboard and the computer proper. The wedge passes signals from the keyboard to the computer, so the keyboard remains operational, and uses the same lines to send ASCII characters decoded from the bar code. Thus the computer cannot differentiate between keyboard and wand input, so no software modification is required to use the bar-code reader. Also, in the case of a failure to read the bar code, information can be typed in. Please see Appendix A for a list of products and their manufacturers.

ii) Software Environment:

The software is based on the **FoxBase+/Mac** Multi-User database program. This program was selected for two reasons: First, it is compatible with the dBase III+ database program for the IBM PC, on which some of the software was originally developed; and second, the multi-user version allows several users to access the database simultaneously. At present, there are two physically separate pools, Bev and 88, corresponding to the two accelerators where the pools are sited, and each pool has its own set of data bases. Combining these pools into a single database could cause problems. First, it would be possible to check out a module belonging to one pool from the other pool, something that could lead to abuses, and second, it would make the search and check out/in times longer. In our implementation, **FoxBase+/Mac** resides on a file server which is a Macintosh with a large hard disk.

The data base files have two-part names, consisting of a prefix "bev" or "88" referring to the pool, and a name that indicates function. (In this paper, files will be referred to without the prefix, indicating function only.) The file server is on a local area network (LAN) connected via a Kinetics Systems FastPath 4 box to the lab Ethernet (see Figure 5). Similarly, the Bevalac and 88-Inch Cyclotron LANs are connected to the same Ethernet. Both the BevPool Mac and the 88Pool Mac each have a special bar code symbol attached to them containing a code number that the program requests when it is run. This ensures that only the pool computers have full access to all functions, while other computers are only able to search the database and list borrowed and late modules.

iii) Programs:

The program **poolmenu** is the main program. It contains the main menu and several procedures that perform the different menu functions. Figure 6 is a chart showing the program and its various functions. Note that the functions fall into two overlapping classes: One set of functions for the user, and another for the pool administrator. The procedures **outgoing** and **incoming** are the primary bookkeeping procedures. The procedure **outgoing** asks the user to input personal information before checking out equipment. This must include his or her name, the new location of the module, phone number, and expected date of return For each module checked out, the computer records the transaction and echoes it back for the user to verify. The **incoming** procedure is simpler and merely accepts the DOE numbers of the returned modules. The **search** procedure allows a user to search for all modules of a specific make and model number, a specific model number alone, a specific DOE number, or a particular user name. Note that **search** can be performed from any Macintosh on any LAN connected to the LBL Ethernet, and both pools can be searched. The procedure called **extend** may be used when a user underestimates the time needed to complete an experiment and return the modules. After the user is contacted as a reminder, **extend** can be used to give the user an extra amount of time to return the modules. It is a good compromise between strict enforcement of the due dates (which are originally supplied by the user) and total apathy. The procedure **transfer** is used to transfer modules from the responsibility of one user to another. It is nearly identical to **outgoing**, the main difference being that the modules are not physically at the pool, so a handwritten list of all the DOE numbers must be taken to the computer. This procedure was included so that users did not have to move a large number of modules into the pool to check them in and back out, but its ability is limited by how many numbers the user wants to accurately copy down on paper. A portable bar-code scanner with memory would make this procedure more useful.

The procedures **borrowed** and **late** are mainly used by the pool administrator. **Borrowed** outputs all information about modules currently on loan, giving some indication of the amount of activity at the pool and who is responsible for it. **Late** generates an overdue list for posting. **Search**, **borrowed**, and **late** each give the user an option to count the modules before listing them, and they can be listed on the screen and/or printed. Finally, the **notify** procedure is an administrative routine for making personally addressed notices to users with overdue modules.

The auxiliary program **codemake** prints bar codes on a laser printer in a format such that they can be photocopied onto a sheet of labels. It reads the module numbers from the database **barcode**, which is created by the administrator. The program **invent** is used to take a complete inventory and compare it with the computer's database. This program generates a file called **report** that lists all discrepancies, such as modules that are listed as checked in but are really missing.

The coding for procedure **outgoing** is listed in Appendix E to give one a feeling for the Foxbase code. Hard copy of all the programs and/or listings on magnetic media may be obtained by writing one of the authors (RJM).

iv) Database Files:

There are six files associated with the various tasks of the system as shown in Figure 7. The **userdb** file is the main data base. It contains all the information about every item in the pool. The first five fields in the database describe the module and are unchanged once entered. They are called DOE, MFG, MOD, DESCRIPT, and OWN. DOE contains the property accounting number, MFG is a three-letter code for the manufacturer, MOD is the model number, DESCRIPT is a description of the item, and OWN is a three-letter code for the owner. The last four fields, NAME, LOCATION, PHONE, and DATE_IN, contain information on the latest user and change as modules are checked in and out. **Userdb** has an index file (**.idx** extension) by the same name. Indexing allows rapid search by DOE number, which is of primary concern in this application. The **search** function also has an index file (search.idx) associated with it. This speeds up the search procedure significantly.

Update is the second most important file. Its records are a subset of those in **userdb**, belonging to those modules which are currently out. Although it varies in size, **update** is usually quite a bit smaller than **userdb**, which makes it much easier and faster to manipulate. If something disastrous should happen to **update**, one may be able to recreate it using information in **userdb**.

Outlog is identical to **update**, except that it is cumulative and thus can be used to determine pool activity. **Outlog** is used by the administrator to see which modules have the most activity, indicating that perhaps more should be bought, and which modules have the least activity, indicating that perhaps some should be discarded. **Outlog** is reset utilizing the procedure **notify**. The files **temp** and **temp2** are scratch databases used by the **notify** procedure. They are empty except when the procedure is running.

v) User interaction:

To check out modules, the user selects the needed items, proceeds to the computer, and activates the program. Figure 8a shows a picture of the main menu from the program.

He or she then selects the check-out option by clicking the corresponding "radio button." Figure 8b shows the resulting screen. After filling in the requested information, the user scans the bar codes on the modules selected and exits the program, returning to the main menu. To check in modules, the user selects the check-in option, scans the bar codes, and returns to the main menu, leaving the modules to be shelved or stored by the administrator. It is important that only the administrator or his representative do all the re-shelving or storing of modules in order to maintain order in the pool.

If the user can not find a particular module, he or she may search for it using the **search** option in the main menu. This brings up a sub menu with various search options. Note that all of these procedures are menu-driven, so even the casual user will be able to walk through them. Complete instructions, as attached to the computers at each pool, may be found in Appendix C.

vi) Maintenance:

Besides being familiar with all the user functions, the administrator must know how to add and delete records from the various databases using **FoxBase+/Mac** commands. This is typically done when a module is added to or permanently removed from the pool. Also, in order to make labels with the **codemake** program, the old records in **barcode** must be deleted and the new records added. Finally, the administrator should know how to modify the programs in order to make changes, for example, in passwords. Some of the relevant **FoxBase+/Mac** commands can be found in Appendix D.

Of course, the administrator must also physically maintain the pool. This involves making sure modules are stored in their proper place and stay grouped together by function or manufacturer. Storage and grouping are made easy with the custom storage racks. Also, modules must be checked periodically to determine if they need new bar code labels, which often take a lot of abuse. Modules that need to be repaired should be checked out to the repair group. Failure to maintain the physical organization of the pool negates the benefits of the electronic bookkeeping.

3) Summary:

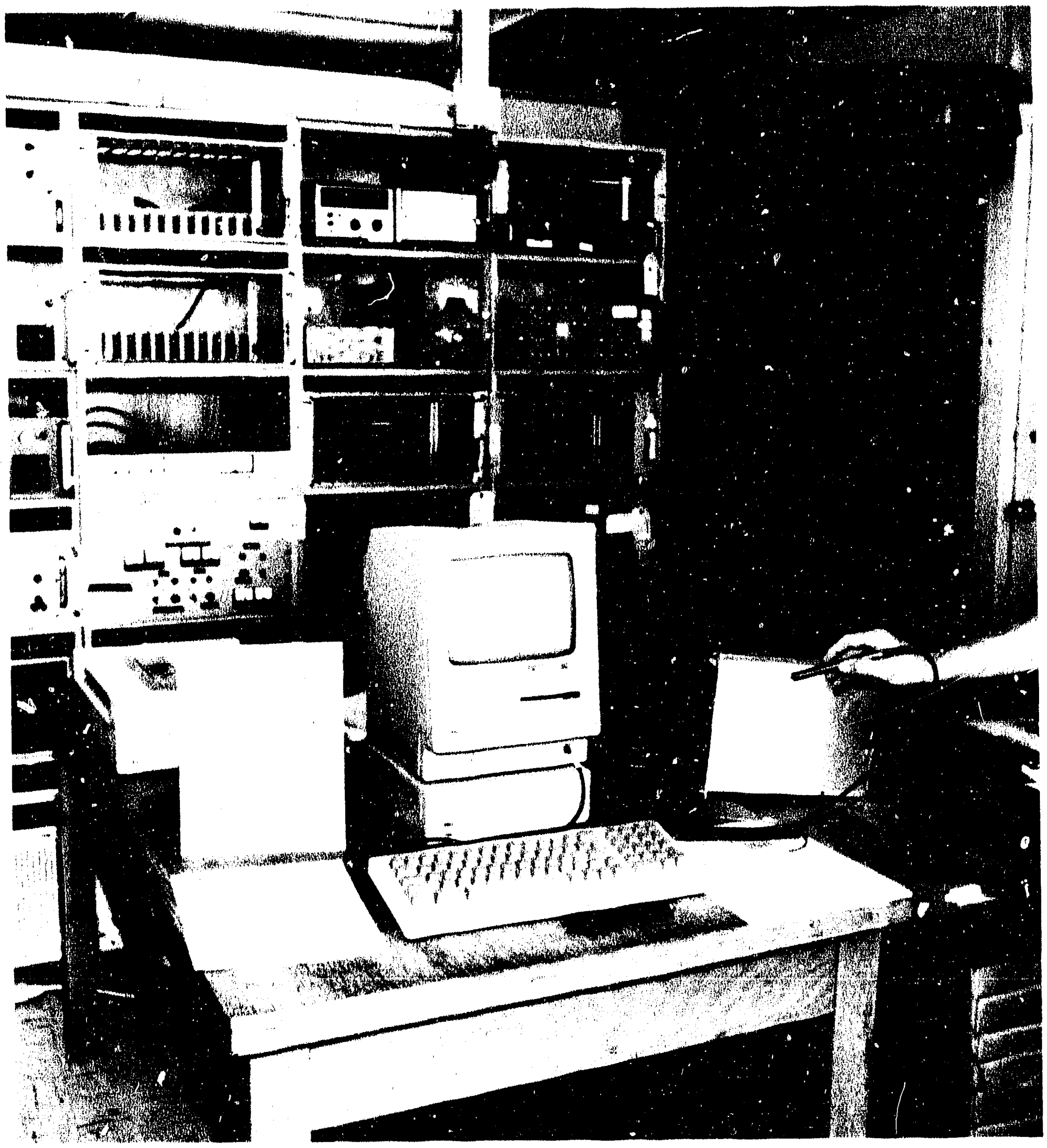
Two pools of CAMAC and NIM modules and miscellaneous related equipment have been set up at each of LBL's particle accelerators. Custom storage racks were fabricated to provide orderly storage of these valuable pieces of equipment. Data bases containing information on these items were constructed on a file server and linked via Macintosh networks to the pool computers. Utilizing bar-code technology, **FoxBase+/Mac** software allows users to easily check items in and out and search for them, as well as providing tracking capabilities for the administrator. This system may be applied to any pool of items whose movements need to be tracked, such as a collection of shop tools, or a library of reference and operation manuals.

