Canadian Digital Telemetered Seismic Networks
Central Acquisition System
Version 4.1

by

J. A. Lyons

June 20, 1988

Geophysics Division
Geological Survey of Canada
Department of Energy, Mines, and Resources
Ottawa, K1A 0Y3 Canada

Internal Report
88-3

Contents

List of Tables

List of Figures

1 Introduction

The Canadian Digital Telemetered Seismic Networks (CDTSN's) are earthquake detection systems that have been developed by the Geophysics Division, GSC over the last 15 years. They consist of a network of micro-computer based outstations that sample seismic ground motion at a variety of rates between 1 and 80 Hz, digitize it, format each sample into a 16-bit word, and send it via a combination of dedicated telephone line and/or UHF radio telemetry to a central processing site. At the central site, the data is scanned by an event detection algorithm and digital files containing potential earthquake signals are created and transferred to a (Micro)VAX system for further analysis.

This report discusses the central site hardware and software portion of the CDTSN systems. The current version, dubbed the Mark 4 CDTSN system, represents the fourth generation of central acquisition system. It incorporates many of the novel features developed for the Sudbury Local Telemetered Network (SLTN), now resident at the Science North Museum in Sudbury. In addition, it incorporates the following functional improvements:

- increased processing capacity to handle up to *32* 60 Hz input channels plus 6 components from the GAC borehole seismometer
- up to *12* channels of analogue (helicorder) output
- processing of concentrator timekeeping information and incorporation of this into the output files
- for ECTN, direct processing of GAC long-period data from the memory-resident ring buffer
- output Time Series Files (TSF's) formatted in the MK2 TSF format

Most of the early conversion work was done under contract by Dan Sharon, then of Roy Ball Associates. His contribution is gratefully acknowledged.

The base level established at contract end was dubbed ECTN Mark IV, and is denoted herein as the MK4 system. Since completion of the conversion contract in July 1985, there has been continued development and improvement of the CDTSN systems. New features include:

1. support for the extended dynamic range MK3 outstations
2. provision of programmable sample rates for the portable networks, which currently use 80 Hz sampling
3. post-trigger processing to categorize detected events as possible earthquakes, calibration pulses, or noise, with provision to delete noise signals
4. automatic transfer of all potential seismic event files to an analysis (Micro)VAX
5. automatic software configuration on system startup based on new entries in the network configuration file so that precisely the same application code can run in all seven CDTSN network configurations
6. support for the Seismic Long-Period Digitizer, to provide PGC with continuous PGC LP data capture in a fashion analogous to GAC

The current level of software and hardware is formally denoted as CDTSN Version 4.1. However, the "MK4 system" will continue to be used within this report to denote the current version unless specific emphasis is being drawn to the timing of later developments.

There are currently five incarnations of the MK4 CDTSN system: the 20–station Eastern Canada Telemetered Network (ECTN) centered at Ottawa (with primary and backup processors); the 18–station Western Canada network (WCTN) based at Sidney, BC (also with primary and backup processors); the 3–station Sudbury Local Telemetered Network (SLTN) system in Sudbury; the 6–station 18–component Charlevoix network (CLTN) based at La Pocatière, Québec, and the the 6–station 18–component Pacific LTN to be installed in the west. All have essentially the same hardware configuration, and all run a common version of the software.

Section two outlines the MK4 hardware configuration; section three provides an overview of the application software and presents software structure charts; section four outlines the command procedures and data files used by the MK4 system; appendix A presents details of the software changes; appendix B describes the standard CDTSN event detection algorithm. For brevity, the scope of appendix A and much of the rest of this report has been limited to a detailed description of the changes made to the previous MK3 CDTSN software in order to bring about the MK4 system. The reader is referred to the documents: *"Overview of the proposed LSI-11 Front End Processor System"* by J. A. Lyons, Internal Report 80-9, and *"The ECTN Mark III (LSI-11 Front End) System"* by J. A. Lyons and A. Vesa. for a description of the previous MK3 system. Much of the detailed material in appendix A and the software structure charts have been adapted from the RBA document entitled: *"ECTN MARK IV System Manual"*.

2 Hardware Configuration

Figure 1 shows a generic block diagram of the MK4 hardware layout. With the exception of the memory board and Medley disk/tape unit, all components are manufactured by Digital Equipment Corporation (DEC).

The system hardware consists of the following major components.

- KDJ11-AC (PDP-11/73) CPU with floating point option and power-fail auto restart
- 2 Mbyte RAM memory (Christlin)
- KWV11-C real-time clock (to control data gathering interrupt service routine)
- EMULEX Medley storage sub-system comprising:
  - a high-performance Winchester disk with 110 Mbyte formatted storage capacity and DEC MSCP emulation using a UC03 SCSI host adapter
  - 1/4" streaming/start-stop cartridge tape with 60 Mbyte formatted capacity emulating a DEC TS-11 and media compatible with the DEC TK25 using a TC05 coupler
- 1–4 DZQ11's providing up to 16 multiplexed asynchronous serial input telemetry ports
- 1–3 AAV11-C's providing up to 12 channels of analog output
- a DLVJ1 providing 4 asynchronous serial terminal ports
- one of the following synchronous communications controllers:
  - DEQNA Ethernet controller (ECTN/WCTN LAN-based systems)
  - DMV11 controller (remote systems connected by dedicated telephone lines)
- LA-120 system console teleprinter

With the exception of the SLTN system, these boards are mounted in a Transduction Superblue box containing backplanes, power supply, cooling fans, and key-switched on-off front bezel control. The Superblue box and Medley unit are rack-mounted, either in standard 19" racks for the Lab environments or in aluminum transit cases for the portable networks. SLTN started life as an orphan PDP-11/23 (KDF11 CPU) mounted in a KOM MicroPAC chassis. However, to maintain hardware commonality, (and hence software compatibility), it has been upgraded over time to match the components of the MK4 configuration.

The lab-based systems (ECTN, WCTN) are connected to their respective analysis VAX system(s) over an Ethernet Local Area Network. Remote systems are connected in a Wide Area Network via a dedicated telephone line and 9600 bps Gandalf SuperModem. In addition, a terminal port is reserved for dial-up modem access at remote sites.

3 SOFTWARE OVERVIEW

The MK4 software system consists of approximately 4700 lines of MACRO-11 (PDP-11 assembler) code distributed among some 93 modules, and approximately 6400 lines of FORTRAN code comprising some 102 modules. In general, the assembly-level code is used for time-critical and device-dependent code sequences, while the higher-level language is used for everything else.

The MK4 system incorporates most of the novel features developed for the SLTN system, most notably, a high-performance single-processor configuration, and a memory resident ring-buffer to improve I/O efficiency and reduce disk wear and tear. In this environment, the inter-processor communication logic required to link the host and front-end processors of the MK3 system is no longer required. Hence the MK3 HOST task was eliminated, and the front-end software was repackaged into the acquisition task SECBLD (same name as in the MK2 system).

Since much of the data acquisition software is table driven, the former "front-end" tables were merged with the existing host system tables and data structures to form a new comprehensive CTNCOM dynamic common region containing all system table data. SECBLD, TRIGGR, and many of the support utilities map directly to the CTNCOM and RNGBUF dynamic commons and use the data as part of their own virtual task space.

The increased channel capacity of the MK4 system demanded a substantial increase in the size of the input buffers (IOBUF's) and analog output buffers (DABUF's) required. The new IOBUF size (see Table 1) of 8 KB now spans one entire Active Page Register (APR); i.e., too large to include in a common that must be mapped by a single APR. Hence, the I/O buffers were removed from CTNCOM and became resident in their own dynamic common: IOBUFS. It was felt that institution of a separate region presented the best solution since:

- it allows single APR mapping to the full CTNCOM
- it allows future expansion space in CTNCOM before the entire 8KB page is filled
- it provides easy access to de-bugging programs like DMPINP

Similarly, the D/A buffers have seen an increase in size to support 12 channels of D/A output [12x60x2(bytes/sample) = 2400 bytes]. Since they need not be shared by other programs, they now form part of the SECBLD task's own virtual address space.

During the software development phase, the process of removing hard-coded configuration parameters to make the software more universal was continued. New features implemented include:

- the size of a SECBUF is now calculated and rounded up to the nearest 32 word block for RNGBUF access, and SAVBLK is determined to the nearest 512 byte block for event file storage
- the size of the ring buffer is now calculated from the size of a SECBUF (SECBLK) and the number of seconds on the ring buffer (RNGSEC, read in from the NCF.ASC configuration file)

Although these modifications provide full configuration-independent operation, there are several parameters that must still be hard coded for the maximum possible channel capacity:

1. IOBUF 8KB size definitions per buffer [SECBLD, MAKCOM, (DMPINP)]
2. SECBUF 4KB (maximum possible) size definition [SECBLD, TRIGGR, DMPSEC]
3. CTNCOM 8KB (maximum possible) size definition [SECBLD, TRIGGR, CTNCOM.ICL for all support utilities]

3.1 DATA STRUCTURES

The data flow and table structures of CTNCOM are virtually identical to those of MK3 except for the entries dealing with the I/O and D/A buffers (see next section). Similarly, the overall structure of a SECBUF has been preserved.

3.1.1 CTNCOM

The MK4 system configuration tables incorporate all the SLTN changes necessitated by conversion (actually reversion) to single processor operation: "front-end" and "host" tables have been combined into a single dynamic common, interprocessor support removed, SECBUF pointers adjusted for the new memory resident ring buffer location (APR 5 mapping), and the Error Summary Table (ERSUMY) is now resident in CTNCOM. With the move to 32-bit missed sample counters, the other UART status fields instituted in the MK3 system were eliminated (parity, carrier detect, framing, data overrun). These proved to be so little used that it was judged not worth expending the space and processor cycles required to maintain them.

As mentioned, in the MK4 system the I/O and D/A buffers no longer reside in CTNCOM due to their increased size. In order to provide a better understanding of the CTNCOM structure, the following breakdown describes the various tables comprising the dynamic common:

Table Name	Comments
SBTBL	one of; has pointer to ERSUMY and a pointer to a SECBUF
DATBL	one of; has pointers to D/A buffers and currently monitored components' SCD's
ZDTBL	one of; pointers to DZ tables, I/O buffers, and various ISR parameters
CCTBL	one of; pointers to each channel's (4 chans/DZ) configuration table (CCT'ch)
DZTBL	one for each DZ unit; pointers to DZ registers and some parameter data
CCT'ch	one for each serial channel (4 per DZ unit); pointers to other data structures for this channel (PMT, SDT, SCD'cmp), and some DZ and channel parameters. (Channel Configuration Table)
PMT'ch	one for each channel; packet synchronization parameters and pointer to PAKBUF's (Packet Management Table)
SDT'ch	one per Longitudinal Code; pointers to next SDT and first SCD of this channel (SECBUF Dispatch Table)
SCD'cmp	one per component (up to 4 components per channel.); pointers into SECBUF locations for this component, link

	pointers and DA gain for displayed components (SECBUF Component Directory)
PKTBUF	one per channel; stores raw bytes of a packet before (data) synchronization
ERSUMY	one of; contains counters for missing data on an active channel (one per component)
USRTBL	one of; pointers to SATBL and SECBUF, as well as RNGBUF head/tail parameters and data, and system parameters
SATBL'cmp	one per component; contains station name, chan. _& comp. number, static time correction (ms), as well as SECBUF access information (SECBUF Access Table)
TTBL'cmp	one per (short period) component; contains trigger and filter (DSP) parameters and data (Trigger Table)

Table 2 shows the makeup of the MK4 CTNCOM and Figure 4 gives a generic layout. Of the various linked lists and tables of CTNCOM, 4 structures were modified for the new implementation (not including changes common with SLTN):

- DATBL - size increased to accommodate the required 12 channels (from SLTN's 4)
  - D.IN and D.OUT buffer pointers now initialized by SECBLD to point to the D/A buffers now resident in that task's own virtual (ZDISR) space (previously initialized via CONFIG)
- ZDTBL - all pointers used by ZDISR that reside in this structure are now initialized by SECBLD (instead of CONFIG) to point at the various buffers (I/O and D/A)
- SATBL - added an entry that contains the static time correction for the component (value is read in from the NCF.ASC configuration file)
- USRTBL - removed entries CURBUF, FUNDER, MISTSB (not applicable in the single processor FPP environment), and added 3 more Mask words for component monitoring: MASK2, MASK3, MASK4 for a total of 64 monitorable components.

3.1.2 RNGBUF

The memory-resident ring buffer (RNGBUF) comprises a concatenation of SECBUF's, where each SECBUF contains one second of data from all seismic components in the system. Figure 5 shows the structure of a SECBUF, along with pointers used to access the data; the header and the time tag portions are fixed in size and are unchanged. The data portion is of variable length depending on the number of components processed.

New with CDTSN MK4 is auto configuration of the size of a SECBUF and the size of the RNGBUF. The CONFIG program rounds up the size of a SECBUF to the nearest 32 word boundary, and

computes the RNGBUF size from the SECBUF size SECBLK and RNGSEC, a new configuration parameter giving the desired ring buffer size in seconds.

As seen in Table 3, the size of a SECBUF for the MK4 system is approximately 4KB.

3.2 Program Descriptions

This section provides a high level look at the tasks that make up the data acquisition, event detection, and system configuration portions of the CDTSN code. Those programs that comprise a multitude of subroutines are given in a hierarchal breakdown to more clearly show their internal structure. The verbal descriptions emphasize new features and changes from the MK3/SLTN systems.

3.2.1 SECBLD

The SECBLD task performs the following functions:

- unscramble serial data input streams from simple and multiplexed outstations (up to four per serial channel) and time-tag it
- produce one-second data buffers (SECBUF's) and manage a 5-minute memory-resident ring buffer
- accumulate counts of missed data samples (outstation downtimes)
- produce up to 12 channels of analog output

As in the Sudbury system, the changeover to a single processor configuration permitted simplification of the data acquisition program, including elimination of support for inter-processor communications and the software clock. SECBLD now writes directly to the memory-resident ring-buffer (RNGBUF).

The hierarchal structure of the program is unchanged from SLTN. The program is an RSX non-checkpointable, privileged task. To support non-60 Hz sampling rates required for the portable networks, the data acquisition interrupt routine ZDISR ... (see following structure charts). SECBLD maps to all the dynamic commons from its INIT routine, and uses the RSX connect to interrupt system service (CINT$) to establish interrupt servicing, including a fork level routine used to "wake up" the task level program.

In addition to the SLTN changes to SECBLD, the MK4 implementation requires three further modifications:

1. the I/O buffers receiving the raw input data now reside in their own IOBUFS dynamic common (to facilitate data sharing with the program DMPINP),
2. the D/A buffers became part of SECBLD's own virtual task space (since only SECBLD uses them) residing in ZDISR for D/A output,
3. the interrupt handling had to be modified to interact with the new KWV-11 hardware (as was the program CLOCK).

It must be noted that the MK4 version of SECBLD uses seven of the eight available active page registers (APR's) for mapping the task, as follows:

```
APR0 and APR1 for task and D/A buffers
APR3 and APR4 for IOBUFS common
APR5          for RNGBUF access
APR6          for CTNCOM access
APR7          for IOPAGE access
```

Figures 6 through 11 present software structure charts for the main modules that comprise the SECBLD task.

3.2.2 TRIGGR

The event detector program performs the following functions:

- access 5-minute memory-resident ring buffer
- for each vertical seismic component
  - perform recursive digital bandpass filtering
  - compute running short-term and long-term averages (STA, LTA)
  - turn trigger condition on when STA/LTA ratio exceeds a threshold value
  - turn trigger condition off when STA falls below the LTA
- produce digital event files containing data for the entire network, including a pre-detection leader and post-detection trailer, when the minimum number of components have triggered to save an event

SLTN modifications to this program consisted primarily of changes required to bring about operation on a memory-resident ring buffer. The MK4 system retains those changes and adds small modifications required to deal with the new SECBUF size.

The most sweeping new change to this program is the removal of all FIS and support code. Previous versions of TRIGGR contained parallel sections of floating point code since it had to run on both a PDP-11/34 with floating point (FPA-11) hardware, and an older PDP-11/40 with only the more limited FIS instruction set. A sophisticated trap-handling mechanism was incorporated to determine at runtime which processor type the code was executing on, and branch accordingly. Since extended floating point (FPP) instructions are now an integral part of the MK4 processors, this special code is no longer required.

Figures 12 through 17 present structure charts giving a hierarchal description of the modules comprising the TRIGGR task.

3.2.3 CONFIG

This program automates the software configuration process by reading an editable ASCII configuration file and creating binary output files containing all the tables and linked list data structures required by the data acquisition, event detection, and support utilities. It runs as an off-line task that generates two arrays, one containing the "front-end" tables, and the other containing the "host" tables. The use of separate output files has been retained from the previous release.

The MK4 implementation of CONFIG incorporates all the SLTN modifications necessitated by the single processor operation. Additional changes were required for the new system to deal with the new locations of the I/O and D/A buffers and with the auto-sizing of a SECBUF and the RNGBUF. The sizes of various internal arrays were also enlarged to accommodate the increased number of channels processed, and the two output arrays have been dimensioned to fill up the 8 KB of CTNCOM dynamic common (unused areas zero filled). This is to facilitate expansion of CDTSN systems in the future without the need for further software modification.

Later changes to this module were required to support the portable networks and the Seismic Long Period Digitizer for PGC.

