THE ECTN MARK III

(LSI-11 FRONT END) SYSTEM

by

J.A. Lyons and A. Vesa

Seismological Service of Canada

INTERNAL REPORT 81 - 5

September 1981

Division of Seismology and Geothermal Studies

Earth Physics Branch

Department of Energy, Mines and Resources

Contents

Appendices

Figures

Tables

Implementation of the ECTN Mark III
(LSI-11 Front End) System

1.0 Introduction

This report discusses the current status of the new ECTN Mark III multiprocessor system that has recently gone into full production mode in Ottawa. In the new system, an LSI-11 dedicated front end processor receives the telemetered data stream, unscrambles and formats it into one-second data records, and produces four channels of visual (D/A) monitor records. A PDP-11 minicomputer host controls the front end, receives the formatted data records, maintains the ringbuffer, runs the event detection algorithm, and, in Ottawa, continuously saves Glen Almond long period data.

The reader is assumed to be familiar with internal report 80-9, "Overview of the proposed LSI-11 front end processor system", (J.A. Lyons, July 1980), hereinafter referred to as paper I. Since most of the material in paper I remains valid, only information that is new or has changed is incorporated in the present report.

Designing and implementing the Mark III system has entailed a substantial software development effort. Not only did the multiprocessor configuration require a great deal of new communication software, but also much of the existing Mark II code proved too rigid and unwieldy to be readily adapted to the needs of the new system. Essentially all of the design objectives identified in paper I have been met. In addition, during the process of adapting and re-designing some of the existing data structures and module functions, it became possible to build in a number of further significant enhancements to the system. This initial release of the ECTN Mark III software will be designated Version 1.0 (M3V1.0).

This report includes a substantial amount of technical detail. The aim has been to strive for a single, comprehensive repository of information covering most aspects of the software/hardware system. More operational detail will be provided in a separate "Operator's Manual". An attempt will be made to issue documentation updates noting the significant changes and enhancements embodied in future releases of the software.

2.0 Front End Subsystem

2.1 Capsule Module Summary

22	new modules created
1	replaced
10	modified
7	eliminated
1	unchanged

Almost all of the Mark II SECBLD modules have required modification to accommodate the added facility of comprehensive error and line status checking in the front end, and passing this information along with the one-second data buffers to the host. Most of the ZDDZ modules are no longer required.

2.2 Additional Features.

The following features represent significant extentions of what was outlined in paper I:

- the data reception logic has been modified to reject samples flagged by the DZ hardware with framing or overrun errors to prevent potentially bad data from being saved and perhaps replicated.

- a new algorithm for generating software time marks now ensures the correct positioning of hour and day marks on visual records produced by the ROM standalone front end system, even though it has not synched time with the host.

- a new HST function permits the operator to alter the selection of D/A output channels and gain on-the-fly, which will facilitate implementation of a "roving" helicorder monitor when the number of input components exceeds the number of visual monitors available.

- the internal D/A table structure has been modified to permit flexible specification of the number of D/A output channels desired, up to a maximum of 16, which should handle all foreseen ECTN and WCTN configurations.

- a new initialization module produces a pip code on the visual outputs on startup which identifies the D/A channels and distinguishes between the ROM and down-line-load systems, important when the configurations in each may differ (see Figure 2-1).

- a new HST function permits re-booting the front end from software, as opposed to manual switch toggling, making this function available to indirect command files.

2.3 ROM Load System

The LSI front end contains a copy of its run-time code and a small loader module (XBBOOT) in PROM. On re-boot, this code is copied into RAM memory as part of the automatic self test/startup sequence initiated by the

BDV-11 diagnostic/bootstrap hardware module. Control passes to the loader module residing in high memory, which initially attempts to contact the host over the inter-processor link. If contact is established, the current down-line-load program is received from the Host and loaded over top of the ROM code, and then executed. Otherwise, after a timeout interval of about one minute, XBBOOT decides that there is no host processor available and starts executing the ROM code. Should the host go down and subsequently come back up while the front end is running down-line-loaded (ie., up-to-the-minute) code, the front end will not be re-booted. Instead, it will continue to run, thereby maintaining its error/status summary counts and report how long the host was down. However, if the front end is running in standalone (ROM) mode when the HOST program starts up, it will be re-booted in order to send it the current program and synchronize the time.

Generation of both the ROM and down-line-load systems is outlined in section 4.2. The step by step procedure for creating PROM images and burning PROM's forms appendix A of this report.

2.4 - LSI-11/23 CPU Upgrade.

Timing tests on the original LSI-11/2 front end processor have shown the CPU to have marginally sufficient capacity to handle sixteen 60 sample/second input channels with the minimum level of status and error checking considered desirable. A parallel set of tests using an LSI-11/23 CPU card, which is plug compatible with the 11/2, and which has only become available since our original purchase, proved the 11/23 CPU to have twice the throughput. It has therefore been decided to upgrade the hardware configuration to include the LSI-11/23 CPU card. With the capacity to handle

30 plus components, this should eliminate concerns of front end CPU throughput for the foreseeable future.

2.5 Backup/Development LSI System

A second complete front end system, including the LSI-11/23 CPU card, has been delivered in Ottawa. This system will serve both as a backup front end and as a software development facility.

In backup mode, this second system will guarantee the availability of sufficient DZ-11 input ports and D/A visual output channels to run the full Mark III system. This in turn will allow a streamlined hardware configuration on the host minicomputer system. (See section 8.)

Under normal circumstances, the second system will provide a facility on which Mark III software maintenance and development can be carried on after the original LSI-11 system is committed to full production.

3.0 Host Subsystem

3.1 Capsule Module Summary.

20 new modules created
2 modified
2 eliminated
1 unchanged.

Most of the HOST software functions are new to the Mark III system. A good deal of effort has gone into synchronizing front end/host communications, especially important for the case of multiple front end processors when some may be heard from while others are not.

3.2 SECBUF Structure

A complete re-design of the structure of the one-second data blocks was undertaken to fulfill the requirements of binding the front end error and status information with the time series data, and of handling the input of multiple front ends on a single host processor. Figure 3-1 shows the new layout. The design and access concepts have been greatly simplified by considering a SECBUF to be a single record rather than a rigid, linked grouping of physical blocks. The time series data is now densely packed: the continuity word has been moved to the Secbuf Component Directory table and the various other "control words" eliminated.

4.0 Network Configuration and System Commons

4.1 Capsule Module Summary

9 new modules created

6 eliminated

The Mark II approach to creating network configuration tables and data structures, essentially of hard-coding them in at assembly time and linking them with the executable code, has been completely scrapped in favour of a new design philosophy based on table generation at run-time using a suite of FORTRAN routines.

4.2 Table Generation

Figure 4-1 outlines the stages involved in the new Mark III table generation scheme. All system input parameters are now specified in a single ASCII table created and modified using the RSX-11 system editor. Table 4-1 shows the configuration file currently in use for the initial ECTN Mk III system. It comprises fifteen 60 sample/second components, including a 2400 baud 2-station multiplexed input line and a total of nine Glen Almond components. As of this writing, the first twelve lines are presently connected, and twelve components are triggerable. This table file is read by program CONFIG, which combines the input parameters with intelligence of the table structures encapsulated in the FORTRAN subroutines, to produce two output data files. These files then serve as input to the run-time initialization routines of the front end and host subsystems.

One ramification of the new table generation scheme will be greatly simplified software maintenance. Software for the proposed WCTN Mark III system will be nearly identical to that for ECTN. Most of the differences will be incorporated, in an up-front and highly visible manner, within the network configuration file.

4.3 Network Reconfiguration

Making changes to the new system such as adding or deleting channels becomes a very simple matter. The steps involved in reconfiguring the running Mark III system are:

(1) Edit the ASCII configuration file NCF.ASC to make desired modifications.

(2) Run program CONFIG to generate the new front end and host table data files.

(3) Abort and re-run the HOST task.

(4) Re-boot the LSI.

The user may carry out the reconfiguration simply by invoking the indirect command file RECONFIG. Steps 2 to 4 above then proceed automatically. Initialization routines in the front end and host will then use the updated table files and the newly-configured system will come up automatically within one minute.