Acknowledgements:

The authors would like to acknowledge Dr. Gary Krebs for his many contributions to the Bevalac pool and Mr. George Constantian who developed a precursor data base. Mr. Norman Lino of the LBL carpenter shop built both the prototype and production storage racks.

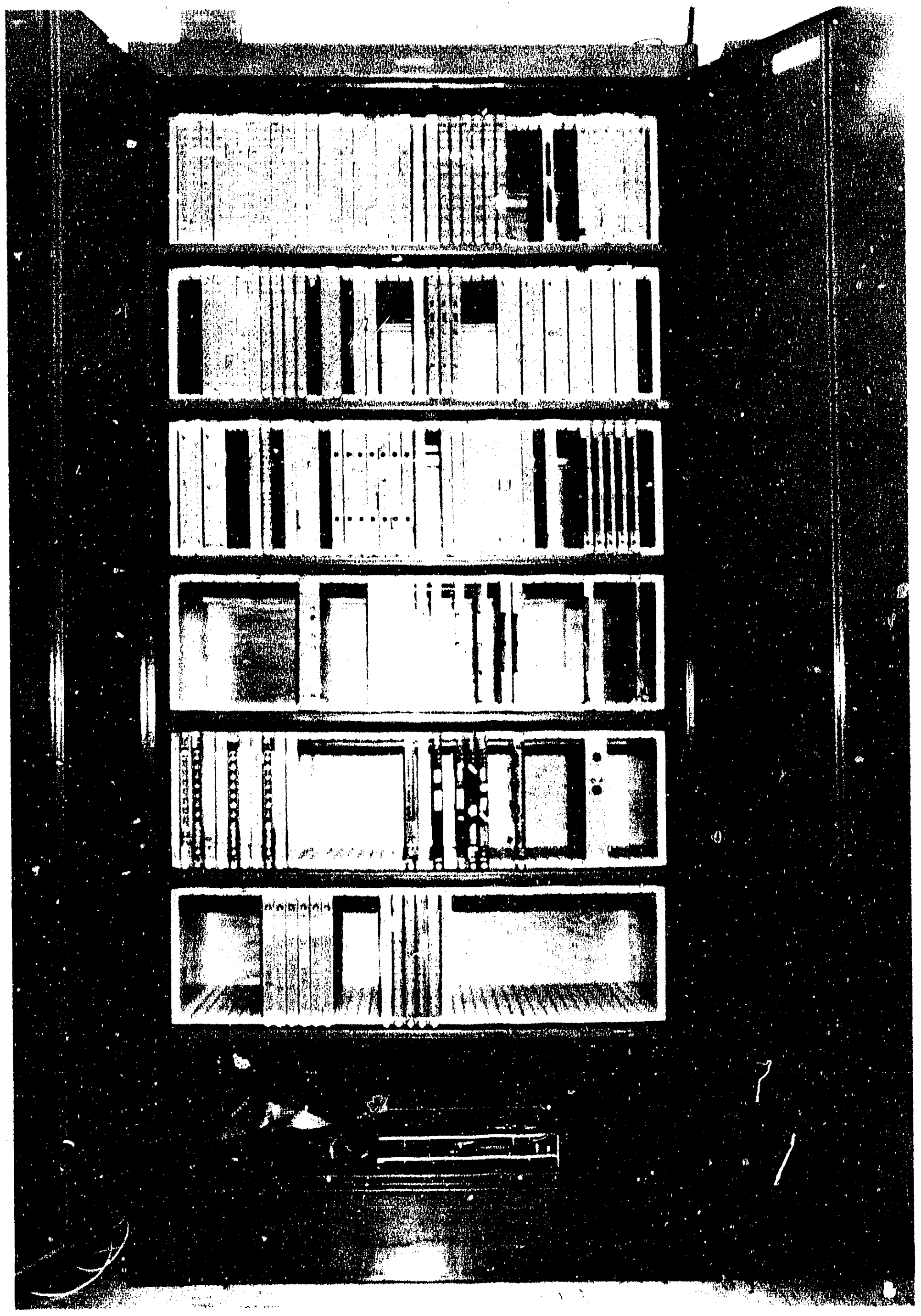
Figure Captions

Figure 1	The Bevalac pool Macintosh. Note the wand used for scanning the bar codes on the sides of the modules.
Figure 2	Six CAMAC module storage racks in a standard 6.5 ft x 3 ft x 1.5 ft steel storage cabinet with doors and lock.
Figure 3.	Seven NIM module storage racks in a standard book case.
Figure 4.	NIM bin and CAMAC crate storage racks. The two lowest spaces can accommodate CAMAC crates.
Figure 5.	Diagram of the computers and their connections via the LBL Ethernet.
Figure 6.	Poolmenu functions and their relationships.
Figure 7.	Database files and their relationships
Figure 8a.	Poolmenu screen
Figure 8b.	check out screen