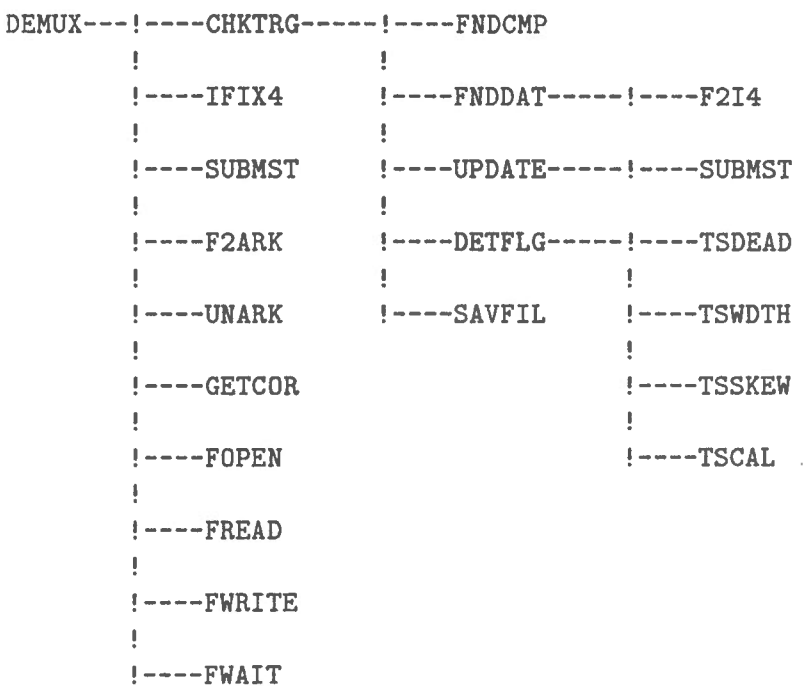
Figures 18 through 20 present structure charts showing the modular breakdown of the CONFIG program.

3.2.4 DEMUX

The DEMUX program has been greatly re-worked and expanded to include the following functions:

- perform post-detection processing to classify triggers as signal, calibration pulses, or noise
- maintain a detection log file of trigger statistics (CURRENT.DLF), including peak amplitude of calibration pulses
- delete event files classified as noise provided the flag file [1,200]NOISE.DEL exists
- convert event files into demultiplexed Time Series Files (TSF's) in standard SAM Mark II, BGR TSF format
- spawn operating system network transfer software to send the file to the analysis (Micro)VAX

The following indicates the subroutines connected to DEMUX:



Briefly, the Mark II TS File structure differs from the previous MK1 format in the following ways:

1. the data representation of entries in the header fields has been converted to 32-bit I*4 (or, where appropriate, R*4). This removes restrictions on such values as number of samples (86,400 for GAC LP files) and non-integral sampling rates (ie., less than 1 per second)

2. data samples are still represented in the traditional 16-bit binary gain ranged (BGR) representation, but the data-valid bit has been re-defined as an extended exponent bit to handle the increased dynamic range of the Mark III outstations

3. the blocking factor has been increased to 2048 bytes (ie., four times Mark 1), which represents a reasonable compromise between large-block I/O efficiency and wasted space

4. several new fields have been added to both the file and component headers that allow more complex processing and better data identification:

   - network name (e.g., 'ECTN','WCTN','SLTN','CLTN','PLTN')
   - file format ('MK02')
   - data format(BGR/I*4/R*4/I*2)
   - time correction
   - trace processing history
   - 5-character ascii station name

See the document entitled "MKIITSF.MEM" for a complete description of the current TSF layout.

Another feature added to DEMUX is the addition of software to handle time correction data: the static correction is simply read in from the input file header's SATBL's, and a subroutine (GETCOR) was added to find the dynamic time correction (if any) for each component in the input file. The routine essentially searches for a repeat of the component's name with a TIM extension to indicate a timekeeping 'pseudo' component, and saves its SECBUF access pointer for later use in the actual extraction of the dynamic correction data.

3.2.5 LPSAVE

Until recently, only the ECTN systems performed continuous long-period data saving of GAC LP data. However, the Lab has developed a Seismic Long-Period Digitizer designed to digitize the PGC LP signals and present them to the WCTN processor in a fashion analogous to GAC LP data. In order to handle this new source of LP data, the LPSAVE program was generalized to determine whether LP data saving is required based on the ascii network name now contained in the memory-resident tables. Also, the time of opening a new data file was changed from 9:00 EST to 00 hours UT for consistency between OTT and PGC.

The MK4 version of LPSAVE differs from the earlier version in the following ways:

- it processes data directly from the (memory-resident) ring buffer
- it runs only once every minute
- it runs independently as a checkpointable task

3.2.6 SUPPORT UTILITIES

The following is a list of the suite of utility programs that form an integral part of the CDTSN software. It provides a high level description of each program and outlines the changes undertaken since the MK3 version:

ADRMON
- monitors the status (contents) of a range of addresses in the I/O page
- used mostly for debugging and testing devices

BOSS
- provides a centralized error reporting facility for the real-time tasks (SECBLD and TRIGGR)
- processes status codes and messages queued by the MSGSND subroutine

CHGPAR
- selectively change system parameters on-the-fly
- increased maximum number of component masks handled to 64

CLERSM
- clear the memory-resident table of accumulated downtime (missed sample counts)
- formerly a HOST-initiated action, now a stand-alone task that maps directly to the ERSUMY table

CLOCK
- reset and maintain system time in synchrony with an external GOES satellite time receiver connected to serial port TT1:
- slightly changed to interact with the new KWV-11 real-time clock source of interrupts

CTNMON
- monitor operation of the critical real-time tasks, free disk space, etc.
- modified to control the optional green/yellow/red status light display panel via serial port TT2:

CULLEV
- free disk space by deleting "low priority" events when disk storage space becomes critically low

DAC
- change analogue helicorder channel and sensitivity settings
- formerly a HOST-initiated action carried out by the "front-end", now a stand-alone task that maps directly to CTNCOM
- modified to specify seismic components by ascii name rather than sequence number

DMPINP
- displays a snapshot of the raw data input buffer IOBUF_1 on-the-fly
- modified to access the new IOBUFS dynamic common region, and optionally accept a parameter on input specifying a particular channel to monitor

DMPSEC
- utility to dump the one-second data buffers
- modified to read SECBUF data directly from the memory-resident RNGBUF, handle a larger size SECBUF (4KB), and to use the new universal CTNCOM mapping

DMPTAB
- a concatenation of the DMPLTB/DMPHTB programs, which prints out the contents of the binary configuration files LSITBL.DAT and HSTTBL.DAT produced by the CONFIG program

LINMON
- maintain a disk-resident file (LINEQUAL.DAT) of accumulated downtime (missed data samples) by component

- modified to support 32-bit missed sample counters

LINSTS
- format and print the contents of the missed sample disk file
- modified to support 32-bit missed sample counters

LISTDF
- produces a list of any non-demultiplexed (EVENT0.DAT) files remaining in EV:[1,200], displaying triggered stations and times, as well as a count of all primary and secondary triggers for each station

MAKCOM
- called on system initialization by the SECBLD task (via RQST$) to create and populate the CTNCOM memory-resident tables and data structures using the configuration files LSITBL.DAT and HSTTBL.DAT

MSGGEN
- generates the direct access file MSGG.DAT for task BOSS from the ascii source file MSGG.ASC
- note that the file UIC's were changed from [6,54] to [6,4]

NCFLST
- produces a formatted listing of the ascii Network Configuration File (NCF.ASC)
- slightly modified to show new RNGSEC parameter (number of seconds of Ring Buffer storage), and to show the new static Time Correction for each component in the network.

RESET
- reset on-the-fly (usually daily) a fixed subset of the system parameters to their initial state as defined in the configuration file to permit continuous system operation

SHOW
- display the memory-resident table of accumulated downtime (missed sample counts) by component
- must now be invoked via "RUN SHOW" rather than just "SHO" to avoid conflict with the DCL SHOW command
- modified to support 32-bit missed sample counters

STATUS
- print a snap-shot report of current status for all triggerable components plus the current ring-buffer positions of SECBLD and TRIGGR

TIMCMP
- test program that, at the time of a Minute Mark interrupt from the GOES satellite receiver, checks RSX time for ss:tt = 00:00, and prints out the difference if the time was not as expected
- modified to interact with KWV-11 as a source of interrupts

TIMER
- new program, still under development, to replace the CLOCK program for the portable networks
- will reset and maintain system time in synchrony with both an external GOES satellite time receiver connected to serial port TT1: and a Model 501 clock connected to port TT2:
- when GOES time is synchronized to the satellite, will periodically rate and update the 501 clock

TIMING
- generate a square wave on one of the D/A output channels for use in the system loading test

TIMMON – program to test time tagging of SECBLD

– runs once per second to monitor the TIC field (60^{th} sec) in an IOBUF's time header, and prints a diagnostic message if the value is non-zero

TRGLST – produces a list of all DEMUX'd event files in EV:[1,201], including triggered station names, trigger times, primary and secondary triggers for each component, etc.

– modified to work with the new MK2 TSF structure, to handle up to 35 triggerable components, and to produce a more compact listing format

TSFDMP – produce a sample by sample listing of data from a selected demultiplexed event file for selected stations and selected times, usually for de-bugging

3.3 CDTSN SOFTWARE GENERATION

The procedure for re-assembling/compiling and linking all or selected portions of the CDTSN software system has been encapsulated in a suite of indirect command files. The highest-level module, CTNGEN.CMD, provides a menu-based user interface with prompts for required input. Options include:

1. building the entire system
2. building only SECBLD
3. building only TRIGGR
4. building all support tasks
5. building selected support tasks

The following sub-menus provide options for:

- building from source files
- building from object library
- assigning disk drives and UIC's for some object,list and task files

NOTE: it possible to create only desired types of files by simply specifying the NULL device (NL:) for unwanted files.

CTNGEN, in turn, calls various lower level command files to execute the selected options:

SECBLDMAC.CMD - assemble all modules comprising SECBLD. All object modules go to [6,5], except DBASE.OBJ which goes to [6,4], since the latter contains no executable code and is more difficult to incorporate into an object library

TRIGGRMAC.CMD - assemble all modules comprising TRIGGR. All object modules to [6,5], except TDBASE.OBJ to [6,4]

NOTE: SECBLD and TRIGGR share some common definitions of data structures. These definitions are found as macros in the Macro Library file [6,4] CTN.MLB, which is referenced in the assembly command files of SECBLD and TRIGGR.

SUPPRTMAC.CMD - assemble all MACRO-11 support task modules

SUPPRTF77.CMD - compile all FORTRAN-77 support task modules

When a particular support task is selected for building, CTNGEN determines whether it is a MACRO or F77 source file and proceeds to assemble/compile it as required. The object module is then used to update the CTN.OLB object library. The object file in [6,5] and list file in [6,4] are then optionally deleted. When the entire system is rebuilt, all object files are merged into one, the old CTNMK4.OLB object library is deleted, and a new one created from the merged object files.

By convention, each task in the CDTSN MK4 system has its own task build (link) command file of the form:

```
[6,1]'task_name'TKB.CMD
```

which is called internally by CTNGEN.

4 Auxiliary Files

This section outlines the indirect command files and auxiliary data files that are required to complete the CDTSN software package.

4.1 Indirect Command Files

This section summarizes the indirect command procedures required to configure and operate the CDTSN systems. *Note:* software generation is dealt with separately in section 3. With the exception of STARTUP.CMD, which resides in UIC [1,2], all command files reside in [1,200], the default CDTSN operator UIC.

STARTUP.CMD - command procedure invoked by RSX automatically on machine startup
- now opens and parses the NCF.ASC file to determine the network name and primary versus backup operational status to perform the following:

  - uses network name to determine the appropriate DECnet node name and brings up the proper flavour of DECnet
  - determines which timing sub-system (GOES clock or model 501) should be used
  - determines whether to initiate automatic event transfers (primary system) or to pool detected event files locally (secondary system) by installing or removing a private copy of the Indirect Command Processor utility (ICP)

CTNDWN.CMD - bring down the CDTSN application software

CTNUP.CMD - bring up the CDTSN application software
- selectively installs, schedules, and runs the various tasks
- includes (pseudo) device assignments previously made by CTNASN.CMD

CTNXFR.CMD - spawned by DEMUX to create a command file CTNXFR.COM containing the complete file specification of a newly-detected event and submit it to the (Micro)VAX hub for further action

DAC.CMD - run by DAILY to reset the D/A channel assignment and sensitivity settings

DACDAILY.CMD - run by DAILY to cycle through the various input channels and produce a short sequence of D/A output on a single helicorder

DAILY.CMD - invoked either automatically via re-scheduling of DAILY.TSK or manually by an operator on a daily basis to de-brief a CDTSN system and reset various operational parameters

ERROR.CMD - run by DAILY to produce an error log summary

FF.CMD - invoked by DAILY to format output listings by providing a form feed

PRIMARY.CMD - designate the system as primary so as to begin automatic event transfer

SAVE.CMD - run by an operator to save (ie., rename to a separate directory) selected events and optionally clean up (delete) the rest

SECONDARY.CMD - designate the system as a secondary processor so as to disable automatic event transfer

4.2 Data Files

The following describes the various ascii and data files required by the CDTSN MK4 system.

[001,200]CTNMON.TMP - temporary file used by CTNMON.TSK to determine disk free space
- normally, these files are deleted by CTNMON and should not appear

[001,200]CTNDALOG.ASC - daily log report form printed by DAILY

[001,200]CTNWKLOG.ASC - thrice-monthly report form

[001,201]CURRENT.DLF - new detection log file created by DEMUX listing triggered event_id's, triggered station-component, classification of trace (Signal, Noise, or Calibration), trigger classification flag, maximum absolute value within a 40-second window centered on the trigger time, and, if not deleted as noise, output filename
- the correspondence between trigger flag and Frank Anglin's old keys from TRGIDC.FOR is as follows:

```
ID = 2       <--> 01 00 00 00   Dead trace

ID = 111111  <--> 00 3F 00 00   Measure of kurtosis in time

ID = 11111   <--> 00 00 1F 00   Measure of skewness in amplitude

ID = 111     <--> 00 00 00 0F   Calibration pulse
```

Note: any value less than 0000000F (i.e., 15) is classified as signal

[006,004]HSTTBL.DAT - "host" tables created by CONFIG.TSK

[001,200]LINEQUAL.DAT - direct access file created by LINMON.TSK containing a current summary of outstation downtimes (missed data samples)

[006,004]LSITBL.DAT - "front-end" tables created by CONFIG.TSK

[006,004]MSGG.ASC - ascii file containing error messages reported through BOSS via the MSGSND Macro

[001,200]MSGG.DAT - binary, direct access file generated by MSGGEN.TSK containing error messages reported through BOSS via the MSGSND Macro

[006,004]NCF.ASC - the ascii Network Configuration File containing the network configuration an operational parameters used to start up and control the CDTSN systems
- has been augmented with the following new entries:

- system time base (= raw interrupt rate from external GOES/Model 501 clock)
- real-time clock interrupt rate (=rate of data gather interrupts)

the ratio of these two denotes the number of raw interrupts to be counted by the real-time clock before interrupting the CPU to gather data

- static time correction
- nominal velocity sensitivity

Note: these last two are incorporated into the TSF by DEMUX

[001,200]NOISE.DEL - if present, signals DEMUX.TSK to delete events

A SOFTWARE MODIFICATION DETAILS

This section provides a more in-depth explanation of the changes made to update the CDTSN software to the MK4 level. Program modules that required modification are listed and the changes to the parameters, variables and structures of each module are detailed.

A.1 SECBLD

DAFILL - removed the embedding of the Minute Mark into the D/A data stream, in favour of setting a Minute Mark bit in a DZQ11 register for separate signal generation. This allows the helicorder signals to be high-pass filtered without interfering with the time mark pulse shape (see also discussion under DBASE)
- added an instruction to do 2's complement inversion of the D/A data to compensate for the inversion required in the operation of the AAV11-C D/A board
- fixed bug in scaling for D/A gain causing signal fold-over for large values

DAFIX - added code to convert MK3 outstation BGR field format to integer
- added scaling to approximate the MK2 outstation helicorder response

DAINI - added initialization of a new variable called DAOFF, which is used to speed up D/A output in ZDISR. DAOFF is defined in ZDISR where it resides because of ISR mapping considerations

DBASE - set CLKCSR to new KWV register address 170420 (factory default)

> NOTE: the Error Summary table has been modified (via BUFDEF macro) to eliminate all inter-processor support and UART status monitoring. The missed sample counters were converted to 32-bit integers, and 128 words were reserved to allow for a expansion to a total of 64 components.

- added new `.PSECT IOBUFS` dynamic common definition to contain the new I/O buffers
- new I/O buffer sizes required modification to the the IODEF macro; IOCAR entry was removed since carrier monitoring eliminated
- in `.PSECT CTNCOM` the entry `SPACE::` was changed to the new size

```
.BLKW 1600.-<<.-BUFTOP>/2>
```

> NOTE: Table 2 shows the actual size of the "Front End" and "Host" tables required for the MK4 implementation. When the same proportion of table sizes is kept for a full 8KB combined size, the "Front End" table size rounds out to 1600 words (Host tables to 2496). These sizes are multiples of 32 word blocks.

- the size of a SECBUF was modified to the new maximum size of 2048 words
- the last four word entries in the MK4 system's USRTBL are now used as mask words for component monitoring to allow monitoring masks for up to 64 components. The deleted USRTBL entries were MISTSB (Missed Secbufs counter) and FUNDER (Floating Underflow counter), neither of which are need the new single processor floating point MK4 system. The USRDEF macro in CTN.MLB was, hence, updated to include

```
MASK1:: .BLKW 1
MASK2:: .BLKW 1
MASK3:: .BLKW 1
MASK4:: .BLKW 1
```

- reverted to MK3 Minute Mark output by way of DTR bit of the first channel of the first DZQ11 unit's Tx Control Register:

```
MOREG == 160014 ;hard coded register address
```

and moved the storage of MMOLD to ZDISR space, where it is used for the output of the bit (ZDISR and DAFILL still determine when the Minute Mark actually occurs. See NOTE 2 in ZDISR description.)