4.4 Memory Layout

The Mark II static common partitions (CCTDEF, USRCOM, SECTRG) have been eliminated. Network configuration information formerly in CCTDEF now resides in the front end tables. The host-side tables now form part of a single dynamic common region, CTNCOM, outlined in Figure 4-2. (For more on dynamic regions, see ch. 3 of the RSX-11M Executive Reference manual.) CTNCOM is created afresh in partition GEN on HOST initialization and disappears automatically whenever the last task connected to it exits. Meanwhile, individual tasks (e.g., TRIGGR, STATUS), can attach and map to selected portions of the common area as required.

There are two main advantages to the dynamic versus static common approach. Firstly, the size of the tables in Host memory may now expand or contract flexibly (and automatically) as required by a particular configuration. (Note that it is not currently possible to add a single additional 60 Hz component to the Ottawa Mark II system without going through a complete RSX-11M system generation to specify larger static common partitions.) Secondly, since when the Mark III system is brought down the dynamic common automatically disappears, there is essentially no memory overhead involved. Hence whenever the ECTN system is not running, more memory will be available to other users.

5.0 Trigger Detection Subsystem

5.1 Capsule Module Summary

3 new modules created

2 replaced

7 modified

4 eliminated

3 unchanged

Much of the logic for ringbuffer access and event file generation proved too unwieldy to permit easy modification for the Mark III SECBUF structure.

5.2 Brief list of enhancements

- TRIGGR now maps to USRTBL, SATBL, and Trigger Tables in CTNCOM
- ringbuffer and event file parameters are now defined dynamically (via entries in USRTBL)
- all file creation and access directives are now run-time only
- initialization code has been collected into a new separate module
- several subroutines have been reorganized and repackaged in a more modular fashion
- ringbuffer access and event file generation logic has been greatly streamlined
- TRIGGR now prints on the console terminal the station name and trigger time for each triggered component as it is detected.

5.3 Event File Header

The event file header record structure, which was never fully implemented under the Mark II system, has been totally re-designed and fleshed out. Figure 5-1 shows the layout of the new header record format. The USRTBL and Secbuf Access Tables are copied from memory (with link addresses converted to offsets) so as to bind important system parameters and network configuration information with the time series data. Next, a Trigger Sequence List, consisting of ASCII station name, trigger time, short- and long-term average, and a slot for peak sample amplitude (not yet implemented) for each triggered component, is created and copied to the header record. (This information is printed out by the Mk III LIST program.) Lastly, hooks for additional entries (e.g., analysis summary section) have been built in.

The main advantage of the new header record structure is that downstream tasks will now be able to obtain critical information such as the ringbuffer and event file attributes, station configuration, time series storage location, and appropriate fix format from the event file itself.

6.0 Glen Almond Long Period Data Capture

6.1 Capsule module summary

4 new modules created

1 eliminated

The original algorithm for picking up GAC LP data from the current SECBUF in memory as it is created has required modification.

6.2 New features

- HOST now picks up the GAC LP data and sends it to task LPSAVE using the SDAT$ executive directive

- LPSAVE now functions asynchronously, keeping track of time and zero - filling any gaps in the input data stream as they occur
- the data is now stored directly in SRO format files to avoid down-stream re-formatting (e.g., LPCOPY is no longer required).

7.0 Summary of ECTN Software Modules

7.1 Real-Time Tasks

Appendix B contains a listing of the indirect command files required to assemble the source modules, create the object module library, and task build the 120 - odd mainline programs and subroutines which collectively comprise the ECTN Mk III software system.

7.2 Summary of Support Tasks

ABSLOD - system loader task used to load binary diagnostic files into LSI memory.

BLDABS - standalone task to convert a specified .TSK file into absolute loader format (for PROM generation).

BOSS - now issues break-through writes to the console
- re-formatted messages now include current time
- new messages added for HOST error conditions

DMPRNG - de-bugging aid producing a quick listing of front end/host error and status information obtained by reading selected records on the ringbuffer

HST - allows an operator on the host processor to interact with the running LSI front end system.

Currently implemented functions are:

BOot - boot the specified LSI front end processor

CLerr - clear accumulative error count on the LSI

DAset - change the selection of D/A output channels

HClerr - clear accumulative error count on the host

PEek - examine up to 16 contiguous memory locations on the LSI

POke - modify up to 16 " " " " " "

INDL11430 - special purpose task to enable a terminal on the host to communicate with the LSI over a DL-11E asynchronous line interface.

LBOMB - mainline task spawned by HOST to produce a formatted dump of the LSI's registers on the host console whenever the LSI front end executes a hardware trap.

LISTDF - modified to output the complete Mk III Trigger Sequence List (i.e., ASCII station name, trigger time, short- and long-term average, and maximum sample value for each triggered comp.)

LSIDLL - mainline routine spawned by HOST on startup to generate and down-time-load the LSI system code

LSIEX - DEC X/11 diagnostic system exerciser for the LSI

LSILDA - mainline routine which generates the LSI system code in .LDA format for blasting into PROM

MAKCOM - mainline routine spawned by HOST on startup to create the dynamic common region CTNCOM

MSGGEN - standalone task producing the BOSS message file from an ASCII input file

NCFLST - new standalone task producing a dated, formatted listing of the current network configuration file

RESET - new production task invoked by @DAILY which re-initializes the TRIGGR ration and status words "on the fly" - i.e., without having to cancel, abort, and re-run TRIGGR

RPAT2 - standalone task to convert a specified .LDA format file (from BLDABS) into hexadecimal, ASCII - coded COSMAC PROM blaster format

RSXPLT - a quick fix allows hardcopy plot on the LV-11

SHORT - de-bugging aid producing a formatted dump of a specified event file including sequence number, SECBUF time, and LSI active bitmap

SHOW - new production task producing a formatted listing of cumulative front end and host error counts (including line downtime) for incorporating into daily logs

SHRTDP - de-bugging aid which produces a formatted dump of the complete ringbuffer including sequence number, SECBUF time, LSI active bitmap, and LSI/host transaction processing times used to determine LSI CPU loading

STATUS - modified for Mk III level

TDZKRX - diagnostic routine toggled ino the LSI front end to test the receiver operation of the K.O. Mair DZK-11 asynchronous multiplexor cards

TDZTX - diagnostic routine run on the host processor which transmits four distinct byte patterns over the DZ-11

TIMING - performance measurement routine which toggles a bit in the DR11-C interface after performing a fixed number of instructions

- used to determine CPU loading on a host processor

Sample outputs of the error/status reporting tasks are shown in appendix C. There is currently no replacement for the ECTN/WCTN Mark II modules DMPINP and DMPSEC. Raw input data in the I/O buffers can be examined using PEEK; one second data buffers can be listed using the RSX DMP utility.

8.0 Hardware Configuration

Figure 8-1 presents a simplified block diagram of the LSI-11 front end system. The LSI backplane configuration and special circuitry that must be added to provide the remote boot and precision 60 Hz timing signals are shown in figure 8-2. This is a slightly modified version of figure 5-2 in paper I.

As mentioned in section 2.5, the availability of a second complete front end system for backup allows a more streamlined hardware configuration to be adopted in the host processor. Figure 8-3 shows the new PDP 11/34 configuration. In particular, at least one of the 8-channel DZ-11 units on each minicomputer can be re-configured to act as terminal ports.

8.1 LSI 11/23 CPU Card Jumper Selection

The 15 jumpers on this circuit card are configured as follows for ECTN:

			JUMPER
W1	Master clock	(ENABLED)	IN
W2, W3	Reserved for DEC Use		Factory installed
W4	Event line enable	(ENABLED)	OUT

W5, W6	Power-up mode selector		
	MODE 2 SELECTED	W5	OUT
	BOOTSTRAP	W6	IN
W7	HALT/TRAP OPTION		
	CONSOLE ODT MODE SELECTED		OUT
W8 - W15	BOOTSTRAP STARTING ADDRESS		
	773000 SELECTED		

W15	W14	W13	W12	W11	W10	W9	W6	W7	W6	W5
IN	IN	IN	IN	OUT	IN	IN	IN	OUT	IN	OUT
7		7			3					

For further details see the DEC "Microcomputer Processor Handbook" (1979-80), Chapter 3.