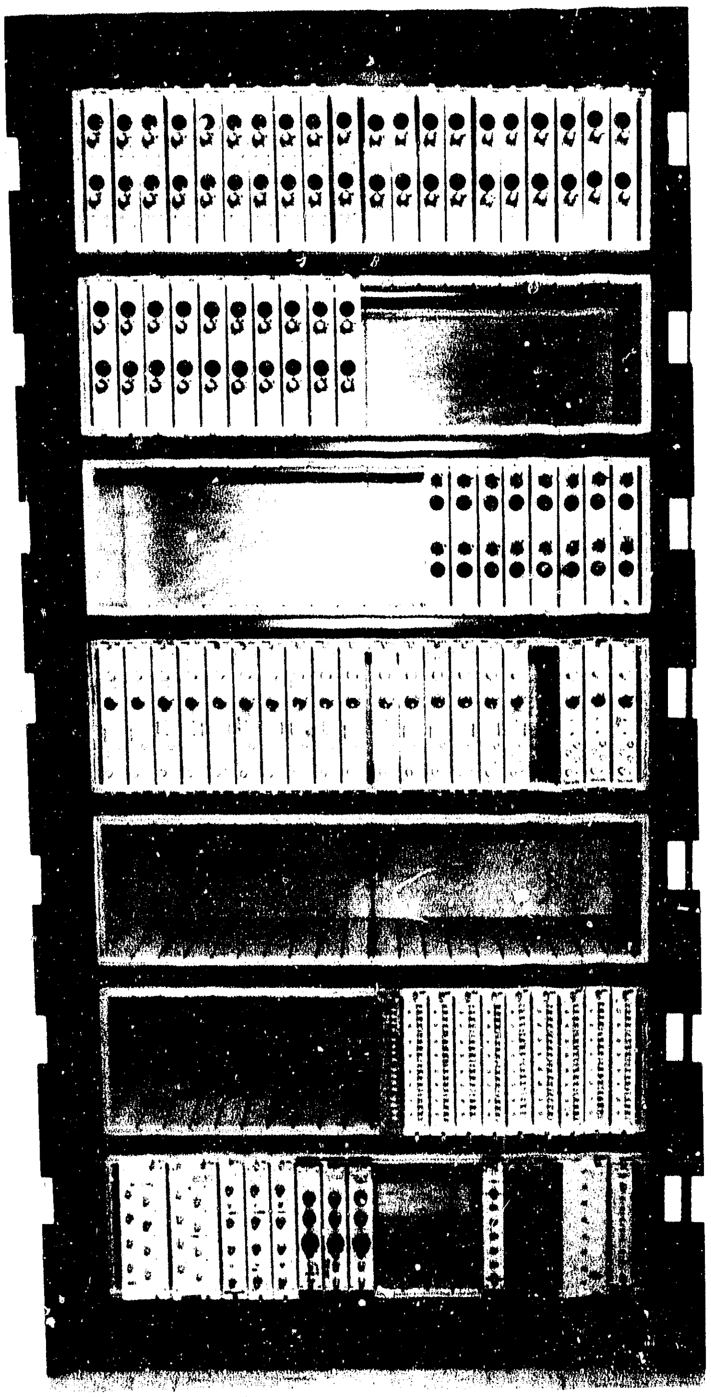
CBB 907-5614

Figure 1



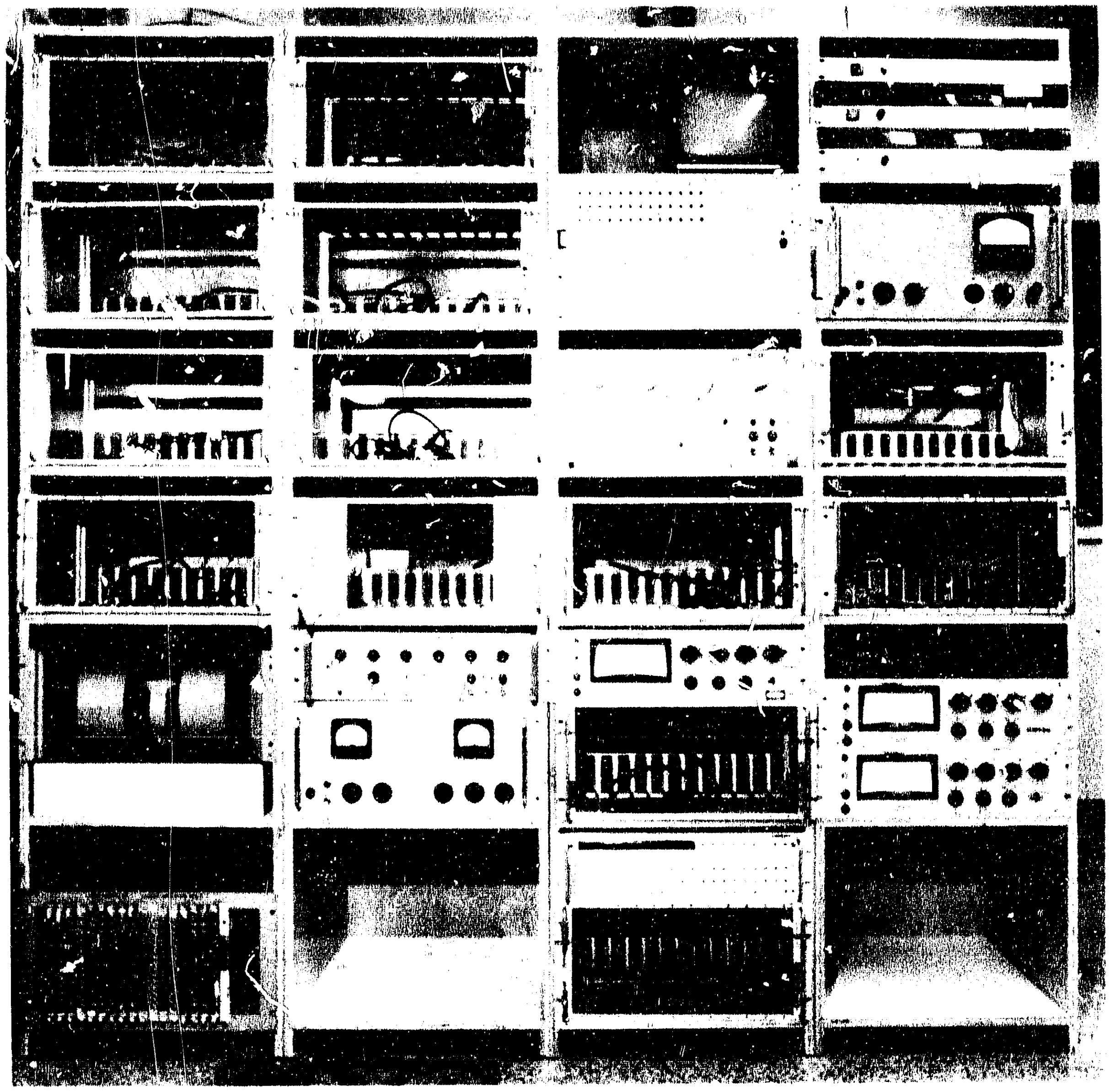
CBB 907-5610

Figure 2



CBB 907-5612

Figure 3



CBB 907-5616

Figure 4

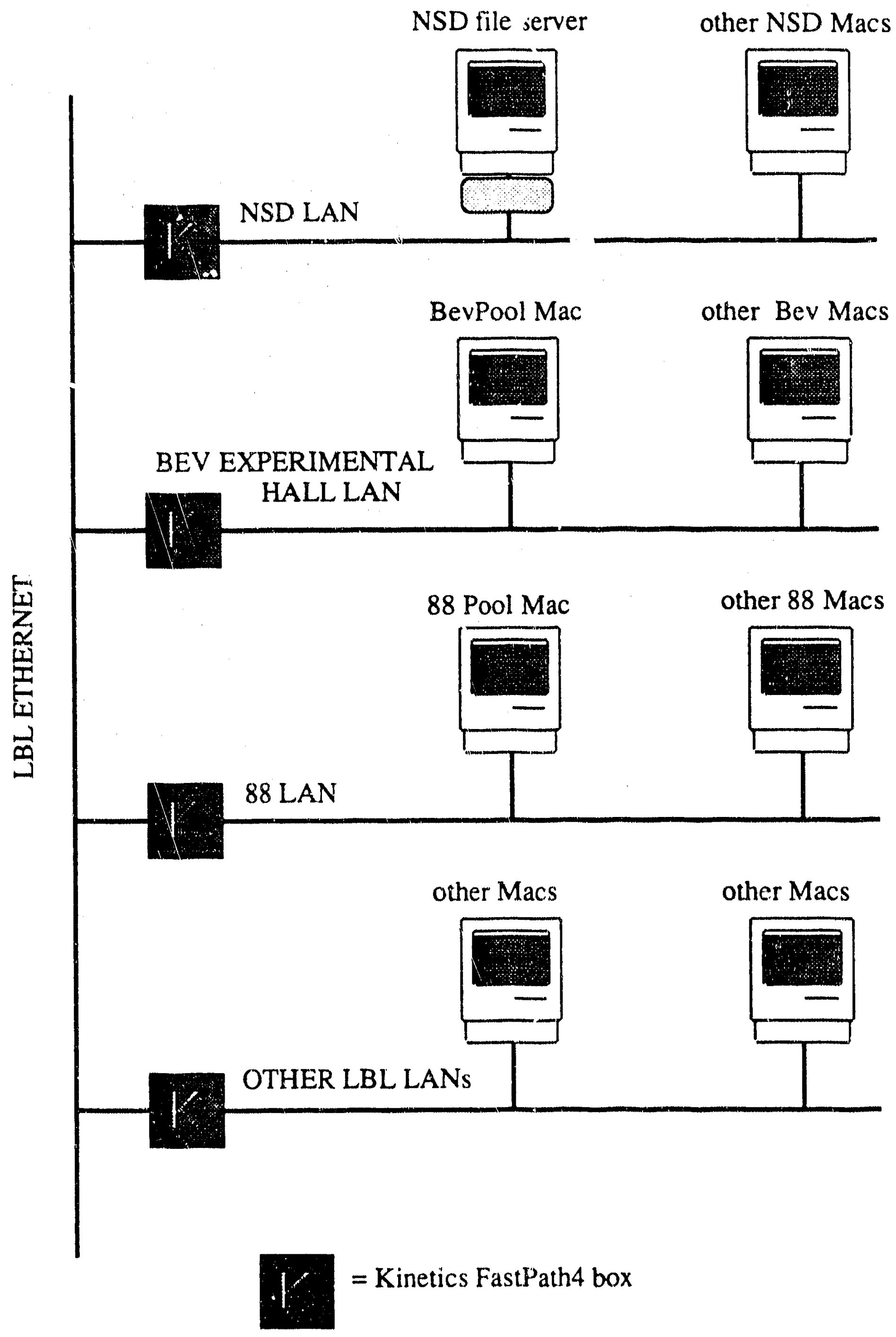

NSD file server
other NSD Macs
NSD LAN
BevPool Mac
other Bev Macs
BEV EXPERIMENTAL HALL LAN
88 Pool Mac
other 88 Macs
88 LAN
other Macs
other Macs
OTHER LBL LANs
LBL ETHERNET
= Kinetics FastPath4 box