DBINI - removed all SLTN specific initialization of RNGBUF access magic numbers, which are now calculated by program CONFIG

DISPAT - replaced code sequence updating the missed sample counters to incorporate a 32-bit add

INIT - modified the startup procedure for SECBLD to become the following:

```
request MAKCOM to run and create all commons
map to CTNCOM (hard coded max. array sizes to
  fill the 8KB; not all the space used)
extract SECBUF size from CTNCOM, and
map to RNGBUF (after attach)
map to IOBUF (after attach)
call INITL (further initialization)
```

- added Region and Window Definition Blocks with hard coded mapping parameters for the MK4 definition

```
IOBRDB:: - region size = 256. (32 word blocks = 16KB)
         - use base APR = 3
IOBWDB:: - use window size and mapping length = 256.,
           thereby mapping entire common
```

INITL - same as SLTN but added initialization of pointers required for new I/O and D/A buffer access since the buffers no longer reside in CTNCOM and can no longer be initialized by CONFIG
- added the saving of mapping registers (3 in the MK4 system) for ZDISR operation (formerly done by ZDINI)
- moved call to ZDINI to the end since it now connects to clock interrupts

PUTPAK - removed all references to ER.NOW since it is no longer relevant in the single processor environment
- removed code for updating counts of UART status errors
- changed initialization of SECCNT from a hard-coded 60 Hz to a value ZD.TPB from the ZD tables to permit variable sampling rates for the portable networks

TOPOFF - modified code updating the missed sample counts

ZDINI - same as SLTN, but moved code to save the contents of User APR3, APR4, and APR6 to ZDON
- now contains code connecting to interrupts from the KWV11 real-time clock moved from SCHED

ZDISR - as in the SLTN system, this Interrupt Service Routine (ISR) is entered by way of the RSX Interrupt Transfer Block (ITB) established by the connect to interrupt (CINT$) directive
- the ISR is mapped in Kernel space via APR5 SLTN saved the USRAPR6 at run time for ZDISR to gain access to CTNCOM. A similar process is used to gain access to IOBUFS in the MK4 system via APR's 3 and 4
- the MK4 version was further modified to interact with the KWV11-C real-time clock interrupt source
- removed code to detect loss of carrier signal on the telemetry lines
- added code to switch to the alternate register set (new feature of the KDJ11 CPU) to avoid repetitive saving and restoring of the primary general purpose register set on each entry and exit of the Interrupt Service Routine
- added check for pending RSX ticks to avoid false indications of synchronization errors when RSX was momentarily too busy to keep the system clock up-to-date
- the D/A output handling had to be modified to reflect the new local location of the buffers in the MK4 system as follows:

  - the ZD.DPT variable had to be changed to become an offset counter rather than an actual data pointer (since the pointer is linked into the task space, at context switch time it would point into the operating system during ZDISR).
  - after the actual D/A output (see listing @DAJTAB::) ZD.DPT is updated by the number of bytes that were just written out. The DAOFF variable allows the updating to be done with a single instruction rather than the 2 previous "ADD DACNT" instructions.

- the DASWAP routine was modified to operate with the new context of the ZD.DPT variable by determining which is the current (for the next second) output buffer and setting ZD.DPT offset to the proper value (recall that the D/A buffers are contiguous).

- Minute Mark handling follows the MK3 method, where the MMARK is output after return from the DASWAP routine. The definition of the MMOLD storage word in ZDISR space is required due to Kernel mapping constraints. (NOTE 2 below describes the system's Minute Mark flow).

NOTE 1: the MTPI/MTPD instructions cannot be used to restore User context in ZDISR because RSX is a multi-tasking operating system. This means that the context restored by those instructions would not necessarily be that of the program SECBLD!

NOTE 2: the MK4 method of Minute Mark output is actually a hybrid of MK3 and SLTN: the actual time keeper is RSX, whose time is monitored by ZDISR for the building and synchronization of the I/O buffers. ZDISR monitors RSX's Seconds and Minutes counts to determine when a Minute Mark or Hour Mark needs to be output. The flags set by ZDISR are then checked by the DAFILL routine which determines the length of mark required (it also checks for possible Day Mark required), and sets the flag on or off. After a DASWAP, ZDISR outputs the value of the Mark whether is it on or not. The actual Mark goes out as the DTR bit of the first serial line in the Tx Control Register of the first DZQ unit.

ZDON - modified to use the KWV11-C real-time clock as the source of data gather interrupts in the following way (see also KWV11 specs):

- set mode to 1 (repeated interval leaves GO bit set)
- load the KWV buffer preset register with the computed countdown value stored by CONFIG into KWVCNT (initially the BPR was loaded with 177777 to get overflow after just one count (immediate) since interrupts were triggered by the BEVNT Q-Bus signal)
- synchronize the ST1 interrupt with RSX 60 Hz using the ST2 Go Enable mechanism triggered by a minute mark signal from the GOES/Model 501 clock

ZDOFF - modified for KWV11 operation by clearing all CSR bits pertaining SECBLD operation (mode,clock source)

A.2 TRIGGR

The SLTN modifications to this program enabled TRIGGR to operate on the memory resident ring buffer. Further changes required for MK4 operation were those dealing with the size of areas mapped by the task: CTNCOM and RNGBUF. Also, because the MK4 hardware uniformly contains an integral floating-point processor (FPP), all FIS code dating back to the PDP-11/40 was removed from the program. Lastly, changes were required to reflect the nominal velocity sensitivity and ascii network name entries in the database.

CIF - this module was scrapped since conversion of a 16-bit integer to 32-bit floating point representation is a single FPP hardware instruction

HEADER - the size of the local buffer used to contain the event file header was increased to the size of a SECBUF, ie., 2048 words. This is easily enough space for all the SATBLs (51 x 10 = 510 words) and the new larger Triggered Sequence List (35 x 15 = 525 words).
- eliminated hard coded parameters by adding local definitions SATSIZ and TSLSIZ (same as in TDBASE) to be used in the copying of the tables

TDBASE - increased size of Triggered Sequence List to handle new maximum of 35 triggered components
- uses new USRDEF value of NETWRK
- removed definition of FPP flag since all FIS code has been removed

TRALG - removed all FIS code

TRBMAP - modified the window size and mapping length entries in the RNGWDB to be extracted from the SECBLK entry of USRTBL in CTNCOM

TRGMAP - changed entries in CTNWDB for mapping to the entire CTNCOM dynamic common:

```
WNDSIZ=128. (32 word blocks=4096 words)
MAPLEN=128.
OFFSET=0.
```

TRINIT - added a $STOP directive after the ringbuffer access initialization to wait for an $UNSTOP from SECBLD. This makes for a cleaner system startup, in that TRIGGR will not start until SECBLD is running
- removed the test for floating point and, hence, deleted the startup trap for floating point initialization. Note that FUNDER is no longer used in the FPP environment (See also USRDEF macro)

TDBASE - modified the size of a SECBUF in the .PSECT RNGBUF directive to `.BLKW 2048`
- since TRIGGR now maps to the entire CTNCOM the .PSECT portion was adjusted to:

```
.PSECT CTNCOM,RW,D,...
```

```
SPACE::.BLKW 1600. ; front end tables
USRTOP::...
```

- added two new component monitoring mask word definitions in the USRDEF macro (see also DBASE)
- increased the number of entries in the Triggered Stations List to 35
- added new time correction (SATIMC) and nominal velocity sensitivity (SANVEL) entries to the Secbuf Access Table offset definitions macro SATDEF. SATSIZ was adjusted accordingly
- removed the flag called FPP since FIS code is no longer used

A.3 CONFIG

The early changes made to CONFIG and its subroutines dealt mostly with the new structure of the system: I/O and D/A buffers removed from CTNCOM, and the increased number of channels processed. Later changes were required to read in and incorporate into the dynamic common regions the new system time base TBASE, and KWV11 interrupt rate IRATE, from the header line of NCF.ASC, plus new static-time-correction TIMECR and nominal velocity sensitivity NVS fields from each component line. Lastly, module DZVPRM was modified to support the PGC LP components.

CONFIG - increased the sizes of arrays containing channel or component data to handle a larger number of channels and components: - increased output array sizes for maximum potential size

```
TBL (2496), PT (600)
```

- modified DATA statements to initialize above sizes of arrays - added code to read and save the ascii network name, system time base, interrupt rate, time correction, and nominal velocity sensitivity from NCF.ASC
- changed variable TXSIZE (no meaning in SLTN/MK4) to new entry SECRNG; the number of seconds (of data that can be stored) on the RNGBUF is now read in from the configuration file
- with the increased number of components handled by the system, the component monitoring mask was increased in size to 64 bits. Two more mask words (16 bits each) were added in USRTBL in place of the current MISTSB and FUNDER entries. In the single processor context MISTSB (Missed Secbufs) has no meaning, as it pertains to SECBUFs not read in time by HOST or SECBUFs not sent in time by the Front End. Two more "IF.THEN.ELSE's" were added to take care of components greater than 32 in the component mask generation part of the module
- modified UIC definitions to work with the latest MK3 conventions: all data files in [6,4]
- added parameter IRATE to the DZVPRM call to support PGC LP data

DZKPRM - no longer required since all MK4 systems use common DEC DZQ11 asynchronous multiplexor hardware

DZVPRM - modified to support the 4800 baud, 1 sample/second/channel, 6-byte per packet, 3 component PGC LP signal from the Seismic Long-Period Digitizer
- updated logic using "IF-THEN-ELSE" constructions

HTBGEN - includes all SLTN changes, and further modified to initialize USRTBL variables dealing with the access to the RNGBUF dynamic common (See also DBINI)
- computes KWV11-C real-time clock interrupt parameter KWVCNT from the values of TBASE and IRATE

```
TBL(top+RNGSEC)=SECRNG  ;read from NCF.ASC file
TBL(top+ENDRNG)=SECRNG-1
TBL(top+SECBLK)=BLKSEC
TBL(top+SECSIZ)=TBL(top+SECBLK)*64
                        ;SECBUF size in bytes
TBL(top+SIZRNG)=BLKSEC*SECRNG
TBL(top+SAVBLK)=(TBL(top+SECSIZ)/512)+1
                        ;SECBUF size in blocks

  where BLKSEC = sum of component sizes
                 + SECBUF header, rounded up to
                 the nearest 32 word block
```

- increased sizes of arrays to handle new network size capability
- added code to insert the "static-time-correction" in each component's SAT table
- added code to handle all 4 mask words (insert into tables)
- renamed TXSIZE variable to RNGSEC

LTBGEN - removed I/O and D/A buffer allocation and pointer initialization since the buffers now reside in SECBLD and it is that task that must do its own pointer initialization
- modified DATBL size parameter to 16, hence the offsets to the beginning of the tables after it become:

```
ZDTBL /20/
CCTBL /40/
```

- increased array sizes to handle maximum network configuration

NCTGEN - included all the SLTN changes
- also increased the sizes of arrays (including SCDSAV) to accommodate new max. number of components
- changed loading of DATBL to ensure a D/A channel was defined for each component in the DALIST array
- replaced definition of ERROF with MISSDP, a direct pointer to the 32-bit integer missed sample counter
- now computes the number of packets/sample from the new parameter TBASE

ZDTGEN - changed DZCSR back to the MK3 value of "160010

A.4 DEMUX

As mentioned in section 3, the DEMUX program has undergone significant changes and now provides greatly enhanced functionality. It is considered to be a re-issue, and earlier audit notes in the code have been removed.

Within DEMUX itself, the most serious ramification of the size-related changes was that the virtual output buffer OUTBUF had to be split into two in order to handle the increased number of components being processed. The previous OUTBUF was double buffered via an offset into the same buffer. The new virtual double buffer consists of two arrays, both VIRTUAL(30000), called OUTBF1 and OUTBF2.

To facilitate the handling of I*4/R*4 header data, the new output buffers are declared as I*4 arrays dimensioned for a minimum of 17920 (=35x512) elements, giving a 2KB buffer for each of the 35 short period components handled. Since DEMUX still writes the data samples in Binary Gain Ranged format, every I*4 output location must be packed with two I*2 BGR samples.

Section 3 outlined the subroutine calling hierarchy for DEMUX. The purpose of each of the subroutines is as follows:

CHKTRG check whether file contains valid triggers worth saving

FNDCMP find component corresponding to an entry in the Triggered Sequence List

IFIX4 convert field format auxiliary data words to parameter value and aux data type (used for timing packets)

FNDDAT return a block of data centered around the trigger time

F2I4 convert data samples from MKII Binary Gain Ranged (BGR) format to Integer*4 (32-bit integer) representation

SUBMST subtract a time vector of the form MM:SS from a given time

UPDATE update the file header, number of seconds saved, file name, and start time of the file in the case where the first detected trigger was false but the file is to be saved

F2ARK convert data samples from field (outstation) format to MKII BGR format

DETFLG classify triggers by calling lower-level routines to perform individual checks and set the trigger flag value

TSDEAD test for dead trace by finding 30 or more consecutive values that are identical

UNARK convert data samples from BGR to Real*4 format

SAVFIL determine if files determined to be noise should be deleted automatically

TSWDTH test if trigger was a narrow event (glitch)

GETCOR applies any time correction due to fixed transmission delays

TSSKEW test if data skewed toward positive or negative values

TRILST has been eliminated, since we now allow the Triggered Sequence List to contain multiple triggers from the same component

TSCAL test if trigger was in fact a calibration pulse

The DEMUX suite of routines also uses the following five INCLUDE files:

DATEQV.ICL equivalence's two large arrays used in different places

HDRINF.ICL contains header information needed to be passed by common block

INBUFF.ICL contains input buffer used by both DEMUX and FNDDAT

USRDEF.ICL provides definitions for all entries in User Table (USRTBL)

SATDEF.ICL offset definitions for second buffer access table

The decision of whether or not to delete events classified as noise is controlled by the presence or absence of the new file [1,200]NOISE.DEL. Normally present, this file causes noise to be deleted. However, during system testing, remote calibrations, and other times when automatic deletion may not be desirable, this file may be removed (or renamed) to prevent automatic deletion. *Note: the contents of the file are not important – only it's presence or absence.*

Earlier versions of DEMUX created an event-list file, [1,201]CURRENT.EVL. This file was unused, except at Sudbury, for use with auto-SAM. It has been replaced in the new DEMUX by the detection log file [1,201]CURRENT.DLF. This file contains a printable listing of event_id (date/time), station name, band, and orientation, event classification (one of: S, N, or C corresponding to Signal, Noise, or Calibration), hexadecimal trigger flag, maximum sample value within a 40 second window centered on the trigger time, and, if a file has been created, the output file name.

A.5 LPSAVE

The MK3 version of the program was closely coupled to the program "HOST", in that LPSAVE was woken up once a second by an event flag set by HOST, closely followed by a message from HOST containing the LP data. The MK4 LPSAVE runs as a stand-alone task that maps directly to the memory resident ring buffer (RNGBUF dynamic common). To make the program more efficient, it runs once every minute, and assimilates 60 seconds worth of data per run (before doing a wait for one minute).

To support the Seismic Long-Period Digitizer output for PGC, LPSAVE now checks the ascii network name in USRTBL to determine if LP data capture is required, and if so, which components. GAC LP data continues to go to the file [1,200]GACLP.SRO; PGC LP data goes to the new file [1,200]PGCLP.SRO.

The output of LPSAVE is unchanged, viz., the same SRO format file. Hence, the buffer structure and buffering strategy of the earlier LPSAVE was maintained. Similarly, the SRO data and time conversion subroutines (ISRO, SROTIM) were kept intact, as was the buffer padding strategy. However, the new file creation time logic was changed from 13:00 UT (09:00 EST) to 00:00 UT for generality and commonality between ECTN and WCTN (for example, to facilitate future merging of digital LP data).

Three existing CTN subroutines were added to facilitate the access of LP data. MAPRNG is an SLTN routine (originally developed for the DMPSEC program) that maps to a SECBUF on the ring buffer. SATIDX is a MK3 routine that returns the index (in the "Host Tables" part of CTNCOM) of the SATBL of the requested station name. MAPCTN is used to gain access to the CTNCOM dynamic common.

NOTE: LPSAVE assumes that the three LP components are contiguous within the SECBUF, and that the order of the components is fixed at Z,N,E.

Further modifications included the changing of all IF...GOTO's to IF...THEN...ELSE statements to make the code more readable, and the addition of some initialization code to check that SECBLD is active before proceeding any further.

Arrays and COMMON declarations are used to gain access to RNGBUF and CTNCOM, and EQUIVALENCE statements are used to extract SECBUF time and sequence number. The program also makes used of the new CTNCOM.ICL file to make its CTNCOM access universally identical to all other support tasks.

The following pseudo-code describes the new algorithm for LPSAVE:

```
LPSAVE
   set type flag = 'NEW'
   map to CTNCOM dynamic common
   map to RNGBUF dynamic common
                     (most recent SECBUF)
   if network = ECTN
      get GACLP access information (call SATIDX)
      name output file [1,200]GACLP.SRO
   elseif network = WCTN
      get PGCLP access information (call SATIDX)
      name output file [1,200]PGCLP.SRO
   endif
   wait one minute
   if SECBLD is not active then
      loop on test for active SECBLD
                     (every minute)
   endif
   copy SECBUF time to new SRO record (call SROTIM)
   copy SECBUF(pointer) data to SRO record
   save sequence number
   loop forever
   :  map to next SECBUF
   :  if caught up on the Ring Buffer then
   :  :  wait one minute
   :  endif
   :  if seq. number is out of order then
   :  :  use the time to calculate the
   :  :  index [SUBTIM]
   :  else
   :  :  add 3 to the index
   :  endif
   :  if index indicates a sample before
   :           end of current buffer then
   :  :  if there is a gap between this
   :  :        sample and the previous then
   :  :  :  fill the gap with zeros
```

```
   :  :  endif
   :  :  convert the sample and put it in
   :  :        the buffer [ISRO]
   :  else
   :  :  Zero the rest of the buffer
   :  :  Open file (new or old specified
   :  :        by type flag)
   :  :  Write the record to the file
   :  :  Close the file
   :  :  if the time has just become 00:00 UT
   :  :  :  Set the type flag to 'NEW'
   :  :  else
   :  :  :  Set the type flag to 'OLD'
   :  :  endif
   :  :  Convert time to SRO format and
   :  :        put in record header [SROTIM]
   :  :  Convert the sample and put it in
   :  :        the record
   :  endif
   Endloop
END LPSAVE
```

A.6 SUPPORT UTILITIES

The CDTSN software incorporates a suite of support programs that may be categorized as follows:

1. system configuration support; CONFIG, MAKCOM, NCFLST, DMPTAB, MSGGEN
2. run time monitoring; BOSS, CTNMON, DMPSEC, LINMON, LINSTS, SHOW, STATUS, WDTIMR
3. run time modification; CHGPAR, CLERSM, CLOCK, DAC, RESET
4. diagnostic routines; DMPINP, TIMCMP, TIMING

Several other programs provide off-line support for such utilities as listing of TRIGGR files on disk (LISTDF and TRGLST) and listing of data in a demultiplexed file (TSFDMP).