8.2 LSI BDV11-A Boot Card Jumper Selection

8.2.1 Socket Selection

The socket selection logic is controlled by jumpers W1-W4 and W9-W12 and these can be configured in seven different ways.

Group A assigns the PCR pages and socket selection.

Group B-G allow the user a choice of where the program execution begins, such as having the processor execute instruction directly from a system ROM or EPROM when power is turned on, rather than from the diagnostic/bootstrap ROM.

We use the standard configuration designated Group A, selected as follows.

W1	W2	W3	W4	W9	W10	W11	W12
OUT	IN	IN	OUT	IN	OUT	OUT	IN

8.2.2 Chip Selection

The system ROM sockets can be occupied by either 2K ROMS or 1K ROMS. The ROM socket logic uses jumpers W5-W8 and W13 to select the type of ROM that can be used on the BDV11 card.

We use type 2716 (2Kx8) PROMS which are selected by jumpers as shown:

W5	W6	W7	W8	W13
OUT	OUT	IN	OUT	IN

Appendix A outlines the procedures for burning ROM's, and figure A-1 shows the placement of ROM chips on the BDV-11 circuit card.

8.2.3 Control Registers

There are five hardware registers that are software addressable. These registers are assigned individual addresses that cannot be changed or modified.

The registers are described on pages 97-99 of the DEC "Microcomputer Interfaces Handbook" (1980). Only the Configuration Register at address 777524 needs further explanation.

The BDV11-A contains two dip switches designated E15 and E21 that permit the user to select diagnostic and bootstrap programs for execution. These two switches are configured to read 4023 octal for our ECTN use.

E21				E15								
ON	OFF	OFF	OFF	OFF	OFF	OFF	ON	OFF	OFF	ON	ON	SWITCH SETTING
B4	B3	B2	B1	A8	A7	A6	A5	A4	A3	A2	A1	SWITCH
11	10	9	8	7	6	5	4	3	2	1	0	DATA BIT
1	0	0	0	0	0	0	1	0	0	1	1	OCTAL

A1 ON Executes CPU test upon power-up or restart

A2 ON Executes memory test upon power-up or restart

A5-B1 Coded to read 00001 to select ROM BOOT

B2-B4 Coded to read 100 to select Program ROM

E21-5 When closed permits program control of the LSI line-time clock (LTC)

This switch configuration provides automatic loading and executing of the code stored in user EPROM immediately after the power up/self test sequence.

8.2.4 General

The selected diagnostic and bootstrap programs are monitored by the four red LED'S that indicate when a program fails. A green LED monitors the +12VDC and +5VDC supplies and is illuminated when power is on. The four RED LED'S flash on and off when power is switched on or when the Restart switch S2 is activated indicating that diagnostic programs for the CPU and memory are running. Should a failure occur while executing a diagnostic or bootstrap program, the four red LED'S indicate the area of the failure as shown below:

D4	D3	D2	D1	COMMENTS
BIT3	BIT2	BIT1	BIT0	TYPE OF ERROR
ON	ON	ON	ON	System Hung: halt switch on or power-up mode wrong
OFF	OFF	OFF	ON	CPU fault or configuration error
OFF	OFF	ON	OFF	Memory Error; R1 points to bad location
OFF	OFF	ON	ON	Console SLU will not Tx
OFF	ON	OFF	OFF	Waiting for response from operator
OFF	ON	OFF	ON	Load device fault
OFF	ON	ON	OFF	Secondary boot incorrect (location 0 not a NOP)

OFF	ON	ON	ON	DECNET; waiting for response from host
ON	OFF	OFF	OFF	DECNET; received done flag set
ON	OFF	OFF	ON	DECNET; message received
ON	OFF	ON	OFF	ROM bootstrap error

For further information see the Digital micro-computer handbook series called Memories and Peripherals.

9.0 Fault Diagnosis and Troubleshooting

9.1 Monitoring and troubleshooting on-the-fly

The Mark III system contains a number of error/status summary reporting programs. Problems with the outstations and transmission lines can be determined by examining the tabular listing of cumulative line error counts provided by task SHOW. The total accumulated number of missed data samples, however caused, is also printed. Task STATUS lists in tabular form the current state of the TRIGGR task. Examples of these output reports appear in appendix C.

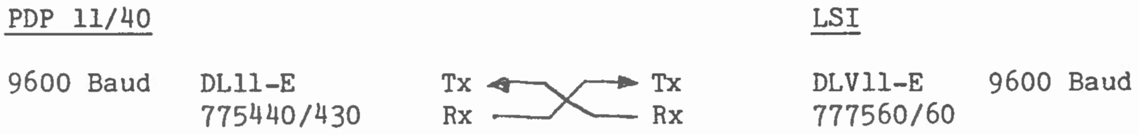
An operator may use the PEEK and POKE functions from any terminal connected to the host processor to examine and modify memory locations or device registers in the LSI front end. This requires, of course, that the Mark III system be running and inter-processor communications be established.

9.2 Console ODT

More serious problems require connecting a de-bugging terminal to the DLV-11 serial link (9600 baud) and switching the "enable/halt" switch on the LSI front panel to halt. Halting the LSI-11 causes the CPU to execute an internal (microcoded) ODT and issue an " " (ampersand) prompt character. A summary of the console ODT commands appears on pages 78-87 of the DEC "Microcomputer Processor Handbook".

9.3 The INDL11430 Program

An option has been added to the INDL11 program (INDL11430.TSK) which allows an operator on the host to communicate with the front end over a serial DL-11 link, simulating a directly-linked de-bugging terminal. This works provided a DL11-E interface card is installed in the host processor at address 775440, Vector 430 and set to 9600 Baud. The LSI DLV11-E is then connected to that DL11-E by Jumper Cable, crossing of course the transmit and receiving lines in that Jumper Cable as shown below.



PDP 11/40

9600 Baud DL11-E 775440/430

LSI

DLV11-E 9600 Baud 777560/60

These back-to-back DL devices and the INDL11430 program allow you to flip flop the terminal from Q-Bus to Unibus. This Q-Bus/Unibus DL link also facilitates loading DEC X/11 diagnostics into the LSI from the host system disk. One can load and run programs in the LSI as one of the multi-users on the host computer.

To run the INDL11 program, respond to the menu listing and prompt with a carriage return; respond to the second prompt with a control P. Your host terminal should be now connected to the Q - Bus via the DL link.

9.4 Static Tests

A number of Static Tests can be carried out using the internal ODT. To verify backplane configuration and wiring, examine the following addresses:

BDV11-A	Register	Size	Address/Vector
	Page control	16 Bits R/W	777520
	Read/Write	16 " "	777522
(Should read 4023_8)	Configuration	12 Bits R	777524
	Display (LED's)	4 Bits W	777524
	BEVNT	1 " W	777546*

* Deposit 100s to 777546 to enable clock.

DLV11-E (Set to 9600 Baud)		777560-4/60
DRV11-B		7772430-6/120
DZV11-B	1	760010/500
	2	760020/510
	3	760030/520
	4	760040/530

ADAC Channel	1	770440
	2	770442
	3	770444
	4	770446

To remote boot the LSI system from the host computer set and then clear bit #8 in the DR11-A output register. (Deposit 400_8 into 767772 followed by depositing 0 into the same register. Boot will occur on clearing bit #8.)

Note: 18 Bit addresses are required with memory management unit.

Another good use of ODT is to check the interprocessor link cables. In the host computer one can use the "Open" command to deposit and examine device registers; use ODT in the LSI system. First clear the DR11-B registers on both the host and LSI side as follows:

```
              PDP 11/40 SIDE                    LSI SIDE

          OPE 772430/XXXXXX  0             772430/XXXXX  0
                  32/        0                 32/  "    0
                  34/        0                 34/  "    0
                  36/        0                 36/  "    0

                WRITE                              READ

        Deposit 177777 into 772436         Examine 772436/177777
        Deposit    0    into   "           Examine    "      0
```

and vice versa.