Figure 5

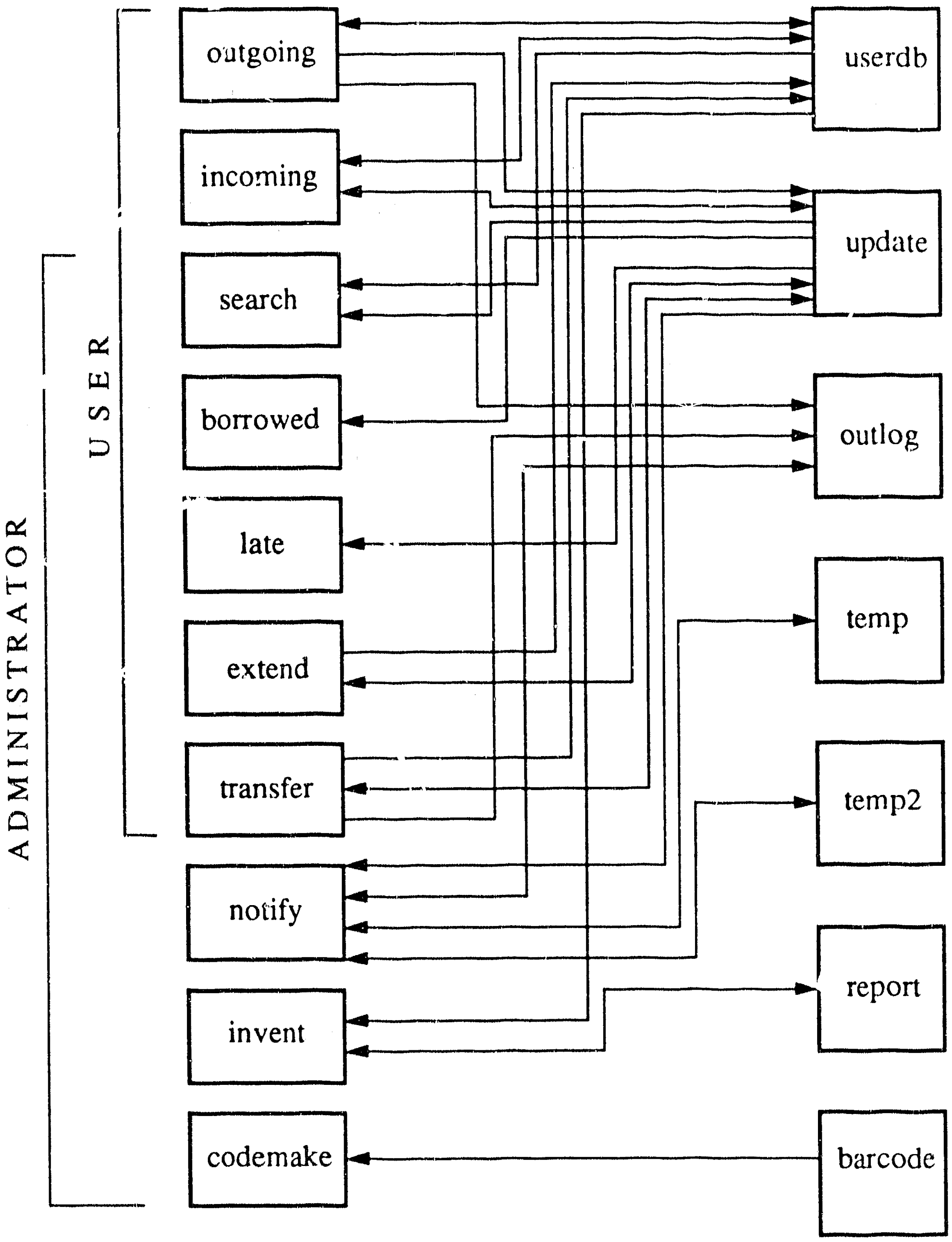
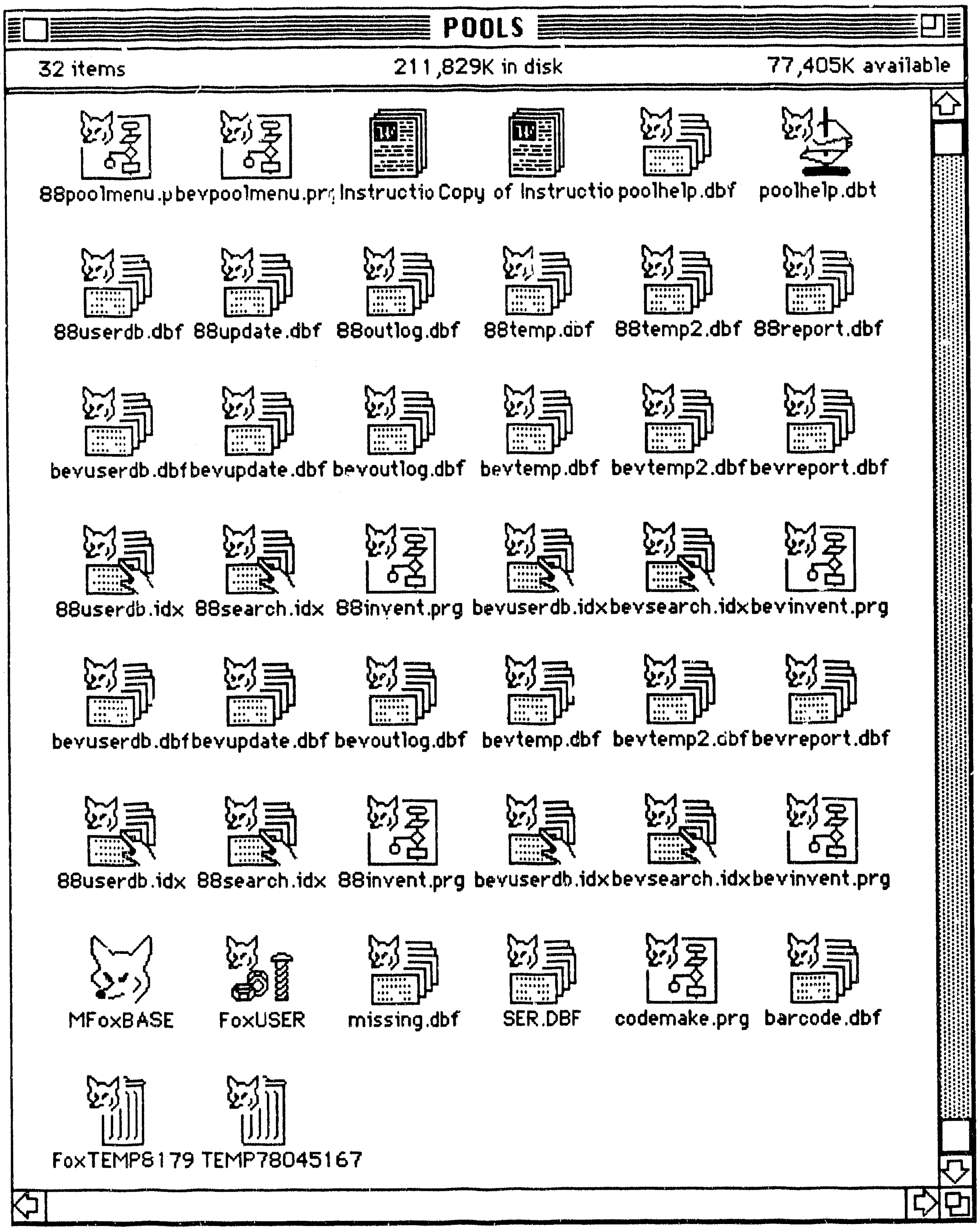

ADMINISTRATOR
USER
outgoing
incoming
search
borrowed
late
extend
transfer
notify
invent
codemake
userdb
update
outlog
temp
temp2
report
barcode

Figure 6



POOLS

32 items 211,829K in disk 77,405K available

88poolmenu.p bevpoolmenu.pr Instructio Copy of Instructio poolhelp.dbf poolhelp.dbt

88userdb.dbf 88update.dbf 88outlog.dbf 88temp.dbf 88temp2.dbf 88report.dbf

bevuserdb.dbf bevupdate.dbf bevoutlog.dbf bevtemp.dbf bevtemp2.dbf bevreport.dbf

88userdb.idx 88search.idx 88invent.prg bevuserdb.idx bevsearch.idx bevinvent.prg

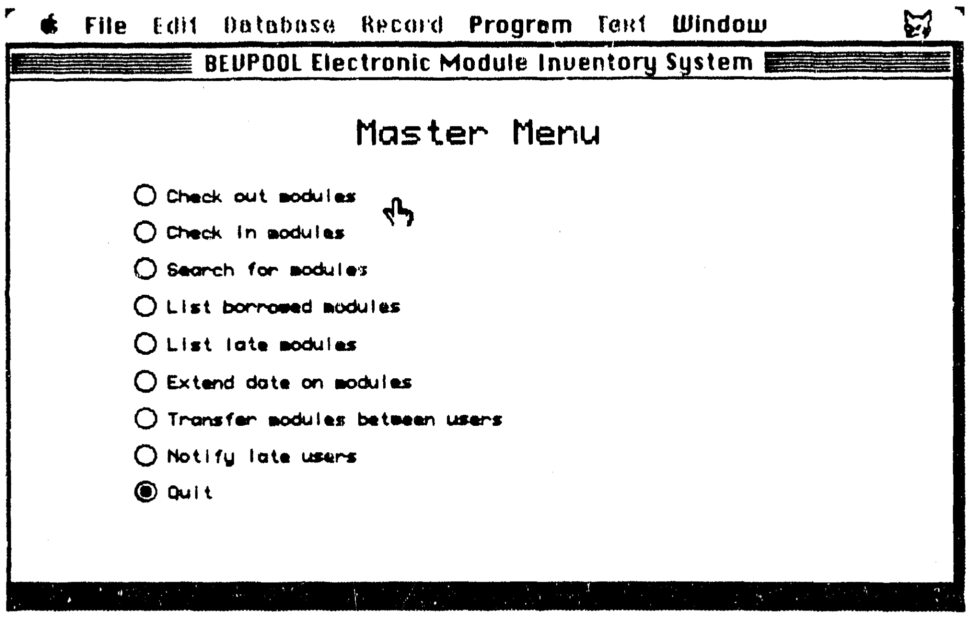
bevuserdb.dbf bevupdate.dbf bevoutlog.dbf bevtemp.dbf bevtemp2.dbf bevreport.dbf