Much of the MK3 support software was directly usable in the SLTN system, and a large part of the programs that were modified, only required changes in their mapping to the new CTNCOM structure. To facilitate the latter change to MK4, and to make CTNCOM access universally identical to all programs that need it, a new 'INCLUDE' file CTNCOM.ICL was developed that contains full definitions and mapping of CTNCOM data, and MAPCTN now maps the entire common.

The following description outlines the contents of the new CTNCOM.ICL:

```
COMMON/CTNCOM/SECTBL (1600), USRTBL(32),
              HSTTBL (2464)
define BASADR="140000 for address conversions
  (CA2I)
set mapping context variables (OFFSET,
                       WNDSIZ, MAPLEN)
define all USRTBL entries with EQUIVALENCE
          statements (via new USRDEF.ICL)
define SATBL indices (via new SATDEF.ICL)
define TTBL indices (via TTDEF.ICL)
define offsets for various ERSUMY parts
```

where the new USRDEF.ICL includes all changes made to USRTBL (see also DBASE), and the new SATDEF.ICL includes the added time correction definition.

The new CTNCOM.ICL standardizes access to the Common for all programs that use it, and also standardizes the variable naming conventions for CTNCOM resident data.

The following is an alphabetical list of those support programs that were modified and the changes that were required for the MK4 implementation (programs that only required the inclusion of CTNCOM.ICL are not listed):

CHGPAR - modified to used the new 'CTNCOM.ICL' and renamed all the variables to conform with naming conventions established by it (especially CTN array to HSTTBL array)
- modified component mask word processing to handle all 4 mask words (see also CONFIG)
- removed the MSKGET, MSKSET function subroutines from the module. They must now be explicitly declared in the link command file

CLERR - added 'CTNCOM.ICL' for the definition of WNDSIZ
- modified via BUFDEF macro to support 32-bit missed data counters

CLOCK - now reads in the time string using an in-line asynchronous QIO (replacing FORTRAN routine GTGOES) so that it now properly times out when the time string not available
- decreased the timeout for the initial minute-mark interrupt from 70 (overcautious) to 61 seconds
- redefined hardware parameters for the new KWV-11 source of interrupts

```
MMVEC==444
MMCSR==170420
```

where the MMVEC is globally defined in the program's link command file CLOCKKTB.CMD, and MMCSR is defined in the interrupt routine MMISR (.MAC)
- the interrupt enable/disable routine MMEDIR was modified to set and clear the INT2 enable bit in the KWV-11 CSR and clear the ST2 FLAG bit (pending interrupt if any)
- added explicit call to the FORTRAN OTS initialization routine OTI$ to avoid periodic mysterious TRAP messages

CTNMON - deleted missed SECBUF processing since it is no longer being monitored by program HOST (that USRTBL entry is now used as an additional mask word for component monitoring)
- downtime messages now report ascii station name rather than line and component numbers
- added 'CTNCOM.ICL'
- increased QUAL array size to 64 for maximum number of components that can be handled by a MK4 system
- added code to handle all 4 mask words for the component down-time monitoring
- restored the MK3 context for disc space constraints

> Note that the GREEN, YELLOW and RED subroutines calls are unchanged even though they work with a new serial interface (See also PANEL module)

- added system calls to disable AST recognition while CTNMON.TMP files are open to prevent corruption of the file system and the generation of mysterious "lost files"
- added a WRITE to the temporary files, again attempting to combat "lost files"

DAC - added 'CTNCOM.ICL'
- modified parsing of string to allow for parameter values that require more than a single character
- now uses ascii station names and accepts blanks as well as the comma delimiter

DAILY - new routine to provide automatic, once-per-day scheduling of DAILY.CMD
- under RSX, commands files cannot be scheduled to run; only tasks can. DAILY is a task that is scheduled at system startup to run at a fixed time every day (currently 7:15 local time)
- internally, the following code:

```
call spawn(MCR,,,,,,,'@DAILY',6,0,,)
```

spawns the indirect command processor with the argument 'DAILY.CMD'

DMPINP - removed initial SLTN/MK4 restriction of 500 samples; can now map to and dump the entire I/O buffer (8192 bytes)
- now scans for end-of-data as signalled by 4 contiguous zero bytes, and only prints out this much
- added GETMCR system call to permit optional command line parameter indicating a single channel to monitor
- now is installed with taskname ...DIP
- replaced call to CWTOB with equivalence of a word and byte array and modified output FORMAT statement
- added declarations for IOBUFS dynamic common:

```
parameter (iolen = 8182)
...
COMMON/IOBUFS/IOBUF1(iolen)
```

- added RDB and WDB arrays for mapping to the common:
`COMMON/FMAP/IOBRDB(8), IOBWDB(8)`
- modified the algorithm to the following pseudo-code equivalent:

```
DMPINP
   initialize RDB and WDB parameters
   determine if a specified channel was selected (GETMCR)
   attach and map to IOBUFS common
   copy IOBUF1 to TEMPBF array
   if no channel specified
     scan TEMPBF for end-of-data
     print entire buffer to end-of-data
   else
     scan TEMPBF for specified channel
     print this channel data
   endif
END DMPINP
```

DMPSEC - added `COMMON/RNGBUF/SECBUF(2048)` declaration
- added 'CTNCOM.ICL'
- added in-line algorithm to calculate position of the second most recent SECBUF on the ring buffer (formerly done in RNGSEC.ICL) followed by a call MAPRNG to map to it
- deleted the call to CA2I as it was no longer needed
- changed references to array SAT with array HSTTBL to conform to the naming conventions used in the universal CTNCOM access

DMPTAB - configuration files now reside in UIC [6,4])

- increased TBL and PT arrays to the new maximum potential sizes

```
TBL (2496) and PT (600)
```

- changed file LUN assignments to avoid bug whereby all output came to the screen, even when a file name was specified

LINMON - added 'CTNCOM.ICL'

- major revision to reflect dropping of UART status monitoring and conversion of missed sample counters to 32-bit integers
- increased MISSED, QUAL, MDS and SRATE array sizes to 64 (max. components handled), and similarly adjusted their initialization DATA statements
- renamed all references to array SAT with array HSTTBL to conform with the new universal access conventions
- modified the RECORDSIZE parameter statement (and corresponding task build parameter in LINMONTKB.CMD) to reflect the new data size written to the LINEQUAL.DAT file
- added call to CA2I to dynamically get the pointer to the Error Summary Table. This makes the ERSUMY accessible, wherever it may reside in the Front End tables (its position varies with system configuration)

LINSTS - expanded array sizes to handle the maximum of 64 components

- updated to reflect dropping of UART status monitoring and conversion of missed sample counters to 32-bit integers

MAKCOM - modified the CTNCOM dynamic common declaration to read

```
COMMON/CTNCOM/SECTBL (1600), HSTTBL (2496)
```

for the new Front End and Host table maximum sizes, respectively

- modified pointer array dimension to new maximum potential size: `POINTR (600)`
- changed CTNCOM mapping parameters to create and attach the entire 8KB common:

```
region size = 128. (32 word blocks = 8KB)
window size = 128.
 map length = 128.
      offset = 0
```

- deleted second mapping call (from SLTN version) to CTNCOM since already mapped to entire common, and renamed arrays to conform with the new naming conventions
- added COMMON declarations and associated RDB and WDB arrays for creating and attaching to the IOBUFS and RNGBUF dynamic commons
- added code to create the IOBUFS region 16KB (hard-coded for max. size) = 256. 32 word blocks, and to calculate the RNGBUF region size = SECBLK * RNGSEC (32 word blocks)

NCFLST - changed variable TXSIZE to RNGSEC and modified FORMAT statement to print out corresponding message 'NO. OF SECONDS ON THE RINGBUFFER:'
- added code to print out the new static time correction field
- updated to read and print new NCF.ASC fields TBASE, IRATE, and NVS

RESET - modified to conform with new UIC conventions (HSTTBL.DAT file in [6,4])
- adjusted the mapping parameters to map the entire CTNCOM dynamic common

SHOW - essentially a rewrite for the MK4 system, incorporating 32-bit missed sample counters and removal of the UART status counters
- added 'CTNCOM.ICL'
- modified to use ERSUMY offset declaration and to call CA2I to gain dynamic access to the Error Summary Table
- increased array sizes dealing with components to the maximum of 64
- added code to handle multi-component serial lines by testing the 'LINE NUMBER' SAT Table entry

STATUS - added 'CTNCOM.ICL'
- changed CTN array references to read as references into the new HSTTBL array, and kept the call to CA2I that is now used to gain TTBL access
- deleted references to NOPOOL and FUNDER which are no longer used
- modified to reflect replacement of TRGLAG and TRGPRI in USRTBL with the 4-byte ascii network name

TIMCMP - identical to changes made to CLOCK program, except interrupt routine is called TCISR (Time Compare ISR)

TRGLST - modified to work with the new structure of the MK2 Time Series Files by way of adjusted parameter, variable, and array declarations
- modified the print format statements to handle the new longer station names
- modified the calculation of event duration to remove hard coded parameters

TSFDMP - modified to allow the program to read TSF MK2 headers
- because of the mix of I*2 and I*4 header data, array sizes and pointers were changed, as were the size of variables needed to handle the two sizes of integers
- since this program uses the FIO routines to do block I/O of a demultiplexed file, it had to be modified to differentiate between the 512 byte blocks used by the FIO routines and the 2KB block pointers used inside the DEMUX'd file
- parameter definitions were modified to reflect the structure of the MK2 headers
- the following is a list of modified or new parameters, variables, and arrays:

```
BLKSIZ = 1024;I*2 word size of an I/O block
STRTIX = 2    ;I*4 index to start of data index
                in the component header
TIME=9        ;I*4 offset in component
```

```
                header to data start time
NUMCMP=22      ;I*4 index in the file header to
                the number of components in the
                file
FSTCMP=28      ;new, I*4 index in the file
                header to the first component
                descriptor block
COMPBK = 4     ;I*4 index to station start
                block pointer in the component's
                descriptor block
THRBLK         ;deleted
TWOBLK=4096    ;no. of bytes in 2 I/O blocks (FREAD)
ONEBLK=2048    ;no. of bytes in 1 I/O (FREAD)

I*4 STNNAM(2,8)
               ;8 characters per name
                (8 stns.)
I*4 SPLCNT     ;number of samples for
                the component
I*2 COMBUF(2048)
               ;general purpose two
                block buffer
I*4 STASHN(2) ;8 character station
                name used in operator
                query
I*4 HDRBUF(512)
               ;MK2 format file
                header buffer
REAL FREQ      ;R*4 sampling rate
```

- several EQUIVALENCE statements were added to facilitate extraction of the I*4 component header, the component sample count, and the new R*4 sampling frequency

- algorithmic modifications to handle the new structure include:

  - get the starting block number of the first component via:

```
HDRBUF(FSTCMP+COMPBK-1)
```

  - get number of seconds of data in the file via:

```
MAXSEC=SPLCNT/ifix(FREQ)
```

- read station name from operator and handle it in 2A4 format
- adjust for discrepancy in file's internal record pointers and FIO's block pointers in PAGEHD.ICL, and use new I*4 COMBUF to adjust time (via minutes and seconds offset) for printout on page header
- modify calculation of required blocks and indices for data extraction, to use the new defined parameter declarations

WDTIMR - was eliminated in the MK4 system

A.7 SUPPORT UTILITY SUBROUTINES

Most of the CDTSN support programs map to the CTNCOM dynamic common to extract or modify run-time data. All those programs use the MAPCTN subroutine to gain access to the region. In addition, there are several other subroutines used by one or more of the support utility programs. The following is a list of these subroutines with a brief description of each:

CA2I - Convert memory address to an index (unchanged)
- used to access parts of CTNCOM normally accessed by an address (needed for F77 programs)

CLERR - Clear Error Summary Table
- assembler routine (SLTN) modified to used the BUFDEF definition of the size of the Error Summary Table, with 32-bit missed data counters
- called only from (F77) CLERSM program to do the actual clearing

CWTOB - eliminated in later changes to DMPINP as its function of separating the high and low bytes of raw data words acquired from the DZQ11 hardware silo buffer could be more simply accomplished by equivalencing a word and byte array and modifying the output FORMAT statement in the main program

DAPTNS - D/A Parameter Test and SET
- SLTN (F77) routine modified to check for new number of DZs=4 and new number of DACs=12
- called only from program DAC to check for correct limits on parameters before changing the D/A gain
- added support for ascii station name input

DASET - Set the new D/A gain (or reassign D/A channel)
- called only from DAPTNS (F77) routine to do the actual work
- modified for the new (CTNCOM) Front End table structure, which can handle 12 D/A channels and 16 serial lines (4 per DZV unit):

```
        .PSECT CTNCOM,RW,...
SBTBL:: .BLKW 3
DATBL:: .BLKW 16.
ZDTBL:: .BLKW 20.
CCTBL:: .BLKW 17.
```

FLT2 - eliminated with move to 32-bit missed sample counters

GETCOR - new F77 subroutine called by DEMUX to extract SECBUF pointer to the pseudo-component containing the dynamic time correction data for a given station (See also DEMUX description and listing)

GETSTN - F77 routine called by TSFDMP to extract data start information about a named component
- modified to work with the new TSF header structure: I*4 array and new longer station names

GSTNAM - return ascii station name given sequential component number
- called only by CTNMON

GTGOES - eliminated in favour of in-line QIO in module CLOCK

IFIX4 - unchanged macro routine called from DEMUX to convert field format auxiliary data word to an integer value and return data type

ISRO - unchanged macro routine called from LPSAVE to convert a data sample from field format to SRO format

IVAL - unchanged macro function subroutine called from ADRMON to return the value of a specified address

MAPCTN - Map to CTNCOM dynamic common
- called by most support programs
- unchanged F77 routine (ECTN MK3/SLTN)

MAPRNG - Map to the RNGBUF dynamic common
- F77 routine unchanged from SLTN

MMISR - Minute Mark Interrupt Service Routine
- used only by the CLOCK (Assembler) program to read time directly from the RSX executive and then zap the seconds and ticks to zero
- modified (from SLTN) to operate with new KWV-11 interface

MSKGET - unchanged F77 function subroutine, but now in stand alone format; previously bundled with CHGPAR and CTNMON (now explicitly specified in the link command files)
- returns the bit value of the specified bit position in the input mask word

MSKSET - slightly modified F77 function subroutine now in stand alone format (previously bundled with CHGPAR)
- changed to actually work (tests showed that the previous version didn't work) via use of the intrinsic IEOR function

OPTIMC - Output the Time in Comparison format
- F77 routine called only from TIMCMP program to display the expected and actual RSX time found at the instant of the minute mark interrupt
- unchanged from SLTN

OPTIME - Output Time
- similar to OPTIMC but called only from CLOCK program to output the time found at the

instant of the minute mark (before the seconds and ticks were zapped to zero)
- unchanged from SLTN

PAGEHD.ICL - slightly changed to extract data start time from new name buffer
- included in-line in program TSFDMP to pretty print a page header before a data dump

PANEL - new F77 module containing 6 subroutines that perform I/O to the serial interfaced monitor panel
- 4 of the subroutines (GREEN, YELLOW, RED, MONOFF) are called from CTNMON to set the appropriate indicator light
- the other 2 were planned for the un-implemented WATCHDOG TIMER program for cross-monitoring and re-booting a malfunctioning companion computer

PINC - eliminated in the change to 32-bit missed sample counters

SATDEF.ICL - modified to reflect new static-time-correction entry

SATIDX - get Secbuf Access Table Index
- called by several (F77) support program to get the SAT index of a desired station (by name)
- F77 routine modified to remove any hard coded control parameters in favour of testing for end of SAT tables
- added 'SATDEF.ICL' and changed variable names to conform with it

SROTIM - unchanged macro routine called from LPSAVE to convert time from CDTSN format to SRO format

TCISR - Time Compare Interrupt Service Routine
- used only by TIMCMP (Assembler) program to read (without zapping) RSX time directly from the executive
- assembler routine slightly modified to work with the new KWV-11 hardware

TTDEF.ICL - unchanged 'INCLUDE' file defining Trigger Table offsets

UNARK - macro subroutine to decode BGR format data and return integer mantissa and gain
- called by DEMUX to determine maximum sample value for a trace
- modified to reflect conversion of the former data valid bit to an extended exponent bit for the extended dynamic range MK3 outstations

USRDEF.ICL - new 'INCLUDE' file reflecting the MK4 status of USRTBL (FUNDER and NOPOOL removed and 2 MASK words added)

*60Hz components x 32	7680 bytes
*30Hz components x 3	360 bytes
* 1Hz components x 6 (GACLP _& GACAX)	24 bytes
IOTIM	14 bytes
IOCAR(min. 16 chans.)	32 bytes
* 2Hz timekeeping x 10	80 bytes

	8190 bytes

*2 words/sample & 2 bytes/word
(upper DZQ byte contains status bits)