9.5 Loading and running diagnostics

DEC X/11 diagnostic routines normally stored on the host system disk may be selectively down-loaded and executed in the front end using console ODT. Unfortunately, LSI-11/23 console ODT lacks the bootstrap loader facility (L-command). It is therefore necessary to manually enter the loader program before attempting to load and run diagnostics. Type in the Bootstrap Loader starting at address 157744.

```
157744/016701    LF
157746/000026    LF
157750/012702    "
    52/   352    "
    54/  5211    "
    56/105711    "
    60/100376    "
    62/116162    "
    64/     2    "
    66/157400    "
    70/  5267    "
    72/177756    "
    74/   765    "
157776/777560    CR
```

Type: 157744G ; This starts the LSI.

" Control V

CPU prints: Enter the name of the file to be transmitted:

Type: ABSLOD.DAT CR ; this loads the absolute loader. This will give you a large number of prompt characters. To save paper use a CRT terminal. After the end of file message start the LSI CPU by typing 157500G then control V. After the prompt for file name, type: LSIEX.DAT.

LSIEX is a long program and will take about one minute to load. It is self starting.

Table 9-1 presents a tentative fault diagnosis and troubleshooting guide for the front end system.

TABLE 9-1

Fault Diagnosis and Troubleshooting Guide

MODULE	SYMPTOMS	DIAGNOSIS	FIX
CPU & MEMORY	Program halt No SECBUF's received No visual outputs	Check what is still running Self-Test Diagnostics on Re-Boot	Reboot Replace cards
BDV-11	Cannot Re-boot	DEC X/11 diagnostics	Load XBBOOT from Host
ADAC's	No visual output Still rx SECBUF's	Poke values into D/A registers	Replace cards
DRV-11B	No interprocessor communication but still have visuals	Use ODT DEC X/11 diagnostics	Static tests Run stand-alone Replace card
DZK-11	Large missing data counts Bad data Many false triggers	Error/Status Reports Examine CSR's Examine IOBUF's Diagnostics	Replace cards
Precision 60 Hz down	System stalls	Check connections Use ODT to look at software clock	Fix hardware clock Restore connections
DLV-11	Used only for maintenance testing		Replace Card
Host Down	SECBUF's not received	Accumulate downtime in Error/Status Report	Run standalone

10.0 Summary

The new Mark III system is now in full production in Ottawa. In addition to the design objectives outlined in paper I, this system incorporates greatly enhanced flexibility and capacity through such mechanisms as the ASCII Network Configuration File, run-time generation of tables, the dynamic memory common, and the new event file header record. Another objective identified in paper I, that of unloading the host minicomputer system, has been achieved. Timing tests on the Ottawa PDP 11/34 show that running the full ECTN Mark III system (RSX-11M, HOST, LPSAVE, TRIGGR) and triggering on 12 components requires only 39% of CPU throughput. Linear extrapolation to 16 triggerable channels indicates a CPU free time still in excess of 50%.

The challenge now is to fully utilize this expanded capacity in designing and implementing a set of software tools that will facilitate the handling and analysis of ECTN/WCTN data by user scientists.

Appendix A

Procedures for Creating and Blasting the LSI Mark III System in PROM

1. Invoke the taskbuilder with command file [6, 61] LSILDABLD to create the output files LSILDA.TSK and LSILDA.STB.
2. Run task LSILDA. It will prompt for the LSI number and output file name. The output file will contain the same code normally loaded by the down-line-load procedure, but in .LDA format. This step requires the front-end configuration data file LSITBL.DAT.
3. Build the other ROM resident tasks (XBBOOT, MESS, LOWCOR, BADINT) by invoking the command file [6,61] BDVBLD. Note that the order of these files must be as specified in BDVBLD.CMD since several of them generate .STB files which are in turn referenced when building others.
4. Convert the tasks built above into a .LDA format file by running the task BLDABS. Since BLDABS stops prompting for input task names after it finds one containing a transfer address, XBBOOT must be the last input file name specified.
5. Use PIP to concatenate the output .LDA format files produced in steps (2) and (4). Note that the file created in step (2) must be specified first.
6. Run task RPAT2 to generate PROM images in hexadecimal ASCII format from the file specified in step (5). When it asks the size of ROM used, enter 2048. Next it will prompt for a file name for each PROM image file needed (e.g., ROMFEB810.HEX). The first record in the file is !M0000, which is a command to the RCA 1802 system.

7. Hook up a 4800 baud link between the PDP 11/40 DL-11E and the RCA 1802 system.

8. Install and run program INDL11430.

9. After the menu listing and prompt, type CR followed by CNTR P to set up communications, then CNTR T to transmit an ASCII file. The program will prompt for filename, and print a message when transmission is complete.

10. First specify the checksum program CHECK.HEX, which loads at location 1000 and verifies error-free transmission of the PROM images.

11. Repeat for each PROM image to be burned:

    - exit (CNTR C) and re-run INDL to get back to command mode
    - repeat the CR , CNTR P, CNTR T sequence
    - enter filespec for the PROM image file
    - run the CHECK program i.e., type SP1000 CR
    - if file transmission was successful (00 is printed at the terminal), enter the PROM burn command i.e., type = P CR
    - after prompt for LO and HI ADR, type 0 7FF CR
    - after about one minute, the response

      OK

      RAM ADR # 0800

      will appear if the burn was successful
    - label the PROM chips sequentially.

    Endrepeat

12. Mount the PROM's onto the user PROM slots of the BDV-11 diagnostic/bootstrap module as shown on the accompanying diagram (figure A-1).