88userdb.idx 88search.idx 88invent.prg bevuserdb.idx bevsearch.idx bevinvent.prg

MFoxBASE FoxUSER missing.dbf SER.DBF codemake.prg barcode.dbf

FoxTEMP8179 TEMP78045167

Figure 7

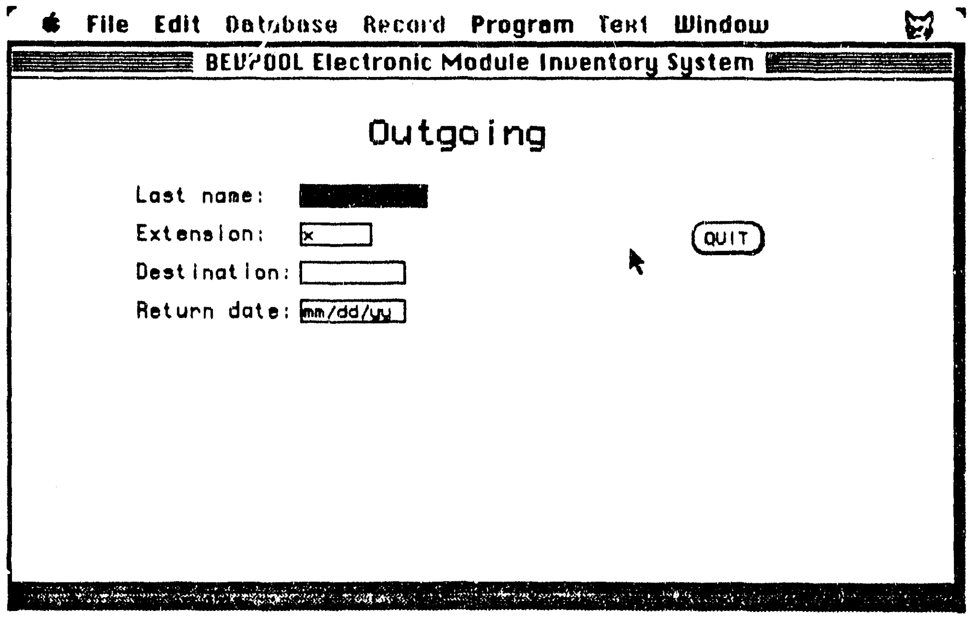


BEVPOOL Electronic Module Inventory System

Master Menu

- ○ Check out modules
- ○ Check in modules
- ○ Search for modules
- ○ List borrowed modules
- ○ List late modules
- ○ Extend date on modules
- ○ Transfer modules between users
- ○ Notify late users
- [✓ Quit]

Figure 8a



BEVPOOL Electronic Module Inventory System

Outgoing

Last name:

Extension: x

Destination:

Return date: mm/dd/yy

Figure 8b

APPENDIX A

Hardware, Software, and Vendors

Macintosh+	Apple Computer, Inc. 20525 Mariani Avenue Cupertino, CA 95014 (408) 996-1010
FoxBase+/Mac	Fox Software, Inc. 134 W. South Boundary Perrysbourg, OH 43551 (419) 874-0162
PrintBar™	Bear Rock Software Co., Inc. 6069 Enterprise Dr. Placerville, CA 95667 (916) 662-4640
WDP Reader	Worthington Data Solutions 417-A Ingalles St. Santa Cruz, CA 95060 (408) 458-9938

APPENDIX B

Storage

The main types of equipment in the pool are CAMAC and NIM modules. In addition, there are NIM bins, CAMAC crates, and other 19-inch rack-compatible units such as power supplies. Holders for these three classes of units were constructed. Set-up work is time consuming in fabricating these pieces, and thus the economies of scale make it beneficial to fabricate several of each item at a time. Typically, 16 of the NIM or CAMAC module racks, enough for two bookcases or cabinets, with spares, is a good number. The bin and crate storage racks are convenient in batches of 4. Note that for small pools, the CAMAC module storage racks will also hold NIM modules, and one cabinet full of CAMAC module racks may meet the needs of a small pool of mixed CAMAC/NIM modules.

1) CAMAC Module Storage Racks:

CAMAC modules pack densely and are relatively expensive, so they are stored in 6.5 ft high by 3 ft wide by 1.5 ft deep steel cabinets with doors and locks. Six or seven inserts may be fit into one of these cabinets. These racks are meant for CAMAC modules with single side rails. CAMAC storage racks are fabricated as follows:

1) Top and bottom panels (Figure B-1) for CAMAC modules:

a) cut 1/2 inch thick plywood to 32 3/8 inches long by 13 3/4 inches wide.

b) cut 3/32 inch deep slots at 1 1/8, 1 7/8, 2 5/8, 3 3/8 .... inches from both sides. (Note that this procedure reduces the number of settings for the rip fence on the table saw.) Cut an additional 3/32 inch slot at 15 5/8 inches for the center brace.

c) cut one end off so the total length is 31 1/8 inches.
d) cut this piece to 12 1/2 inches wide for the BOTTOM panel. The remaining (~1 1/8 inches) will be TOP. Note that the slotting process may warp the pieces, and an additional stiffener may be needed on top, as shown in the drawing.

2) End Pieces: Cut 1/2 inch plywood to size (12 1/2 x 8 11/16 inches) and add 1/4 inch x 1/2 inch rabbits at top and bottom. Note that the critical dimension is the 7 11/16 inches inside the rabbits. See Figure B-2.

3) Assemble by attaching top and bottom pieces to side pieces as shown in Figure B-2. Add a 1/8 thick back plate and a 1/8 x 1 x 7 7/8 center brace for support. Add an additional 1/2 x 1/2 x 31 1/8 inch block at the rear to support the back plate. Note that a thicker center brace (1/2 inch wide) may be needed if the top and bottom pieces were warped by cutting the slots.

4) Spray with a clear finish.

2) NIM Module Storage Racks:

For the NIM modules, normal 78 inch high by 38 inch wide by 10 inch deep steel bookcases were chosen for the overall structure and wooden inserts were fitted to them, seven inserts per bookcase. These inserts are constructed from plywood as follows:

1) Top and bottom panels (Figure B-3) for NIM modules:

a) cut 1/2 inch thick plywood to 32 3/8 inches long by 12 inches wide.

b) cut 3/32 inch deep slots at 1 7/8, 3 3/8 ....inches from both sides. Note that this procedure reduces the number of settings for the rip fence on the table saw. Cut an additional 3/32 inch slot at 15 5/8 inches for the center brace.

c) cut one end off so the total length is 31 1/8 inches.
d) cut this piece to 10 3/4 inches wide for the BOTTOM panel. The remaining (~1 1/8 inches) will be TOP. Note that the sawing process may warp the pieces, and an additional stiffener may be needed on top, as shown in the drawing.

2) End Pieces: Cut 1/2 inch plywood to size (10 3/4 x 8 11/16 inches) and add 1/4 inch x 1/2 inch rabbits at top and bottom. Note that the critical dimension is the 7 11/16 inches inside the rabbits. See Figure B-4.

3) Assemble by attaching top and bottom pieces to side pieces as shown in Figure B-4. Add a 1/8 thick back plate and a 1/8 x 1 x 7 7/8 center brace for support. Add an additional 1/2 x 1/2 x 31 1/8 inch block at the rear to support the back plate.

4) Spray with clear finish.

3) NIM Bin and CAMAC Crate Storage Racks

Besides modules, one needs to store (NIM) bins and (CAMAC) crates, as well as various 19-inch rack-compatible components, such as power supplies. The storage racks shown in Figure B-5 may be used for that purpose. They are constructed as follows:

1) Using a piece of 4 ft by 8 ft by 3/4 inch thick plywood, cut out the two sides 67 3/4 by 24 inches each, and two of the shelves 18 3/4 by 24 inches each. Note that the "24 inch depth" is only nominal. The other 5 shelves can be cut from another piece of 3/4 inch thick plywood. Six of the shelves should be covered with a hard plastic laminate.

2) Cut six dadoes 3/8 inch deep by 13/16 inch wide (wide enough for the 3/4 inch plywood and the plastic sheet) in the sides to hold the shelves. Cut a 3/8 inch deep by 3/4 inch rabbit at the top, and glue (and nail) the shelves into place. A piece of 1/4 inch plywood should be nailed into place as a backing.

3) Spray with a clear finish.

Note that by using 4 ft by 8 ft by 3/4 inch thick plywood, each piece will make two sides and two shelves. Thus, 4 pieces will make all the sides and the top and bottom shelves for 4 units. The other 5 x 4 =20 shelves may be made from two additional pieces of 4 ft by 8 ft by 3/4 inch thick plywood. Thus, 6 pieces of plywood will make 4 units.

Figure Captions

Figure B-1 Top and Bottom pieces for CAMAC module storage racks.

Figure B-2 Side pieces for CAMAC module storage (lower right) and assembly drawing for CAMAC module storage racks.

Figure B-3 Top and Bottom pieces for NIM module storage racks.

Figure B-4 Side pieces for NIM module storage (lower right) and assembly drawing for NIM module storage racks.

Figure B-5 Construction and assembly drawing for bin and crate storage racks.