Table 1: IOBUF Size Requirements

"Front End" Maximum Table Sizes (words)*

	SBTBL	3	
	DATBL	16	
	ZDTBL	20	
	CCTBL	17	
4	DZTBL @ 12 ea	48	(one per DZQ unit)
16	CCT'ch @ 5 ea	80	(one per serial channel)
16	PMT'ch @ 6 ea	96	(one per serial channel)
16	SDT'ch @ 3 ea	48	(one per longitudinal code) (at least one per channel)
51	SCD'cmp @ 11 ea	561	
16	PKTBUFs @ 8 ea	128	(4 comp/chan maximum)
	ERSUMY	128	
		1145	

"Host" Maximum Table Sizes (words)*

	USRTBL	32	(shared by "Front End")
51	SATBL'ch @ 10 ea	510	
35	TTBL'ch @ 38 ea	1330	(only SP can trigger)
		1872	

*worst case accounts for 32-60Hz components
3-30Hz components
6-1Hz components
10-2Hz "pseudo" components

Table 2: CTNCOM Table Size

```
32 x 60 = 1920 (60Hz components)
 3 x 30 =   90 (30Hz components)
 6 x  1 =    6 (1Hz components including aux.
               data)
10 x  2 =   20 (2Hz "pseudo" components)
  SBTIM =    7 (SECBUF time)
 HEADER =    4

          --------
          2047 words
```