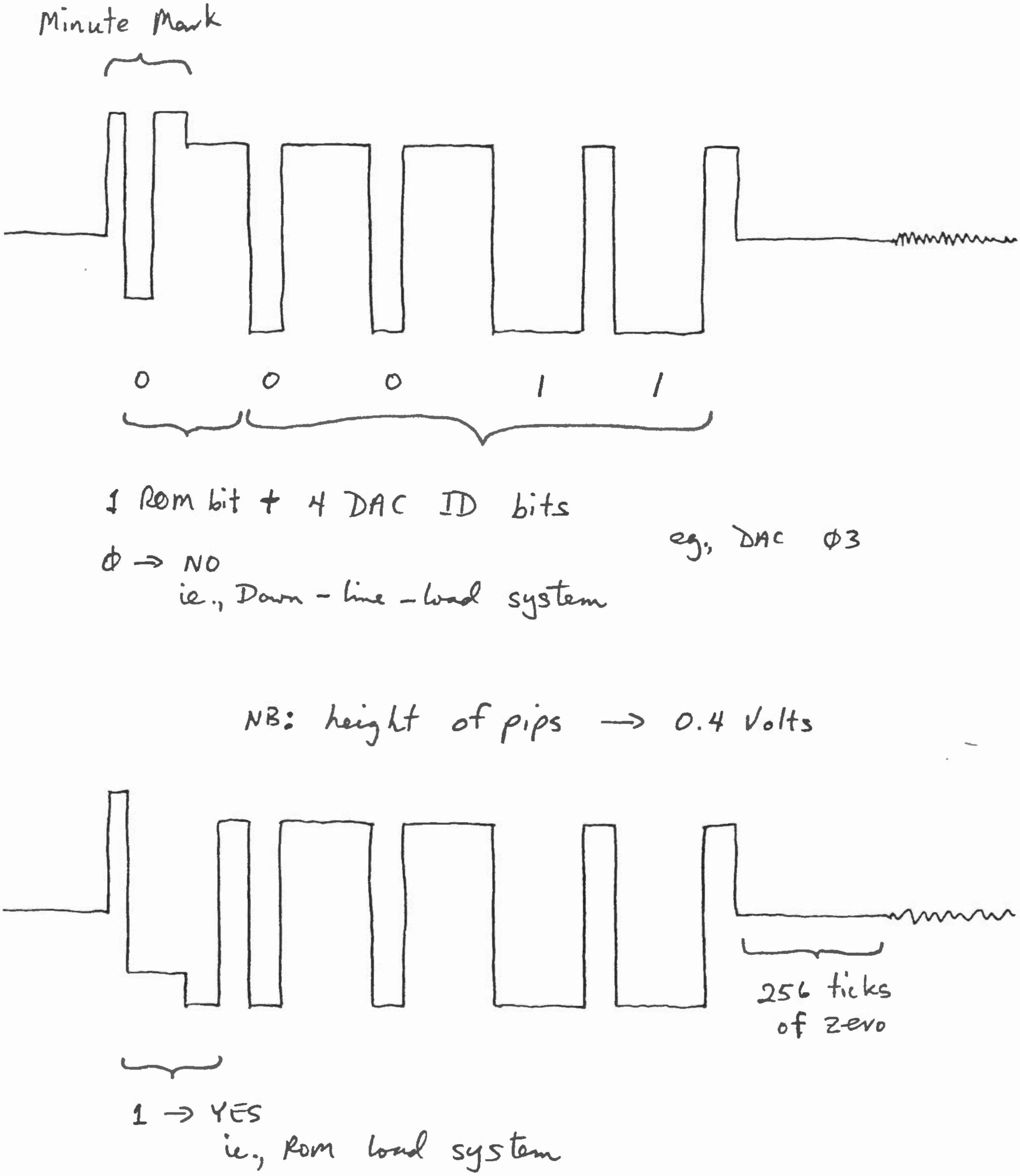
Appendix B

Listing of the indirect command files used to generate ECTN M3V1.

The following pages contain listings of the indirect command files required to assemble the source modules, create the object module library, and task build the complete ECTN Mark III software system.

Appendix C

Sample outputs of the error/status reporting tasks.



NB: height of pips → 0.4 Volts

Figure 2-1

DAC CALIBRATION AND

IDENTIFICATION PIP CODES

# WORDS		
4	ASCII 'SB' ID WORD # LSI'S IN CONFIGURATION ACTIVE LSI BITMAP SB SEQUENCE NO.	Ring Record Header (appended by HOST)
19	LSI → HOST MESSAGE BUFFER	repeat for each LSI in configuration
128	LSI ERROR / STATUS SUMMARY SECTION	
7	SB TIME BLOCK (YYMMDDHHMMSSTT)	
60	COMPONENT 1 TIME SERIES	
60	COMPONENT 2 " "	
60	COMPONENT 3 " "	
• • •	• • •	
HEROSY::	RINGBUFFER WRITE ERRORS " " TIMEOUTS	Ring Record Trailer (appended by HOST)
10	ERROR THIS SECOND FLAG HOST XB: TX ERRORS HOST XB: RX ERRORS MISSING SECBUFS HOST CHECKSUM ERRORS UNSOLICITED MESSAGES FROM LSI UNDEFINED FUNCTION CODES SPARE	repeat for each LSI in configuration
	• • •	