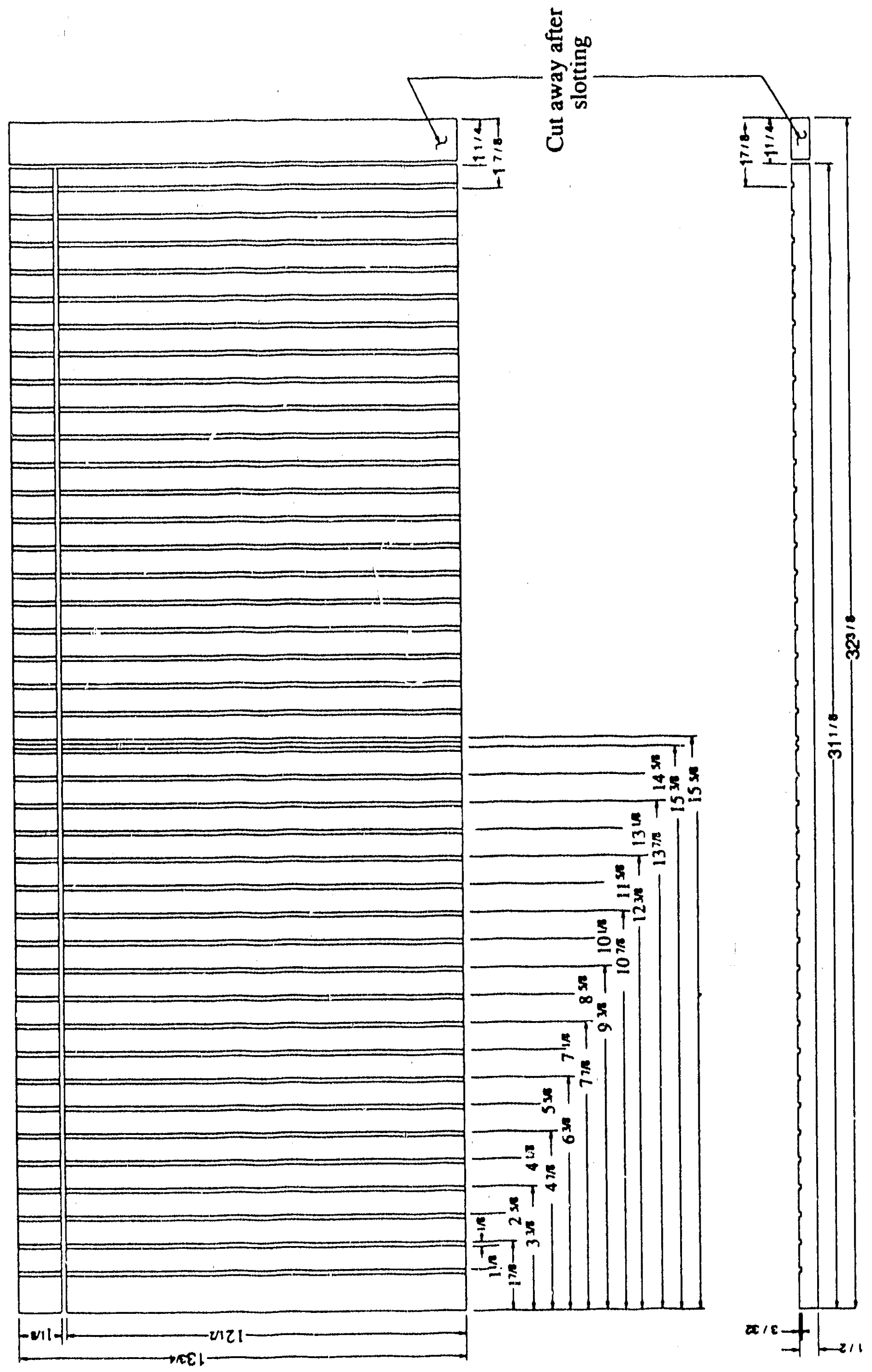

Cut away after slotting
1 1/4
1 7/8
32 3/8
31 1/8
1/8
12 1/2
13 3/4
3/32
1/2
1 1/8
1 7/8
1/8
2 5/8
3 3/8
4 1/8
4 7/8
5 5/8
6 3/8
7 1/8
7 7/8
8 5/8
9 3/8
10 1/8
10 7/8
11 5/8
12 3/8
13 1/8
13 7/8
14 5/8
15 3/8
15 5/8

Figure B-1

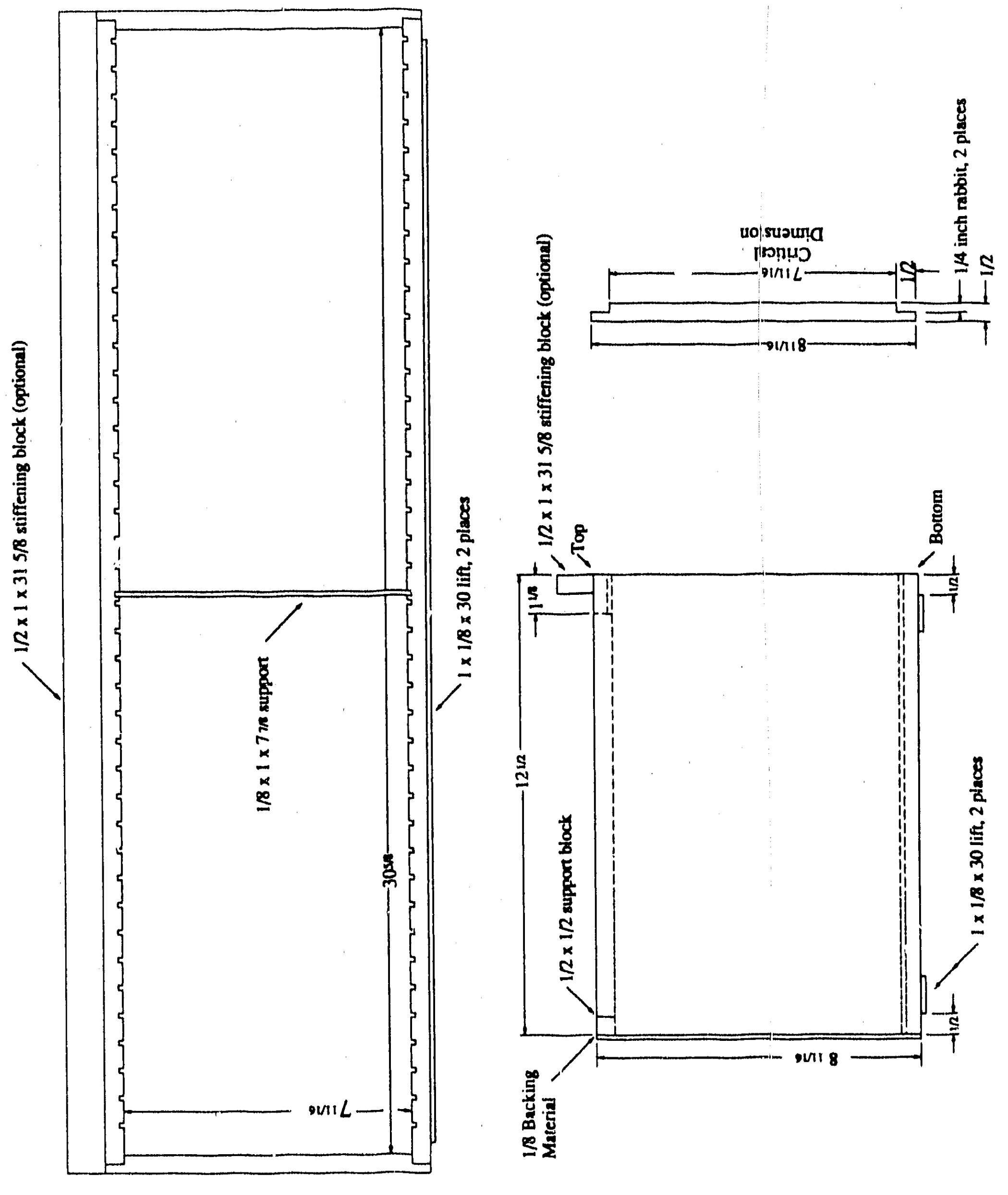
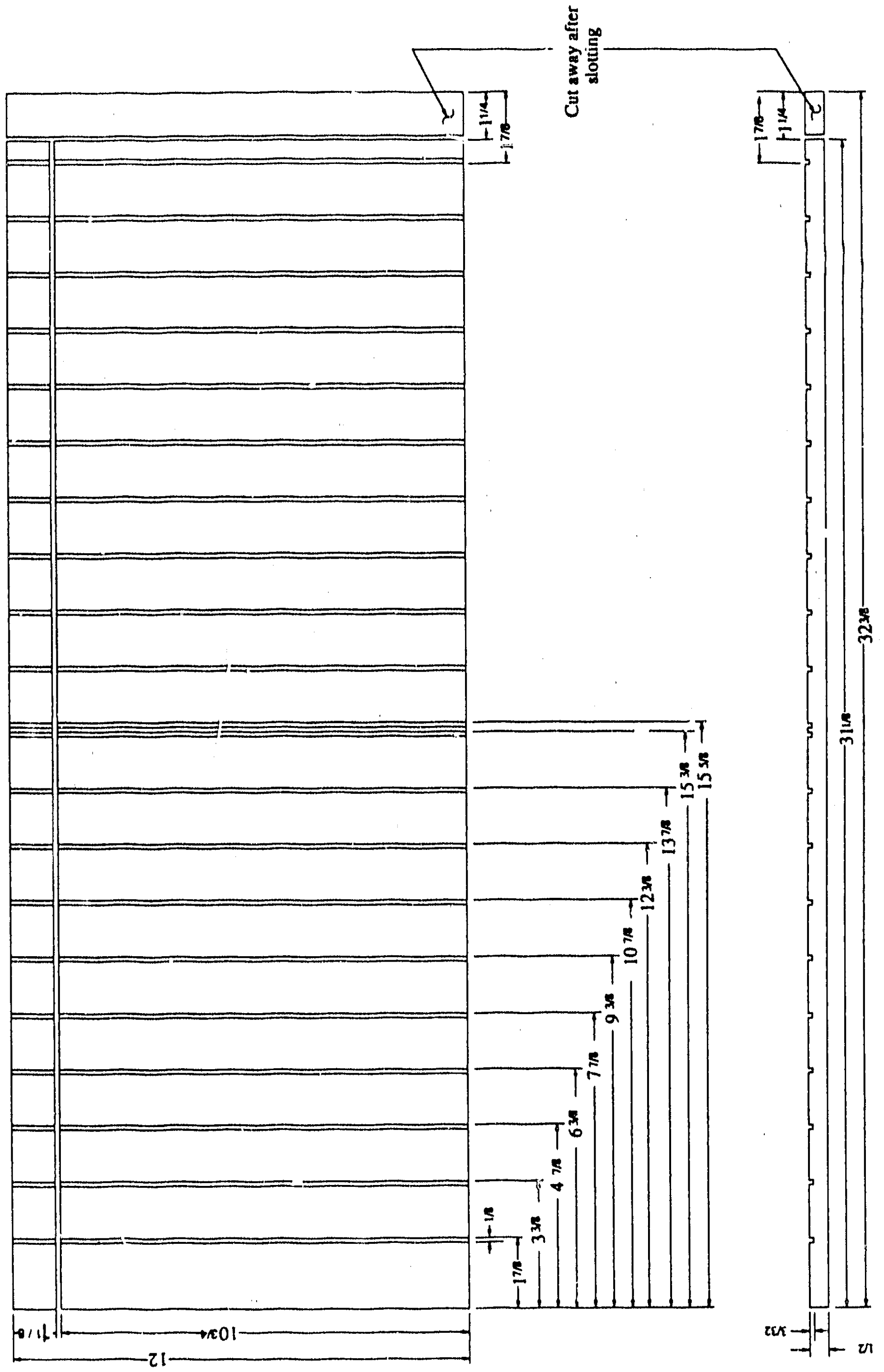