Table 3: Ring-buffer Size Determination

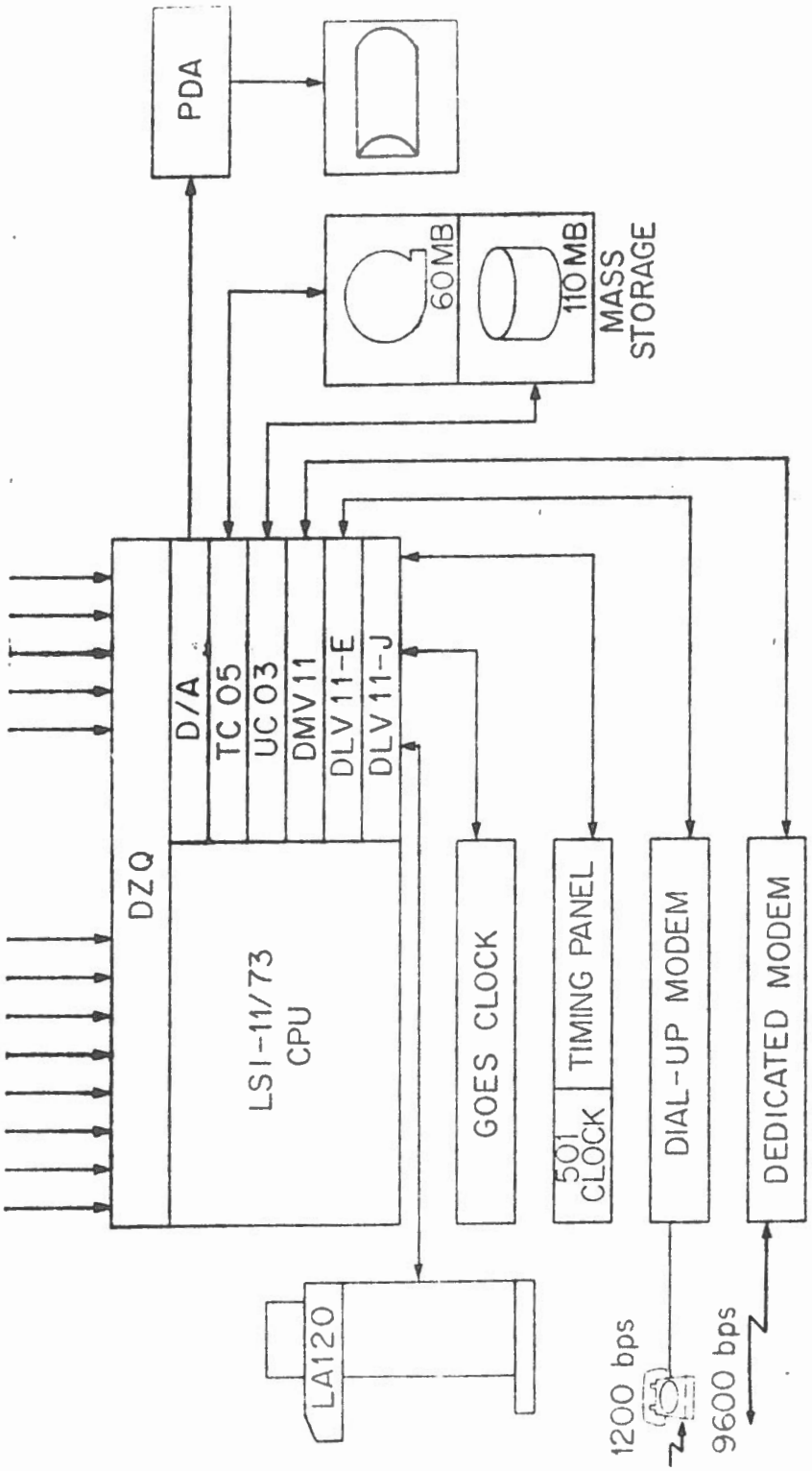


Figure 1: Mark IV Hardware Block Diagram

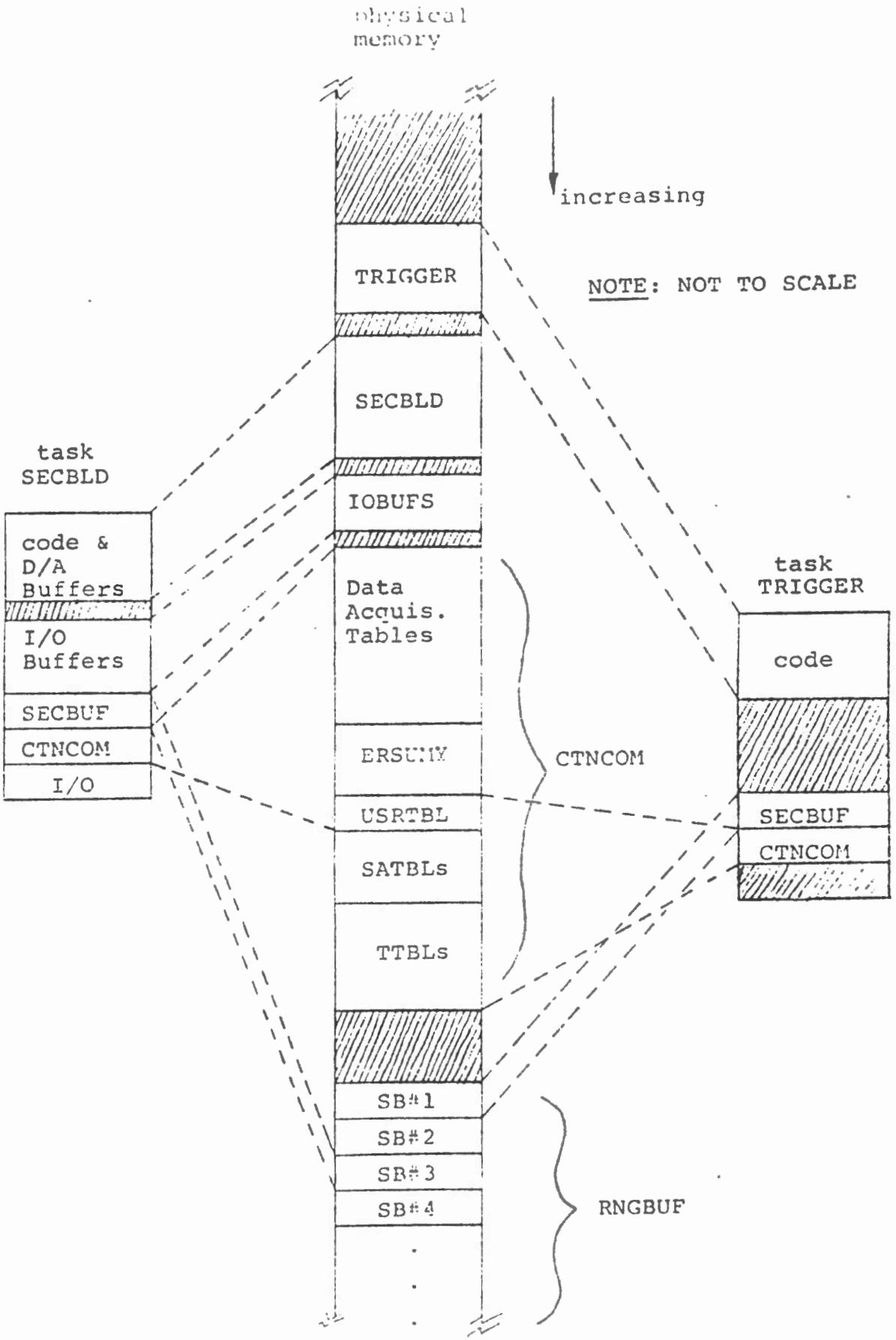


Figure 2: Typical Memory Layout

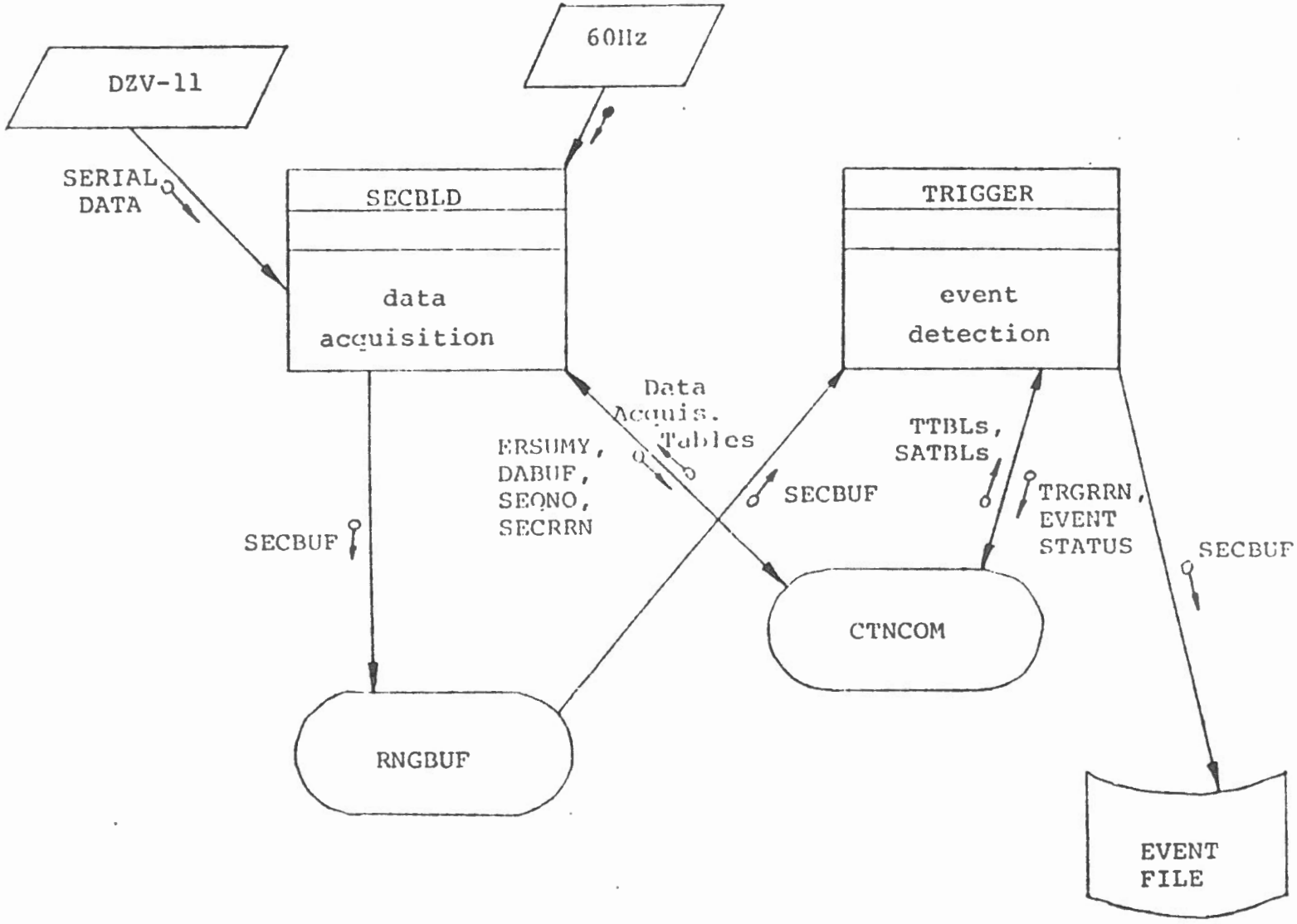


Figure 3: Task-database Interaction

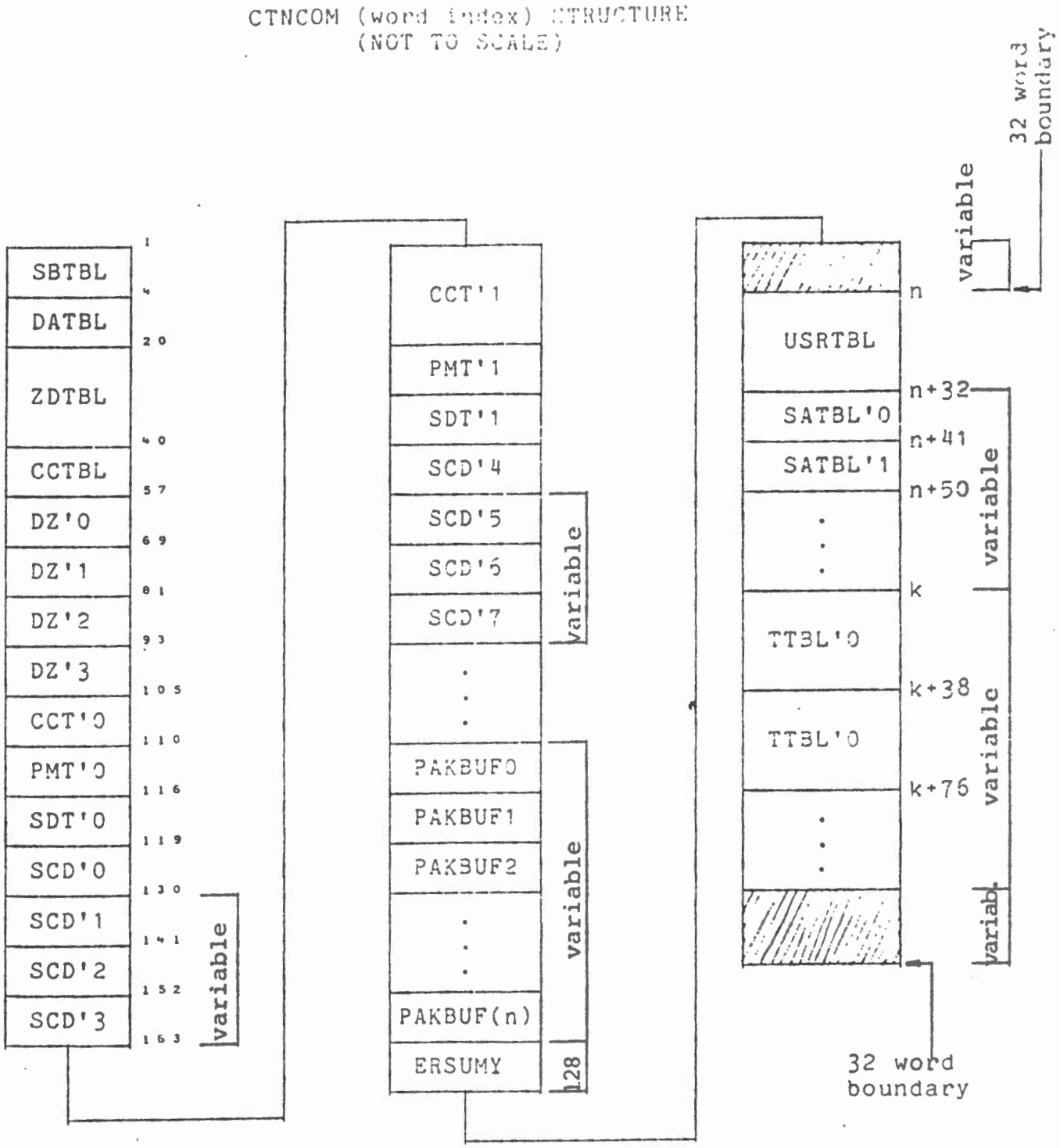


Figure 4: CTNCOM structure

SECBUF (Word Index) STRUCTURE

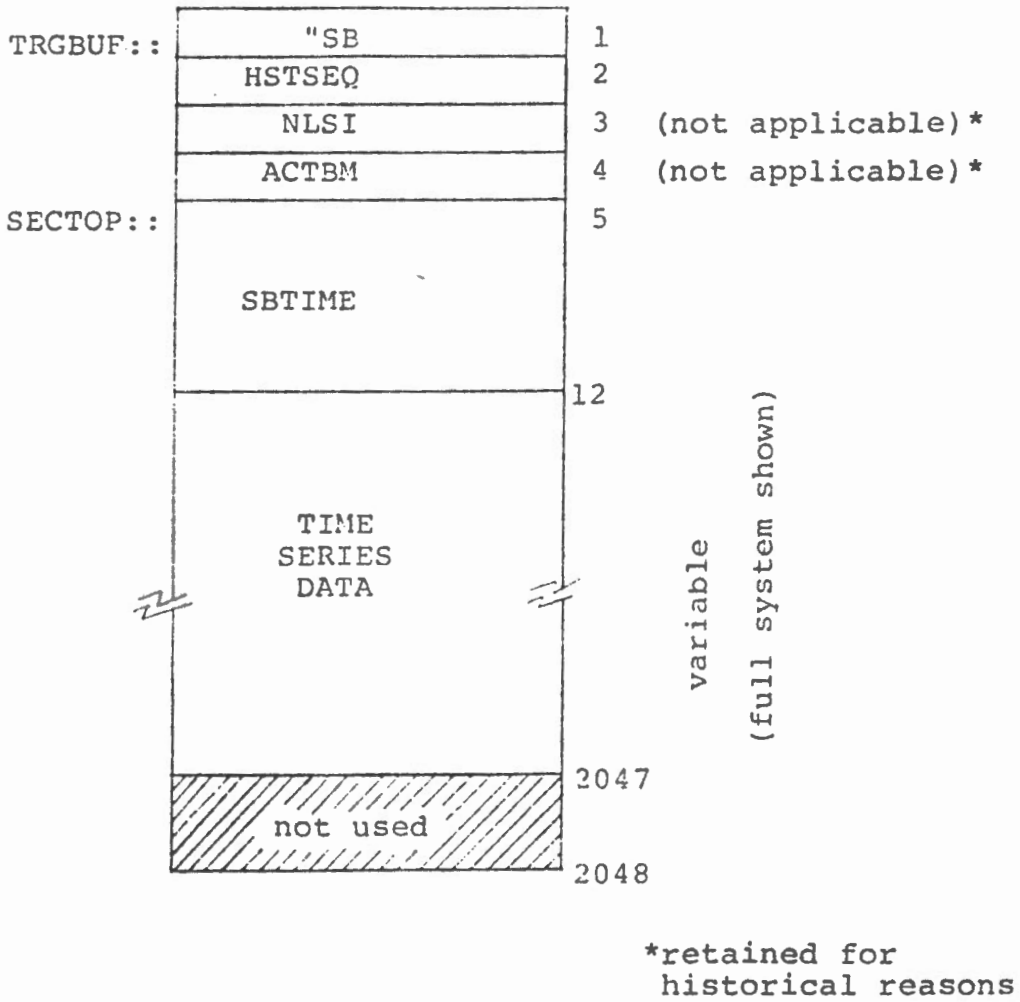


Figure 5: SECBUF Structure

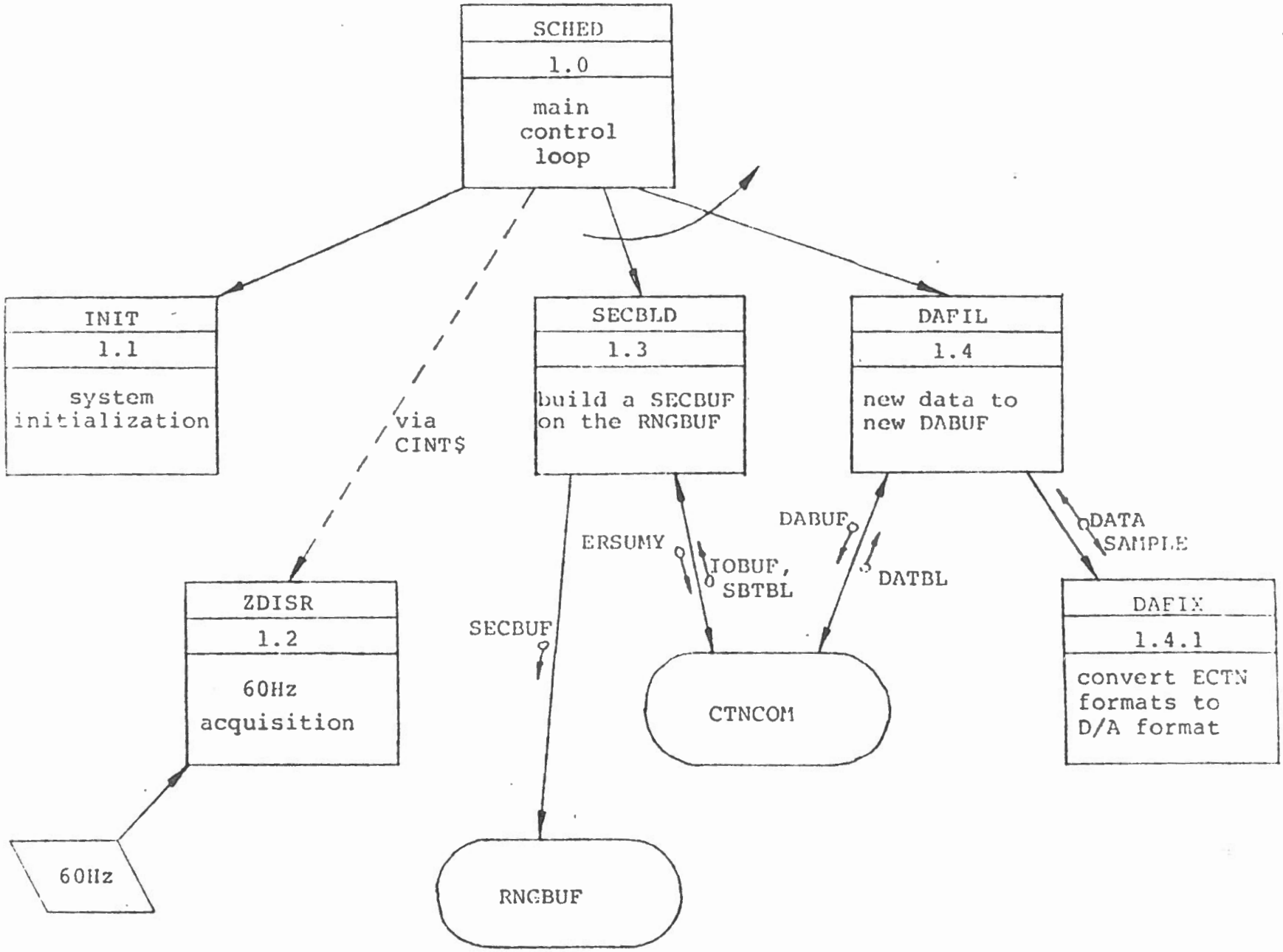


Figure 6: SCHED

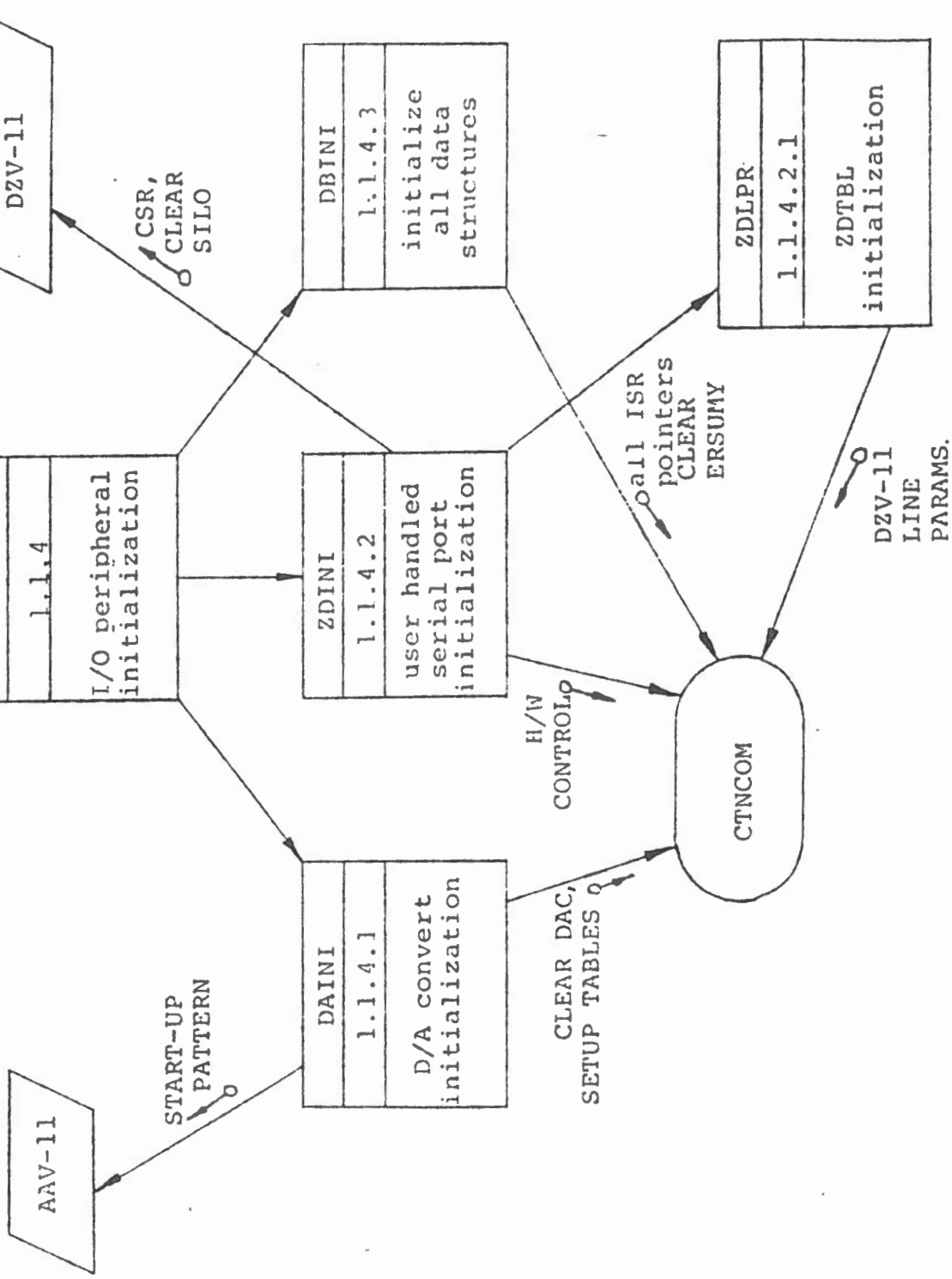


Figure 7: INITL

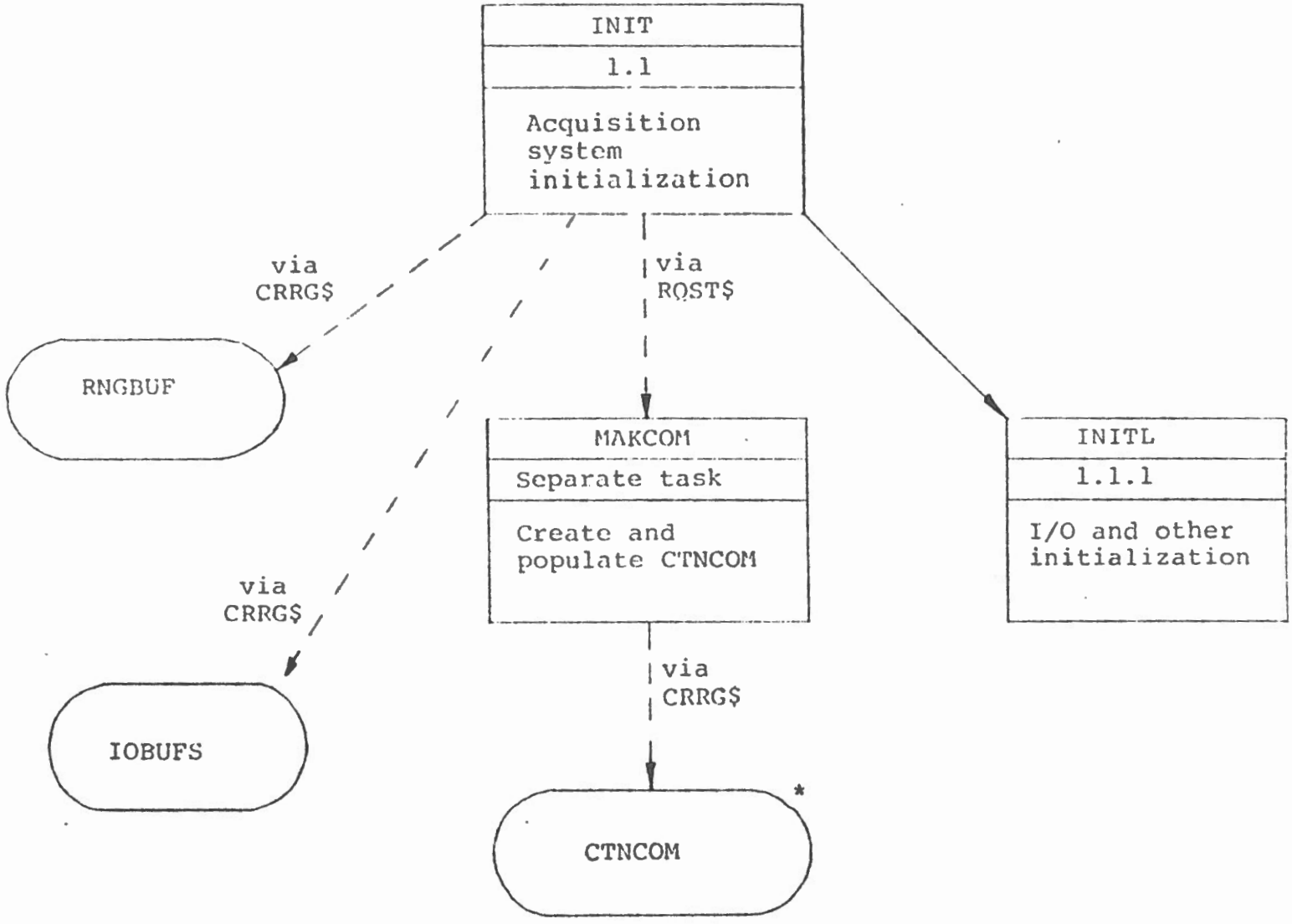
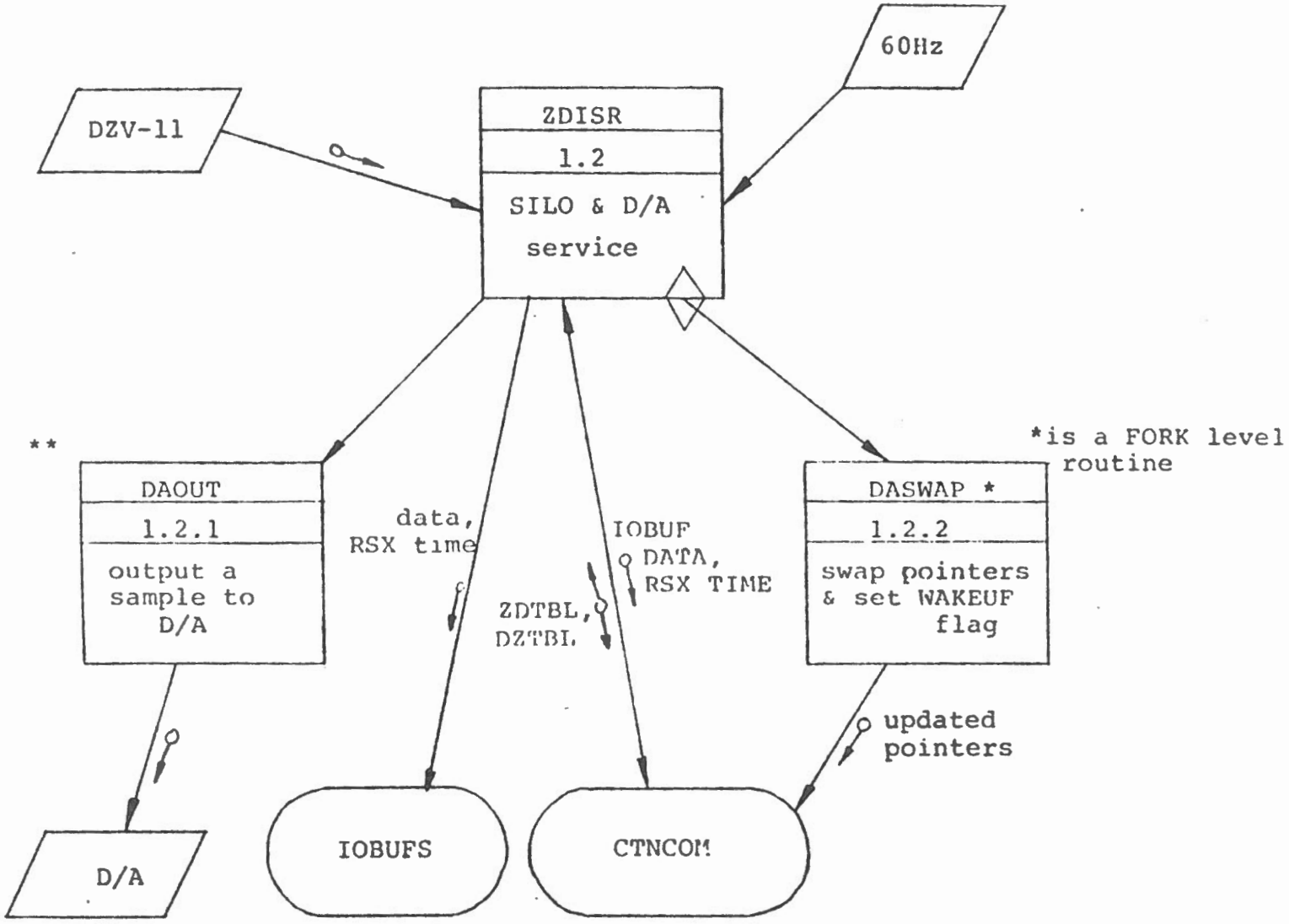