Figure 3-1

SECBUF STRUCTURE

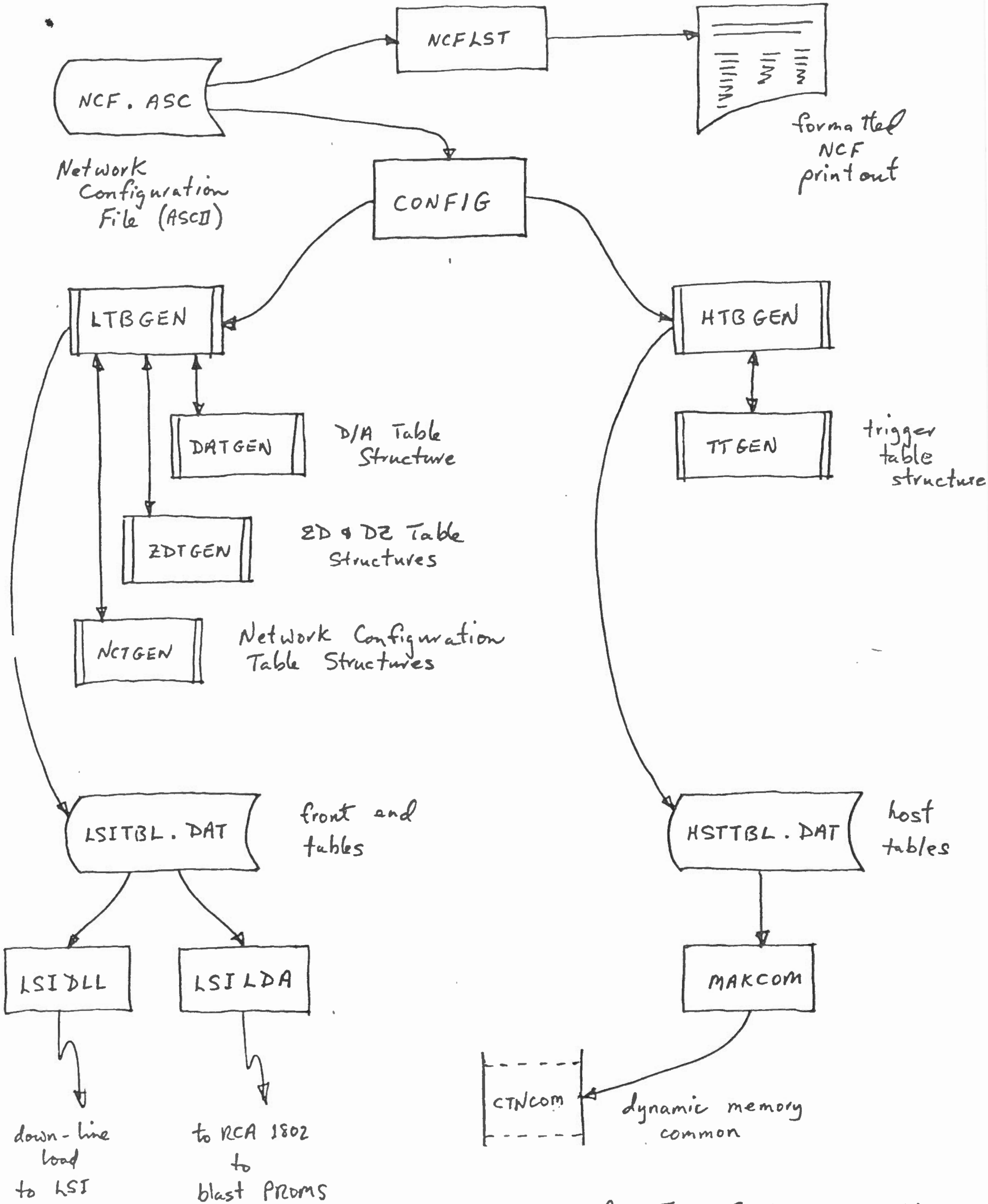


Figure 4-1 Run-Time Table Generation

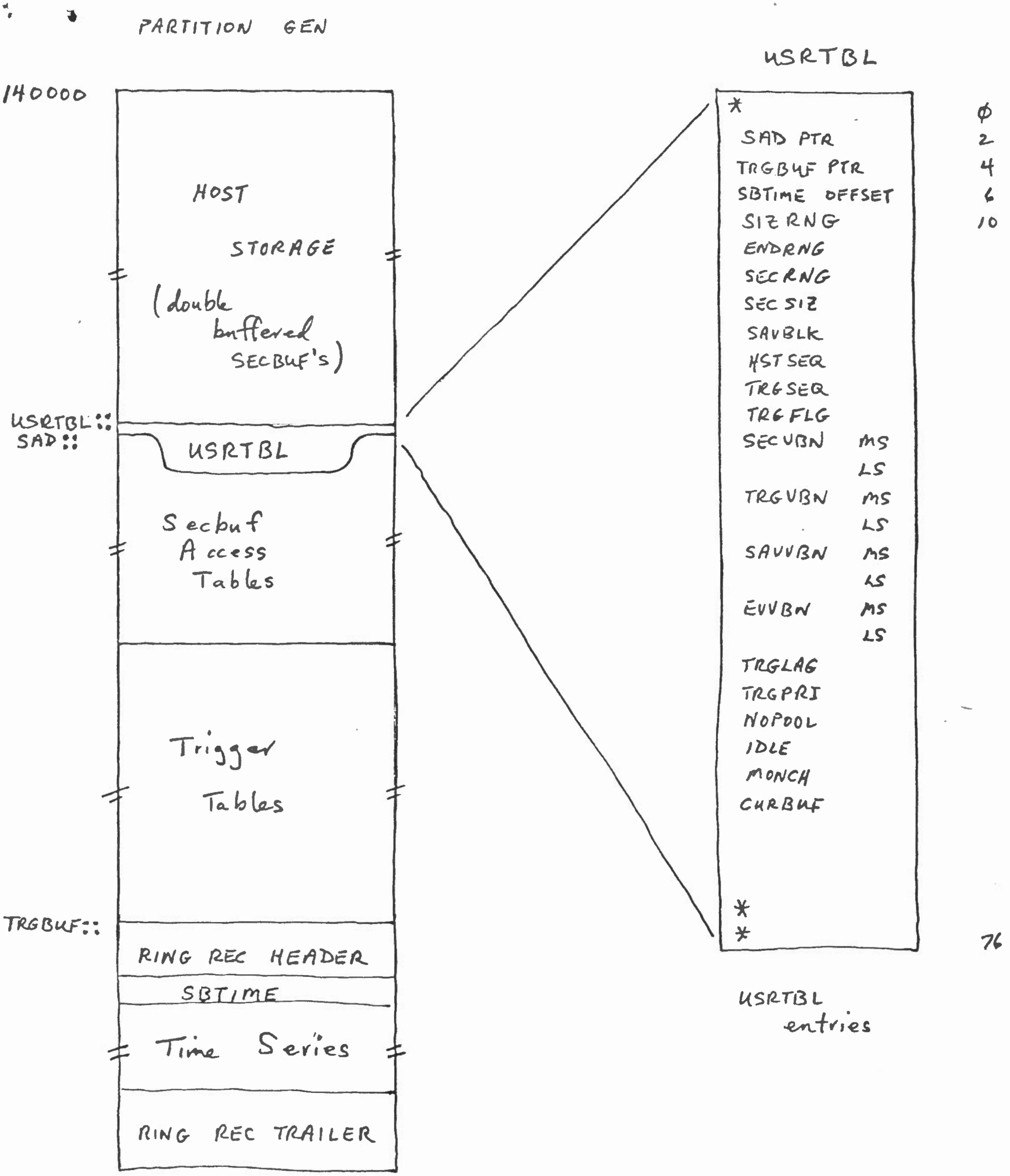
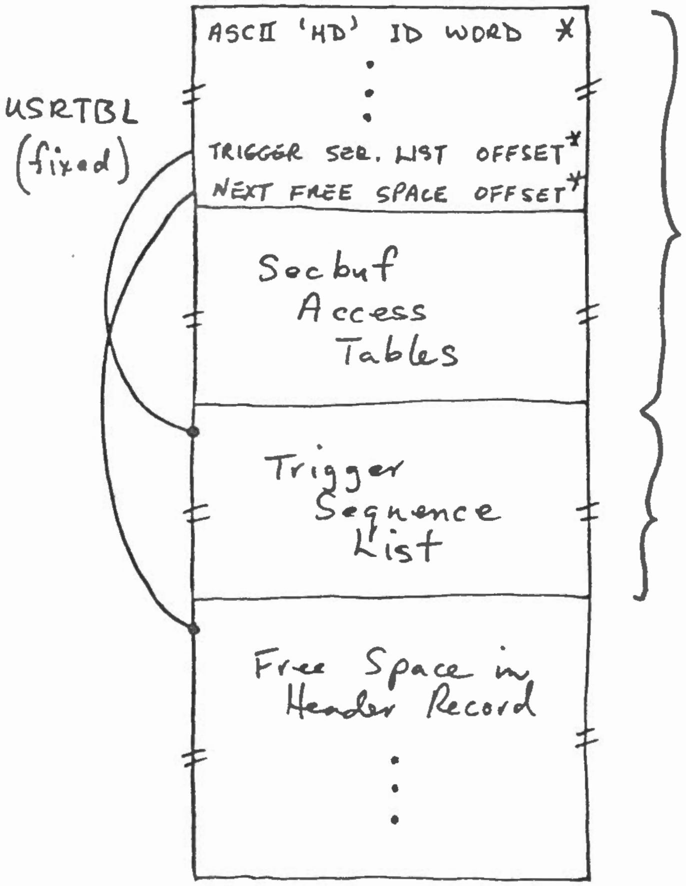


Figure 4-2

Memory Layout of CTNCOM Dynamic Common

* Reserved for use in Event File Header

HEADER RECORD



copied from CTNCOM with link addresses converted to offsets

copied from TDBASE

* Generated at run-time by HEADER module

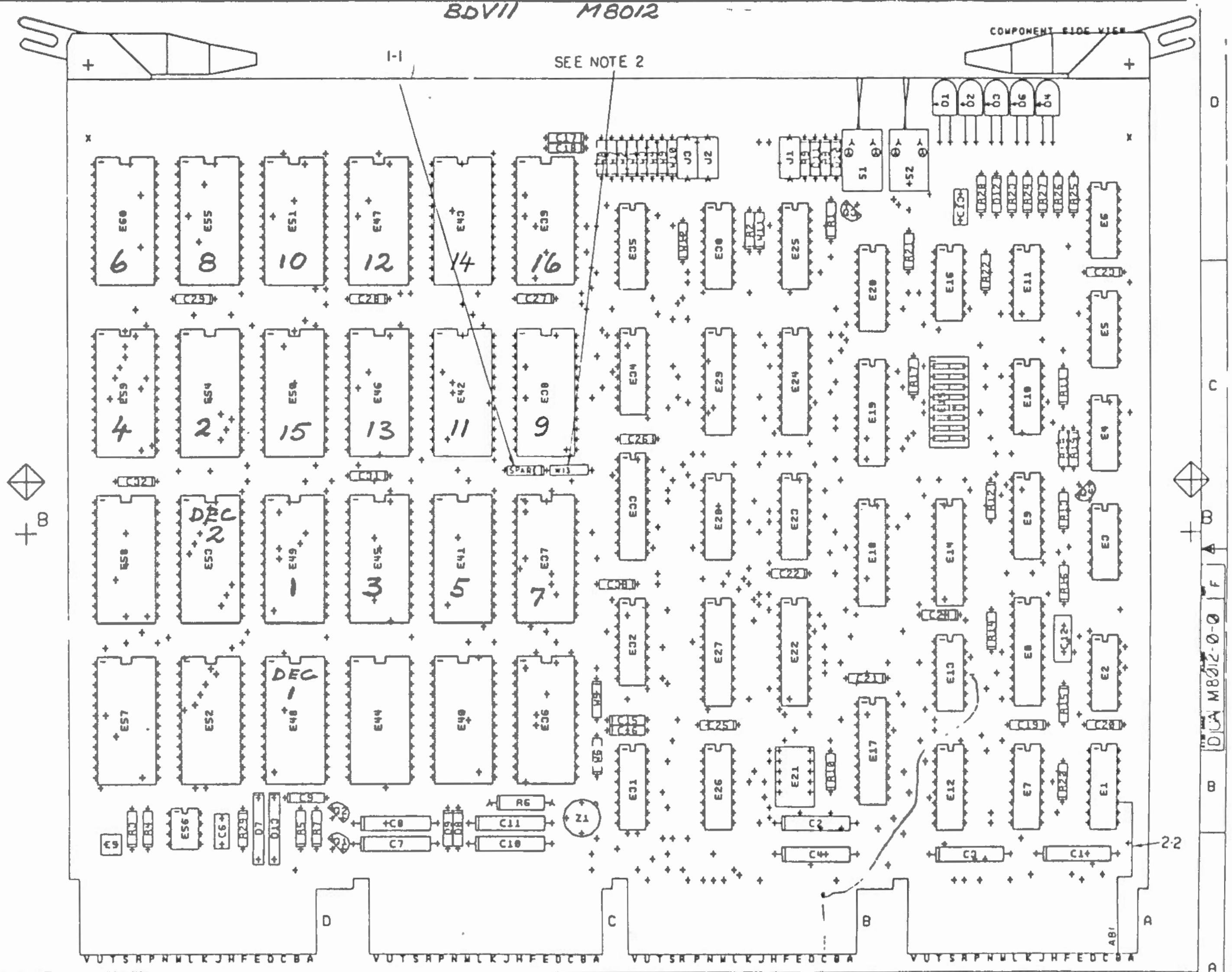
Figure 5-1

Event File Header Record Structure

DIGITAL 4K
DIAGNOSTIC/BOOTSTRAP
1) E48   2) E53

16K USER ROM

1) E49   2) E54
3) E45   4) E59
5) E41   6) E60
7) E37   8) E55
9) E38   10) E51
11) E42   12) E47
13) E46   14) E43
15) E50   16) E39