1/2 x 1 x 31 5/8 stiffening block (optional)
1/8 x 1 x 7 7/8 support
1 x 1/8 x 30 lift, 2 places
30 5/8
7 11/16
1/2 x 1 x 31 5/8 stiffening block (optional)
Top
1 1/8
12 1/2
1/2 x 1/2 support block
1/8 Backing Material
8 11/16
Bottom
1/2
1 x 1/8 x 30 lift, 2 places
1/2
8 11/16
7 11/16
Critical Dimension
1/2
1/4 inch rabbit, 2 places
1/2

Figure B-2



Cut away after slotting
1 1/4
1 7/8
1/8
3 3/8
4 7/8
6 3/8
7 7/8
9 3/8
10 7/8
12 3/8
13 7/8
15 3/8
15 5/8
1/8
10 3/4
12
32 3/8
31 1/8
3/32
1/2

Figure B-3

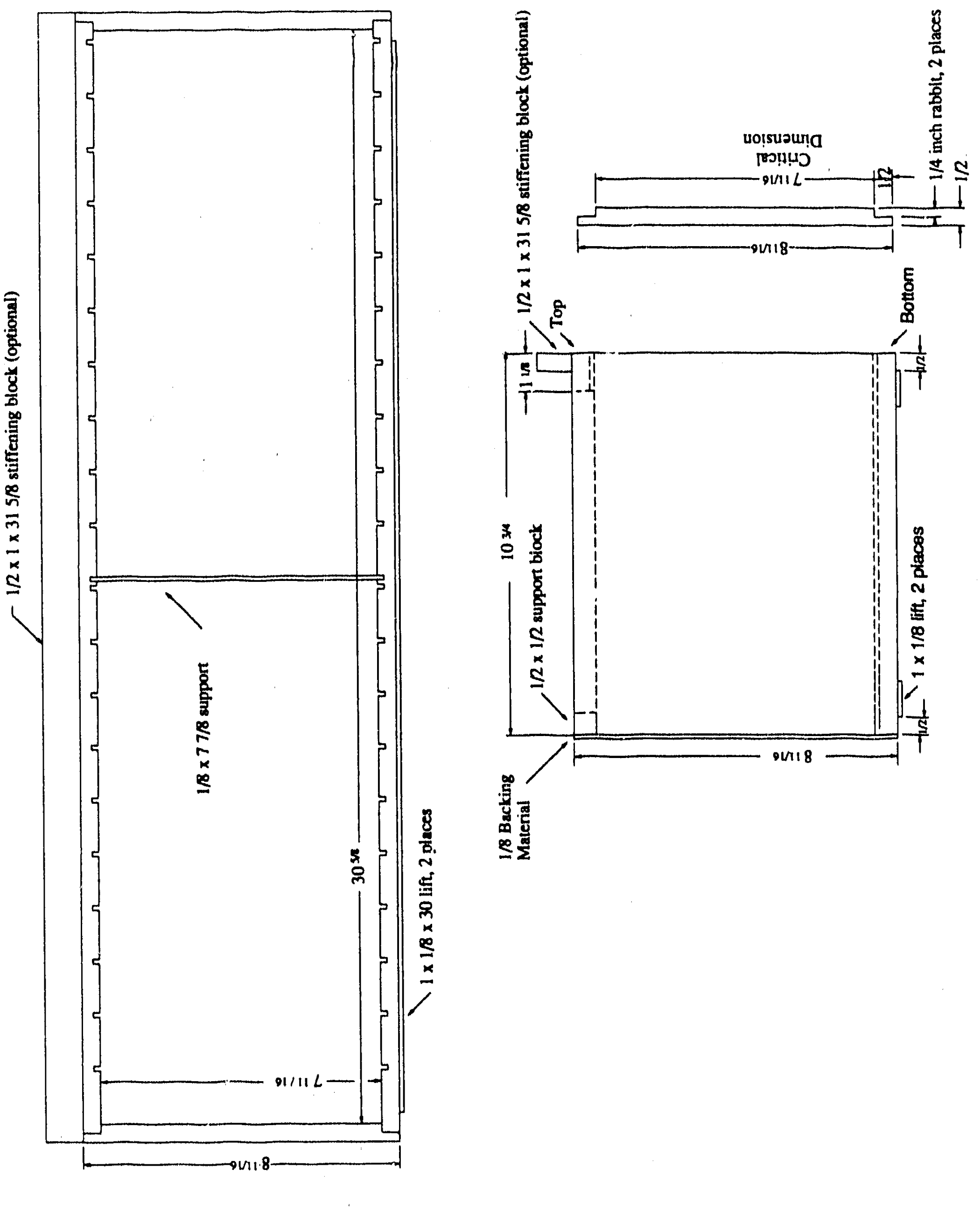
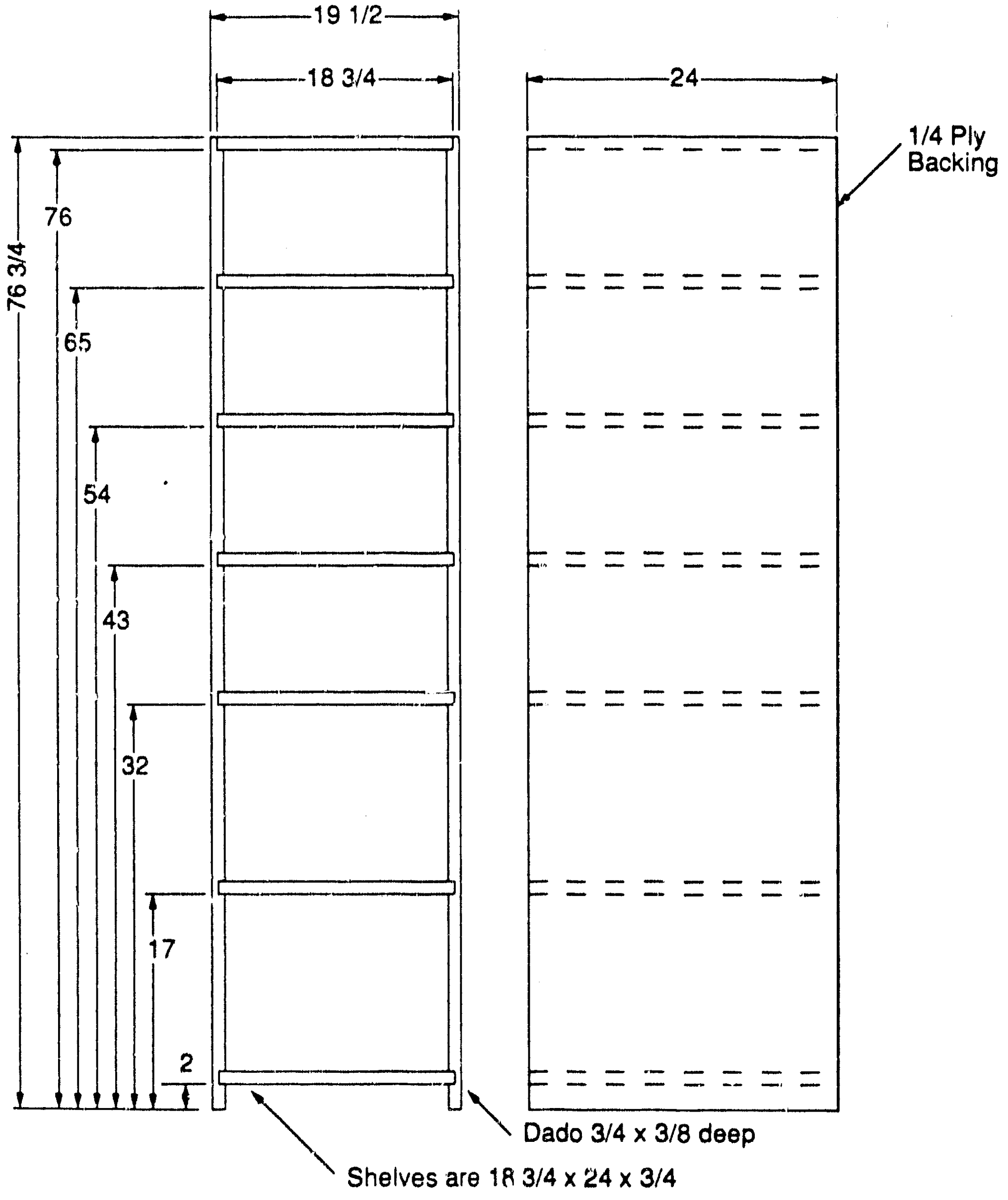