Figure 8: INIT



** at start-up, two routines DSTART & DZERO are executed, to output a pattern to the DAC

Figure 9: ZDISR

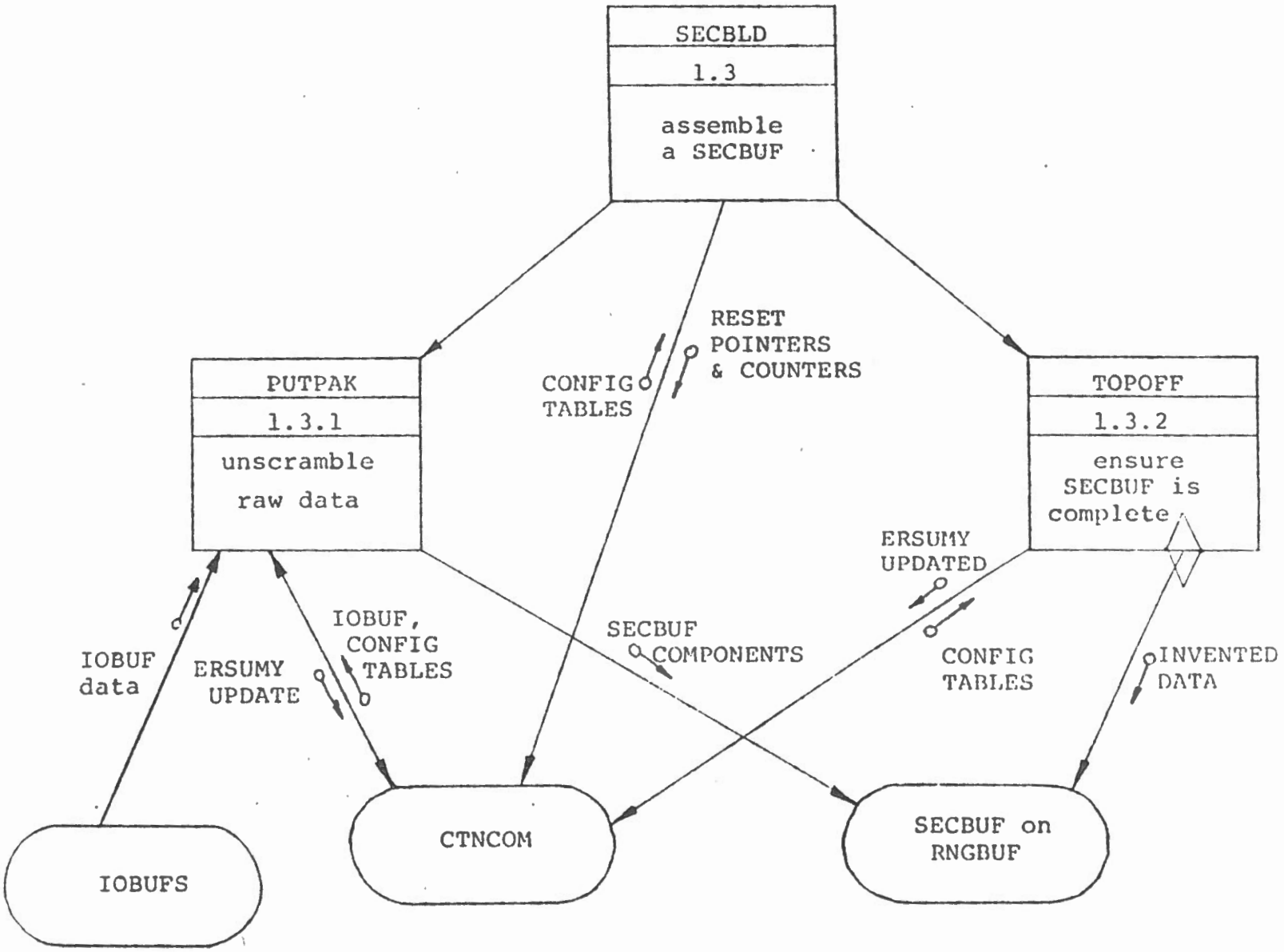


Figure 10: SECBLD

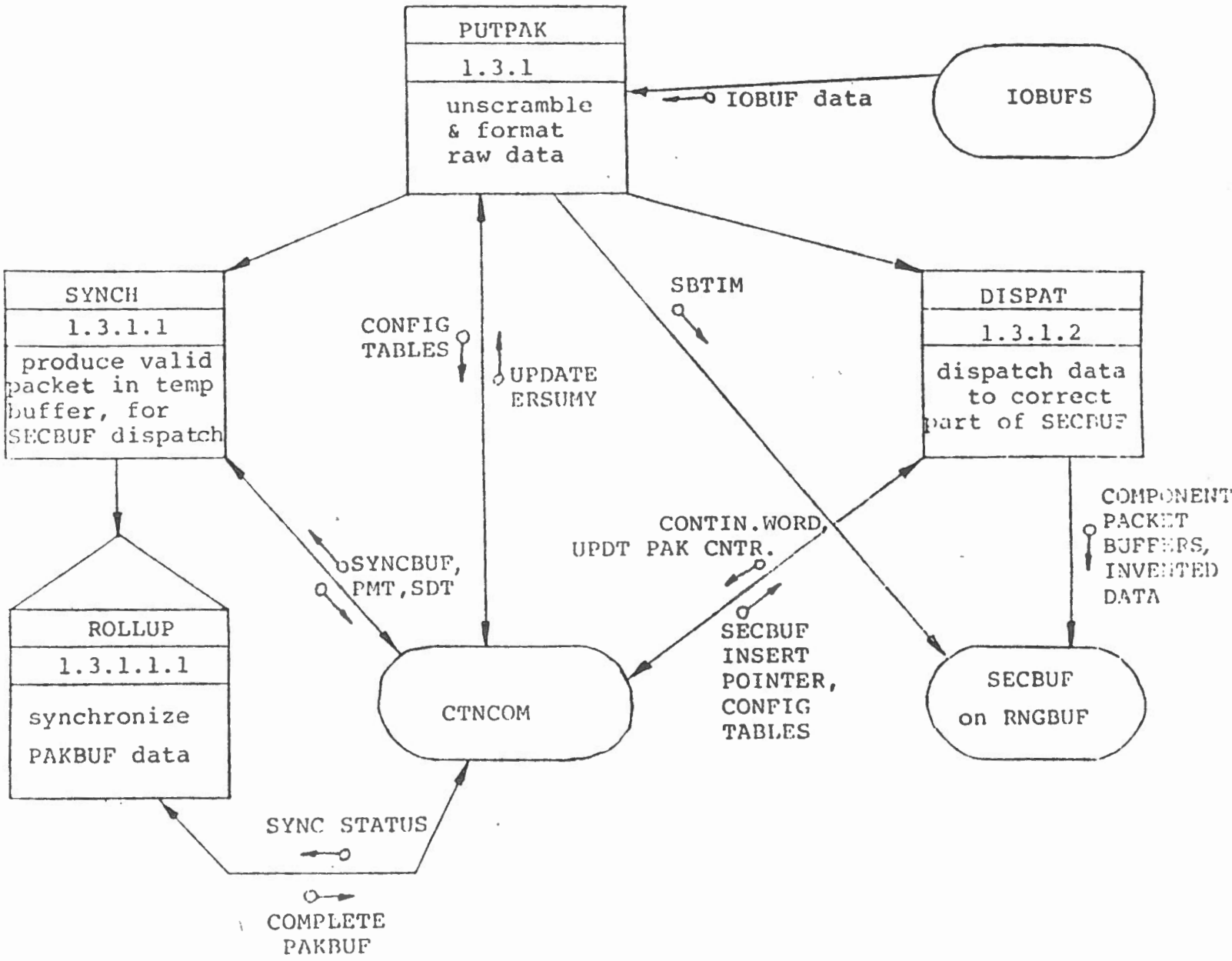


Figure 11: PUTPAK

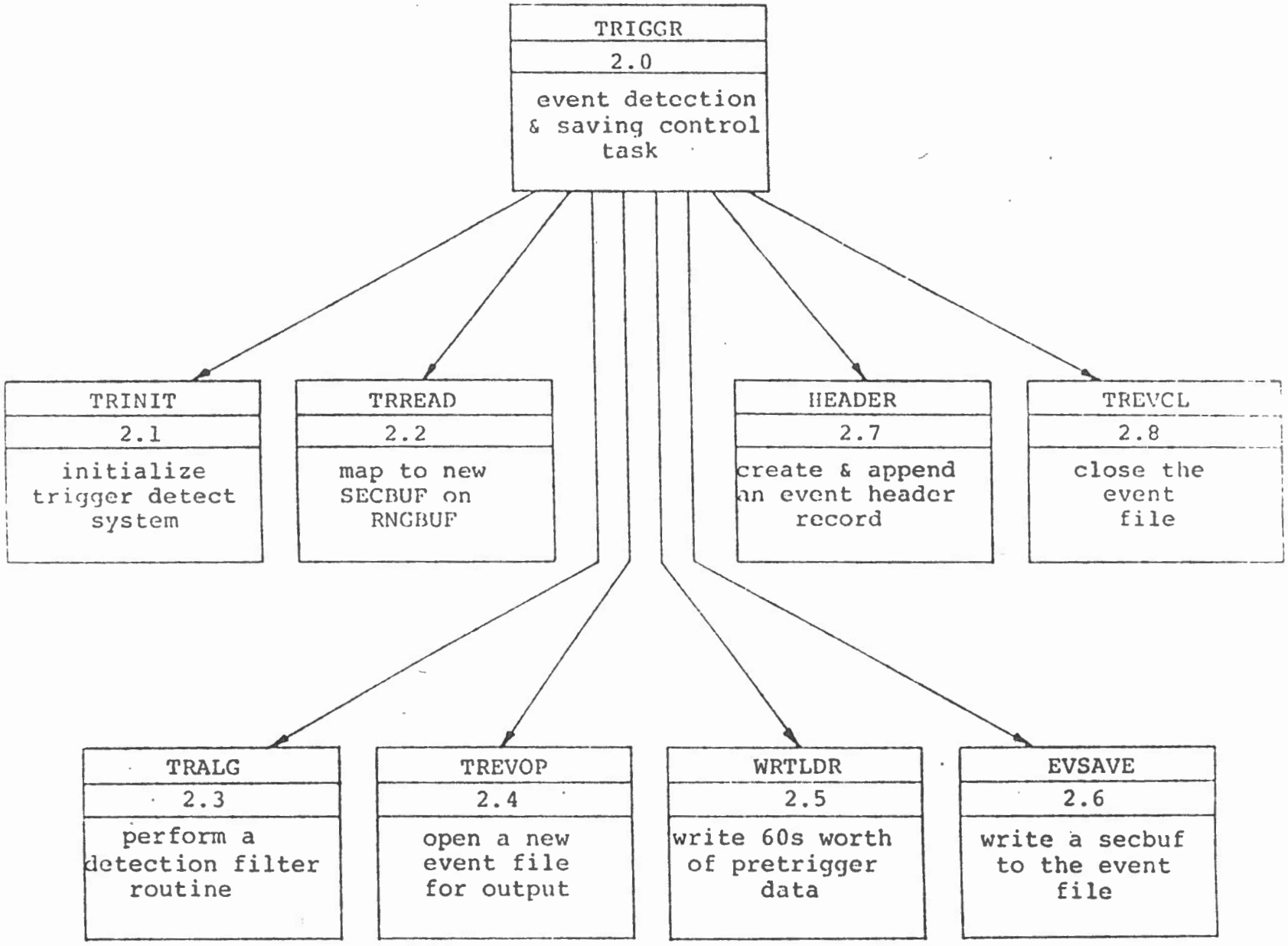
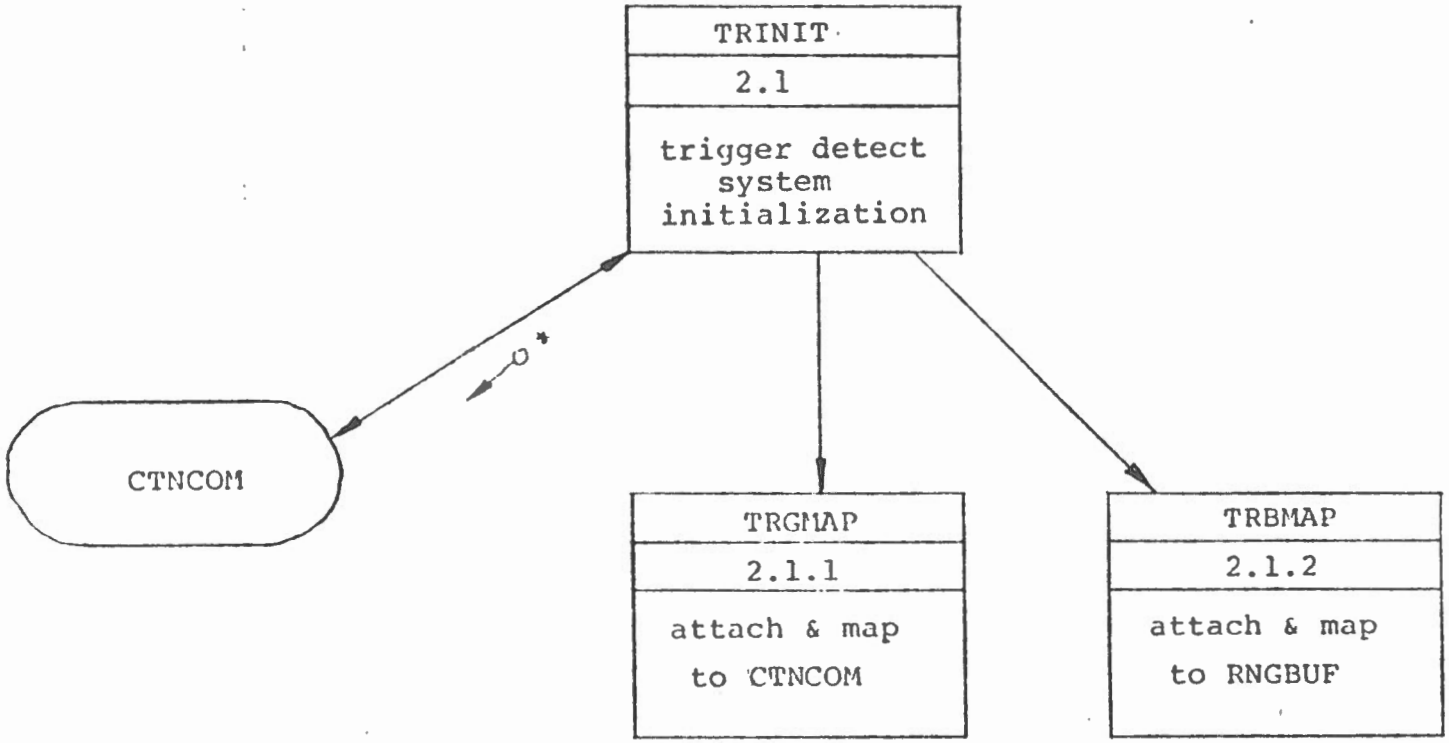


Figure 12: TRIGGR



* initialize: event save data, trigger tables, fix routines (ECTN format conversion), event file allocation, FPP trap routine