NOTES
1. 8 HOLES .065 (+.000-.005) MAYNARD ONLY
2. W14 FIGURE ARE OPTIONAL DO NOT INSTALL
[illegible] FOR E35 [illegible] E57 ONLY

ETCH REV.	C
P.C. DESIGN DATA BASE REV.	C

SIGNATURES	DATE	digital			
DRN		TITLE BOOT TERMINATOR ANC DIAGNOSTIC ROM			
CHK'D.					
ENG.	9-7-77				
PROJ.ENG.	8-7-77				
PROD.					
SCALE 2/1		SIZE	CODE	NUMBER	REV
SHT. 1 OF 4		D	UA	M8012-0-0	F
NEXT HIGHER ASSY. B-00-M8012-0					

1 MS#104054

Figure A-1 BDV-11 circuit card layout showing ROM placemen

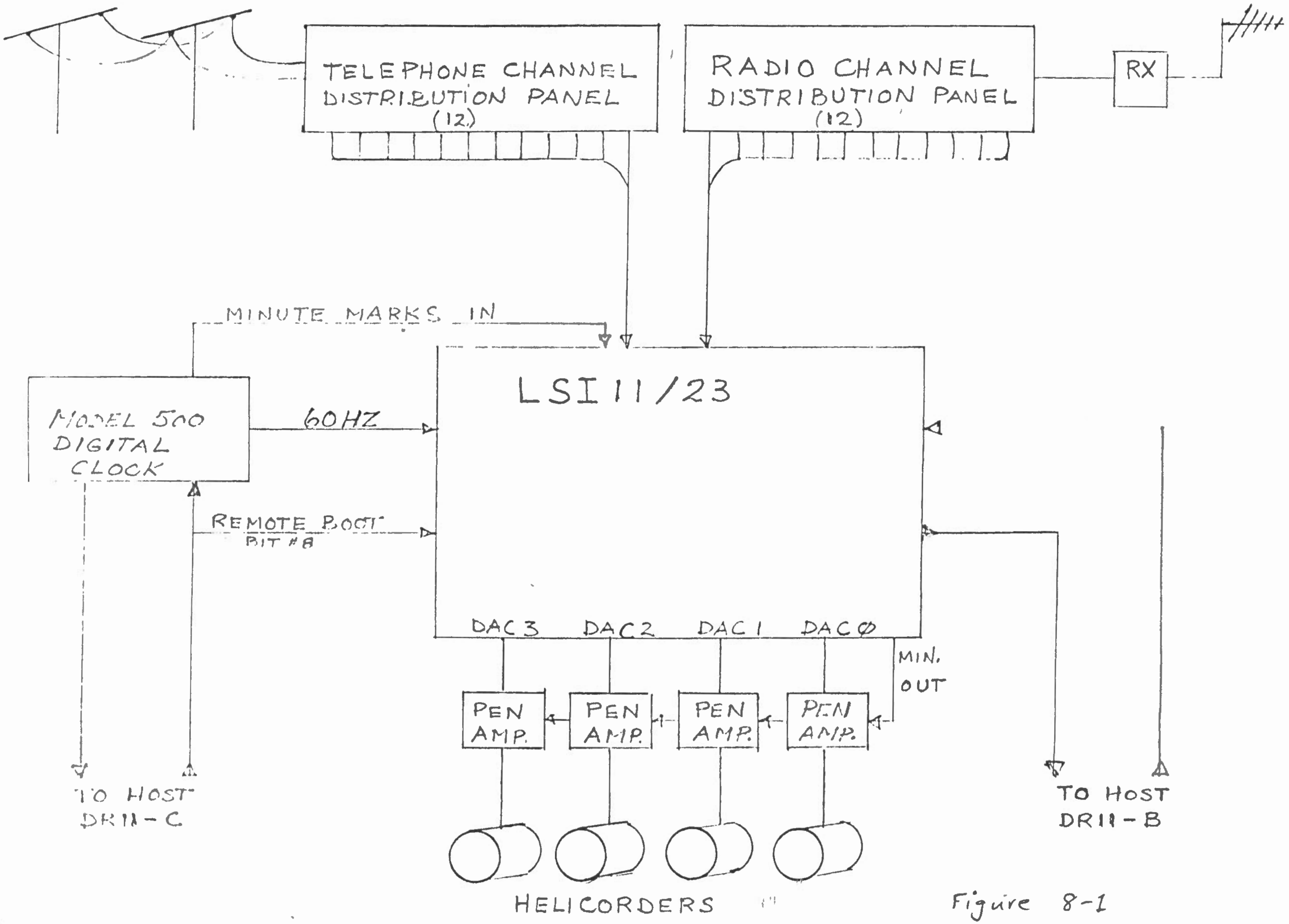
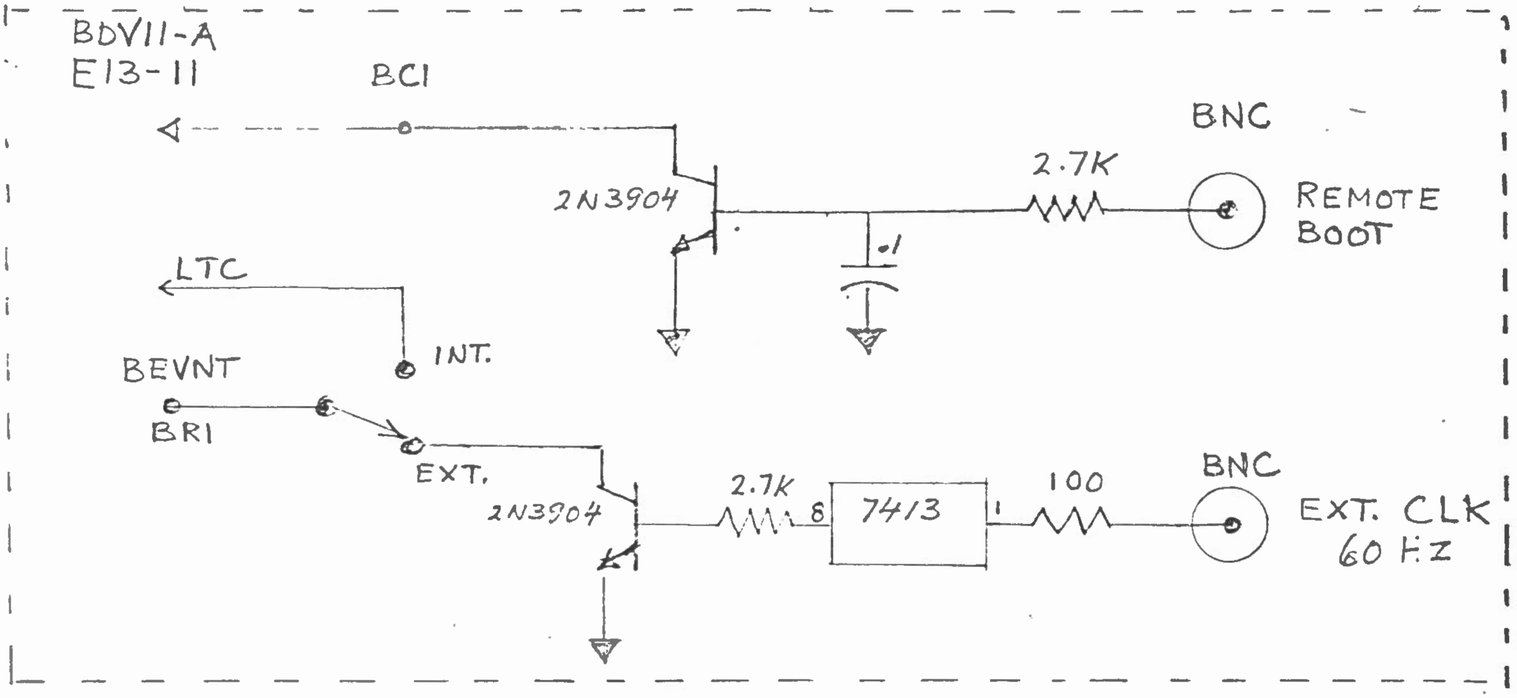