1/2 x 1 x 31 5/8 stiffening block (optional)
1/8 x 7 7/8 support
30 5/8
7 11/16
8 11/16
1 x 1/8 x 30 lift, 2 places
1/2 x 1 x 31 5/8 stiffening block (optional)
Top
1 1/8
10 3/4
1/2 x 1/2 support block
1/8 Backing Material
8 11/16
Bottom
1/2
1 x 1/8 lift, 2 places
7 11/16
Critical Dimension
1/2
1/4 inch rabbit, 2 places
1/2
8 11/16

Figure B-4



19 1/2
18 3/4
24
1/4 Ply Backing
76 3/4
76
65
54
43
32
17
2
Dado 3/4 x 3/8 deep
Shelves are 18 3/4 x 24 x 3/4

Figure B-5

Appendix C

Operating the CAMAC/NIM Pool Inventory System on the Macintosh

To start the system:

01. Turn on the hard drive and the computer.
02. Go to the menu and select Chooser.
03. Choose the nsd zone and click on the Appleshare icon.
04. Select nsd FS.
05. Connect as a guest.
06. Open the nsd FS volume.
07. Close the Chooser window.
08. The nsd FS icon should now be on the desktop. Open it.
09. Open the Public Access folder.
10. Open the POOLS folder.
11. Double-click on the **bevpoolmenu** or **88poolmenu** program icon.
12. Wait for the FoxBase program to load. It takes a few seconds.
13. If you are at the pool, scan the code at the center top of the Macintosh when requested.
14. The menu program should now be running.
15. If the FoxBase program is loaded but for some reason the menu program is not running (blank white window), go to the PROGRAM part of the menu bar and select DO... . Then select the **bevpoolmenu** or **88poolmenu** program.

Using the menu:

01. Use the mouse to select options by single-clicking on the radio buttons, text buttons, etc. Hitting a key when the program expects a button click will usually result in the default option, which is usually quit.
02. When filling out text boxes, use the keyboard and terminate your responses with the <return> key. The program will automatically step through the boxes. If you use the mouse to skip a box, the computer is likely to squeak at you and make you do it all over.
03. Use the wand when entering DOE numbers. The reader will beep for a valid read. A one-second sweep across the bar code is about ideal. In the event of a failure of the bar-code reader, the number may be entered on the keyboard. Give it several tries before you resort to the keyboard.
04. Wand entries automatically include a <return>, and completely satisfy the computer. Note that the search on DOE number option does not accept a wand entry, since you obviously don't have the bar code for a module you are searching for.
05. There are many places in the program where you can quit and go back if you enter something wrong. Feel free to use the quit buttons if you have made a mistake or if you are not sure what to do.
06. Users should never use the **notify** option, and should not use **quit** when at the pool since the program should be left running.
07. If you are not at a pool computer you will only be able to **search**, list **borrowed** modules, and list **late** modules.

Appendix D

Relevant FoxBase+/Mac Commands for the Administrator

A database is opened with the **FoxBase+/Mac** command **use** file name, as in **use update**. For **userdb**, the command must be **use userdb index userdb** to insure that the index file is loaded along with the database. The command **browse** allows one to browse through the database with a scrolling window. Each record takes one line. The **edit** command is similar, but the format is different and each record takes about a full screen. To add records, one must use the **append** command. This brings up a blank **edit**-style record for editing. Leaving the last field with a return ends editing for that record. If the database is indexed, the record will be put in its proper place and the command will then be terminated. Otherwise, a new blank record will be presented for editing. When all additions have been made, the database is closed with the command **close database**, **close all**, or **use**. An indexed file must have the command **reindex** before closing.

For deleting all the records in **barcode**, use the command **zap** when it is open. For deleting individual records in a database, open it and go to the record using **browse** or **edit**. Clicking in the thin bar to the left of the record will cause it to be marked for deletion. This can also be accomplished by clicking on the record, going back to the command window and typing **delete**. Once all the records to be deleted have been marked, the command **pack** will remove them permanently. Then the database can be closed. The command **modify command** opens the specified program for editing. Clicking on the close window ends the editing and brings up a save dialog box. Note that all of these commands are available in some form in the menu bar.

Appendix E

Listing of function outgoing

```
procedure outgoing
on escape return to master
set talk off
x=0
xx=0
qc=0
n="name"
p="phone"
l="location"
d="date"
m="module number"
z="MFG"
zz="MOD"
zzz="DESCRIPT"
zzzz="OWN"
do while x=0
   clear
         @ 2,31 say "Outgoing" font "Monaco",18
   xx=0
         n="          "
         p="x      "
         l="        "
         d="mm/dd/yy "
   do while xx=0
                  qc=0
                  @ 5,10 say "Last name:" font "Monaco",12
                  @ 5,25 get n size 1,11
                  @ 7,10 say "Extension:" font "Monaco",12
                  @ 7,25 get p size 1,6
                  @ 9,10 say "Destination:" font "Monaco",12
                  @ 9,25 get l size 1,9
                  @ 11,10 say "Return date:" font "Monaco",12
                  @ 11,25 get d size 1,9
                  @ 6,60 get qc picture "@* QUIT"
                  read
                  n=trim(n)
                  p=trim(p)
                  l=trim(l)
                  d=trim(d)
                  xx=1
         if qc=1
            return to master
         endif
      if len(n)=0
                           alert stop 2 "I need a name!"
                           n="          "
```

```
                xx=0
  endif
  if len(p)=0
    alert stop 2 "I need an extension number!"
                p="x     "
                xx=0
  endif
          if len(l)=0
    alert stop 2 "I need a destination!"
                l="        "
                xx=0
  endif
          if substr(d,3,1)<>"/" .and. substr(d,3,1)<>"-"
                d="0"+d
          endif
          if substr(d,6,1)<>"/" .and. substr(d,6,1)<>"-"
                d=substr(d,1,3)+"0"+substr(d,4)
          endif
  if len(d)<>8
    alert stop 2 "I need a date!"
                d="mm/dd/yy "
                xx=0
          endif
          if len(d)=8 .and. date()>ctod(d)
                alert stop 2 "I need a FUTURE date!"
                d="mm/dd/yy "
                xx=0
          endif
enddo
    if year(ctod(d))>year(date())+1
  alert note 2 "I doubt you can have it for that long!"+;
                "  But I'll let it go...for now."
endif
    on escape ? chr(7)
    on error do err3 with ERROR(),MESSAGE()
    use bevuserdb.dbf index bevuserdb.idx
    select 2
    on error do err3 with ERROR(),MESSAGE()
    use bevupdate.dbf exclusive
    select 3
    on error do err3 with ERROR(),MESSAGE()
    use bevoutlog.dbf exclusive
    select 1
    on error
    @ 15,10 say "Enter doe numbers of outgoing modules:" font "MONACO",12
xx=0
do while xx=0
          m="             "
          qc=0
          @ 17,30 get m size 1,13
          @ 17,60 get qc picture "@* QUIT"
          read
          @ 19,10 say "                              "
```

```
            @ 20,10 say "                              "
m=trim(m)
            if qc=1
   xx=1
   x=1
                     exit
endif
if len(m)>7
   m=right(m,7)
endif
go top
seek m
 if eof() .or. doe="0000001"
   alert stop 2 "That module does not exist!  Please make"+;
                              " a note on a receipt."
else
   if location<>"BEVPOOL"
      alert stop 8 "That module has not been signed in yet."+;
                                      " You can take it if you sign it in and check"+;
                                      "  it out in your name."
   else
      do while .not. rlock()
                                      @ 19,10 say "One moment please..."
                              enddo
                              replace location with l
      replace name with lower(n)
      replace phone with p
      replace date_in with ctod(d)
                              m=doe
      z=mfg
      zz=mod
      zzz=descript
                              zzzz=own
      @ 19,10 say z+" "+trim(zz)+" "+trim(zzz)
      unlock
      select 2
      append blank
      replace doe with m
      replace mfg with z
      replace mod with zz
      replace descript with zzz
                              replace own with zzzz
      replace location with l
      replace name with lower(n)
      replace phone with p
      replace date_in with ctod(d)
      select 3
      append blank
      replace doe with m
      replace mfg with z
      replace mod with zz
      replace descript with zzz
                              replace own with zzzz
            replace location with l
            replace name with lower(n)
            replace phone with p
            replace date_out with date()
            select 1
                  @ 20,10 say "OK - signed out"
                           endif
      endif
   enddo
         select 3
         use
         select 2
         use
         select 1
         use
         on escape return to master
enddo
set talk on
on escape
```