Figure 13: TRINIT

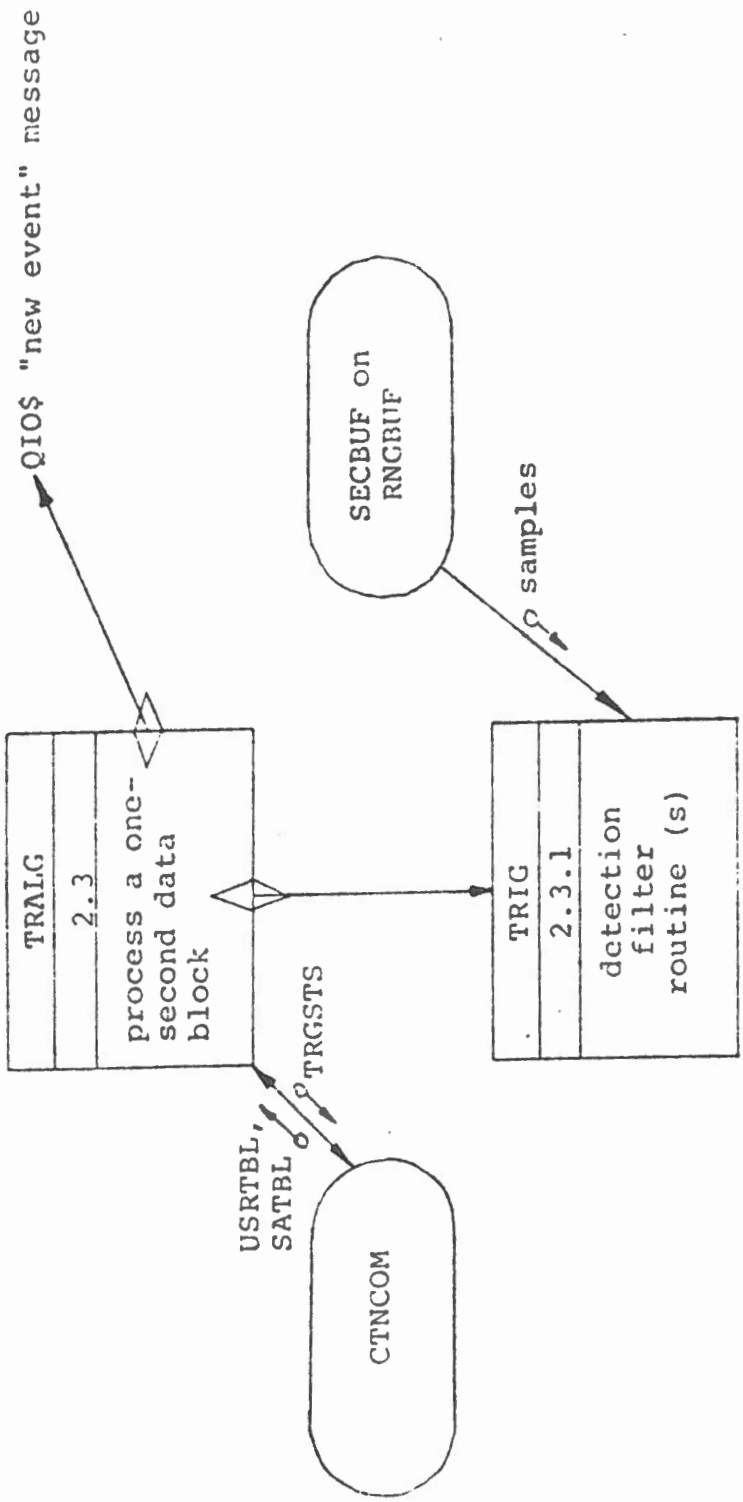


Figure 14: TRALG

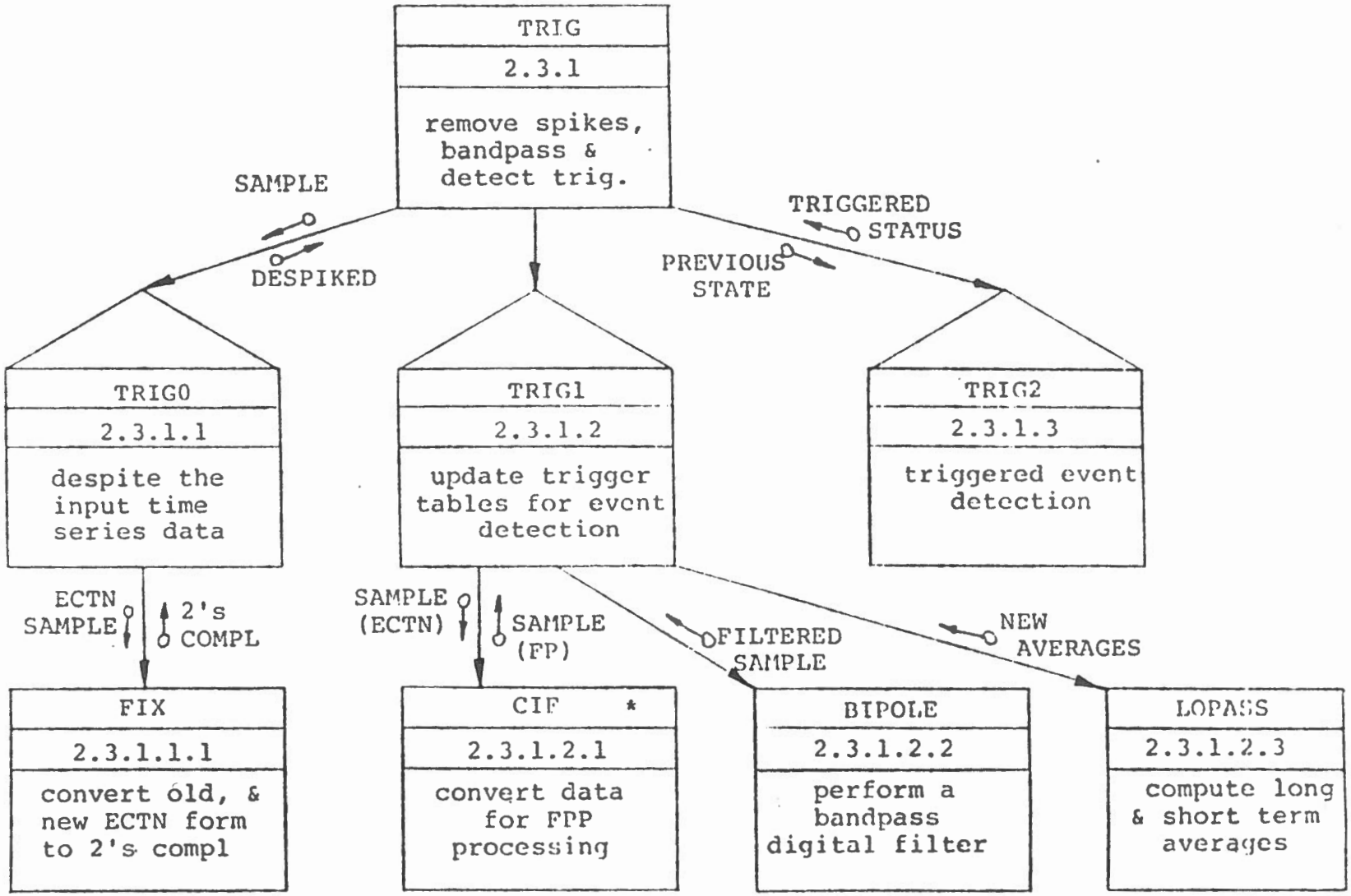


Figure 15: TRIG

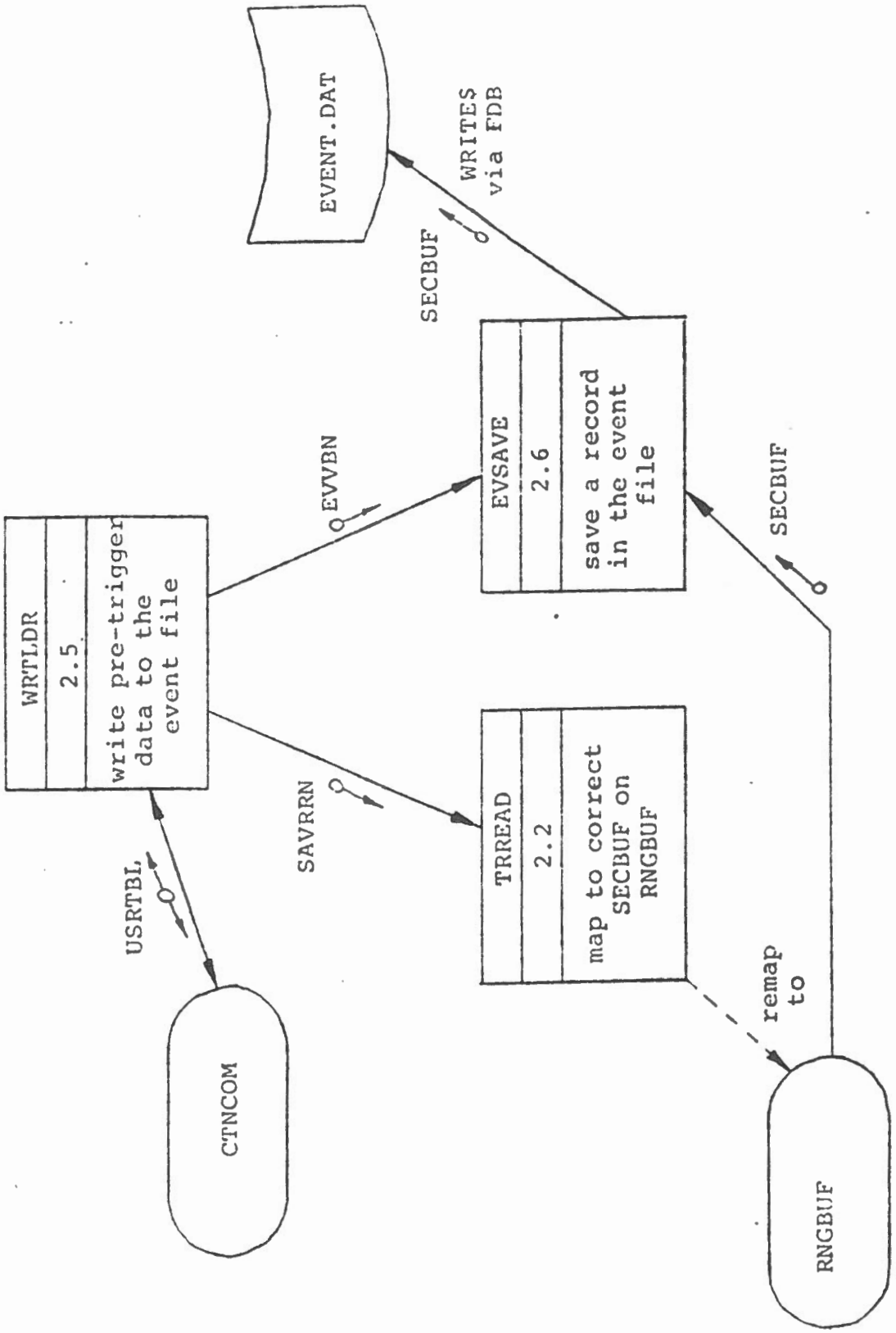


Figure 16: WRTLDR

* includes: USRTBL, SATBL, TRGSLT (sequence list)

Figure 17: HEADER

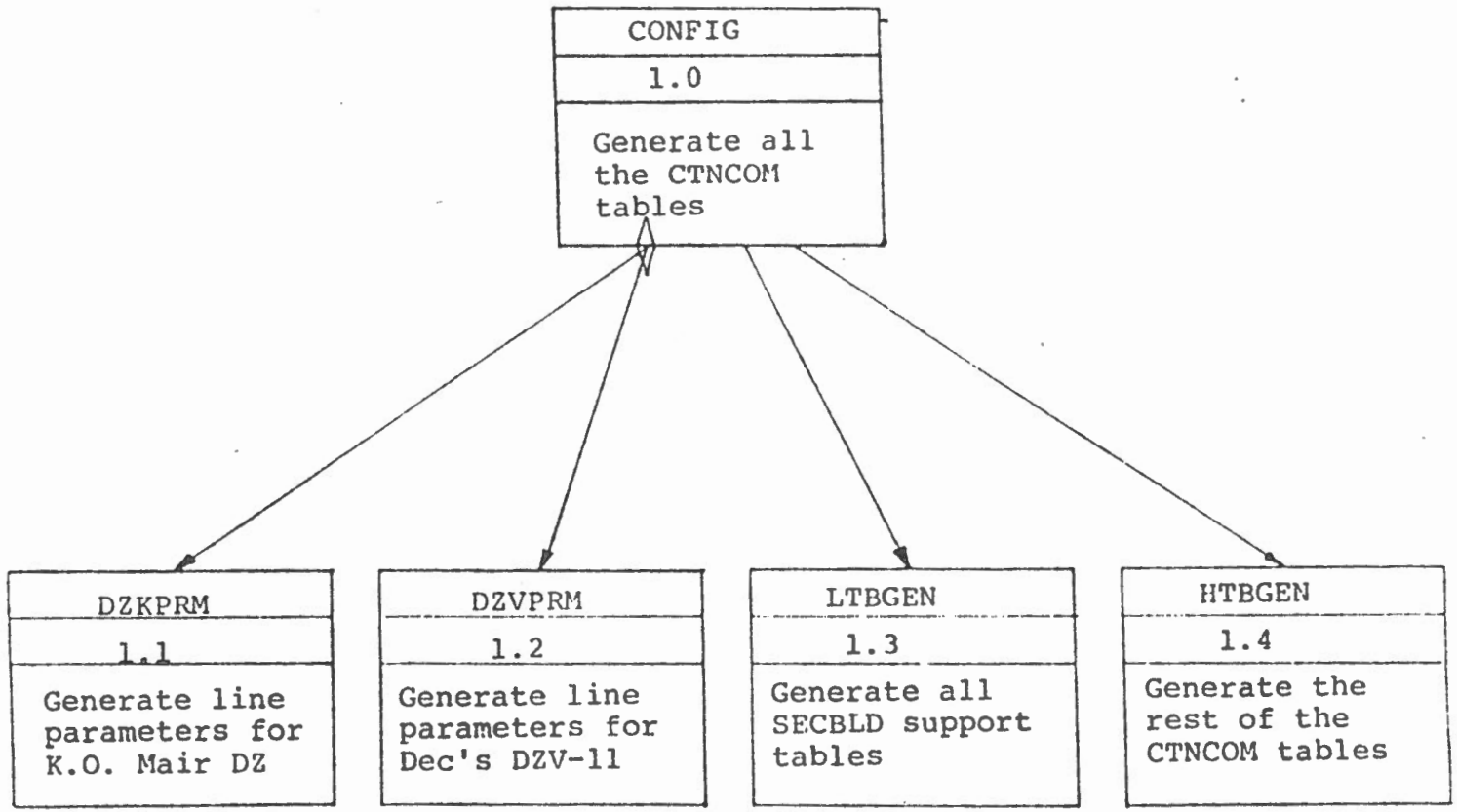


Figure 18: Config

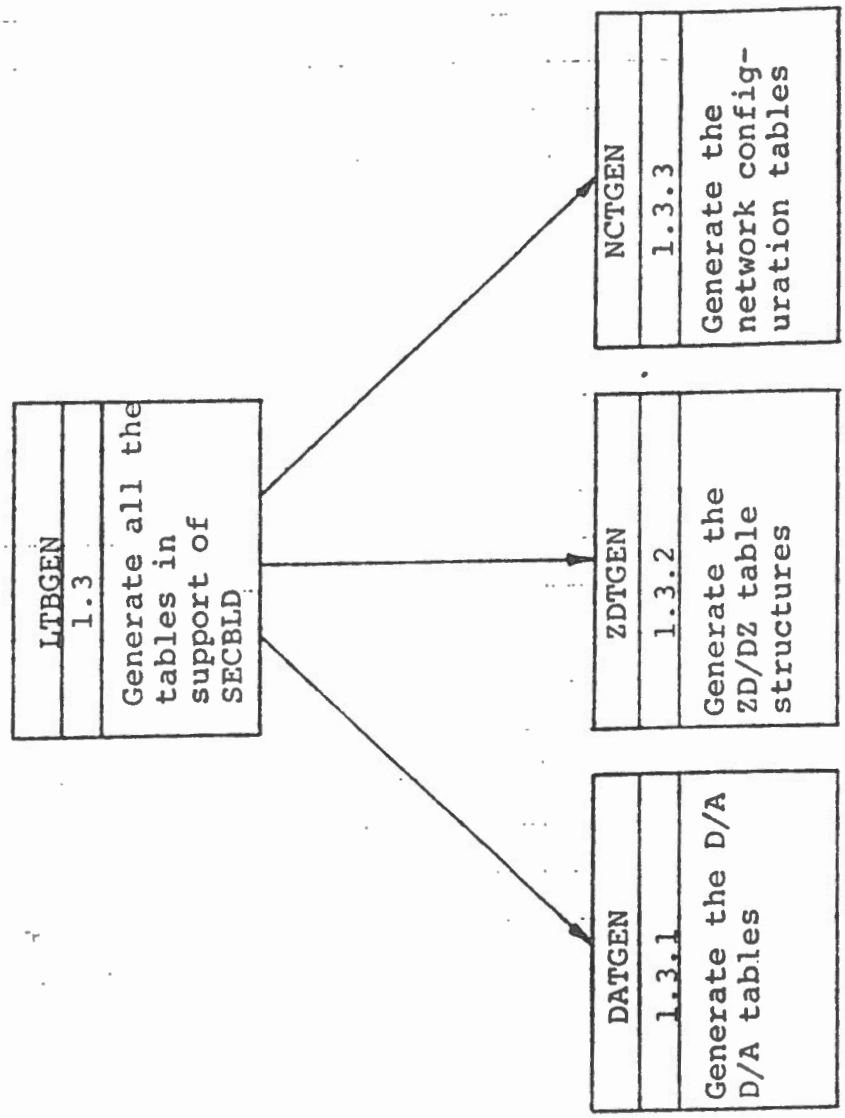


Figure 19: LTBGEN

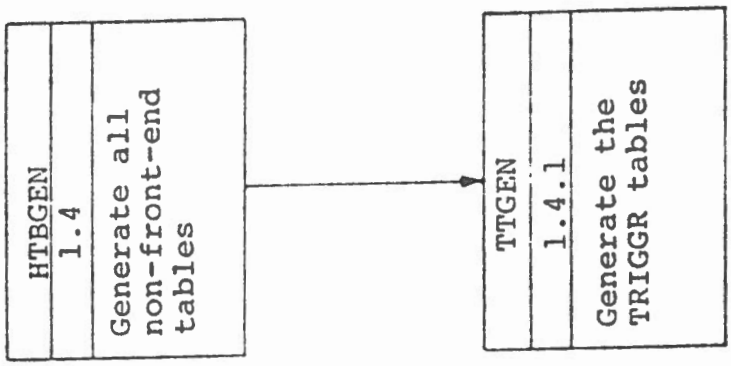


Figure 20: HTBGEN

A Description of Standard CDTSN Event Detection Algorithm

```
for each triggerable component
   for each one-second data buffer
      for every second sample (i.e., decimation by 2)
         convert sample from field Binary Gain Ranged to I*2 representation
            with clipping at +/- 15 bits
         compute absolute value of the second difference (SD):

            (|SD| = |x(i-1) + x(i+1) - 2*x(i)|)

         if |SD| exceeds a threshold = n1*LTSD   where n1 = 8 and LTSD is
         the long-term second difference
            if sample (spike) occurs between two valid samples
               interpolate new sample value to replace spike:

                  x(i) = (x(i-1) + x(i+1)) /2

            else, update the LTSD:

               LTSD(i) = LTSD(i-1) + (|SD| - LTSD(i-1)) /n2   where n2 = 8

         convert current sample from I*2 to DEC R*4 representation
         perform fourth-order digital bandpass detection filtering by
            cascading the output of two passes of a second-order recursive
            filter (BIPOLE):

               y(i) = (x(i) - x(i-1) - y(i-2) + y(i-1)*k1) * k2

               z(i) = (y(i) - y(i-1) - z(i-2) + z(i-1)*k3) * k4

         compute (update) short-term average (STA) using a first-order
            digital filter (LOPASS), essentially an exponentially weighted
            sum of previous values:

               STA(i) = (|z(i)| - STA(i-1))*k5 + STA(i-1)

      end_for every second sample
```

```
      compute (update) long-term average (LTA) using a first-order
         digital filter (LOPASS), again an exponentially weighted
         sum of previous values:

         LTA(i) = (STA(i) - LTA(i-1))*k6 + LTA(i-1)

      if this component not currently triggered
         compute trigger threshold = LTA*factor where 2 < factor < 5 typically
         if STA exceeds threshold
            set flag to declare new trigger
            increment triggered component count
      else, if currently triggered
         if STA still exceeds LTA
            maintain triggered status
         else, if STA < LTA
            clear flag declaring trigger finished
            decrement triggered component count
            if this component's ration has expired
               set flag to disable trigger on this component
      if a new trigger was found
         output message to console
         decrement trigger ration
   end_for each one-second data buffer
end_for each triggerable component
```

On startup, the LTSD is initialized to a value of 30; the STA is initialized to zero; the LTA is initialized to 200.