Figure 8-1

SIMPLIFIED BLOCK DIAGRAM OF LSI-11 FRONT END SYSTEM

	A	B	C	D
1	LSI 11/23 CPU M8186		MEMORY 64KB M8044-DB	
2	DZK11 #2 4 CHAN. MUX.			
3	DZK11 #3 4 CHAN. MUX.			
4	DZK11 #4 4 CHAN. MUX.			
5	ADAC 4 CHAN. D/A		DLV11-E M8017	
6	BDV11-A M8012-YA			
7	DRV11-B (ABLE) INTERPROCESSOR LINK			
8	DZK11 4 CHAN. MUX.			

LSI 11/23 CONFIGURATION ECTN



CIRCUIT TO BE ADDED FOR EXTERNAL SIGNALS

Figure 8-2 LSI-11 BACKPLANE LAYOUT AND SPECIAL BOOT/TIMING CIRCUITS

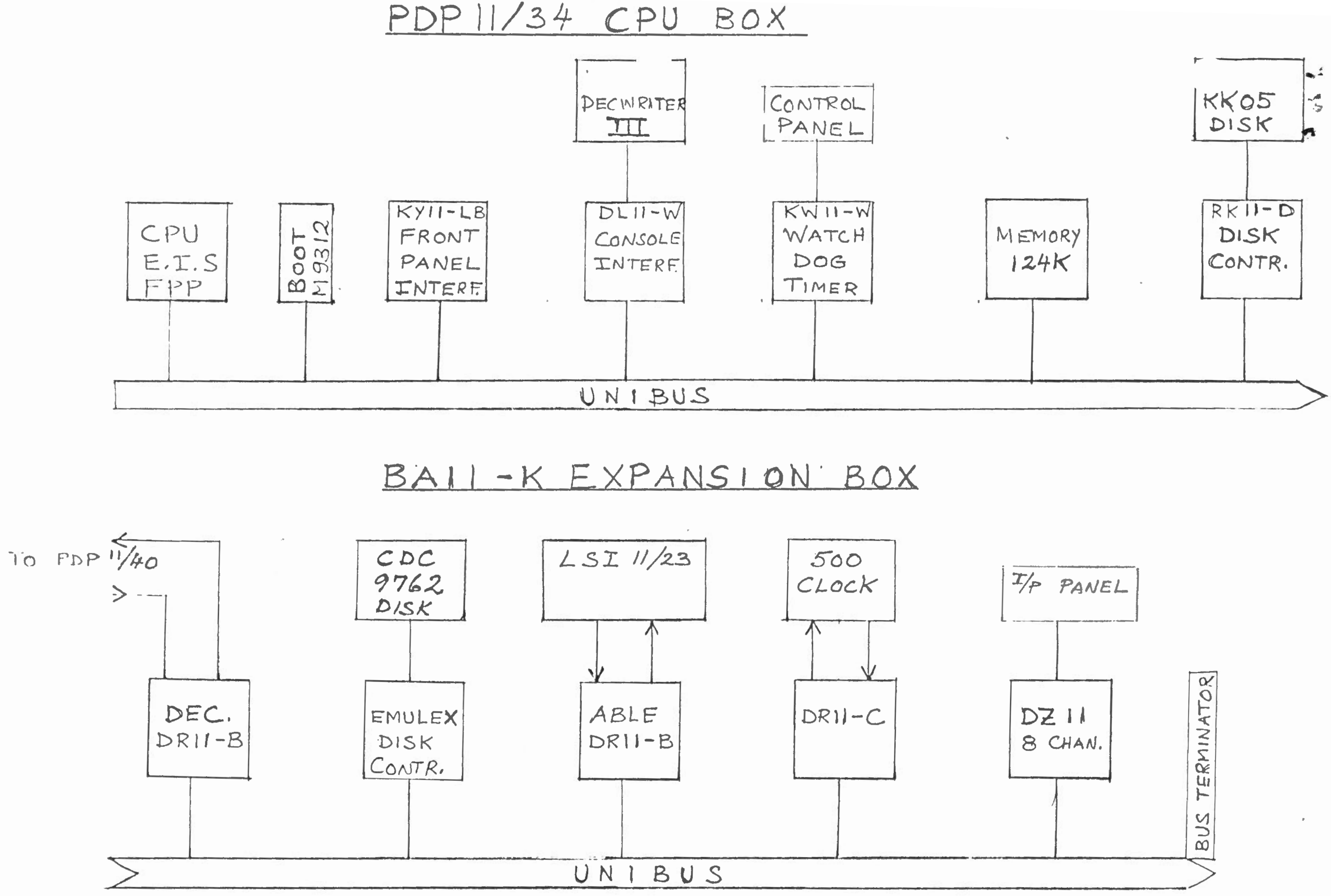


Figure 8-3 Mark III PDP 11/34 CONFIGURATION

```
                              FORMATTED LISTING OF [6,54[illegible]F.ASC      18-DEC-8021:03:10

LINE  CODE    STNAME  SRATE  FIX  DZLPR  PKTSIZ  TPP  DAGAIN  TRGLVL  RATION    FILTER CONSTANTS ...
****  ****    ******  *****  ***  *****  ******  ***  ******  ******  ******    *****************

  1     2     OTTSPZ   60     1    5530     2     1      0      4.0      8     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
  2     2     MNTSPZ   60     1    5531     2     1      0      4.0      8     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
  3     2     MIQSPZ   60     1    5532     2     1      0      5.0     20     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
  4     2     MNQSPZ   60     1    5533     2     1      0      3.0     12     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
  5     2     GNTSPZ   60     1    5530     2     1      0      3.0     20     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
  6     2     SHQSPZ   60     3    5531     2     1      2      0.0     20     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
  7     2     LPQSPZ   60     3    5532     2     1      2      4.0     20     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
  8     2     LDQSPZ   60     1    5533     2     1      0      4.0     20     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
  9     2     FHQSPZ   60     1    5330     2     1      0      2.0     23     2.06192 0.86224 1.67283 0.64000 0.00391 0.00391
 10     1     GACSPZ   30     3    5131     6     2      1      2.0     20     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
 10     0     GACSPN   30     3       0     0     0      0      0.0      0     0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
 10     0     GACSPE   30     3       0     0     0      0      0.0      0     0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
 10    34     GACLPZ    1     3       0     0     0      1      0.0      0     0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
 10     0     GACLPN    1     3       0     0     0      0      0.0      0     0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
 10     0     GACLPE    1     3       0     0     0      0      0.0      0     0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
 10    19     GACAX1    1     3       0     0     0      0      0.0      0     0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
 10     0     GACAX2    1     3       0     0     0      0      0.0      0     0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
 10     0     GACAX3    1     3       0     0     0      0      0.0      0     0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
 11     2     PHQSPZ   60     3    5532     2     1      0      0.0      8     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
 12     2     WDQSPZ   60     3    5533     2     1      0      0.0      8     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
 13    14     GRQSPZ   60     3    6130     4     1      0      0.0      8     2.06409 0.90605 2.19250 0.69836 0.00391 0.00391
 13     0     THQSPZ   60     3       0     0     0      0      0.0      0     0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
 14    14     TEST01   60     3    6131     4     1      0      0.0      0     0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
 14     0     TEST02   60     3       0     0     0      0      0.0      0     0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
```

TABLE 4-1

FORMATTED LISTING OF THE ASCII NETWORK CONFIGURATION FILE